

**Rational Software Development Platform**



# **VisualAge Generator to EGL マイグレーション・ガイド**

**バージョン 6 リリース 01**



**Rational Software Development Platform**



# **VisualAge Generator to EGL マイグレーション・ガイド**

**バージョン 6 リリース 01**

ご注意

本書および本書で紹介する製品をご使用になる前に、467 ページの『特記事項』に記載されている情報をお読みください。

本マニュアルに関するご意見やご感想は、次の URL からお送りください。今後の参考にさせていただきます。

<http://www.ibm.com/jp/manuals/main/mail.html>

なお、日本 IBM 発行のマニュアルはインターネット経由でもご購入いただけます。詳しくは

<http://www.ibm.com/jp/manuals/> の「ご注文について」をご覧ください。

(URL は、変更になる場合があります)

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原 典： SC31-6830-03  
Rational Software Development Platform  
VisualAge Generator to EGL Migration Guide  
Version 6 Release 01

発 行： 日本アイ・ビー・エム株式会社

担 当： ナショナル・ランゲージ・サポート

第1刷 2005.8

この文書では、平成明朝体™W3、平成明朝体™W7、平成明朝体™W9、平成角ゴシック体™W3、平成角ゴシック体™W5、および平成角ゴシック体™W7を使用しています。この(書体\*)は、(財)日本規格協会と使用契約を締結し使用しているものです。フォントとして無断複製することは禁止されています。

注\* 平成明朝体™W3、平成明朝体™W7、平成明朝体™W9、平成角ゴシック体™W3、  
平成角ゴシック体™W5、平成角ゴシック体™W7

© Copyright International Business Machines Corporation 2004, 2005. All rights reserved.

© Copyright IBM Japan 2005

---

## まえがき

本書は、VisualAge Generator 4.5 から Enterprise Generation Language (EGL) にマイグレーションする必要があるユーザーを対象にしています。

---

## 本書の対象読者

本書は、VisualAge Generator 4.5 から Enterprise Generation Language (EGL) にコードをマイグレーションする必要があるプログラマー、またはシステム管理者を対象にしています。

---

## 関連情報

関連情報は、次のどちらかまたは両方のフォーマットで提供されています。

- 製品 CD-ROM に収録のオンライン・ブック・ファイル (.pdf)。マニュアルをオンラインで表示して、必要なページを印刷するには、Adobe Acrobat Reader を使用します。
- 製品 CD-ROM に収録の HTML ファイル (.htm)。

本書の最新版は、次の Web サイトからオンライン・ブック・ファイルとして入手できます。

<http://www.ibm.com/developerworks/rational/library/egl/doc.html>



# 目次

まえがき	iii
本書の対象読者	iii
関連情報	iii

## 第 1 部 マイグレーションの概要 . . . 1

### 第 1 章 マイグレーションの概要 . . . 3

本書で使用される用語	3
マイグレーションを必要とする EGL の新機能	3
マイグレーションの計画	4
EGL にマイグレーションできるかどうかの判別	8
EGL で使用できない VisualAge Generator 機能	10
用語の違い	11

### 第 2 章 マイグレーション・ツールの考え方 . . . 17

VisualAge Generator to EGL マイグレーション・ツールの概要	18
マイグレーション・ツールの用語	19
ステージ 1 の詳細	20
ステージ 2 の詳細	23
ステージ 3 の詳細	24
単一ファイル・マイグレーションの概要	26
マイグレーションの考慮事項	29
EGL 構文の厳密さ	29
パーツ名が解決される時期と方法	31
共通コードのシナリオ	31
VisualAge Generator to EGL マイグレーション・ツールが使用する技法	35
技法の概要	35
エディターとビルド記述子の設定	36
プログラム・プロパティ	37
EGL のビルド・パスと import ステートメント	38
containerContextDependent プロパティ	40
EGL の予約語リスト	41
EGL ファイル内でのパーツの配置	42
プログラムを使用したマイグレーション	46
関連パーツを使用したマイグレーション	47
関連パーツを使用しないマイグレーション	48
ファイルの上書きとマージ	49
一般規則	52
EGL 5.1.2 以降の VAGen マイグレーション・ツールの新機能	55
EGL 6.0 iFix 以降の VAGen マイグレーション・ツールの新機能	56
EGL 6.0.0.1 以降の VAGen マイグレーション・ツールの新機能	56
マイグレーション・ツールに関する既知の制限事項	57
一般	57
Java および Smalltalk のステージ 1	57

ステージ 2 および 3	57
構文のマイグレーション	58

## 第 3 章 未確定状態の処理 . . . 59

データ項目の未確定状態の処理	59
偶数の長さの PACK データ項目	59
共用編集とメッセージ	61
共用データ項目のマッピング編集ルーチン	63
共用データ項目の充てん文字	64
レコードの未確定状態の処理	65
再定義レコード	65
レコード内のレベル 77 項目	66
代替仕様レコード	68
同じレコード名をもつ異なる定義	70
予約語と UI レコード名	71
テーブルの未確定状態の処理	72
予約語とテーブル名	72
マップ・グループとマップの未確定状態の処理	73
予約語と formGroup 名	73
マップ・グループと formGroup の要件	74
浮動域と開始位置	75
マップ・グループ、マップ、および装置サイズ	76
マップ名とヘルプ・マップ名	77
数値変数フィールド	79
マップ変数フィールドと編集ルーチン	80
マップ・フィールドと数値ハードウェア属性	81
マップ配列と属性	82
名前なしマップ変数フィールド	83
無保護マップ定数	84
row=0, column=0 にあるフィールド	84
プログラムの未確定状態の処理	86
プログラム名と予約語	86
プログラム内の暗黙データ項目	86
関連プログラム・パーツ	87
呼び出しパラメーター・リストに EZEDLPCB が含まれるプログラム	89
マイグレーションに必要な中間変数	90
関数 (入出力ステートメントを含む) の未確定状態の処理	92
マップの DISPLAY ステートメント	92
入出力エラー・ルーチン	93
SQL 入出力ステートメント	95
SQL 入出力と必須 SQL 文節の欠落	97
SQL 入出力と !itemColumnName	99
複数の更新を行う SQL 入出力	100
DL/I I/O および比較値項目	101
その他のステートメントの未確定状態の処理	103
ステートメントの暗黙データ項目	103
ステートメントのレベル 77 項目	103
代入ステートメント	103
FIND ステートメント	104

RETR ステートメント . . . . .	105
SET map PAGE ステートメント . . . . .	106
SET mapItem 属性 . . . . .	106
IN がリテラルかスカラーかの検査 . . . . .	107
SQL 項目とマップ項目が NULL かどうかの検査 . . . . .	108
入出力エラー値 UNQ および DUP . . . . .	110
入出力エラー値 LOK . . . . .	112
EZE ワードの未確定状態の処理 . . . . .	113
EZELTERM . . . . .	113
EZESYS . . . . .	114
EZEWAIT . . . . .	116

## 第 2 部 VisualAge Generator 4.5 on Java から EGL へのマイグレーション . . . . . 119

### 第 4 章 ステージ 1 — Java からの抽出 . . . . . 121

VisualAge for Java へのステージ 1 マイグレーション・ツールのインストール . . . . .	121
マイグレーション・フィーチャーの追加 . . . . .	122
マイグレーション・データベースの作成 . . . . .	122
ステージ 1 設定の実行 . . . . .	123
「プランの作成 (Build Plans)」ページ . . . . .	124
「マッピング (Mapping)」ページ . . . . .	127
「名前変更」ページ . . . . .	129
「実行 (Execution)」ページ . . . . .	130
MigPreferences.xml ファイルのサンプル . . . . .	133
ステージ 1 ツールを実行する前に — ヒント . . . . .	134
パフォーマンスの向上 . . . . .	135
ワークスペースの保管 . . . . .	135
ステージ 1 ツールの実行 . . . . .	136
マイグレーション・プランと上位 PLP プロジェクト . . . . .	137
上位 PLP プロジェクトの作成 . . . . .	138
マイグレーション・プラン・ファイルの手動作成 . . . . .	139

## 第 3 部 VisualAge Generator 4.5 on Smalltalk から EGL へのマイグレーション . . . . . 143

### 第 5 章 ステージ 1 — Smalltalk からの抽出 . . . . . 145

VisualAge Smalltalk へのステージ 1 マイグレーション・ツールのインストール . . . . .	145
マイグレーション・フィーチャーのロード . . . . .	146
マイグレーション・データベースの作成 . . . . .	147
ステージ 1 設定の実行 . . . . .	147
「プランの作成 (Build Plans)」ページ . . . . .	148
「マッピング (Mapping)」ページ . . . . .	150
「名前変更」ページ . . . . .	153
「実行 (Execution)」ページ . . . . .	154
MigPreferences.xml ファイルのサンプル . . . . .	155

設定からのファイル名の取得 . . . . .	157
ステージ 1 ツールを実行する前に — ヒント . . . . .	158
パフォーマンスの向上 . . . . .	158
イメージの保管 . . . . .	159
ステージ 1 マイグレーション・ツールの実行 . . . . .	159
マイグレーション・プランと上位構成マップ . . . . .	162
上位構成マップの作成 . . . . .	163
マイグレーション・プラン・ファイルの手動作成 . . . . .	164

## 第 4 部 ステージ 2 および 3 — 共通のマイグレーション・ステップ . . . 167

### 第 6 章 ステージ 2 — EGL 構文への変換 . . . . . 169

ワークベンチの設定 . . . . .	169
始動パラメーター . . . . .	169
必要な EGL 設定 . . . . .	170
推奨される設定 . . . . .	170
VAGen マイグレーションの設定 . . . . .	171
その他の推奨設定 . . . . .	177
ステージ 2 VAGen マイグレーション・ファイルの設定 . . . . .	178
ステージ 2 の実行 . . . . .	183
ユーザー・インターフェースからのステージ 2 の実行 . . . . .	183
バッチ・モードでのステージ 2 の実行 . . . . .	184

### 第 7 章 ステージ 3 — インポート . . . 187

ステージ 3 ツールの実行 . . . . .	187
バッチ・モードでのステージ 3 の実行 . . . . .	190
一時ディレクトリーに書き込まれたマイグレーション・セットの使用 . . . . .	191

### 第 8 章 単一ファイル・モードでのマイグレーションの実行 . . . . . 193

ユーザー・インターフェースを使用した単一ファイル・マイグレーションの実行 . . . . .	194
バッチ・モードを使用した単一ファイル・マイグレーションの実行 . . . . .	195

## 第 5 部 マイグレーションの完了 199

### 第 9 章 マイグレーションの完了 . . . 201

設定のエクスポート . . . . .	201
EGL プロジェクトとパッケージに関するベースラインの保管 . . . . .	202
単一ファイル・マイグレーションを完了するための予備手順 . . . . .	202
ステージ 1 から 3 のマイグレーションと単一ファイル・マイグレーションに共通する手順 . . . . .	203
EGL ソース・コードの検討 . . . . .	203
EGL ビルド記述子パーツの検討 . . . . .	203
EGL リンケージ・オプション・パーツの検討 . . . . .	207
EGL リソース関連パーツの検討 . . . . .	209



テンプレートとして使用するバインド制御パーツ の設定 . . . . .	209
プログラム固有のバインド制御パーツの設定 . . . . .	211
リンク・エディット・コマンドの検討 . . . . .	212
ご使用の VGWebTransactions の検討 . . . . .	212
デバッグの準備 . . . . .	213
COBOL 生成を行う場合の生成とテスト . . . . .	214
Java 生成を行う場合の生成とテスト . . . . .	215
標準の検討 . . . . .	215
VisualAge Generator 互換モードの使用を中止す る場合 . . . . .	216

## 第 6 部 言語と実行時に関する違い 223

### 第 10 章 言語と実行時に関する違い 225

言語に関する違い . . . . .	225
実行時の違い . . . . .	225
一般的な違い . . . . .	225
デバッグに関する違い . . . . .	226
生成される COBOL に関する違い . . . . .	228
生成される Java に関する違い . . . . .	228
ホスト環境とワークステーション環境の違い . . . . .	229
分散 CICS 環境とネイティブ・ワークステーシ ョン環境の違い . . . . .	229
生成される C++ と生成される Java の違い . . . . .	233

## 第 7 部 付録 . . . . . 237

### 付録 A. 予約語 . . . . . 239

EGL 予約語 . . . . .	239
EGL 列挙型ワード . . . . .	239
SQL 予約語 . . . . .	243
特殊な処理を必要とする SQL 予約語 . . . . .	244
Java 予約語 . . . . .	245

### 付録 B. VisualAge Generator と EGL の言語エレメントの関係 . . . . . 247

一般的な構文規則 . . . . .	248
データ項目 . . . . .	249
レコード . . . . .	257
テーブル . . . . .	274
マップ・グループ . . . . .	276
マップ . . . . .	280
プログラム . . . . .	297
関数 . . . . .	305
ステートメント . . . . .	326
EZE ワード . . . . .	342
プログラム・フローの EZE ワード . . . . .	342
SQL EZE ワード . . . . .	344
DL/I EZE ワード . . . . .	345
日付と時刻の EZE ワード . . . . .	346
その他のデータの EZE ワード . . . . .	347
一般的な関数の EZE ワード . . . . .	350
ストリング EZE ワード . . . . .	351
数学 EZE ワード . . . . .	351

ユーザー・インターフェースの EZE ワード . . . . .	352
オブジェクト・スクリプトの EZE ワード . . . . .	353
サービス・ルーチン . . . . .	353
PSB . . . . .	354
制御パーツ . . . . .	357
生成オプション・パーツ . . . . .	359
リンケージ・テーブル・パーツ . . . . .	379
リソース関連パーツ . . . . .	388
リンク・エディット・パーツ . . . . .	392
バインド制御パーツ . . . . .	393
シンボリック・パラメーター . . . . .	393

### 付録 C. マイグレーション・ツールから のメッセージ . . . . . 397

VisualAge Generator から EGL マイグレーション・ ツールへのメッセージ — ステージ 1 . . . . .	397
ステージ 1 共通メッセージ . . . . .	397
VisualAge for Java のステージ 1 . . . . .	401
VisualAge Smalltalk のステージ 1 . . . . .	403
VisualAge Generator から EGL マイグレーション・ ツールへのメッセージ — ステージ 2 . . . . .	405
VisualAge Generator から EGL マイグレーション・ ツールへのメッセージ — ステージ 3 . . . . .	426

### 付録 D. 「問題」ビューのメッセージ 427

### 付録 E. 「問題」ビューの IWN.xxx メ ッセージ . . . . . 433

IWN.VAL メッセージ . . . . .	433
IWN.XML メッセージ . . . . .	437
JSP の Java メッセージ . . . . .	438

### 付録 F. 外部ソース形式の誤りが原因で EGL の作成に問題が生じる状態 . . . . . 439

### 付録 G. マイグレーション・データベー ス . . . . . 441

DB2 マイグレーション・データベースの作成 . . . . .	441
DB2 7.2 に対する JDBC レベルの設定 . . . . .	441
DB2 8.1 以上に対する JDBC レベルの設定 . . . . .	441
Windows XP 上での DB2 の使用 . . . . .	441
マイグレーション・データベースの作成 . . . . .	441
マイグレーション・データベースのリセット . . . . .	443
DB2 を使用したリモート・データベースのカタロ グ . . . . .	444
DB2 を使用したリモート・データベースのアンカ タログ . . . . .	445
便利な照会 . . . . .	446

### 付録 H. マイグレーション・ツールのパ フォーマンス . . . . . 449

プロジェクト、パッケージ、パーツ、およびプログ ラムの数 . . . . .	450
マイグレーション・セットおよびその他のマイグレ ーション・オプションの数 . . . . .	451

プロセッサ速度 . . . . .	452
演算部の行数 . . . . .	453
ステージ 1 の新規 Java ワークスペース . . . . .	454
ディスク・スペース要件 . . . . .	454

## 付録 I. 前のバージョンのマイグレーション・ツールを使用してマイグレーションを行った場合に必要の変更 . . . . . 457

一般的な変更 . . . . .	457
@ 記号による変更 . . . . .	457
一部の EZE ストリング関数用の追加 EGL 置換表現 . . . . .	457
IMS および DL/I サポートによる変更 . . . . .	458
プログラム・パーツ . . . . .	458
PSB パーツおよび DL/I セグメント・レコード関数入出力 - PSB 名、データベース ID、親の スキャン、更新のスキャン、および SSA . . . . .	459

EZEDL* 特殊機能語および CSPTDLI サービス・ルーチン . . . . .	460
生成オプション・パーツ . . . . .	461
リンケージ・テーブル・パーツ . . . . .	462
リソース関連パーツ . . . . .	462
Web トランザクション・サポートによる変更 . . . . .	463
DataItem パーツ - ヘルプおよびラベルのテキスト . . . . .	463
Web トランザクション・プログラムおよび UI レコード・パーツ . . . . .	464
XFER ステートメント . . . . .	464
生成オプション・パーツ . . . . .	464

## 特記事項. . . . . 467

商標 . . . . .	469
--------------	-----

## 索引 . . . . . 471

---

## 第 1 部 マイグレーションの概要



---

## 第 1 章 マイグレーションの概要

Enterprise Generation Language (EGL) コンポーネントを組み込んだ Rational® Developer または WebSphere® Developer 製品は、VisualAge Generator の後継製品です。EGL に転換するためには、VisualAge Generator (VAGen) ソース・コードをマイグレーションする必要があります。

本マイグレーション・ガイドでは、マイグレーションの計画、マイグレーション・ツールを使用したソース・コードの変換、およびマイグレーション・ツールの実行後にマイグレーションを完了するために必要なその他の手順について説明します。

本マイグレーション・ガイドのほかに、追加情報や最新情報について次の資料を確認してください。

- EGL のオンライン・ヘルプ・システム。
- *EGL Reference Guide*。
- VisualAge Generator に関する Web サイトとニュース・グループ。Web サイトは以下のとおりです。

<http://www.ibm.com/software/awdtools/visgen/>

- EGL に関する Web サイトとニュース・グループ。Web サイトは以下のとおりです。

<http://www.ibm.com/developerworks/rational/products/egl/>

- 使用する製品の Web サイトおよびニュース・グループ。

---

### 本書で使用される用語

EGL は、幾つかの製品のコンポーネントとして出荷されます。本書では、以下の用語を使用しています。

#### 開発者製品

EGL をコンポーネントとして組みこむ製品。以下の製品があります。

- Rational Web Developer
- Rational Application Developer
- WebSphere Development Studio Client Advanced Edition for iSeries™
- WebSphere Developer for zSeries

#### EGL 開発環境

EGL をコンポーネントとして組みこむ製品のいずれかを始動した後に表示されるワークベンチおよび他のウィンドウ。

---

### マイグレーションを必要とする EGL の新機能

EGL は、以下のような、VisualAge® Generator からの主要な変更および機能強化が行われています。

- VAGen 言語の変更。例えば、新しいデータ型、多次元構造化フィールド配列、動的配列、*case* ステートメント、Web サポートの改良など、数多くの機能が拡張されています。
- コンテンツ・アシスト、パーツ作成用のコード・テンプレート、ほとんどのパーツ型用のテキスト・エディターなど、プログラムの開発に使用するユーザー・インターフェースの変更。
- 生成および準備のプロセスの変更。例えば、分散プラットフォームの場合に C++ と Java™ を生成する代わりに Java のみを生成し、COBOL 生成の場合に JCL テンプレートを準備する代わりにビルド・サーバーを使用します。
- ランタイムの変更。z/OS 環境向けのランタイム・サービス用 Enterprise Developer Server の使用も含まれます。
- ライブラリー管理に対する変更。例えば、EGL とのインターフェースに独自のソース・コード・リポジトリを選択する機能などがあります。

VAGen 言語と EGL の違いは広範囲に及びます。以前、システム共通プロダクトまたは VisualAge Generator のバージョンを新バージョンにアップグレードしていたときには、言語の変更は小規模でした。従来のマイグレーション・ツールは、他のパーツと独立して各パーツをマイグレーションできました。一方で、VisualAge Generator to EGL マイグレーション・ツールは、2 つの言語の違いが大きいため、次のことを判別して、他の参照先パーツや関連パーツのコンテキストに従ってそれぞれのパーツをマイグレーションする必要があります。

- 参照先パーツのパーツ型
- 新しい EGL の構文に従って参照側パーツに移動する必要がある情報
- ワークスペース内での参照先パーツのロケーション

このように、1 つのパーツのマイグレーションが他のパーツに依存するという状況を表すために、パーツ間マイグレーション という用語が使用されます。パーツ間マイグレーションは、VAGen 言語から EGL への変換を可能な限り最適に行うために必要です。さらにこのためには、一緒にマイグレーションするパーツのグループを慎重に検討する必要があります。

VisualAge Generator と EGL との相違点、およびパーツ間マイグレーションが必要なことを考慮すると、このマイグレーションはかなりの作業であり、慎重な計画が必要です。

---

## マイグレーションの計画

マイグレーション・プロジェクトを計画する際には、次の作業を検討する必要があります。

- マイグレーションのパイロット・プロジェクトの計画:
  - パイロット・プロジェクトに参加する開発者とシステム・サポート担当者を選出します。
  - パイロット・プロジェクトに使用するソース・コードの小サブセットを選択します。この小サブセットを使用して、環境のセットアップ、およびライブラリー管理の手順とツールを検証します。
  - フィックスパック 4 を適用して VisualAge Generator 4.5 にアップグレードします。VisualAge Generator を VisualAge for Java と組み合わせて使用してい

る場合は、さらに APAR PQ88461 をインストールして VisualAge Generator ユーティリティー機能をアップグレードする必要があります。APAR の修正を入手するには、IBM® サポートにご連絡いただくか、VAGen Web サイト (<http://www-306.ibm.com/software/awdtools/visgen/support>) にアクセスして、「Download」セクションのリンクをたどってください。また、ユーザー個々の状況に応じてさらに必要になる可能性がある VisualAge Generator APAR については、439 ページの『付録 F. 外部ソース形式の誤りが原因で EGL の作成に問題が生じる状態』を参照してください。

注: フィックスパック 5 には、マイグレーション・ツールに必要なすべての APAR が含まれる予定です。

- まだ使用可能でない場合は、DB2® をインストールします。DB2 は、マイグレーション・データベース用に必要です。
- 使用する予定の開発者製品の機能を確認します。必要な機能が含まれていることを確認します。例えば、次のようになります。
  - z/OS 環境用の COBOL の生成を予定している場合は、zSeries® 用 COBOL 生成を組み込んだ開発者製品を使用する必要があります。
  - iSeries の使用を検討している場合は、iSeries 用 COBOL 生成を組み込んだ開発者製品を使用する必要があります。
- 使用する予定の製品の前提条件を確認します。また、ランタイム環境の前提条件を確認します。例えば、次のようになります。
  - z/OS 環境用の COBOL の生成を予定している場合は、Enterprise Developer Server for z/OS 製品の前提条件を確認します。Build Server の前提条件も確認します。
  - VSE 環境用の COBOL の生成を予定している場合は、「*VisualAge Generator EGL Plug-in for VSE Reference Manual*」に記載された前提条件を確認します。
  - UNIX® システム・サービス (USS) 環境用の Java を生成する予定の場合は、Enterprise Developer Options for z/OS コンポーネントの前提条件を確認します。
  - iSeries 環境での生成を予定している場合はご使用の開発者製品のランタイム・コンポーネントの前提条件を必ず確認してください。
  - ワークステーション環境用の Java を生成する予定の場合は、使用する予定の開発者製品の前提条件を必ず確認してください。
- パイロット・プロジェクトを実施するチームの研修を行います。
  - Developer 製品環境
  - EGL 言語
  - VisualAge Generator to EGL マイグレーション・ツール
- 次のことを行うためのパイロット・プロジェクト計画を実施します。
  - パイロット・チーム用の開発者製品をインストールします。インストール先のマシンの地域設定値が、VAGen プログラムの開発に使用した設定値と同じであることを確認してください。例えば、次のようになります。

- ドイツ語マシン上で VAGen プログラムを開発した場合は、ドイツ語マシンに開発者製品をインストールする必要があります。これにより、小数点としてコンマが使用され、ドイツ語のウムラウト文字が正しくマイグレーションされるようになります。
- 中国語マシン上で VAGen プログラムを開発した場合は、同じコード・ページを使用する中国語マシンに開発者製品をインストールする必要があります。これにより、DBCS 文字が正しくマイグレーションされるようになります。
- パイロットのコード・セットに対して VAGen マイグレーション・ツールを実行します。マイグレーション・ツールの詳細については、次のセクションを参照してください。
  - 17 ページの『第 2 章 マイグレーション・ツールの考え方』
  - 119 ページの『第 2 部 VisualAge Generator 4.5 on Java から EGL へのマイグレーション』
  - 143 ページの『第 3 部 VisualAge Generator 4.5 on Smalltalk から EGL へのマイグレーション』
  - 167 ページの『第 4 部 ステージ 2 および 3 — 共通のマイグレーション・ステップ』
  - 199 ページの『第 5 部 マイグレーションの完了』
- ライブラリー管理プロセスの作成:
  - 開発用ワークステーションからのアクセスなどの、ソース・コード・リポジトリを選択し、インストールします。
  - 企業標準と選択したソース・コード・リポジトリが連動する変更管理手順を定義します。
  - 変更管理手順に必要なすべてのツールを開発します。これには、以下のものがあります。
    - チェックインおよびチェックアウト手順
    - バージョン管理手順
    - パッチ生成を使用する場合は、ソース・コード・リポジトリからソース・コードを検索し、ワークスペースまたはディレクトリ構造をロードするツール。
- iSeries COBOL ターゲット環境:
  - 「*EGL Server Guide for iSeries*」の指示のとおりに行います。
- z/OS COBOL ターゲット環境:
  - Enterprise Developer Server for z/OS V5.0 製品の前提条件 (COBOL コンパイラおよびランタイムに対する変更を含む) をインストールします。
  - Enterprise Developer Server for z/OS V5.0 製品をインストールします。
  - Enterprise Developer Server for z/OS に対する最新の PTF をインストールします。
  - ビルド・サーバーおよび必須 PTF をすべてインストールします。また、擬似 JCL ビルド・スクリプトをカスタマイズします。ビルド・サーバーは Enterprise Developer Options for z/OS® に同梱されています。
- VSE COBOL ターゲット環境:



- 「VisualAge Generator EGL Plug-in for VSE Reference Manual」に記載された指示に従います。
- Java ターゲット環境:
  - プラットフォームを変更する場合 (例えば、Windows® CICS® から Windows ネイティブへ) は、ランタイム・プラットフォームの相違点を検討します。これによるコード変更を行います。オリジナル・プラットフォームおよび新規ランタイム・プラットフォームに応じて、225 ページの『第 10 章 言語と実行時に関する違い』内の適切なセクションを参照し、相違点のリストを確認してください。C++ 生成から Java 生成に変更する場合も、同じ章を参照してください。
  - 現在 ODBC サポートを使用している場合は、ベンダーから JDBC サポートを取得します。
- プログラムの生成と準備:
  - ビルド・パーツ (ビルド記述子、リンケージ・オプション、リソース関連、リンク・エディット、バインド制御パーツ) を確認、修正します。VAGen マイグレーション・ツールによって処理できない変更の詳細については、ランタイム環境で使われるビルド・パーツに応じて、201 ページの『第 9 章 マイグレーションの完了』内の適当なセクションを参照してください。
  - 必要により、EGL バッチ生成サーバー・マシンをビルドします。これには、ソース・コード・リポジトリを使用し、また生成に必要なすべてのパーツが入ったディレクトリをロードするツールを作成する必要があります。
- テスト:
  - 使用する開発環境をテストして、プログラムを正常にデバッグできるか確認します。開発環境でデバッグを行うには、DB2、リモート VSAM ファイル、およびランタイム環境のみで稼働する非 EGL プログラムへのアクセスが必要になる場合があります。
  - 少なくともプログラムの代表例をテストして、すべてのランタイム相違点を把握したことを確認します。相違点のリストについては、225 ページの『実行時の違い』を参照してください。
  - EGL ソース・コードに行う可能性のある標準的な変更を使用して、ライブラリー管理手順とツールをテストします。それぞれのターゲット環境ごとに、共通のコード、書式、dataTable、およびプログラムを変更する手順を必ずテストしてください。また、それぞれのターゲット環境ごとに、共通のコード、書式、dataTable、およびプログラムを追加する手順もテストしてください。
- パイロット・プロジェクトの結果に基づいて、ライブラリー管理手順およびツールを改善します。
- 以下の事項を含むパイロット・プロジェクトでの調査結果を文書化します。
  - 必要なコード変更 (特に、ターゲット環境を変更する場合)。
  - 開発者がすべての個人用ビルド記述子パーツに行う必要のある変更。
  - パイロット・プロジェクト中に発生した問題に基づいた、開発者に特に役に立つマイグレーション・ガイドのセクションの参照。
  - エンド・ユーザーが気付くランタイム動作の変更。

- パイロット・プロジェクトで分かったことに基づいて、マイグレーションを完了するための計画の作成
- プロジェクト外の開発者への教育の実施:
  - Developer 製品環境
  - EGL 言語
  - 新規のソース・コード・リポジトリ
  - 新規のライブラリー管理プロセス
  - 新規の生成プロセス

---

## EGL にマイグレーションできるかどうかの判別

EGL は、VisualAge Generator のお客様がマイグレーションする必要がある、Rational Developer または WebSphere Developer 製品の戦略的コンポーネントです。ただし EGL サポートは、VisualAge Generator Developer 4.5 (VAGen) がサポートするすべての機能とプラットフォームを完全に置き換えるものではありません。ターゲット環境および VisualAge Generator で開発したプログラムのタイプによっては、EGL の今後のリリースを待つ必要があります。

次のリストに、EGL でサポートされる VisualAge Generator ターゲット環境を示します。

- MVS™ CICS
- MVS Batch
- IMS/VS
- IMS BMP
- Unix システム・サービス
- Windows ネイティブ
- AIX® ネイティブ
- Intel® プラットフォーム上の Linux™
- HP-UX
- Solaris
- iSeries

さらに、VisualAge Generator EGL Plug-in for VSE V1.0 (プログラム番号 5724-L93) は以下のターゲット環境をサポートします。

- VSE CICS
- VSE Batch

注: VisualAge Generator は一部のプラットフォーム上で Java と C++ を生成しますが、EGL は Java のみを生成します。

これらのサポート環境のその他の考慮事項については、以下を参照してください。

- EGL へのマイグレーションに関する特別な考慮事項 — ファイルおよびデータベース・アクセス、9 ページの表 1
- EGL へのマイグレーションに関する特別な考慮事項 — ユーザー・インターフェース、9 ページの表 2

- 10 ページの『EGL で使用できない VisualAge Generator 機能』

次の表に、サポート環境に関する特別な考慮事項をリストします。

表 1. EGL へのマイグレーションに関する特別な考慮事項 — ファイルおよびデータベース・アクセス

VAGen ファイルおよびデータベース・アクセス	特別な考慮事項
SQL	EGL でサポートされています。
シリアル、索引付き、および相対レコード	EGL でサポートされています。
メッセージ・キュー・レコード	EGL でサポートされています。
DL/I	z/OS 環境用の COBOL 生成は EGL 内でサポートされています。VSE 環境用のデバッグ、Java 生成、または COBOL 生成はサポートされていません。
GSAM	EGL でサポートされています。
IMS™ メッセージ・キュー	EGL でサポートされています。
Btrieve	EGL ではサポートされていません。
ローカル VSAM	次の場合にサポートされます。 <ul style="list-style-type: none"> <li>• AIX 環境での Java 生成</li> <li>• COBOL 生成</li> </ul> デバッグ、または他の環境での Java 生成の場合はサポートされません。
リモート VSAM	次の場合にサポートされます。 <ul style="list-style-type: none"> <li>• リモート・ファイルが CICS for z/OS、VSE CICS、または iSeries 上にある場合のデバッグ。</li> <li>• リモート・ファイルが CICS for z/OS、VSE CICS、または iSeries 上にある場合の Windows 環境用の Java 生成。</li> <li>• CICS for z/OS または VSE CICS 環境用の COBOL 生成。</li> </ul>

表 2. EGL へのマイグレーションに関する特別な考慮事項 — ユーザー・インターフェース

VAGen ユーザー・インターフェース	特別な考慮事項
テキスト・ユーザー・インターフェース (印刷を含む)	COBOL 生成と Java 生成のどちらの場合も、EGL でサポートされます。
Web トランザクションおよびユーザー・インターフェース (UI) レコード	COBOL 生成と Java 生成のどちらの場合も、環境に応じて EGL でサポートされます。IMS/VS、VSE CICS または iSeries ではサポートされません。

表 2. EGL へのマイグレーションに関する特別な考慮事項 — ユーザー・インターフェース  
(続き)

VAGen ユーザー・インターフェース	特別な考慮事項
VAGen Java ラッパーを使用する JSP と Java サブレット	<ul style="list-style-type: none"> <li>製品に付属の情報を使用して、JSP と Java サブレットを新規の開発者製品にマイグレーションできます。</li> <li>本書「VAGen マイグレーション・ガイド」を使用して、VAGen サーバー・プログラムを EGL にマイグレーションできます。EGL を使用して、Java ラッパーを生成できます。</li> </ul>
フリーフォーム域で VAGen パーツを使用せず、VAGen Java ラッパーを使用する Java GUI アプリケーションまたはアプレット。	<ul style="list-style-type: none"> <li>製品に付属の VisualAge for Java から Java コードをマイグレーションするための情報を使用して、Java アプリケーションまたはアプレットを新規の開発者製品にマイグレーションできます。</li> <li>本書「VAGen マイグレーション・ガイド」を使用して、VAGen サーバー・プログラムを EGL にマイグレーションできます。EGL を使用して、Java ラッパーを生成できます。</li> </ul>
フリーフォーム域で VAGen パーツを使用する Java GUI アプリケーションまたはアプレット	現行リリースではサポートされていません。
Smalltalk GUI ビューまたはビジュアル・パーツ	<p>現行リリースではサポートされていません。</p> <p>VAGen パーツのビューは、Java ベース・ソリューションにマイグレーションする必要があります。EGL には、Smalltalk ベースのソリューションはありません。</p>

## EGL で使用できない VisualAge Generator 機能

表 1 および 2 にリストした特別な考慮事項に加え、次のリストのいずれかの機能が必要な場合は、マイグレーションを現時点で行うことと将来的に行うことから受ける影響を評価する必要があります。

- 生成時に作成されるプログラムのリストなど、特殊なエディターおよびリスト
- 特殊な機能
  - 選択したパーツのセットから参照を検索し、その参照リストをプログラムに関連する項目に限定する機能
  - パーツ型別またはサブタイプ別によるパーツのフィルター操作。EGL には検索機能が用意されているので、特定のパーツ型またはサブタイプを検索できます。
- 以下のような特殊なデバッグ・サポート
  - DL/I データベース I/O
  - IMS/VS 環境内のプログラム呼び出し
- 以下の環境における Web トランザクション・サポート
  - IMS/VS
  - VSE CICS

- iSeries
- VisualAge Generator テンプレート。
- Java 生成を使用する予定の場合は、ネイティブ・ランタイム環境に変換できない CICS 固有の機能を使用しているかどうかを判別してください。相違点の詳細については、229 ページの『分散 CICS 環境とネイティブ・ワークステーション環境の違い』を参照してください。

## 用語の違い

VisualAge Generator Developer on Java、VisualAge Generator Developer on Smalltalk、および EGL は、すべて異なる用語を使用します。VAGen 用語と EGL 用語の関連が分かりやすいように、次の 6 つの表に 3 種類の用語を示します。

表 3. コード編成用語の相違点

VisualAge Generator on Java	VisualAge Generator on Smalltalk	エンタープライズ開発言語 (EGL)
ワークスペース	イメージ	ワークスペース
プロジェクト	構成マップ	EGL プロジェクト
パッケージ	アプリケーション	1 つ以上の EGL ファイルを含む EGL ソース・フォルダーおよび EGL パッケージ
(同等の概念なし)	(同等の概念なし)	ファイル (一般に、Java パッケージまたは Smalltalk アプリケーションは複数のファイルに分割される)。EGL ファイルには、1 つ以上のパーツ型の EGL パーツ (複数可) が格納されます。
クラスまたはタイプ	クラス	EGL パーツ型
メソッドまたはメンバー	メソッド	(同等の概念なし)
VAGen パーツ	VAGen パーツ	ファイル内の EGL パーツ

表 4. VAGen のパーツと概念に関する用語の違い

VisualAge Generator on Java	VisualAge Generator on Smalltalk	EGL
共用データ項目	共用データ項目	DataItem パーツに対する型定義
非共用データ項目	非共用データ項目	プリミティブ項目定義
データ項目パーツ	データ項目パーツ	DataItem パーツ
レコード・パーツ	レコード・パーツ	レコード・パーツ 注: マイグレーション・ツールは、VAGen の振る舞いを保持するために、VAGen のレコード定義をすべて EGL の固定レコードに変換します。
PSB パーツ	PSB パーツ	PSBRecord パーツ

表 4. VAGen のパーツと概念に関する用語の違い (続き)

VisualAge Generator on Java	VisualAge Generator on Smalltalk	EGL
ユーザー・インターフェース (UI) レコード	ユーザー・インターフェース (UI) レコード	VGUI レコード
構造項目 (レコード内のフィールドの構造)	構造項目 (レコード内のフィールドの構造)	構造化フィールド
配列 (レコードまたはマップに複数回出現する項目)	配列 (レコードまたはマップに複数回出現する項目)	構造化フィールド配列
マップ・グループ・パーツ	マップ・グループ・パーツ	FormGroup
マップ・パーツ: <ul style="list-style-type: none"> <li>表示マップ</li> <li>プリンター・マップ</li> </ul>	マップ・パーツ: <ul style="list-style-type: none"> <li>表示マップ</li> <li>プリンター・マップ</li> </ul>	書式: <ul style="list-style-type: none"> <li>textForm</li> <li>printForm</li> </ul>
入出力オプションと入出力オブジェクト	入出力オプションと入出力オブジェクト	EGL 入出力ステートメント
Java アプリケーションまたはアプレット (GUI)	Smalltalk ビューまたはビジュアル・パーツ (GUI)	<ul style="list-style-type: none"> <li>Smalltalk ビューおよびビジュアル・パーツはサポートされていません。</li> <li>Java アプリケーションとアプレットは、フリー・フォーム域で VAGen パーツを使用していない場合にサポートされます。フリー・フォーム域で VAGen パーツを使用していた場合、Java アプリケーションまたはアプレットは現行リリースでサポートされません。</li> </ul>
生成オプション・パーツ	生成オプション・パーツ	ビルド記述子パーツ
生成オプション	生成オプション	ビルド記述子オプション
リンケージ・テーブル・パーツ	リンケージ・テーブル・パーツ	リンケージ・オプション・パーツ

表 5. VAGen と IDE Windows の用語の違い

VisualAge Generator on Java	VisualAge Generator on Smalltalk	EGL
<p>ログ</p> <ul style="list-style-type: none"> <li>エラー・メッセージを表示します</li> <li>ログとワークベンチの両方を閉じた場合のみ、製品は閉じます</li> <li>製品を閉じるとワークスペースは常に保管されます</li> </ul>	<p>システム・トランスクリプト</p> <ul style="list-style-type: none"> <li>エラー・メッセージを表示します</li> <li>システム・トランスクリプトまたは VisualAge Organizer のどちらかを閉じると、製品が閉じます。</li> <li>製品を閉じると、イメージはオプションで保管されます</li> </ul>	<p>コンソール</p> <ul style="list-style-type: none"> <li>メッセージを表示します。「問題」ビュー</li> <li>メッセージを表示します (特に構文の検証に関連するメッセージ)。</li> <li>製品を閉じると、ワークスペースは常に保管されます。</li> </ul>

表 5. VAGen と IDE Windows の用語の違い (続き)

VisualAge Generator on Java	VisualAge Generator on Smalltalk	EGL
ワークベンチ <ul style="list-style-type: none"> <li>ワークスペース内のプロジェクトとパッケージを表示します。</li> </ul>	VisualAge Organizer <ul style="list-style-type: none"> <li>イメージ内のアプリケーションを表示します。</li> </ul>	EGL および Web パースペクティブ: <ul style="list-style-type: none"> <li>「ナビゲーター」ビューと「プロジェクト・エクスプローラー」ビューに、ワークスペース内のプロジェクト、ソース・フォルダー、パッケージ、およびファイルが表示されます。</li> </ul>
スクラップブック	ワークスペース	スクラップブック・ページ・エディター
リポジトリ・エクスプローラー	アプリケーション・エディション・ブラウザー	リポジトリの使用を決定した場合、リポジトリに同等の概念がある場合があります。
VAGen パーツ・ブラウザー <ul style="list-style-type: none"> <li>3 つのペインに、パッケージ、パーツ型、および VAGen パーツが表示されます</li> <li>ブラウザーにはフィルター操作とソートが組み込まれています。</li> </ul>	VAGen パーツ・ブラウザー <ul style="list-style-type: none"> <li>3 つのペインに、アプリケーション、パーツ型、および VAGen パーツが表示されます。</li> <li>ブラウザーにはフィルター操作とソートが組み込まれています。</li> </ul>	EGL および Web パースペクティブ: <ul style="list-style-type: none"> <li>「ナビゲーター」ビューと「プロジェクト・エクスプローラー」ビューに、ワークスペース内のプロジェクト、ソース・フォルダー、パッケージ、およびファイルが表示されます。</li> <li>「アウトライン」ビューに、ファイル内のパーツが表示されます。</li> <li>「EGL パーツ・リスト」ビューではフィルター操作とソートを実行できます。</li> </ul>
VAGen オプション	VAGen 設定	EGL 設定
VAJava オプション	VASmalltalk 設定	その他の製品設定
「参照 (References)」ツールによる、特定のパーツ名またはテキスト・ストリングを使用するパーツの検索	「参照 (References)」ツールによる、特定のパーツ名またはテキスト・ストリングを使用するパーツの検索	EGL 検索またはファイル検索
「関連 (Associates)」ツールによる、特定のパーツが参照するパーツすべての検索	「関連 (Associates)」ツールによる、特定のパーツが参照するパーツすべての検索	EGL パーツ参照

表 6. VAGen のワークスペース管理に関する用語の違い

VisualAge Generator on Java	VisualAge Generator on Smalltalk	EGL
リポジトリ	ライブラリー	なし。使用する製品によって、CVS および Clear Case LT が提供されます。独自のリポジトリ管理システムを選択できます。
追加/削除	ロード/アンロード	リポジトリの使用を決定した場合、リポジトリに同等の概念がある場合があります。
置換	別のエディションのロード	ローカル・ヒストリーとの置換 注: 使用を決定したリポジトリに追加機能が存在することがあります。
次と比較	変更点のブラウズ	ローカル・ヒストリーとの比較 注: 使用を決定したリポジトリに追加機能が存在することがあります。

表 7. VAGen のリポジトリ管理に関する用語の違い

VisualAge Generator on Java	VisualAge Generator on Smalltalk	EGL
管理者	ライブラリー・スーパーバイザー	リポジトリの使用を決定した場合、リポジトリに同等の概念がある場合があります。
リポジトリ管理: • パージ/復元 • 圧縮	ライブラリー管理: • パージ/サルベージ • クローン	リポジトリの使用を決定した場合、リポジトリに同等の概念がある場合があります。

表 8. VAGen ソース・コード管理の用語の相違点

VisualAge Generator on Java	VisualAge Generator on Smalltalk	EGL
所有権: • プロジェクト所有者 • パッケージ所有者 • クラス所有者	所有権: • 構成マップ管理者 • アプリケーション管理者 • クラス所有者	リポジトリの使用を決定した場合、リポジトリに同等の概念がある場合があります。
バージョンとリリース	バージョンとリリース	リポジトリの使用を決定した場合、リポジトリに同等の概念がある場合があります。



表 8. VAGen ソース・コード管理の用語の相違点 (続き)

VisualAge Generator on Java	VisualAge Generator on Smalltalk	EGL
<p>プロジェクト:</p> <ol style="list-style-type: none"> <li>1. プロジェクトは必須。</li> <li>2. VAGen プロジェクト・リスト・パーツが、プロジェクト間の関係を指定します。</li> <li>3. パッケージ所有者は、常にパッケージをプロジェクトにリリースできます。</li> </ol>	<p>構成マップ:</p> <ol style="list-style-type: none"> <li>1. 使用はオプション。</li> <li>2. 必須のマップが、構成マップ間の関係を指定します。</li> <li>3. オプションで、アプリケーションのリリースを委任したり、リリースを構成マップ・マネージャーのみに制限したりすることができます。</li> </ol>	<p>プロジェクト:</p> <ol style="list-style-type: none"> <li>1. プロジェクトは必須。</li> <li>2. プロジェクトの EGL ビルド・パス・プロパティ。ただし、これによってプロジェクトがまとめて自動的にワークスペースにロードされることはありません。</li> <li>3. リポジトリに存在しない限り、同等の概念はありません。</li> </ol>
<p>パッケージ:</p> <ol style="list-style-type: none"> <li>1. 同等の概念なし</li> <li>2. 同等の概念なし</li> <li>3. 同等の概念なし</li> <li>4. グループ・メンバー</li> <li>5. プロジェクトのバージョン管理により、組み込まれたパッケージのバージョン管理が自動的に行われます。</li> </ol>	<p>アプリケーション:</p> <ol style="list-style-type: none"> <li>1. 前提アプリケーション</li> <li>2. サブアプリケーション</li> <li>3. 特権</li> <li>4. グループ・メンバー</li> <li>5. 構成マップのバージョン管理を行う前に、アプリケーションのバージョン管理を行う必要があります。</li> </ol>	<p>フォルダーまたはパッケージ:</p> <ul style="list-style-type: none"> <li>• リポジトリの使用を決定した場合、リポジトリに同等の概念がある場合があります。</li> </ul>
<p>クラスまたはタイプ:</p> <ul style="list-style-type: none"> <li>• パッケージまたはプロジェクトのバージョン管理により、組み込まれたクラスのバージョン管理が自動的に行われます。</li> </ul>	<p>クラス:</p> <ul style="list-style-type: none"> <li>• アプリケーションのバージョン管理を行う前に、クラスのバージョン管理とリリースを行う必要があります。</li> </ul>	<p>EGL パーツ型</p> <ul style="list-style-type: none"> <li>• EGL に同等の概念はありません。</li> </ul>

表 8. VAGen ソース・コード管理の用語の相違点 (続き)

VisualAge Generator on Java	VisualAge Generator on Smalltalk	EGL
<p>VAGen パーツ:</p> <ul style="list-style-type: none"> <li>• それぞれのパーツごとに日時スタンプが付いています。</li> <li>• 重複するパーツ名を含むパッケージをワークスペースに追加できます。</li> <li>• 重複するパーツを見付けるための重複パーツ・ツールが存在します。</li> </ul>	<p>VAGen パーツ:</p> <ul style="list-style-type: none"> <li>• それぞれのパーツごとに日時スタンプが付いています。</li> <li>• 重複するパーツ名を含むアプリケーションは、イメージにロードできません。</li> </ul>	<p>EGL パーツ:</p> <ul style="list-style-type: none"> <li>• パーツは EGL ファイル内にあり、EGL ファイルのみに日時スタンプが付いています。</li> <li>• ワークスペース内に重複するパーツがあっても構いません。EGL は、プロジェクトの EGL ビルド・パス、ファイルの import ステートメント、および <code>containerContextDependent</code> プロパティの組み合わせを使用して、パーツ名の参照を解決するために検索されるネーム・スペースを判別します。パーツ名は、ネーム・スペースの中で固有である必要があります。プロジェクトの EGL ビルド・パスにより、パーツ名を検索する際に対象になる他のプロジェクトが限定されます。ファイルの import ステートメントにより、パーツ名を検索する際に対象になる、EGL ビルド・パスに含まれる他のパッケージまたはパーツ (あるいはその両方) が限定されます。レコードまたは関数に対して <code>containerContextDependent</code> プロパティを指定すると、EGL は、レコードまたは関数を含むファイルではなく、プログラムを含むファイルの EGL ビルド・パスと import ステートメントを使用します。</li> </ul>

---

## 第 2 章 マイグレーション・ツールの考え方

VisualAge Generator to EGL マイグレーション・ツールは、実際には一連の複数のツールからなります。この章では、これらのツールの概要を説明し、ツールが使用している技法について説明します。

VisualAge Generator to EGL マイグレーション・ツールは、主に次のことを目的として設計されています。

- プログラムの振る舞いを、VisualAge Generator から EGL へのマイグレーション後も保持する。
- Java プロジェクトとパッケージの構造を、VisualAge Generator から EGL へのマイグレーション後も保持する (該当する場合)。
- Smalltalk 構成マップとアプリケーションの構造が、VisualAge Generator から EGL へのマイグレーション後も保持されます (該当する場合)。
- サブシステムのインクリメンタル・マイグレーションを、1 つのサブシステムごとに実行する。
- サブシステムの複数バージョンのマイグレーションを実行する。

VisualAge Generator to EGL マイグレーション・ツールの設計は、次のような副次的な目的も考慮しています。

- 可能な限りバッチ・モード処理を使用し、重要なポイントでは、次のステップに進む前にユーザーがマイグレーションの計画を検討する機会があります。
- 複数のプロジェクト・バージョン、および複数のサブシステム間で情報を保持できるように、計画したマイグレーションに関する情報をデータベースに保管します。またこれにより、中間結果をバックアップとして保管できます。これは、リポジトリに多数のパーツがある場合に重要です。
- VAGen ソースをリポジトリから抽出して、マイグレーション・データベースをロードする、一連のサンプル・プログラムがツールのために用意されています。環境を正確に反映するように、オプションでサンプル・プログラムを作り替えることもできます。

VisualAge Generator to EGL マイグレーション・ツールは、次の前提に基づいて設計されています。

- VisualAge Generator 4.5 からのマイグレーションが、VisualAge Generator 4.5 によって生成された外部ソース形式を使用して行われる。
- マイグレーション対象のパーツは、有効な **VisualAge Generator** パーツである。プログラム、テーブル、およびマップ・グループは、VisualAge Generator 4.5 内で検証または生成 (あるいはその両方) できる。

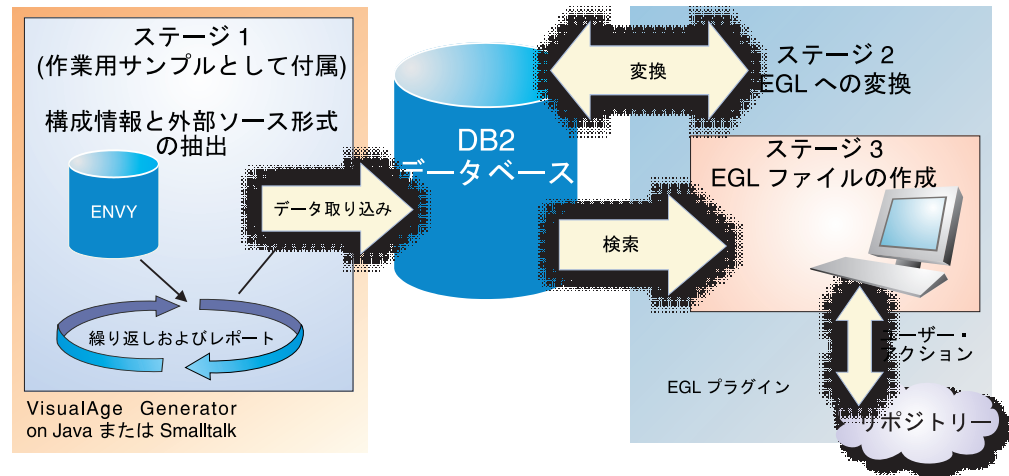
VisualAge Generator to EGL マイグレーション・ツールを使用するには、次の 2 つの方法があります。

- ステージ 1 から 3 のマイグレーション。これについては、18 ページの『VisualAge Generator to EGL マイグレーション・ツールの概要』で説明します。これは、ソース・コードをマイグレーションするための基本手法です。

- 単一ファイル・マイグレーション。これについては、26 ページの『単一ファイル・マイグレーションの概要』で説明します。この手法は、環境が正しく作動することを確認するために、少数のプログラムをマイグレーションする場合に便利です。

## VisualAge Generator to EGL マイグレーション・ツールの概要

前述の目的を果たすために、VisualAge Generator to EGL マイグレーション・ツールは実際には複数のツール群で構成され、次の図に示すように 3 つのステージに編成されています。



- ステージ 1 用のツールは、VAGen 環境で実行されます。ステージ 1 ツールは、ソース・コードの編成、およびソース・コード自体に関する情報を、Java リポジトリまたは Smalltalk ライブラリーから抽出します。ステージ 1 ツールは、この情報をマイグレーション・データベースにロードします。VAGen ソース・コードは、外部ソース形式で保管されます。
- ステージ 2 ツールは、EGL 環境で実行します。ステージ 2 ツールは、マイグレーション・データベースに保管されている情報を使用して、ステージ 1 の実行中にマイグレーション・データベースに保管された VAGen パーツの EGL 構文を作成します。ステージ 2 ツールは、得られた EGL ソース・コードをマイグレーション・データベースに保管します。
- ステージ 3 ツールも EGL 環境で実行します。作成するそれぞれの EGL プロジェクトごとに、ステージ 3 ツールはそのプロジェクトに属するパーツの EGL ソースを抽出して、EGL プロジェクトをファイル・システム内に作成します。プロジェクト・セットの 1 つのバージョンのみを使用する場合、オプションでステージ 3 ツールでプロジェクトをご使用のワークスペースにインポートできます。

プロジェクトがワークスペースに配置された後は、任意のソース・コード・リポジトリを使用して、プロジェクトのバージョン管理を行うことができます。そのソース・コード・リポジトリが提供するツールを使用して、ソース・コードを管理できます。

## マイグレーション・ツールの用語

パーツ間マイグレーションを問題なく行うには、パーツ自体だけでなく、そのパーツが参照するパーツすべてを用意する必要があります。例えば、プログラムをマイグレーションする場合は、そのプログラムだけでなく、プログラムが参照するパーツすべてを提供する必要があります。プログラムの場合、プログラムからマイグレーションする際に必要なパーツのセットは、プログラムを VisualAge Generator 内で生成するときに必要なパーツのセットと同じです。このパーツのセットは、プログラムの関連リストです。

VisualAge Generator の場合、生成用のパーツをすべて提供するための一般的な技法は次のとおりです。

- VisualAge Generator on Java のプロジェクト・リスト・パーツ (PLP)
- VisualAge Generator on Smalltalk の構成マップ

マイグレーション・ツールは、これら 2 つの技法を使用します。このツールは、次の用語を使用します。

- VisualAge Generator on Java からのマイグレーションの場合:
  - 上位 PLP プロジェクト は、プロジェクト・リスト・パーツ (PLP) を含む Java プロジェクトで、他の PLP からは参照されません。
  - マイグレーション・セット は、Java 上位 PLP プロジェクト内で参照されるすべての VAGen プロジェクトからなり、上位 PLP プロジェクトから始まる PLP チェーン全体の VAGen プロジェクトをすべて含みます。
- VisualAge Generator on Smalltalk からのマイグレーションの場合:
  - 上位構成マップ は、他の構成マップに必須マップとしてリストされていない Smalltalk 構成マップです。
  - マイグレーション・セット は、必須マップとして Smalltalk 上位構成マップにリストされているすべての Smalltalk 構成マップからなります。マイグレーション・セットには、上位構成マップから始まる Smalltalk 必須マップのチェーン全体にある構成マップすべてが含まれます。
- マイグレーション・プラン は、1 つ以上のマイグレーション・セットに関する情報を指定するファイルです。ステージ 1 の設定中にマイグレーション・プラン・ファイル名を指定した場合は、リポジトリ・フィルターにマッチングするマイグレーション・セットがすべて同じマイグレーション・プラン・ファイル内に置かれます。マイグレーション・プラン・ファイル名を指定しない場合は、それぞれのマイグレーション・セットが別々のマイグレーション・プラン・ファイルに置かれます。

注: VisualAge Generator on Java からのマイグレーションを行う場合に、PLP プロジェクトを現在使用していなければ、マイグレーションのみに使用する PLP を作成できます。代わりに、次のいずれかを行うこともできます。

- Java プロジェクトのバージョンに関連して、生成に必要なものを指定する情報がデータベースや他のシステム内にある場合は、データベースからマイグレーション・プラン・ファイルを自動的に作成するツールをユーザーが開発できます。
- マイグレーション・プラン・ファイルを手作業で作成できます。

VisualAge Generator on Java からのマイグレーションを行う場合、詳細については 137 ページの『マイグレーション・プランと上位 PLP プロジェクト』を参照してください。

## ステージ 1 の詳細

ステージ 1 ツールは、サンプル・プログラムとして Rational Developer または WebSphere Developer 製品に付属しています。現在使用している VisualAge Generator Developer 4.5 製品に応じて、VisualAge Generator Developer on Java、または VisualAge Generator Developer on Smalltalk 上で稼働するサンプル・プログラムをインストールします。Java 環境と Smalltalk 環境の違いにより、2 つのサンプル・プログラムは多少異なります。ただし、ステージ 1 サンプル・プログラムを使用するための基本的な手順は、どちらの環境でも同じです。ステージ 1 の基本手順は、次のとおりです。

- ステップ 1. ステージ 1 マイグレーションを指示するための規則と設定を定義する。
- ステップ 2. ツールを実行して、次のうち 1 つ以上の出力を生成する。
  1. 1 つ以上のマイグレーション・プラン・ファイル
  2. それぞれのマイグレーション・プラン・ファイルのマイグレーション状況を示すレポート
  3. 検出された問題に関するメッセージを含むログ・ファイル
  4. マイグレーション・データベース

### ステップ 1

次のように、マイグレーションする対象に関する情報をステージ 1 ツールに対して指定する規則と設定を定義します。

1. マイグレーションしたいプロジェクトのみが処理対象になるように、Java プロジェクト名をフィルターに掛ける方法。Smalltalk の場合は、Smalltalk 構成マップ名をフィルターに掛ける方法を指定します。これにより、フィルターにマッチングする Java プロジェクトまたは Smalltalk 構成マップのみをツールが処理するようになるので、ステージ 1 のパフォーマンスが向上します。
  - フィルターにマッチングする Java プロジェクトから、ステージ 1 ツールは上位のプロジェクト・リスト・パーツを含む Java プロジェクトを選択します。Java プロジェクトが他の PLP によって参照されていない場合、その Java プロジェクトは上位のプロジェクト・リスト・パーツ (PLP) を含んでいます。
  - フィルターにマッチングする Smalltalk 構成マップから、ステージ 1 ツールは上位構成マップを選択します。上位構成マップは、他の構成マップに必須マップとしてリストされていないものです。
2. フィルターに基づいてマイグレーション可能な対象をすべて反映したマイグレーション・プランを 1 つ作成するか、複数のマイグレーション・プラン (それぞれの Java 上位 PLP プロジェクトのバージョンごと、または Smalltalk 上位構成マップのバージョンごとに 1 つずつ) を作成するか。
3. Java プロジェクト名とパッケージ名から、または Smalltalk 構成マップ名とアプリケーション名から、EGL プロジェクト名、パッケージ名、およびファイル名を作成する方法。指定できる情報には、次のものがあります。



- 共通コードを含む Java プロジェクトとパッケージ、または Smalltalk 構成マップとアプリケーションを指示する規則。
  - EGL プロジェクト名とパッケージ名を作成する際に使用される名前変更規則。
  - 共通パーツ、または未使用パーツを含む EGL ファイルに使用される名前。
4. マイグレーション・データベースの名前、およびデータベースにアクセスするために必要なユーザー ID とパスワード。
  5. ステージ 1 ツールがステップ 2 で作成する出力。すべての出力を単一ステップで作成するように指示することもでき、または出力を順次作成して、長時間かかる次の出力の作成前に規則と設定を検討する機会を設けることもできます。

## ステップ 2

定義した規則と設定に基づいて、ステージ 1 ツールは次の出力を生成できます。

1. **マイグレーション・プラン・ファイル (複数可)**。マイグレーション・プラン・ファイルは、マイグレーション・セットを格納しています。それぞれのマイグレーション・セットは、Java リポジトリからの上位 PLP プロジェクトの 1 バージョン、または Smalltalk ライブラリーからの上位構成マップの 1 バージョンです。従属 Java プロジェクトのバージョン、または必須 Smalltalk 構成マップのバージョンが、マイグレーション・セット内で指定されます。
  - マイグレーション設定ファイルにマイグレーション・プラン・ファイル名のオプションの値が指定されていない場合は、複数のマイグレーション・プラン・ファイルが作成されます。Java 用の上位 PLP プロジェクトのバージョンごとに、マイグレーション・セットを 1 つ含むマイグレーション・プラン・ファイルが 1 つずつ作成されます。同様に、Smalltalk 用の上位構成マップのバージョンごとに、マイグレーション・セットのバージョンを 1 つ含むマイグレーション・プラン・ファイルが 1 つずつ作成されます。
  - マイグレーション設定ファイルにマイグレーション・プラン・ファイル名のオプションの値が指定されている場合は、Java 用の上位 PLP プロジェクトのバージョンごとに、単一のマイグレーション・プラン・ファイル内にマイグレーション・セット項目が 1 つずつ作成されます。同様に、Smalltalk 用の上位構成マップのバージョンごとに、単一のマイグレーション・プラン・ファイル内にマイグレーション・セット項目が 1 つずつ作成されます。
  - 例えば、5 つの Java プロジェクトと、他の 5 つのプロジェクトのバージョンを指定する PLP を含む 6 番目の Java プロジェクトからなる受注システムがあるとして、複数のマイグレーション・プランと 3 つのバージョンを要求すると、PLP パーツを含む Java 受注プロジェクトのバージョンごとに 1 つずつ、3 つのマイグレーション・セットが作成されます。同様に Smalltalk の場合は、受注システムを構成するコードを反映する構成マップのバージョンを 3 つマイグレーションすると、この上位構成マップのバージョンごとに 1 つずつ、3 つのマイグレーション・セットが作成されます。

この時点でステージ 1 ツールに停止を指示することにより、マイグレーション・プラン・ファイルを検討して、マイグレーションしたい Java プロジェクトのバージョン、または Smalltalk 構成マップのバージョンがマイグレーション・プラン・ファイルに正しく反映されているかどうか確認できます。

2. **それぞれのマイグレーション・セットのマイグレーション状況を示すレポート。**  
ステージ 1 ツールは、**データベースをロードせずに**このレポートを作成できます。このレポートにより、Java プロジェクトまたは Smalltalk 構成マップの正しいセットがフィルターと設定によって選択されていること、および名前変更規則を適用した結果の EGL プロジェクト、パッケージ、およびファイルの命名規則に問題がないことを確認できます。レポートを検討すれば、提示された EGL 構造に不備がある場合に、ステージ 1 ツールによって実際にデータベースがロードされる前に規則と設定を改良できます。マイグレーションされる対象と提示される EGL 構造に問題がなくなるまで、これまでの手順を必要な回数だけ繰り返すことができます。レポートには、次の情報が示されます。

- Java の場合は、それぞれのマイグレーション・セットごとに、組み込まれるプロジェクトのバージョンがリストされます。それぞれのプロジェクト・バージョンごとにパッケージ・バージョンが示され、それぞれのパッケージ・バージョンごとに VAGen パーツのリストが示されます。
- Smalltalk の場合は、それぞれのマイグレーション・セットごとに、組み込まれる構成マップのバージョンがリストされます。それぞれの構成マップ・バージョンごとにアプリケーション・バージョンが示され、それぞれのアプリケーション・バージョンごとに VAGen パーツのリストが示されます。

それぞれの VAGen パーツごとに、そのパーツが配置される対応 EGL プロジェクト、パッケージ、およびファイル名が示されます。また、それぞれの VAGen パーツごとに、VisualAge Generator によって作成された関連パーツのリスト、および関連パーツが配置される EGL ファイルも示されます。ステージ 1 から 3 までのマイグレーション中に、VAGen パーツがどのようにファイルに割り当てられるかについては、42 ページの『EGL ファイル内でのパーツの配置』を参照してください。

3. **VAGen プログラム、テーブル、マップ・グループ、または制御パーツの名前が EGL 予約語リストのいずれかと競合する場合は、ログ・ファイルにメッセージが示されます。**これらのパーツの名前は、マイグレーション中には変更されません。VisualAge Generator 内でパーツの名前を変更するか、EGL へのマイグレーションが済むまで待つことができます。

またこのログ・ファイルには、EGL 予約語リストと矛盾する UI レコード名、あるいは「#」または「@」で始まる UI レコード名に関するメッセージも含まれます。UI レコードは、ステージ 2 のマイグレーション中に名前変更されます。

4. **マイグレーション・プラン・ファイルに基づいて情報と VAGen ソース・コードがロードされたマイグレーション・データベース。**データベースのロードに使用するマイグレーション・プランを 1 つ選択することも、ディレクトリー内のマイグレーション・プラン・ファイルをすべて選択することもできます。ステージ 1 マイグレーション・ツールは、データベースに次のデータをロードします。

- 選択したマイグレーション・プラン・ファイル (複数可) 内の各マイグレーション・セットに関する情報。
- マイグレーション・セットに関連した Java プロジェクトまたは Smalltalk 構成マップのセット。
- Java プロジェクトまたは Smalltalk 構成マップのセットに含まれるそれぞれの VAGen パーツごとの、外部ソース形式の VAGen パーツ定義。



- それぞれの Java プロジェクト、パッケージ、および VAGen パーツごと、またはそれぞれの Smalltalk 構成マップ、アプリケーション、および VAGen パーツごとの、対応する EGL プロジェクト、パッケージ、およびファイル名。
- マイグレーション・データベースにロードされたものを完全に記録するために、このステップでもレポートが作成されます。このレポートは、前述のレポートと同じ形式です。

ステージ 1 ツールは、VisualAge Generator の Java と Smalltalk の両バージョン用サンプル・プログラムとして付属しています。ステージ 1 ツールは「現状のまま」使用でき、ご使用の環境に適合するようにサンプル・プログラムを変更することもできます。例えば、現在は構成情報を Java リポジトリまたは Smalltalk ライブラリーの外部に保管していて、この構成情報は生成に必要なソース・コードのバージョンを指定しているとします。この状況では、サンプル・プログラムをガイドとして使用して、構成情報と Java リポジトリまたは Smalltalk ライブラリーの組み合わせからマイグレーション・データベースをロードするためのツールをユーザーが独自に作成できます。

ステージ 1 サンプル・プログラムを変更した場合は、マイグレーション・データベースを変更して、コードの分析に役立つ追加情報を組み込むことができます。既存の SQL 表に列を追加したり、マイグレーション・データベースに表を追加したりすることができます。ただし、これらの新しい列と表は、マイグレーションのステージ 2 と 3 には使用されません。また、ステージ 1 サンプル・プログラムを変更する場合は、サンプル・プログラムに示されている情報を SQL 表に必ず取り込む必要があります。このようにしなければ、ステージ 2 と 3 でコードをマイグレーションできません。

ステージ 1 ツールを VisualAge Generator Developer on Java にインストールして実行する場合の詳細については、121 ページの『第 4 章 ステージ 1 — Java からの抽出』を参照してください。ステージ 1 ツールを VisualAge Generator Developer on Smalltalk にインストールして実行する場合の詳細については、145 ページの『第 5 章 ステージ 1 — Smalltalk からの抽出』を参照してください。

## ステージ 2 の詳細

ステージ 2 ツールは、Eclipse プラグイン `com.ibm.etools.egl.vagenmigration` で出荷され、EGL 環境で稼動します。この時点で、マイグレーションする情報はマイグレーション・データベース内にあるので、VisualAge Generator on Java または VisualAge Generator on Smalltalk のどちらからマイグレーションするかに関係なく、同じステージ 2 ツールを使用します。ステージ 2 ツールを実行するための基本手順は、次のとおりです。

1. 次のように、マイグレーション対象をステージ 2 ツールに指示する規則と設定を定義します。
  - EGL ソース・コードの作成方法に関する具体的な詳細。例えば、ステージ 2 マイグレーション・ツールは、VAGen 作業用ストレージ・レコードを次の 2 つの EGL 基本レコードに分割する必要があります。
    - a. オリジナルの作業用ストレージ・レコードと同じ名前の、レベル 77 以外の項目をすべて含むレコード。

- b. オリジナルの作業用ストレージ・レコードと同じ名前に接尾部が付いた、レベル 77 項目をすべて含む 2 つ目のレコード。

ステージ 2 のマイグレーション設定を使用して、レベル 77 項目を含むレコードを新規作成する際にステージ 2 ツールが使用する接尾部を指定できます。

- マイグレーションするマイグレーション・セット。例えば、受注システムの 3 つの異なるバージョンをマイグレーションするために 3 つのマイグレーション・セットを作成した場合、最初は 1 バージョンのみをマイグレーションできます。この機能により、マイグレーションの制限を柔軟に設定でき、マイグレーション・データベース内のものをすべて同時にマイグレーションする必要がなくなります。
  - マイグレーション・データベースの名前、およびデータベースにアクセスするために必要なユーザー ID とパスワード。データベースのユーザー ID とパスワードが明示的に指定されていない場合、ステージ 2 とステージ 3 のマイグレーション・ツールは、Windows マシンへのログオンに使用されたユーザー ID とパスワードを使用してデータベースへのログオンを試行します。
  - ステージ 2 の完了後、ステージ 3 ツールを自動的に開始するかどうか。ステージ 3 を自動的に実行する場合は、EGL プロジェクトの 1 バージョンをワークスペースにロードするか、または後でソース・コード・リポジトリとインターフェースを取ることができるように、EGL プロジェクトを一時ディレクトリにロードするかを選択できます。
2. 定義した規則と設定に基づいて、ステージ 2 ツールは次の処理を行います。
    - a. データベースから 1 つのマイグレーション・セット用のパーツを検索します。
    - b. 外部ソース形式のソース・コードを EGL ソース・コードに変換します。
    - c. EGL ソース・コードをマイグレーション・データベースに保管します。パーツのマイグレーションに関連したメッセージも、マイグレーション・データベースに保管されます。これにより、ステージ 2 のパフォーマンスが向上します。同じパーツ・エディションが別のマイグレーション・セット内で使用されている場合、EGL ソース・コードがすでに使用可能で再変換が行われないからです。
    - d. 検出された潜在的な問題のログ・ファイルを作成します (例えば、生成可能パーツと EGL 予約語リストの競合や、マイグレーション・ツールが解決できない未確定状態など)。
    - e. 次に選択されたマイグレーション・セットに対してプロセスを繰り返します。

ステージ 2 マイグレーション・ツールは、バッチ・モードで実行できます。EGL 開発環境での、ステージ 2 ツールのインストールおよび実行の詳細については、169 ページの『第 6 章 ステージ 2 — EGL 構文への変換』を参照してください。ステージ 2 マイグレーション・ツールを変更することはできません。

## ステージ 3 の詳細

ステージ 3 ツールは、ステージ 2 ツールと同じ Eclipse プラグイン (com.ibm.etools.egl.vagenmigration) で出荷され、同様に EGL 環境で稼動します。こ

の時点で、マイグレーションする情報はマイグレーション・データベース内にあるので、VisualAge Generator on Java または VisualAge Generator on Smalltalk のどちらからマイグレーションするかに関係なく、同じステージ 3 ツールを使用します。ステージ 3 ツールを実行するための基本手順は、次のとおりです。

1. 次のように、マイグレーション対象をステージ 3 ツールに指示する規則と設定を定義します。
  - マイグレーションするマイグレーション・セット。例えば、受注システムの 3 つの異なるバージョンをマイグレーションするために 3 つのマイグレーション・セットを作成した場合、ステージ 2 ツールを使用して 3 つのバージョンすべてをマイグレーションする一方で、ステージ 3 ツールを使用してマイグレーションするバージョンは 1 つだけにしたい場合があります。ステージ 3 で 1 バージョンだけを処理する理由として最も一般的な状況は、ソース・コード・リポジトリ内でこのコードのバージョン管理を行い、その後でステージ 3 ツールを使用して次のバージョンをマイグレーションし、ソース・コード・リポジトリ内でそのバージョンを管理したい場合です。
  - マイグレーション・データベースの名前、およびデータベースにアクセスするために必要なユーザー ID とパスワード。
2. 定義した規則と設定に基づいて、ステージ 3 ツールは次の処理を行います。
  - a. マイグレーション・セットの「to do」リストを作成します。この「to do」リストは、ステージ 2 で生成されたメッセージを統合したリストで、マイグレーションを完了するために必要になる可能性がある追加作業を示しています。
  - b. ステージ 1 でマイグレーション・データベースに保管された情報に基づいて、EGL プロジェクトとパッケージの構造をワークスペースに作成します。
  - c. ステージ 2 で保管された VAGen パーツの EGL ソース・コードに基づいて、.egl ソース・ファイルを作成します。.egl ソース・ファイルは、EGL パーツ参照の解決に必要な import ステートメントをほとんど含んでいます。import ステートメントについて詳しくは、38 ページの『EGL のビルド・パスと import ステートメント』を参照してください。
  - d. ステージ 2 で保管された VAGen 制御パーツの EGL XML ソースに基づいて、.eglbld ファイルを作成します。制御パーツは、生成オプション (EGL ビルド記述子パーツ)、リンケージ・オプション、リソース関連、バインド制御、およびリンク・エディット・パーツです。
  - e. ワークスペースをリフレッシュして EGL 検証が実行されるようにします。
3. この時点で、次のことを行う必要があります。
  - a. オプションで、EGL プロジェクトをソース・コード・リポジトリにバージョン管理またはコミットして、マイグレーションされたときとまったく同じコードを反映するベースラインを確立する。
  - b. ワークスペースの「問題」ビューにあるメッセージを検討して、検証エラーがないかどうか確認する。この確認は、ステージ 2 で生成されたログ、またはステージ 3 で作成された「to do」リストを併用して行うことができます。
  - c. ターゲット環境へのマイグレーションが正しく行われるように、すべてのプログラムと dataTable を生成 (準備せずに) する。プログラムを生成する際には、formGroup がすべて生成されるように、必ず genFormGroup と

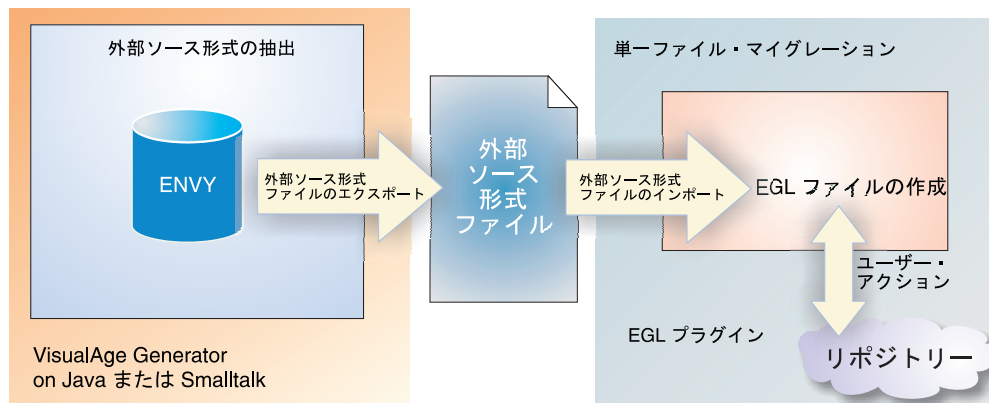
genHelpFormGroup のビルド記述子オプションを使用します。このステップはオプションですが、強く推奨されます。

- d. EGL プロジェクトをソース・コード・リポジトリにバージョン管理またはコミットし、問題を解決するために行ったすべてのコード変更を反映する新しいベースラインを確立する。
- e. マイグレーション済みコードを生成し、テストする。このステップもオプションですが、強く推奨されます。

ステージ 3 マイグレーション・ツールは、パッチ・モードで実行できます。EGL 上でのステージ 3 ツールの実行について詳しくは、187 ページの『第 7 章 ステージ 3 – インポート』を参照してください。ステージ 3 ツールは、ステージ 2 ツールをインストールすると同時に自動的にインストールされます。ステージ 3 マイグレーション・ツールを変更することはできません。

## 単一ファイル・マイグレーションの概要

EGL に慣れてきて初めて環境をセットアップする場合、少数のプログラムのみをマイグレーションして使用する環境を検証し、生成および準備が正しく動作することを確認して、EGL に関するランタイム環境が正しく構成されていることを確認できます。この場合に、ステージ 1 から 3 のマイグレーションを完全に行う必要はありません。ステージ 2 マイグレーション・ツールには、次の図に示す、単一ファイル・マイグレーションと呼ばれるものを使用してプログラムをマイグレーションするメカニズムが用意されています。



単一ファイル・マイグレーションは、ステージ 1 から 3 のマイグレーションに比べて手操作の部分が多いプロセスです。単一ファイル・モードでは、VisualAge Generator を使用して、外部ソース形式のソース・コードをファイルにエクスポートします。その後、EGL 開発環境を使用して EGL プロジェクトと EGL パッケージを作成します。「インポート」ウィザードを使用して、外部ソース形式ファイルを EGL ソースにインポートできます。単一ファイル・マイグレーション・ツールが実行され、次の処理を行います。

- 存在しない場合は、ターゲット EGL ソース・ファイルを作成します。そのファイルが存在する場合は、上書きするか、ファイルに追加するかを選択できます。使用している設定と、外部ソース形式ファイルに含まれるパーツによっては、追加の EGL ファイルがマイグレーション・ツールによって作成される場合があります。

- 外部ソース形式のソース・コードを EGL ソース・コードに変換します。
- 検出された潜在的な問題のログ・ファイルを作成します (例えば、生成可能パーツと EGL 予約語リストの競合や、マイグレーション・ツールが解決できない未確定状態など)。

単一ファイル・マイグレーション中に行われる外部ソース形式から EGL への変換は、ステージ 1 から 3 のマイグレーションのステージ 2 で行われる構文変換と本質的には同じです。ただし、単一ファイル・マイグレーションにはいくつかの制限があり、大規模マイグレーションには適していません。次のような制限があります。

- 単一の外部ソース形式ファイルにあるパーツのみが、マイグレーション中に処理されます。可能な最良のマイグレーションを行うには、外部ソース形式ファイルにプログラムとその関連パーツをすべて含めます。
- VAGen パーツのファイルへの配置は、ステージ 1 から 3 のマイグレーションの配置とは異なります。単一ファイル・モードでは、ターゲット EGL ファイル名として *targetFile.egl* を指定したと想定すると、次のようになります。
  - *targetFile.eglbld* という名前のファイルに、すべての制御パーツが配置されます。
  - 各 UI レコードはそれぞれ単独で *uiRecordName.egl* という名前のファイルに格納されます。 *uiRecordName* は UI レコードの名前です。
  - 生成可能パーツを EGL ファイルに分離する設定を選択していない場合、残りのパーツはすべて *targetFile.egl* という名前のファイルに配置されます。
  - 生成可能パーツを EGL ファイルに分離する設定を選択した場合は、次のようになります。
    - それぞれのプログラム・パーツが、*programName.egl* という名前のファイルに配置されます (ここで、*programName* はプログラムの名前)。
    - それぞれのテーブル・パーツが、*tableName.egl* という名前のファイルに配置されます (ここで、*tableName* はテーブルの名前)。
    - それぞれのマップ・グループと、マップ・グループ内のマップすべてが、*mapGroupName.egl* という名前のファイルに配置されます (ここで、*mapGroupName* はマップ・グループの名前)。
    - 残りのパーツはすべて、*targetFile.egl* という名前のファイルに配置されます。残りのパーツすべてをプログラムと同じファイルに配置したい場合は、*targetFile* を *programName.egl* と同じにすることができます。複数の生成可能パーツに共用されているパーツの判別は試行されません。
  - すべてのファイルが、同じ EGL プロジェクト、ソース・フォルダー、およびパッケージに配置されます。
- すべての出力ファイルが同じ EGL パッケージに配置されるので、マイグレーション・ツールは `import` ステートメントを組み込みません。また、すべてのパーツが同じ EGL パッケージに配置されるので、オリジナルの Java プロジェクトとパッケージの構造は保持されません。同様に、オリジナルの Smalltalk 構成マップとアプリケーションの構造は保持されません。
- 同じパーツが外部ソース形式ファイルに複数回現れる場合は、最後の定義のみがマイグレーションされます。



- 単一ファイル・モードでマイグレーションを行う場合に、共通パーツを処理するための技法を 4 つの中から選択できます。いずれかを選択する前に、それぞれの技法の欠点について理解してください。4 つの技法は、次のとおりです。
  - 複数のプログラムとその関連パーツからなる大容量の外部ソース形式ファイルをマイグレーションする場合は、1 つの **targetFile** 名しか指定できません。マイグレーション設定をパーツを *EGL* ファイルに分離に選択した場合、マイグレーション・ツールはデータ項目、機能、PSB、および非 UI レコードを単一のターゲット・ファイルに格納します。複数のプログラムがパーツを共有しているため同じ関連が複数回このファイルに現れたとしても、マイグレーション・ツールがマイグレーションするのは各パーツについて 1 つの定義のみです。したがって、この技法を使用することによってパーツが重複することはありません。ただし、複数のプログラム内に非常に多くの関連パーツがある場合は、**targetFile** がきわめて大きくなることがあります。
  - 2 つの外部ソース形式ファイルを同じ *EGL* パッケージにマイグレーションする場合に、2 つのファイルに同じパーツが含まれていて、**指定した targetFile 名が異なっていると**、*EGL* によって解決できない重複パーツが生じます。この問題は、Program1 とその関連パーツを含むファイルと、Program2 とその関連パーツを含む別のファイルをマイグレーションする場合に、2 つのプログラムが共通パーツを共用していると起こります。マイグレーション・ツールは、共通パーツを両方の **targetFile** に配置します。それぞれのプログラムとその関連パーツを別々の *EGL* パッケージにマイグレーションすれば、この問題を回避できます。この場合もワークスペース内に重複パーツが生じますが、これらのパーツは別々のパッケージにあるので、*EGL* はパーツ参照を解決できます。
  - 2 つの外部ソース形式ファイルを同じ *EGL* パッケージにマイグレーションする場合に、2 つのファイルに同じパーツが含まれていて、**指定した targetFile 名が同じであるときは**、ワークスペース内の既存 **targetFile** を上書きするかどうか尋ねるプロンプトが出されます。
    - 既存の **targetFile** への上書きを希望しないことを指定した場合、2 番目のインポートのデータ項目、関数、PSB、および非 *VGUI* レコードは、**targetFile** に追加されます。2 番目のインポートにある共通パーツは、**targetFile** 内で重複パーツになります。
    - 既存の **targetFile** への上書きを希望することを指定した場合、2 番目のインポートのデータ項目、関数、PSB、および非 *VGUI* レコードは、既存の **targetFile** を完全に置き換えます。このため、最初のインポートに含まれていて、2 番目のインポートに含まれていないパーツはすべて失われます。
    - マイグレーション設定として「パーツを *EGL* ファイルに分離」を選択した場合、マイグレーション・ツールはプログラム、*formGroup*、および *dataTable* 用に作成されたファイルを上書きします。マイグレーション設定を選択しなかった場合、これらのパーツは **targetFile** に配置され、上書きプロンプトに対する応答に従って追加または上書きされます。
    - マイグレーション・ツールは、*VGUI* レコードと *.eglbld* ファイル用のファイルを常に上書きします。

最初の手法と同様に、それぞれのプログラムとその関連パーツを別々の *EGL* パッケージにマイグレーションすれば、この問題を回避できます。この場合もワークスペース内に重複パーツが生じますが、これらのパーツは別々のパッケージにあるので、*EGL* はパーツ参照を解決できます。

- 共通パーツを別の外部ソース形式ファイルに分割すると、単一ファイル・ベースで VisualAge Generator から EGL へのマイグレーションを正常に行うために必要な情報がすべてそろわない場合があります。例えば、1 つの外部ソース形式ファイルに SQL レコードがあり、そのレコードの変更された SQL を使用する関数が別のファイルにある場合、マイグレーション・ツールはその関数の入出力ステートメントのビルドを完全に行うことができません。また、共通パーツが別のパッケージに存在する場合、共通パッケージを参照する必要のある各ファイルに EGL import ステートメントを追加する必要があります。
- ステージ 2 と 3 で行われる追加処理は、単一ファイル・モードでは実行されません。いくつかの例を次に示します。
  - 書式は、formGroup 内でネストされません。
  - パーツは、ファイルのパーツ型の中でパーツ名順にソートされません。
  - 出力ファイルの中で、パーツ間には 1 行のみが入ります。

単一ファイル・マイグレーションとステージ 1 から 3 のマイグレーションとの違いについて詳しくは、29 ページの『マイグレーションの考慮事項』、および 35 ページの『VisualAge Generator to EGL マイグレーション・ツールが使用する技法』を参照してください。

---

## マイグレーションの考慮事項

ソース・コードを作成および管理するためのアプローチには、VisualAge Generator と EGL の間でいくつかの相違点があります。特に、次の相違点はマイグレーションには重要です。

- EGL の構文は VisualAge Generator よりも厳密な場合がある
- パーツ参照が解決される時期と方法の違い
- 共通コードの処理の違い

次に、これらの相違点について詳しく説明します。

### EGL 構文の厳密さ

2 つの言語の構文は大きく異なりますが、VAGen 言語の大部分は、オリジナルの VAGen プログラムと同じ振る舞いを保持しながら EGL 言語にマイグレーションできます。ただし、状況によっては、EGL の構文が VisualAge Generator よりも厳密または制限的になることがあります。このような状況は、標準的なプログラムではあまり起こりません。ただし、この状況が発生した場合、マイグレーション・ツールはパーツ間マイグレーションを行って、VisualAge Generator 内で必要だった振る舞いを保持する正確な EGL 構文を判別する必要があります。パーツ間マイグレーションを行うには、現行パーツを正しくマイグレーションするために、参照されている他の 1 つ以上のパーツをマイグレーション・ツールが使用できるようにする必要があります。次に、いくつかの例を示します。

- VisualAge Generator の場合は、表示 (テキスト) マップとプリンター・マップの両方に DISPLAY 入出力オプションを使用します。EGL は、テキスト書式用の display ステートメントと、印刷書式用の print ステートメントを提供しています。VisualAge Generator からのマイグレーションを容易にするために、VisualAge Generator との互換性が必要であることを示す EGL 設定があります。VisualAge Generator 互換の設定を使用すると、印刷書式に display ステートメントを使用で

きます。マイグレーション中に、プログラム、そのマップ・グループ、およびマップがすべて使用可能ならば、マイグレーション・ツールは `display` または `print` のどちらのステートメントにマイグレーションするかを判別できます。ただし、プログラムを含めずに `DISPLAY` 関数をマイグレーションする場合、マイグレーション・ツールは `display` または `print` のどちらの EGL ステートメントを使用するか確実に判別できません。この場合、マイグレーション・ツールは `display` ステートメントを使用します。これは、VisualAge Generator 互換モードでは印刷書式にこのステートメントを使用することが許容されるからです。

- VisualAge Generator では、表示 (テキスト) マップおよびプリンター・マップの両方について `SET map PAGE` ステートメントを使用します。このため、次の `CONVERSE` または `DISPLAY` の対象が表示マップである場合は画面が消去され、次の `DISPLAY` の対象がプリンター・マップである場合はページ替えが行われます。EGL は、テキスト書式用には `clearScreen` システム・ライブラリー関数、印刷書式用には `pageEject` システム・ライブラリー関数を提供しています。VisualAge Generator 互換の設定は、`clearScreen` または `pageEject` の使用には影響しません。マイグレーション中に、プログラム、そのマップ・グループ、およびマップがすべて使用可能ならば、マイグレーション・ツールは `clearScreen` または `pageEject` のどちらのシステム・ライブラリー関数にマイグレーションするかを判別できます。ただし、プログラムなしでマイグレーションされる関数で `SET map PAGE` ステートメントが使用されると、`clearScreen` または `pageEject` システム・ライブラリー関数のどちらを使用するかを、マイグレーション・ツールで明確に判別できません。この場合、マイグレーション・ツールは意図的に無効な EGL 構文である `EZE_SETPAGE` を使用します。この結果、「問題」ビューにエラーが示されるので、関数の訂正が必要であることが分かります。
- VisualAge Generator の場合は、マップ変数フィールドの編集ルーチンとして、編集テーブルまたは編集関数のどちらかを指定できます。両方は指定できません。EGL の場合は、`validatorDataTable` と `validatorFunction` の両方のプロパティを指定できます。編集テーブルまたは編集関数がマイグレーション時に使用できる場合、マイグレーション・ツールは `validatorDataTable` または `validatorFunction` のどちらのプロパティを設定するか判別できます。ただし、編集ルーチンによって指定されているパーツが使用不可ならば、マイグレーション・ツールは EGL の `validatorDataTable` または `validatorFunction` のどちらのプロパティを設定するか確実に判別できません。この場合、マイグレーション・ツールは、編集ルーチン名の長さ、編集メッセージの存在などの情報を使用して、編集ルーチンがテーブルまたは関数のどちらであるか判別を試みます。それでも判別できない場合、マイグレーション・ツールは `validatorFunction` プロパティを使用します。`validatorFunction` が関数でないか、検出できない場合のみ、「問題」ビューにエラーが示されます。

マイグレーション・ツールは、マイグレーション・セット内で使用可能なパーツをすべて使用して、未確定状態を解決します。こうした未確定状態を最小限にするために、マイグレーションの際には関連パーツを必ず組み込んでください。例えば、プログラムをマイグレーションする場合は、VAGen 内でプログラムを生成するために必要なパーツをすべて組み込みます。これにより、可能なかぎり最適なパーツのマイグレーション結果が得られます。マイグレーション・ツールが未確定状態を解決する方法の概要については、次に示す項を参照してください。

- 46 ページの『プログラムを使用したマイグレーション』



- 47 ページの『関連パーツを使用したマイグレーション』
- 48 ページの『関連パーツを使用しないマイグレーション』

正しいマイグレーションのためにマイグレーション・ツールがパーツ間マイグレーションを行う必要がある状況すべてのリストと、追加のパーツが使用できない場合にマイグレーション・ツールがインテリジェントに選択を試行する手順については、59 ページの『第 3 章 未確定状態の処理』を参照してください。

## パーツ名が解決される時期と方法

定義時に、VisualAge Generator はすべてのパーツの存在を必要としません。プログラム構造ダイアグラムの中で、VisualAge Generator は欠落しているマップ、レコード、テーブル、および関数に疑問符 (?) を付けて示します。ただし、共用データ項目を使用する場合など、他の場所では現在パーツが存在していないことを示すものではありません。パーツを VisualAge Generator に保管するときに基本的な構文の検証が行われますが、パーツ間での検証は、テスト、検証、または生成のときまで行われません。EGL の場合は、ファイルを保管するたびに幅広い検証が行われ、すべてのパーツ名が解決可能かどうか検証されます。このため、問題があれば可能なかぎり早期に警告が示されます。

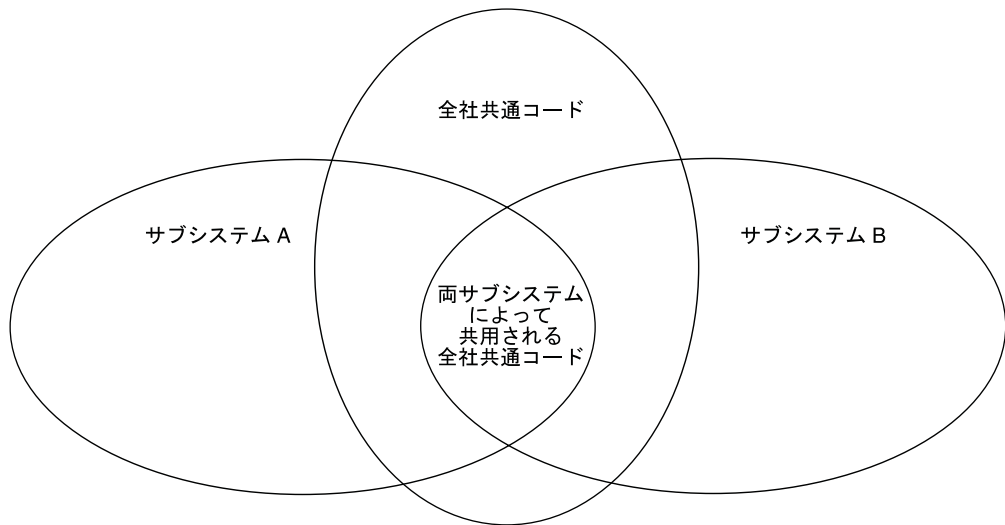
VisualAge Generator は、特定のパーツ名を見付けるために、ワークスペース内のパーツすべてを検索します。VisualAge for Java 内で重複するパーツ名がある場合は、重複パーツの問題が修正されるまで、テストと生成は停止します。VisualAge for Smalltalk の場合は、イメージに重複するパーツをロードすることは許されません。EGL の場合は、ワークスペース内で重複するパーツ名が許されます。EGL は、プロジェクトの EGL ビルド・パス、ファイル内の `import` ステートメント、およびレコードと関数の `containerContextDependent` プロパティの組み合わせによって、使用するパーツの定義を判別します。

ステージ 1 から 3 のマイグレーションを使用してマイグレーションを行う場合、マイグレーション・ツールは、マイグレーション・セット内にあるパーツに基づいて、プロジェクトの EGL ビルド・パスを設定し、ファイルに `import` ステートメントを組み込みます。正しい EGL ビルド・パスと `import` ステートメントが得られるように、マイグレーションの際には必ず関連パーツすべてを組み込むようにしてください。例えば、プログラムをマイグレーションする場合は、VisualAge Generator 内でプログラムを生成するために必要なパーツをすべて組み込みます。これにより、可能なかぎり最適なパーツのマイグレーション結果が得られます。詳しくは、次に示す項を参照してください。

- 38 ページの『EGL のビルド・パスと `import` ステートメント』
- 40 ページの『`containerContextDependent` プロパティ』

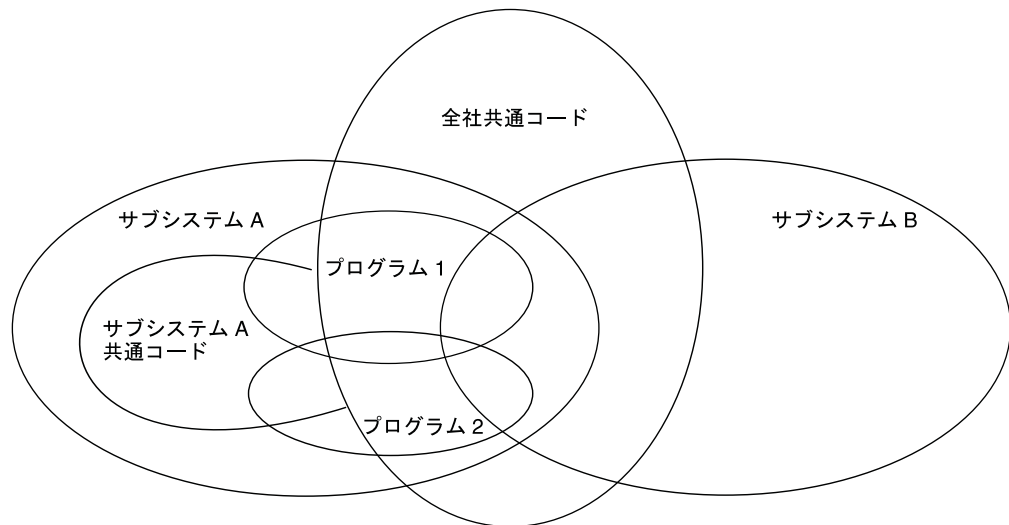
## 共通コードのシナリオ

共通コードは、サブシステムまたはプログラムの間で共用されるコードです。次の図に、2 つのサブシステム間で共用される共通コードを示します。



この例では、全社共通コードを含む Java プロジェクトまたは Smalltalk 構成マップが 1 つ以上あります。これらのプロジェクトまたは構成マップ内のコードは、複数のサブシステム間で共用できます。この例では、サブシステム A とサブシステム B が共通コードのサブセットを使用しており、全社共通コードの一部が両方のサブシステムによって使用されています。例えば、全社共通コードには、多数のサブシステム間で使用される SQL レコード定義を組み込むことができます。

次の図に、2 つのサブシステム間で全社共通コードを共用する基本的な方法を、サブシステム A の詳細とともに示します。



サブシステム A には、サブシステム A 共通コードがあります。このコードは、サブシステム A 内の複数のプログラムによって使用されますが、サブシステム A 内のプログラム以外には使用されません。この例では、プログラム 1 とプログラム 2 がそれぞれ、サブシステム A 共通コードの一部を全社共通コードの一部とともに使用しています。2 つのプログラム間で、サブシステム A 共通コードと全社共通コードの両方がオーバーラップする部分があり、サブシステム B が使用する全社共通コードとのオーバーラップもあります。例えば、サブシステム A 共通コードには、サブシステム A 内のプログラムのみが使用する SQL レコード定義を組み込むことが

できます。さらに、サブシステム A 内の複数のプログラムが使用するマップ・グループ定義も、サブシステム A 共通コードに組み込むことができます。

## 共通コードと VisualAge Generator

共通コードを使いやすくするために、VisualAge Generator はソース・コードの特定部分の解釈方法をテストと生成の時点で決定します。この利点は、それぞれのサブシステムまたはプログラムがコードに小規模の変更を加えるときに、プログラムが使用する特定のマップ・グループを変更するか、生成時にワークスペース内にあるデータ項目またはレコード定義を変更するだけで済むという点です。次に、3 つの例を示します。

- 端末と対話するオンライン・プログラムと、類似したレポートを印刷するバッチ・プログラムとの間で、次のように同じロジックの大部分を共用できます。
  - プログラム A は、HEADER、DETAIL、および TRAILER という名前の表示マップを含む MapGrpA を使用する、メインのトランザクション・プログラムです。プログラム A は、部分的な HEADER マップを表示して、DETAIL 行を浮動域に表示し、ユーザーが次のレポートを要求するために使用する入力フィールドを含む TRAILER マップと対話します。プログラム A は、SET<sup>TM</sup> HEADER PAGE ステートメントを使用して画面を消去します。
  - プログラム B は、HEADER、DETAIL、および TRAILER という名前のプリンター・マップを含む MapGrpB を使用する、メインのバッチ・プログラムです。プログラム B は、プログラム A が端末に表示するものと同じレポートのハードコピー版を作成します。プログラム B は、部分的な HEADER マップを表示して、DETAIL 印刷行を浮動域に表示し、TRAILER マップをページの下部に表示します。プログラム B は、SET HEADER PAGE ステートメントを使用してページ替えを強制します。
  - 浮動域の行数は、メインのトランザクション・プログラムとメインのバッチ・プログラムの間で異なります。ただし、データ検索、データ操作、および HEADER マップと DETAIL マップを表示するためのロジックは両プログラムとも同じです。このため、プログラム A とプログラム B は、共通関数を使用してデータベースからデータを検索し、データを操作し、HEADER マップと DETAIL マップを表示するように設計されています。
  - VisualAge Generator 内では、同じ DISPLAY 入出力オプションを表示マップとプリンター・マップの両方に使用できるので、この共通コードの技法が有効です。また、表示マップとプリンター・マップの両方に同じ SET HEADER PAGE ステートメントを使用できます。VisualAge Generator は、テストまたは生成している特定プログラムに基づいて、DISPLAY 入出力オプションと SET map PAGE ステートメントを解釈します。
  - EGL の場合は、表示書式と印刷書式に対して異なるステートメントを使用する必要があります (テキスト書式に対しては display と clearScreen、印刷書式に対しては print と pageEject)。VisualAge Generator 互換モードでは、印刷書式に対して display ステートメントを使用することが許容されます。ただし、VisualAge Generator 互換モードでも、clearScreen はテキスト書式のみに適用され、pageEject は印刷書式のみに適用されます。
- 一般的ではありませんが、SET-MESSAGE-TEXT という名前の共通エラー・ハンドラー関数を使用する例があります。この関数は、MSGTBLE という名前の

VAGen テーブルからメッセージ・テキストを検索し、MESSAGE-TEXT という名前の関数仮パラメーターに格納します。ここで、MESSAGE-TEXT は共用データ項目です。

- サブシステム A とサブシステム B が、異なる CICS 領域内で稼働しているとします。この場合、2 つのサブシステムが MSGTBLE の定義と、関数仮パラメーターとして使われる MESSAGE-TEXT 共有データ項目の定義を独自に提供することがあります。これは、サブシステムそれぞれのマップ定義に、異なるサイズのエラー・メッセージ・フィールドがある場合に起こります。
- VisualAge Generator は、プログラムを生成するときに、ワークスペースに現在ロードされている定義を使用します。テストまたは生成の前に、それぞれのサブシステムが独自の MESSAGE-TEXT データ項目の定義を必ずワークスペースにロードするので、VisualAge Generator はそのサブシステムに適した定義を使用します。この技法の欠点は、生成時にワークスペース内にある定義を管理する必要があることと、両方のサブシステムが同時にワークスペース内に存在できないことです。
- EGL の場合は、ワークスペース内に両方のサブシステムが同時に存在することが可能です。この状況では、EGL は EGL ビルド・パス、import ステートメント、および SET-MESSAGE-TEXT 関数の containerContextDependent プロパティの組み合わせを使用して、MESSAGE-TEXT dataItem パーツ定義への参照を解決します。
- 少し異なる例として、MESSAGE-TEXT2 という名前の共用データ項目を含む、ERROR-RECORD という名前の共通エラー・レコードを使用する場合があります。
  - サブシステム A とサブシステム B には、異なる MESSAGE-TEXT2 の定義があるとします。この状況は、両サブシステムが異なる画面サイズ用のメッセージ・テキストを作成する必要がある場合に起こります。
  - VisualAge Generator は、プログラムを生成するときに、ワークスペースに現在ロードされている定義を使用します。それぞれのサブシステムが独自の MESSAGE-TEXT2 の定義をロードするので、VisualAge Generator はそのサブシステムに適した定義を使用します。この技法の欠点は、SET-MESSAGE-TEXT 関数の例の場合と同じです。生成時にワークスペース内にある定義を管理する必要があり、両方のサブシステムが同時にワークスペース内に存在することはできません。
  - EGL の場合は、ワークスペース内に両方のサブシステムが同時に存在することが可能です。この状況では、EGL は EGL ビルド・パス、import ステートメント、および ERROR-RECORD の containerContextDependent プロパティの組み合わせを使用して、MESSAGE-TEXT2 dataItem パーツ定義への参照を解決します。

## 共通コードとマイグレーション・ツール

共通コードは、一般には多数のプログラムに使用されるコードです。共通コードはマイグレーション・ツールに次のような影響を及ぼすので、すべてのマイグレーション・セットに共通コードを組み込む必要があります。

- 共通コードが使用できれば、マイグレーション・ツールは大部分の未確定状態を解決できます。これにより、手作業で行う必要があるコードの変更が最小限あるいは不要になります。

- 共通コードが使用できれば、マイグレーション・ツールはプロジェクトの EGL ビルド・パスを正しく設定でき、EGL ファイルに正しい import ステートメントを組み込むことができます。これにより、手作業で行う必要がある EGL ビルド・パスの変更と import ステートメントの追加を最小限にすることができます。
- マイグレーション・ツールがパーツ・バージョンを初めてマイグレーションするとき、ツールはパーツに対して作成された EGL をデータベースに保管します。また、オリジナルの外部ソース形式もマイグレーション・データベースに保存されます。別のマイグレーション・セットが同じパーツ・バージョンを使用している場合、その別のマイグレーション・セット内で新規パーツに対する EGL を作成するときには、マイグレーション・ツールはオリジナルの外部ソース形式を参照して使用し、パーツの EGL への変換は再度行いません。その別のマイグレーション・セット用の EGL プロジェクト、パッケージ、およびファイルを作成する際にも、マイグレーション・ツールはそのパーツ・バージョン用の EGL を使用します。この技法により、パーツ間マイグレーション時に未確定状態を解決するために必要な参照情報がマイグレーション・ツールに提供され、さらにそれぞれのパーツ・バージョンが一度に 1 つだけマイグレーションされるのでパフォーマンスが向上します。

可能なかぎり最適なマイグレーションが行われるように、サブシステムをマイグレーションするには、全社共通コードとサブシステム共通コードをマイグレーション・セットに必ず含めてください。

---

## VisualAge Generator to EGL マイグレーション・ツールが使用する技法

### 技法の概要

対応する EGL 構文を判別し、VisualAge Generator の振る舞いを保持するために、マイグレーション・ツールはいくつかの技法を使用します。これらの技法には、次のものがあります。

- エディターとビルド記述子の設定
- プログラム・プロパティ
- EGL のビルド・パスと import ステートメント
- containerContextDependent プロパティ
- EGL の予約語リスト
- EGL ファイル内でのパーツの配置
- プログラムを使用したマイグレーション
- 関連パーツを使用したマイグレーション
- 関連パーツを使用しないマイグレーション
- ファイルの上書きとマージ

これらの技法について、次に説明します。また、マイグレーション・ツールの動作を規定する一般規則がいくつかあります。



## エディターとビルド記述子の設定

マイグレーションのステージ 2 を開始する前に、使用するワークスペースの *VisualAge Generator* 互換性 設定をオンにします。EGL *VisualAge Generator* 互換性 設定により、次の VAGen の振る舞いがサポートされます。

- ハイフン (-) と各国語文字 @ および # を含むパーツ名の使用。
- プリミティブ・データ型 **numc** および **pacf**。
- 1 次元の構造化フィールド配列の場合に、添え字をデフォルトで 1 に設定。
- **dataTable** の使用宣言に対する **deleteAfterUse** プロパティ (keep after use の置換表現)。
- SQL フィールド・プロパティ **sqlDataCode**。
- **call** ステートメントのオプション *externallyDefined* と *noRefresh* (NONCSP オプションと NOMAPS オプションの置換表現)。
- **transfer** ステートメントと **show** ステートメントの *externallyDefined* オプション (DXFR ステートメントと XFER ステートメントの NONCSP オプションの置換表現)。
- 印刷用の **printForm** ステートメントと同じ、表示用の **printForm** ステートメントがインプリメントされます。
- 書式フィールドの初期値は、値が割り当てられていないフィールドを画面に表示するときのみ使用されます。この設定により、ストレージ内のフィールドの初期値は設定されません。
- **handleSysLibraryErrors** システム変数 (EZEREPLY の置換表現)。
- **handleHardDLIErrors** システム変数 (EZEDLERR の置換表現)。
- 従来の EZESYS の VAGen 値を提供する **getVAGSysType** システム関数。
- **connectionService** システム関数 (EZECONCT の置換表現)。
- **segmentedMode** システム変数 (EZESEGM の置換表現)。
- **sqlIsolationLevel** システム変数 (EZESQISL の置換表現)。
- SQL レコードにないホスト変数は、データベースから検索された値が NULL ならば、プリミティブ型に応じてブランクまたはゼロに初期化されます。SQL レコード内のホスト変数は、フィールドに対して *isNullable=yes* プロパティが設定されている場合にのみ初期化されます。
- SQL WHERE 文節と EGL *prepare* ステートメント内のホスト変数参照の場合を除き、10 進フィールド (VAGen PACK フィールド) の偶数精度は 1 つ増やされます。

VAGen マイグレーション・ツールは、EGL ビルド記述子パーツにマイグレーションするすべての VAGen 生成オプション・パーツに、*vagCompatibility="YES"* オプションを自動的に追加します。*vagCompatibility* ビルド記述子オプションは、生成機能に対して、「*VisualAge Generator* の互換性」設定と同じサポートを提供するように指示します。

注: 将来、*VisualAge Generator* 互換モードの使用を中止することを検討している場合は、**マイグレーションの前に**、216 ページの『*VisualAge Generator* 互換モードの使用を中止する場合』で詳細を参照してください。例えば、ハイフン、@、または # をパーツ名から除去する必要がある場合は、マイグレーション時に名前変更ユーザー出口を使用できます。

マイグレーション・ツールの VisualAge Generator 互換モードの使用を最小化できる設定があります。例えば、マイグレーション・ツールが `vagCompatibility="YES"` ビルド記述子オプションをあらゆる VAGen 生成オプション・パーツに追加しないように指定できます。詳しくは、171 ページの『VAGen マイグレーションの設定』を参照してください。

設定内容に関わらず、マイグレーション・ツールはワークスペースをリフレッシュしたとき、VisualAge Generator 互換モードを常にオンにします。

## プログラム・プロパティ

次の 5 つのプログラム・プロパティは、マイグレーション・ツールによってすべてのプログラムに組み込まれます。

- *includeReferencedFunctions* = *yes*。関数をプログラム内でネストする必要がないように、マイグレーション・ツールはこのプログラム・プロパティを常に組み込みます。これにより、別個のプロジェクトまたはパッケージ内で共通関数のコピーを 1 つだけ保持してインポートでき、それぞれのプログラムに共通関数を組み込まずに済みます。ステージ 1 から 3 のマイグレーションを使用する場合、マイグレーション・ツールは、プログラムと異なるパッケージ内にある必要な関数の `import` ステートメントも組み込みます。
- *allowUnqualifiedItemReferences* = *yes*。フィールド (VAGen データ項目) の参照を修飾する必要がないように、マイグレーション・ツールはこのプログラム・プロパティを常に組み込みます。非修飾フィールドに関する EGL 規則に VAGen 規則を組み込むことにより、非修飾フィールドは VisualAge Generator と同じレコード、dataTable (VAGen テーブル)、または書式 (VAGen マップ) に解決されます。マイグレーション・ツールは修飾を追加しません。
- *localSQLScope* = *yes*。yes がデフォルト値ですが、マイグレーション・ツールはこのプログラム・プロパティを常に組み込みます。マイグレーション・ツールが結果セット ID と `prepare` ステートメント ID の作成に使用する命名規則は、プログラム間での固有性を保証しません。このため、VAGen の振る舞いを保持するために *localSQLScope* = *yes* を指定する必要があります。
- *throwNrfEofExceptions* = *yes*。NRF (noRecordFound) と EOF (endOfFile) がエラー条件として扱われるように、マイグレーション・ツールはこのプログラム・プロパティを常に組み込みます。EGL の場合、NRF と EOF は通常はエラー条件として扱われません。このため、VAGen の振る舞いを保持するために *throwNrfEofExceptions* = *yes* が必要です。
- *handleHardIOErrors* = *no*。VGVar.handleHardIOErrors のデフォルト値が 0 に設定されるように、マイグレーション・ツールは常にこのプログラム・プロパティを組み込みます。VGVar.handleHardIOErrors は EZEFE の互換表現です。EGL の VGVar.handleHardIOErrors のデフォルト値は、通常は 1 です。一方、VAGen の EZEFE のデフォルト値は 0 です。このため、VAGen の振る舞いを保持するために *handleHardIOErrors* = *no* が必要です。

マイグレーション・ツールはすべての DLI プログラムまたは IMS プログラム用の @DLI complex プロパティを組み込んでおり、以下のプロパティを設定します。



- *psb* = "*psbVariableName*". マイグレーション・ツールは、このプログラム・プロパティを組み込んで、PSB レコード・パーツ名を表す変数の名前を指定するようにします。マイグレーション・ツールは常に *psb* を変数名として使用します。
- *callInterface* = *DLICallInterfaceKind.CBLTDLI*。マイグレーション・ツールは、このプログラム・プロパティを組み込んで *CBLTDLI* を呼び出しインターフェースとして使うようにします。*CBLTDLI* は、VisualAge Generator が使用するのと同じ呼び出しインターフェースを備えています。EGL は *AIBTDLI* をデフォルトの呼び出しインターフェースとして使用します。*AIBTDLI* を使用したい場合は、PCB 名情報を IMS PSB と EGL PSB レコード・パーツに追加する必要があります。
- *handleHardDLIErrors* = *no*。マイグレーション・ツールは、このプログラム・プロパティを組み込んで、*dliVar.handleHardDLIErrors* のデフォルト値が 0 に設定されるようにします。*dliVar.handleHardDLIErrors* は EZEDLERR の置換表現です。EGL の *dliVar.handleHardDLIErrors* のデフォルト値は、通常は 1 です。ただし、VAGen の EZEDERR のデフォルト値は 0 です。このため、VAGen の振る舞いを保持するために *handleHardDLIErrors* = *no* を指定する必要があります。
- *pcbParms* = [ PCB パラメーターのリスト ]。VAGen プログラムが EZEDLPCB[n] (*n* は数値リテラル) を呼び出し先パラメーターとして組み込んでいる場合、マイグレーション・ツールは、入力 PCB パラメーターをプログラムの PSB レコード・パーツ内にある PCB にマッピングするために、*pcbParms* プロパティを組み込みます。

## EGL のビルド・パスと import ステートメント

EGL を使用すると、ワークスペース内に複数のパーツ名の定義を同時に設定できます。プロジェクトの EGL ビルド・パスにより、パーツ名を検索する際に対象になる他のプロジェクトが限定されます。ファイル内の *import* ステートメントにより、パーツ名を検索する際に対象になる、EGL ビルド・パスに含まれるパッケージ (現行パッケージ以外) とパーツが決定されます。

ほとんどの場合、パーツ参照の解決には EGL ビルド・パスと *import* ステートメントがあれば十分です。例えば、プログラムのレコード宣言の中でレコードを型定義として使用している場合は、EGL ビルド・パスとプログラムの *import* ステートメントがあれば、レコード名の解決には十分です。また、レコード定義、関数ローカル・ストレージ、または関数仮パラメーター・リストに対して使用できる *dataItem* パーツの定義がただ 1 つならば、*dataItem* パーツ参照の解決には EGL ビルド・パスと *import* ステートメントがあれば十分です。

例えば、サブシステム A とサブシステム B を使用していて、これらに 2 つの異なる RECORDX の定義が存在するとします。サブシステム A 内のプログラムはすべて、RECORDX のサブシステム A による定義を使用する必要があります。このためには、次のようにします。

- サブシステム A 内のプロジェクトに対する EGL ビルド・パスのプロパティに、サブシステム A の RECORDX の定義を提供するプロジェクトを組み込む必要があります。
- レコードの型宣言として RECORDX を使用するサブシステム A 内のプログラム用のファイルに、サブシステム A 内で RECORDX の定義を含むパッケージの *import* ステートメントを組み込む必要があります。

サブシステム A のプロジェクトに対する EGL ビルド・パスのプロパティにより、検索対象になるプロジェクトは、サブシステム A 内にあるプロジェクトと共通プロジェクトのみに限定されます。サブシステム A 内にあるファイルの `import` ステートメントにより、EGL ビルド・パスの中で検索対象になるパッケージが限定されます。RECORDX が `dataItem` パーツ ITEM1 を型定義として使用する場合に、2 つのサブシステムの ITEM1 の定義が異なっている場合でも、ITEM1 への参照を解決するためには、EGL のビルド・パスと `import` ステートメントがあれば十分です。それぞれのサブシステム内の RECORDX を含むプロジェクトは、ITEM1 の対応するサブシステム定義を含むサブシステム・プロジェクトを組み込んで、EGL ビルド・パスのプロパティを指定する必要があります。それぞれのサブシステム内の RECORDX を含むファイルには、ITEM1 の対応するサブシステム定義を含むサブシステム・パッケージを指定する `import` ステートメントが必要です。

ステージ 1 から 3 のマイグレーションを使用する場合は、**マイグレーション・セット内のパーツに基づいて**、マイグレーション・ツールは次の処理を行います。

- プロジェクトが他のプロジェクト内で参照する必要があるパーツに基づいて、それぞれのプロジェクトごとに EGL ビルド・パスを設定します。
- それぞれのファイルごとに、そのファイルを含むプロジェクトの EGL ビルド・パス内で、ファイルが参照する必要がある他のパッケージ内のパーツに基づいて、パッケージの `import` ステートメントの大部分を組み込みます。これらの `import` ステートメントは、マイグレーションのステージ 1 で VisualAge Generator によって判別されたパーツの関連に基づいています。
- マイグレーション・ツールは、データ項目パーツに対する `import` ステートメントを追加します。ステージ 2 のマイグレーション中に、マイグレーション・ツールはデータ項目パーツに編集ルーチンがあるかどうかを判別します。編集ルーチンとして指定されたテーブルまたは関数がマイグレーション・セットに含まれている場合、マイグレーション・ツールはマイグレーション・データベースを更新して、編集ルーチンとして指定されているパーツをデータ項目の関連パーツとして組み込みます。
- マイグレーション・ツールは、UI レコードに関する `import` ステートメントを追加します。ステージ 2 のマイグレーション中に、マイグレーション・ツールは UI レコード内のいずれかのフィールドがプログラム・リンク情報を指定しているかどうかを判別します。UI レコード内のフィールドがプログラム・リンク情報を指定していて、かつ参照先プログラムと最初の UI レコードがマイグレーション・セットに組み込まれている場合、マイグレーション・ツールはマイグレーション・データベースを更新して、参照先プログラムと最初の UI レコードを UI レコードの関連パーツとして組み込みます。
- 次に示す状況では、マイグレーション・ツールは `import` ステートメントを追加しません。これは、VisualAge Generator 内ではこれらのものが関連パーツではないからです。
  - `CALL`、`DXFR`、または `XFER` のいずれかのステートメントを使用してプログラムに転送される関数。Java 用の生成を行っている場合は、その関数を含むファイル内でそのプログラムを含むパッケージの `import` ステートメントを追加するか、プログラム名をパッケージ名によって完全に修飾する必要があります。また、リンクエッジ・オプション・パーツ内の項目を使用して、プログラムがあるパッケージの名前を指定することもできます。

- .eglbld ファイル内のビルド・パーツの場合。生成オプション・パーツなどの VAGen 制御パーツは関連パーツをリストしないので、マイグレーション・ツールは情報を入手できません。また、EGL がビルド記述子パーツを処理する方法が原因で、nextBuildDescriptor 値の再配列が必要になる可能性があります (VAGen /OPTIONS)。さらに、この再配列のためには、マイグレーション・ツールが行ったインポートを変更する必要があります。

注: ステージ 1 マイグレーション・ツールは、マイグレーション・セット内のパーツを分析して、各パーツの関連パーツを判別します。マイグレーション・セット内のパーツのみが分析対象に含まれるように、上位 PLP プロジェクトによって指定されたマイグレーション・セットをロードする前に、ステージ 1 マイグレーション・ツールはすべての Java プロジェクトをワークスペースから削除します。同様に、上位構成マップによって指定されたマイグレーション・セットをロードする前に、ステージ 1 マイグレーション・ツールはすべての Smalltalk 構成マップを削除します。関連パーツの分析対象はマイグレーション・セットに限定されるので、マイグレーション・ツールは、マイグレーション・セットに含まれない EGL プロジェクトを指定する EGL ビルド・パス・プロパティを設定しません。また、マイグレーション・ツールは、マイグレーション・セットに含まれない EGL パッケージの import ステートメントを組み込みません。

単一ファイル・マイグレーションを使用する場合、マイグレーション・ツールの動作は次のようになります。

- 常に単一の既存プロジェクトが出力先になるので、EGL ビルド・パスは設定しません。
- 作成されたファイルはすべて同じパッケージに入れられるので、パッケージの import ステートメントは組み込みません。このため、import ステートメントは不要です。

## containerContextDependent プロパティ

注: ここで説明するのは、EGL バージョン 6.0.1 に部分的にのみインプリメントされている機能です。関数に containerContextDependent を指定すると、その関数内での関数呼び出しの解決は (開発時でなく) 生成時に行われ、呼び出し側関数を使用するプログラムまたは pageHandler のネーム・スペースへの参照を組み込みます。現時点では、レコード・パーツまたは dataItem パーツのネーム・レゾリューションに対して containerContextDependent の効果はありません。

以降の説明は、最終インプリメンテーションとして意図されている内容を反映したものです。

38 ページの『EGL のビルド・パスと import ステートメント』で説明したとおり、一般には EGL ビルド・パスと import ステートメントがあれば、必要なパーツ名の解決には十分です。ただし EGL は、ファイルを保管するたびにパーツ名参照すべての解決を期待します。パーツ名を解決できない場合、EGL は「問題」ビューにエラー・メッセージを追加します。アーキテクチャーによっては、レコードまたは関数の containerContextDependent プロパティを使用する必要があることもあります。

RECORDX が、FUNCTIONY 内で関数仮パラメーターの型定義として使用されるという状況を考えます。RECORDX と FUNCTIONY が異なるプロジェクトとパッケージの中にあるとすると、EGL は次のことを期待します。

- FUNCTIONY を含むプロジェクトの EGL ビルド・パスには、RECORDX の定義を含むプロジェクトが組み込まれている。
- FUNCTIONY を含むファイルには、RECORDX を含むパッケージの import ステートメントが組み込まれている。

すべてのサブシステムにある RECORDX の定義が同じならば、EGL ビルド・パスと import ステートメントがあれば十分で、EGL は FUNCTIONY を含むファイルを保管するときに RECORDX のパーツ参照をいつでも解決できます。

一方で、サブシステム A とサブシステム B が両方とも FUNCTIONY を使用し、ただし異なる RECORDX の定義を使用するという状況を考えましょう。この状況では、EGL ビルド・パスと import ステートメントは両方のサブシステムを同時に指定できません。EGL は、関数の `containerContextDependent` プロパティをサポートします。この状況では、FUNCTIONY に対して `containerContextDependent=yes` を指定できます。このプロパティは、関数仮パラメーターとローカル・ストレージのパーツ名参照が、プログラム内で FUNCTIONY が使用されるまで解決されないように指定します。FUNCTIONY を使用するプログラムをテストまたは生成するとき、プログラムを含むプロジェクトの EGL ビルド・パスと、プログラムを含むファイルの import ステートメントによって、RECORDX の定義を検索する場所が決まります。`containerContextDependent=yes` を使用すれば、関数に対して VisualAge Generator の機能と同じ柔軟性が得られます。サブシステム内の各プロジェクトの EGL ビルド・パスと、サブシステム内のプログラムを含むファイルの import ステートメントが、そのサブシステムの RECORDX の定義を指示します。

`containerContextDependent` プロパティは、レコード・パーツにも対応しています。例えば、SubsystemA と SubsystemB が両方とも RECORDZ の同じ定義を使用しているとします。ただし、RECORDZ は ITEM1 という名前の `dataItem` パーツを参照する型定義を使用します。各サブシステムの ITEM1 の定義は異なります。この場合は、RECORDZ に対して `containerContextDependent=yes` を指定すれば、RECORDZ がプログラム内で使用されるまで、EGL の検証機能は ITEM1 の解決を試行しません。プログラムを含むプロジェクトの EGL ビルド・パスと、プログラムを含むファイルの import ステートメントによって、ITEM1 の定義を検索する場所が決まります。

マイグレーション・ツールは、`containerContextDependent` プロパティの設定を自動的に行いません。これは、マイグレーション・ツールが必ずしもすべてのサブシステムを同時にマイグレーションするとは限らず、重複するパーツ定義があることを判別するために全パーツの定義すべてを完全に分析しないからです。定義時 (EGL の場合のように) でなく、テストおよび生成時 (VisualAge Generator の場合のように) に解決する必要がある重複パーツ名があるとユーザーが判断した場合に、必要に応じて `containerContextDependent` プロパティを追加できます。

## EGL の予約語リスト

EGL には予約語リストがあります。パーツ名は、EGL の予約語と同じであってはなりません。さらに EGL の場合は、EGL パーツ名の先頭文字として # 記号または @ 記号を使用することはできません。VAGen パーツに EGL の予約語と同じ名



前が付いている場合、または VAGen パーツの先頭文字が # または @ である場合は、パーツ型に応じてマイグレーション・ツールは次の処理を行います。

- マイグレーション・ツールは、プログラム、マップ・グループ、またはテーブルの名前を変更しません。これは、これらのパーツが、VAGen 以外のプログラム、またはランタイム環境を参照する場合があります (例えば、CICS PROGRAM 定義)。
- マイグレーション・ツールは、パーツ名に名前変更接頭部を付けることによって、データ項目、レコード、マップ、および関数の名前を変更します。名前変更接頭部は、ステージ 2 または単一ファイル・モードのマイグレーションに関して指定できる、VAGen マイグレーション設定の 1 つです。

**注:** 名前変更について、マイグレーション・ツールは UI レコードを他のレコードと同じに扱います。さらに、UI レコードの名前を変更する必要がある場合は、ステージ 3 マイグレーション・ツールはその UI レコードを含む .egl ファイルの名前を変更して UI レコードの変更後の名前と一致するようにします。

- マイグレーション・ツールは、次の場合を除いて制御パーツの名前を変更しません。
  - マイグレーション・ツールは、バインド制御パーツ名の末尾から .BND 接尾部を除去します。
  - マイグレーション・ツールは、リンク・エディット・パーツ名の末尾から .LKG 接尾部を除去します。
  - マイグレーション・ツールは、制御パーツ名に含まれるその他のドットを下線に変更します。さらにこのツールは、/OPTIONS、/RESOURCE、および /LINKEDIT 生成オプション内で参照されている制御パーツ名のドットを下線に変更します。

ステージ 1 マイグレーション・ツールには、EGL の予約語リストと競合するプログラム名、マップ・グループ名、テーブル名、および制御パーツ名のリストがあります。マイグレーション前にこれらのパーツの名前を変更しなければ、ステージ 2 マイグレーション・ツール (または単一ファイル・モード) からエラー・メッセージが出されます。「問題」ビューにエラーが示されます。プログラム、formGroup、または dataTable の名前を変更し、またオプションで EGL *alias* プロパティを使用して EGL の問題を訂正できます。

**注:** ステージ 2 マイグレーション・ツールは、そのツールで名前変更される UI レコードすべてについて警告メッセージを発行します。ステージ 3 マイグレーション・ツールも .egl ファイル名を変更するため、UI レコードの「問題」ビューにエラーは表示されません。

## EGL ファイル内でのパーツの配置

ステージ 1 から 3 のマイグレーションを使用してマイグレーションを行う場合、それぞれの Java パッケージまたは Smalltalk アプリケーションは、ステージ 1 の名前変更規則に基づいて、対応する EGL パッケージにマイグレーションされます。オリジナルの Java パッケージまたは Smalltalk アプリケーション内の VAGen パーツは、次の条件に基づいて、対応する EGL パッケージ内で 1 つ以上の EGL ファイルに配置されます。

- パーツのタイプ:
  - 生成可能パーツ (プログラム、テーブル、マップ・グループ、または UI レコード)
  - 制御パーツ (生成オプション、リソース関連、リンケージ・テーブル、リンク・エディット、またはバインド制御)
  - その他のマイグレーション可能パーツ (データ項目、マップ、関数、PSB、および UI レコード以外のレコード)
- 共通パーツを含む Java プロジェクトまたはパッケージ名の識別を可能にするステージ 1 設定。同様に、共通パーツを含む構成マップまたはアプリケーション名の識別を可能にする、Smalltalk 用の設定もステージ 1 にあります。
- パーツが他のパーツによって使用されているかどうか。ステージ 1 マイグレーション・ツールは、次の条件に基づいてパーツが使用されているかどうかを判断します。
  - パーツが**マイグレーション・セット内**の生成可能パーツの VAGen 関連リストにあれば、パーツは「使用されている」。
  - ステージ 1 設定に指定されているとおり、パーツが共通 Java プロジェクトまたはパッケージ内、または共通 Smalltalk 構成マップまたはアプリケーション内にあれば、パーツは「使用されている」。

ステージ 1 マイグレーション・ツールは、すべてのパーツの配置を決定します。ステージ 1 マイグレーション・ツールは、単一の Java パッケージまたは Smalltalk アプリケーション内にある VAGen パーツを、対応する EGL パッケージ内の EGL ファイルに次のように配置します。

- すべての制御パーツが、eglPackageName.eglbid という名前の単一ファイルに配置されます (ここで、eglPackageName は対応する EGL パッケージの名前)。
- それぞれのプログラム・パーツが、programName.egl という名前のファイルに配置されます (ここで、programName はプログラムの名前)。
- それぞれのテーブル・パーツが、tableName.egl という名前のファイルに配置されます (ここで、tableName はテーブルの名前)。
- それぞれのマップ・グループと、マップ・グループ内のマップすべてが、mapGroupName.egl という名前のファイルに配置されます (ここで、mapGroupName はマップ・グループの名前)。マップ・グループ・パーツがない場合、ステージ 1 マイグレーション・ツールはダミーのマップ・グループ・パーツを作成します。マップ・グループと、そのマップ・グループ内にあるマップはすべて同じファイルに配置する必要があるため、これらのパーツはグループとして考える必要があります。このため、一部のパーツが元は同じ Java パッケージまたは Smalltalk アプリケーション内になかった場合、そのパーツは異なる EGL パッケージまたはプロジェクトに移動されることがあります。マイグレーション・ツールは、mapGroupName.egl ファイルを配置する場所を次のように決定します。
  - マップ・グループとそのマップすべてが**同じ Java パッケージ内にある場合**、mapGroupName.egl ファイルは対応する EGL パッケージに配置されます。同様に、マップ・グループとそのマップすべてが**同じ Smalltalk アプリケーション内にある場合**、mapGroupName.egl ファイルはその Smalltalk アプリケーションに対応する EGL パッケージに配置されます。この場合、マイグレーション

ン・ツールはプログラムやテーブル・ファイルと同じ方法で  
mapGroupName.egl ファイルを処理します。これは最もよく起こる状態です。

- マップ・グループとそのマップが、**1 つのプロジェクト内で複数の Java パッケージに分散している場合は**、プロジェクト名に接尾部を付加した名前が、mapGroupName.egl ファイルを格納する新規 EGL パッケージ名の作成に使用されます。この新規 EGL パッケージは、オリジナルのプロジェクト内に配置されます。同様に、マップ・グループとそのマップが 1 つの構成マップ内で複数の Smalltalk アプリケーションに分散している場合は、構成マップ名に接尾部を付加した名前が、mapGroupName.egl ファイルを格納する新規 EGL パッケージ名の作成に使用されます。Java と Smalltalk のどちらの場合も、ステージ 1 設定を使用して接尾部を制御できます。
- マップ・グループとそのマップが、**複数の Java プロジェクトに分散している場合は**、マイグレーション・セット名に接尾部を付加した名前が、mapGroupName.egl ファイルを格納する新規 EGL プロジェクト名の作成に使用されます。同様に、マップ・グループとそのマップが複数の Smalltalk 構成マップに分散している場合は、マイグレーション・セット名に接尾部を付加した名前が、mapGroupName.egl ファイルを格納する新規 EGL プロジェクト名の作成に使用されます。Java と Smalltalk のどちらの場合も、ステージ 1 設定を使用して接尾部を制御できます。
- 各 UI レコードはそれぞれ単独で uiRecordName.egl という名前のファイルに格納されます。uiRecordName は UI レコードの名前です。
- その他のパーツはすべて、次のように配置されます。
  - パーツがマイグレーション・セット内の**ただ 1 つのプログラムによって使用されている場合**、パーツは次のように配置されます。
    - パーツがプログラムと同じパッケージ内にある場合、パーツはプログラムと同じファイルに配置されます。例えば、プログラムの main 関数 (ProgramA-MAIN) は、その関数が他のプログラム内で使用されていないければ、プログラム (ProgramA) と同じファイルに配置されます。このファイルは、プログラムに合わせて ProgramA.egl という名前になります。
    - パーツがそのパーツを使用するプログラムとは異なるパッケージ内にある場合、パーツは次のように配置されます。
      - パーツが共通プロジェクトまたはパッケージ内にある場合は、そのパーツのオリジナル・パッケージ内で、commonParts.egl という名前のファイルにパーツが配置されます。commonParts の名前は、ステージ 1 設定によって制御します。
      - パーツが共通プロジェクトまたはパッケージ内に存在しない場合も、そのパーツのオリジナル・パッケージ内で、commonParts.egl という名前のファイルにパーツが配置されます。
  - パーツがマイグレーション・セット内の**複数のプログラムによって使用されている場合**、パーツはオリジナル・パッケージ内で commonParts.egl という名前のファイルに配置されます。例えば、ProgramA が ProgramB を呼び出して RecordR を渡す場合、RecordR が配置される先のファイルは、RecordR を含むオリジナルの Java パッケージまたは Smalltalk アプリケーションに対応する EGL パッケージ内にある、commonParts.egl という名前のファイルです。
  - パーツがマイグレーション・セット内のプログラムによって**使用されていない場合**、パーツは次のようなファイルに配置されます。



- パーツが共通 Java プロジェクトまたはパッケージ内にある場合は、そのパーツを含むオリジナル Java パッケージに対応する EGL パッケージ内で、`commonParts.egl` という名前のファイルにパーツが配置されます。同様に、パーツが共通 Smalltalk 構成マップまたはアプリケーション内にある場合は、そのパーツを含むオリジナル Smalltalk アプリケーションに対応する EGL パッケージ内で、`commonParts.egl` という名前のファイルにパーツが配置されます。
- パーツが共通 Java プロジェクトまたはパッケージ内に存在しない場合は、そのパーツを含むオリジナル Java パッケージに対応する EGL パッケージ内で、`unusedParts.egl` という名前のファイルにパーツが配置されます。同様に、パーツが共通 Smalltalk 構成マップまたはアプリケーション内に存在しない場合は、そのパーツを含むオリジナル Smalltalk アプリケーションに対応する EGL パッケージ内で、`unusedParts.egl` という名前のファイルにパーツが配置されます。`unusedParts` の名前は、ステージ 1 設定によって制御します。
- 次のように特別な考慮事項があります。
  - マップ・レコードまたは UI レコードの編集ルーチンとして使用される関数は、関数が使用されるか使用されないかの判断に影響します。ただしマイグレーション・ツールは、常に関数をプログラムと一緒に配置するか、`commonParts` ファイルに配置します。マイグレーション・ツールが、マップ・グループ用に作成されたファイル、または UI レコード用に作成されたファイルに編集関数を配置することはありません。
  - テーブル内または UI レコード内で使用されている共用項目は、データ項目が使用されるか使用されないかの判断に影響します。ただしマイグレーション・ツールは、常にデータ項目をプログラムと一緒に配置するか、`commonParts` ファイル内に配置します。マイグレーション・ツールは、テーブル用に作成されたファイルまたは UI レコード用に作成されたファイルにデータ項目を配置することはありません。

**注:** マイグレーション・データベースを消去せずに、複数のマイグレーション・セット、またはマイグレーション・セットの複数バージョンをマイグレーションした場合は、いずれかのパーツ・エディションを含む、ステージ 1 で最初に処理されたマイグレーション・セットのバージョンによって、EGL パーツのプロジェクト、パッケージ、およびファイル名が制御されます。マイグレーション・セットのバージョンごとの定義に従ってパーツが配置されるようにするには、バージョンが変わるたびにマイグレーション・データベースを消去する必要があります。また、ステージ 1 における最新バージョンのマイグレーション・セットをマイグレーションすることにより、前バージョンのマイグレーション・セット以降変化していないすべてのパーツ・エディションが、現在使用されているかどうかを基準にして EGL ファイルに配置されるようにします。例えば、以前に 1 つのプログラムでしか使用されていないパーツが複数のプログラムで使用されるようになっていくとします。この場合、最新バージョンのマイグレーション・セットを最初にマイグレーションすると、パーツは、そのパーツの最初の (そして唯一の) ユーザーであったプログラムではなく、共通パーツ EGL ファイル内に配置されます。

Java および Smalltalk 用のステージ 1 マイグレーション・ツールは、サンプル・コードとして提供されています。お客様独自のライブラリー管理方針に基づいてパーツを配置するように、ステージ 1 マイグレーション・ツールを変更できます。例えば、次のようになります。

- ProgramX が ProgramY を呼び出して、レコード ProgramY-Parm と Common-Parm を渡す場合は、ProgramY-Parm を ProgramY と一緒のファイルに配置し、Common-Parm を commonParts ファイルに配置できます。命名規則が分かっている場合は、ステージ 1 マイグレーション・ツールを修正してファイル配置アルゴリズムを変更できます。
- 大規模なパッケージの場合、パーツをパーツ型別、またはパーツ名の最初の数文字別に複数ファイルに分割できます。
- 同じパーツ・エディションがマイグレーション・セットの複数バージョンに存在し、ただしマイグレーション・セットのバージョンに応じて、このパーツ・エディションを異なる EGL プロジェクト、パッケージ、またはファイルに配置する必要がある場合は、マイグレーション・セットのいずれかのバージョンを処理するたびに、パーツごとの新しい EGL プロジェクト、パッケージ、およびファイル名に応じてマイグレーション・データベースを更新できます。この変更を行う場合は、それぞれのマイグレーション・セット・バージョンをステージ 1 から 3 まで完全に処理してから、次のマイグレーション・セット・バージョンのマイグレーションを開始するようにしてください。
- Java パッケージごとに 1 つのプログラム、そしてプロジェクトごとに 1 つのパッケージを配置する方針をとっていた場合、複数の Java パッケージまたはプロジェクトを結合して、ソース・コード・リポジトリで管理する必要のある EGL パッケージおよびプロジェクトの数を減らすことができます。同様にまた、Smalltalk アプリケーションごとに 1 つのプログラム、そして構成マップごとに 1 つのアプリケーションを配置する方針をとっていた場合、EGL プロジェクトおよびパッケージを作成するとき複数の Smalltalk アプリケーションまたは構成マップを結合することができます。

## プログラムを使用したマイグレーション

通常、マイグレーションの際には、グループとしてマイグレーションする必要があるすべての Java プロジェクトまたは Smalltalk 構成マップを識別するマイグレーション・セットを指定します。マイグレーション・ツールは、マイグレーション・セットを使用して、プログラムとその関連パーツを最初にマイグレーションします。これによって、EGL 言語が VisualAge Generator よりも厳密または制限的である場合に、ツールがその状態を解決するための支援情報として特定プログラムのコンテキストを使用できます。最初にマイグレーションされるプログラムと、その関連パーツの組み合わせによって、そのプログラムまたは関連パーツの中で未確定状態に対応する EGL 構文が決まります。プログラムが異なると、共用データ項目、共用レコード、マップ、テーブル、または関数の同じ未確定状態に対して、異なる解決結果が得られる場合があります。パーツ・バージョンのマイグレーションは一度限り行われるので、共通パーツを最初に使用するプログラムが、その関連パーツの未確定状態の解決を制御します。

例として、ProgramA が表示マップを使用するメインのトランザクション・プログラムであり、ProgramB がプリンター・マップを使用するメインのバッチ・プログラムである場合を考えます。これらのプログラムは、HEADER マップと DETAIL マッ

プを表示する共通の関数を共用します。共通関数はまた、SET map PAGE ステートメントを使用して画面のクリアまたはページ替えの強制を行います。この例では、ProgramA を先にマイグレーションした場合、マイグレーション・ツールは、display ステートメントと clearScreen システム・ライブラリー関数を使用する EGL ソースを関数用に作成します。ProgramB を先にマイグレーションした場合、マイグレーション・ツールは、print ステートメントと pageEject システム・ライブラリー関数を使用する EGL ソースを作成します。

プログラムとその関連パーツをマイグレーションするときには必ず、共用データ項目、共通レコード、マップ、テーブル、または関数を最初に使用するプログラムが、EGL コードの生成結果を制御します。ほとんどの場合、プログラムは共通コードを同じ方法で使用するので、この技法によって VAGen ソースの最適なマイグレーションが実現します。ただし、この例から分かるように、共通コードの実行内容として意図された特性が、結果の EGL ソースには反映されない場合があります。この例では、どのプログラムを最初にマイグレーションしたかに関係なく、2 番目にマイグレーションしたプログラムをテストしたり生成したりすることはできません。VisualAge Generator 互換モードでは、display ステートメントを使用して、入出力ステートメントに関する問題を解決できます。ただし、clearScreen または pageEject の選択に関する問題を解決するには、新しい変数 TEXT-OR-PRINT を追加する必要があります。それぞれのプログラムがこの変数を初期化し、共通関数がこの変数をテストして、clearScreen または pageEject のどちらのシステム・ライブラリー関数を実行するか判別します。

## 関連パーツを使用したマイグレーション

プログラムと関連パーツが使用不可ならば、マイグレーション・ツールはマイグレーション・セット（また、単一ファイル・モードでマイグレーションする場合は、外部ソース形式ファイル）内で使用可能なすべてのパーツを使用します。この場合、パーツ間マイグレーションに必要な追加パーツが使用可能ならば、マイグレーション・ツールは正しい選択肢を高い確率で判断できます。

例えば、マップ変数フィールドが編集ルーチンを指定している場合を考えます。編集ルーチンと同じ名前の VAGen テーブルが同じマイグレーション・セット（または外部ソース形式ファイル）内にあれば、マイグレーション・ツールはこれが常に使用されるテーブルであると想定して、validatorDataTable プロパティにマイグレーションします。編集ルーチンと同じ名前の関数が存在すれば、マイグレーション・ツールは validatorFunction プロパティにマイグレーションします。どちらの場合も、編集ルーチンと同じ名前のパーツが存在するので、マイグレーション・ツールが正しい選択を行った確率は高くなります。編集ルーチンと同じ名前のテーブルまたは関数がなければ、マイグレーション・ツールは、関連パーツを使用せずにマイグレーションを行う場合と同じようにマップ変数フィールドを処理します。

多くのケースでは、関連パーツを使用してマイグレーションを行えば、プログラムを関連パーツとともにマイグレーションした場合とほぼ同様のマイグレーションを実現できます。プログラムを使用しないマイグレーションの欠点は、単一の関数内であっても、マイグレーション対象である特定のステートメント、およびマイグレーション・セットに含まれているその他のパーツに応じて、関連パーツを使用したマイグレーションから、関連パーツを使用しないマイグレーションに切り替わりやすいことです。

## 関連パーツを使用しないマイグレーション

場合によっては、プログラムが使用可能であっても、その関連パーツの一部がマイグレーション・セットに含まれていないことがあります。あるいは、マイグレーションする共通パーツが、過去にサブシステムによって使用されていたものの、現在は使用されていない場合もあります。このような場合、マイグレーション対象のパーツに関連するものが使用できなくなる可能性があります。それでもマイグレーション・ツールは、次のいずれかの技法を使用してパーツを変換します。

- **EGL 構文の柔軟性。**例えば、関連マップがない状態で `DISPLAY` 入出力オプションをマイグレーションするとします。この場合、マイグレーション・ツールは `display` ステートメントの使用を選択し、マイグレーション・ログに警告メッセージを書き込みます。マイグレーション・ツールの予測が誤っていたとしても、VisualAge Generator 互換モードを使用していれば、書式が印刷書式であっても `display` ステートメントは受け入れられます。
- **インテリジェント予測。**例えば、マップ変数フィールドで編集ルーチンが指定されているが、その編集ルーチンと同じ名前のパーツがマイグレーション・セットに存在しないとします。この場合、マイグレーション・ツールは他の情報を使用します。ツールは次の情報を調査して、`validatorDataTable` または `validatorFunction` のどちらのプロパティを使用するか判別します。
  - 編集ルーチン名の長さ。8 文字以上ならば、編集ルーチンが関数であることを示しています。
  - 編集ルーチン名が `EZEC10` または `EZEC11` である。これは、編集ルーチンが関数であることを示しています。
  - 編集メッセージも指定されている。メッセージは、編集テーブル、`EZEC10`、または `EZEC11` に対してのみ使用できます。

これらのいずれかの情報があれば、マイグレーション・ツールは `validatorDataTable` または `validatorFunction` のどちらのプロパティを設定するかインテリジェントに選択できます。決定的な選択要因がなければ、マイグレーション・ツールは `validatorFunction` プロパティを使用し、エラー・メッセージをマイグレーション・ログに書き込みます。マイグレーション・ツールの予測が誤っている場合は、「問題」ビューにもエラーが示されます。

- **意図的な無効構文。**例えば、関連マップがない状態で `SET map PAGE` をマイグレーションするとします。この場合、マイグレーション・ツールはテキスト書式用の `EGL converseLib.clearScreen` 関数を使用するか、印刷書式用の `EGL converseLib.pageEject` 関数を使用するかを選択できます。ただし、どちらの選択項目も等しくありえます。このため、マイグレーション・ツールは意図的に誤った構文を作成し、`converseLib.EZE_SETPAGE` に変換します。この結果、「問題」ビューでエラーが発生するので、問題の訂正が必要であることが分かります。
- **関連パーツの欠落による、情報がない状態での直接変換。**(関連パーツが欠落していると、マイグレーション・ツールが検出できない問題が生じる可能性があります。) 例えば、`RecordA` が `RecordB` のストレージを再定義するように指定しているとします。VisualAge Generator の場合、再定義情報は `RecordA` のレコード定義に格納されています。生成時には、`RecordA` と `RecordB` が使用可能であることが必要で、`RecordA` の再定義がプログラム内で行われます。EGL の場合、再定義情報はプログラムのみに格納されます。プログラムのマイグレーション時に `RecordA` が使用できなければ、`RecordA` がプログラム内に `redefines` プロパティ



ーを組み込む必要があるということを、マイグレーション・ツールが検出する手段はありません。redefines プロパティがなければ、EGL のテストおよび生成機能は、RecordA と RecordB を別個のデータ域として扱います。**プログラムは、VisualAge Generator の場合とは異なる動作をします。**データが正しく初期化されず、異常終了が発生する可能性があります。このため、マイグレーション済みのプログラムを生成し、テストすることを強くお勧めします。

## ファイルの上書きとマージ

ステージ 2 および 3 のマイグレーション・ウィザードは、同じマイグレーション・セットの複数バージョンの処理を制御する、いくつかの関連設定を行います。これらの設定には、次のものがあります。

- 残りの VAGen パーツをマイグレーションする
- ワークスペースにインポート（「既存ファイルを上書き」の指定あり、または指定なし）
- マイグレーションしたファイルを一時ディレクトリーに保管

「残りの VAGen パーツをマイグレーションする」は、マイグレーション・ツールがマイグレーション・セット内のパーツすべてを EGL に変換するかどうかを制御します。

- 残りの VAGen パーツをマイグレーションする 設定については、UI レコードは他のレコードと同じように扱われます。この設定では、UI レコードを生成可能パーツとはみなされません。
- 「残りの VAGen パーツをマイグレーションする」を選択しない場合は、生成可能パーツとその関連パーツのみが EGL に変換され、マイグレーション・データベースに保管されます。データ項目、レコード、および関数は、1 つ以上の生成可能パーツに関連していない限りは変換されません。制御パーツは変換されません。サブシステム・プロジェクトをマイグレーションしていて、共通プロジェクトが単一のマイグレーション・セットにある場合は、「残りの VAGen パーツをマイグレーションする」を選択解除すると役に立つことがあります。この状態では、次のことが起こります。
  - サブシステム・プロジェクトの場合は、サブシステム内で実際に使用されているパーツのみが変換されます。
  - 共通プロジェクトの場合は、生成可能パーツとその関連パーツが変換されます。また、サブシステムによって使用されているデータ項目、レコード、および関数も変換されます。それ以外の、他のサブシステムに使用されていても現行サブシステムに使用されていないデータ項目、レコード、および関数は、EGL に変換されません。

「残りの VAGen パーツをマイグレーションする」を選択解除する場合の利点は、次の 2 つです。

- サブシステム・プロジェクトの場合は、実際に使用されているパーツのみがマイグレーション・ツールによって変換されるので、コードをクリーンアップする機会になります。
- 共通プロジェクトの場合は、パーツが別のサブシステムによって実際に使用されるまで、パーツの変換を遅らせることができます。別のサブシステム用のマイグレーション・セットに共通プロジェクトを組み込むと、このサブシステム

によって使用されている追加パーツが EGL に変換され、マイグレーション・データベースに格納されます。この機能は特に、共通プロジェクトがさまざまなサブシステムの関連パーツを含んでいる場合や、さまざまなサブシステム内の生成可能パーツに関連するパーツを含んでいる場合に便利です。サブシステムがパーツを使用するまで共通パーツのマイグレーションを遅らせることにより、「関連パーツを使用して」共通パーツをマイグレーションできます。共通プロジェクトを含む次のマイグレーション・セットをマイグレーションするとき、マイグレーション元の共通パーツ・ファイルに対して行われる処理は、「既存ファイルを上書き」の選択項目によって制御されます。

- 「残りの VAGen パーツをマイグレーションする」を選択した場合は、生成可能パーツとその関連パーツが EGL に変換され、マイグレーション・データベースに保管されます。その後、生成可能パーツに関連していないデータ項目、レコード、および関数すべてが EGL に変換されます。また、制御パーツもすべて EGL に変換されます。「残りの VAGen パーツをマイグレーションする」を選択する場合の利点は、次の 2 つです。
  - サブシステム・プロジェクトの場合は、使用されているかいないかに関係なく、すべてのパーツが EGL に変換されます。このことは、コードを履歴として保持しておく必要がある場合に便利です。
  - 共通プロジェクトの場合は、将来マイグレーションする予定のサブシステム・プロジェクト内に、共通プロジェクト内のパーツに関連するものがないことが分かっている場合に、「残りの VAGen パーツをマイグレーションする」の選択が特に役立ちます。すべての共通パーツを一度に変換でき、マイグレーション・データベースに EGL を格納できます。その後、共通プロジェクトが他のサブシステム用のマイグレーション・セットに含まれている場合に、EGL はすでに変換済みであり、ワークスペースにインポートしたり、新規サブシステムの一時ディレクトリーに保管したりできます。

マイグレーション・セットの最初のバージョンに対して「残りの VAGen パーツをマイグレーションする」を選択した場合は、マイグレーション・セットの他のバージョンに対しても、引き続き「残りの VAGen パーツをマイグレーションする」を選択する必要があります。さらに、「既存ファイルを上書き」を指定する必要があります。両方のオプションを指定することにより、マイグレーション・セット内のパーツすべてが EGL ファイルに含まれるようになります。

「ワークスペースにインポート」は、マイグレーション・ツールが EGL プロジェクト、パッケージ、およびファイルのビルドをワークスペース内で行うかどうかを制御します。「ワークスペースにインポート」を選択すると、選択可能なその他のオプションが示されます。

- マイグレーション・セットの複数バージョンをマイグレーションする場合は、マイグレーションの終了時に、ワークスペースにインポートするバージョンを選択できます。「最新バージョン」(最近のバージョン)、または「最も古いバージョン」のどちらかを選択できます。最新バージョンを選択する場合の利点は、このバージョンが後続のテスト用に生成したいものである可能性が最も高いことです。最も古いバージョンを選択する場合の利点は、最も古いバージョンに対応する EGL プロジェクト、パッケージ、およびファイルが、ソース・コード・リポジトリに最初に格納されるようになることです。
- 現行マイグレーションによって作成されている EGL ファイルがワークスペースにすでに存在する場合に、その状態にどのように対処するか指定できます。

- 「既存ファイルを上書き」を選択すると、EGL ファイルは、現行マイグレーション・セット内のパーツのみを含む新規ファイルに置き換えられます。VAGen パーツ・エディションが前のマイグレーション・セットのためにすでに変換済みの場合、マイグレーション・ツールはパーツ・エディションを再度変換しません。ただし、パーツが現行マイグレーション・セットに含まれている場合、マイグレーション・ツールはそのパーツ・エディション用の EGL をファイルに組み込みます。VAGen マイグレーション設定またはデータベース I/O 設定を変更する場合、あるいは名前変更ユーザー出口設定を変更する場合は、「既存ファイルを上書き」を選択してください。この場合は、データベースをステージ 1 の終了時の状態に復元し、新しい設定を使用してステージ 2 と 3 を再度実行できます。「既存ファイルを上書き」を指定すれば、EGL ワークスペースを最初に消去せずにステージ 2 と 3 を実行できます。「残りの VAGen パーツをマイグレーションする」を指定した場合にも、「既存ファイルを上書き」を選択すると便利です。この場合、マイグレーション・セットの別のバージョンをマイグレーションすると、EGL ファイルは、マイグレーション・セットの現行バージョンに含まれるパーツ・エディションを格納している新規ファイルに置き換えられます。
- 「既存ファイルを上書き」を選択しない場合は、既存の EGL ファイルが変更されて、現行マイグレーション・セット内にある追加パーツをすべて格納します。EGL ファイル内にすでに存在しているパーツは、現行マイグレーション・セットが別のパーツ・エディションを使用していても、変更されません。「残りの VAGen パーツをマイグレーションする」を指定しない場合に、マイグレーションする共通プロジェクトのバージョンがただ 1 つであり、対象とするサブシステムが複数ならば、「既存ファイルを上書き」を選択解除すると便利です。この場合、それぞれのサブシステムをマイグレーションしながら、共通プロジェクト用の EGL ファイルを段階的に徐々に構築できます。初期の EGL ファイルに含まれているものは、最初のサブシステムによって使用されている共通パーツのみです。2 番目のサブシステムをマイグレーションするときに、マイグレーション・ツールは 2 番目のサブシステムに必要な追加パーツを EGL ファイルに追加します。マイグレーション・ツールは、オリジナル・ファイルと追加パーツのマージを行って、引き続きパーツ型の中でパーツがアルファベット順に編成されるようにします。

マイグレーション・セットの複数のバージョンをマイグレーションする場合は、「ワークスペースにインポート」を選択できます。ただし、より良い手法は、「ワークスペースにインポート」を選択解除し、代わりに「マイグレーションしたファイルを一時ディレクトリーに保管」を選択することです。一時ディレクトリーを使用すると、マイグレーション・ツールはすべてのマイグレーション・セット・バージョンを作成できます。

「マイグレーションしたファイルを一時ディレクトリーに保管」を選択すると、マイグレーション・セットの複数のバージョンをマイグレーションし、すべてのバージョンをワークスペースの外部に保管できます。このオプションでは、EGL 開発環境の複数のインスタンスを同時に使用する必要があります。したがって、大量のメモリー・リソースが必要になるので、このオプションはバッチ・モードでのみ使用することをお勧めします。「マイグレーションしたファイルを一時ディレクトリーに保管」を選択する際には、上位ディレクトリーも指定する必要があります。マイグレーション・ツールは、マイグレーション・セットのバージョンごとに、この上位ディレクトリー内にサブディレクトリーを作成します。「マイグレーションした



ファイルを一時ディレクトリに保管」は、「残りの VAGen パーツをマイグレーションする」と一緒に指定した場合に特に効果を発揮します。この状態では、1 つのマイグレーション・セット・バージョンに対応するそれぞれのサブディレクトリに、そのマイグレーション・セット・バージョンに含まれる VAGen プロジェクト・バージョンのパーツがすべて保管されます。

## 一般規則

ソース・コードをマイグレーションする際にマイグレーション・ツールが行う処理には、いくつかの一般規則が適用されます。この規則は次のとおりです。

- EZE ワードやその他のステートメントをサポートするために追加する新規変数は、プログラムに追加する必要があります。関数の中でローカル・ストレージとして追加すると、問題が起こる可能性があります。これは、現在スタック内に存在する関数にローカル・ストレージ、パラメーター、または戻り値がある場合、セグメント化対話がサポートされないからです。関数のコンテキストが不明なので、セグメント化対話に関連付けられた関数スタックにその関数が入っているかどうか判別する手段がありません。このため、新規変数はすべてプログラムに追加され、関数のローカル・ストレージとしては追加されません。詳しくは、90 ページの『マイグレーションに必要な中間変数』を参照してください。
- システム共通プロダクトや VisualAge Generator の旧リリースから VisualAge Generator 4.5 にマイグレーションされたパーツの中で、VisualAge Generator 4.5 内では変更されなかったパーツが存在する可能性があります。場合によっては、外部ソース形式から情報が欠落していたり、EGL ではサポートされない方法で情報が指定されていたりすることがあります。マイグレーション・ツールは、欠落情報の提供や情報の訂正を試行します。次のような処理が行われます。
  - メインのトランザクションの場合、VAGen のセグメンテーション情報が欠落していれば、マイグレーション・ツールは EGL セグメント化プロパティをデフォルトで *no* に設定します。
  - SQL レコードの場合、SQL データ・コードが欠落していれば、マイグレーション・ツールはその項目を固定長項目に変換します。
  - SQL 関数の場合、欠落している必須 SQL 文節があれば、マイグレーション・ツールは入出力オブジェクトとして指定されたレコードに基づいて、その文節の作成を試みます。
- 関数内では、マイグレーション・ツールはすべてのステートメントを何らかに変換します。これにより、IF / ELSE / END と WHILE / END のロジックは保持されます。ただし、得られるステートメントの構文が正しくない可能性があります。例えば、次のようになります。
  - EZESYS 値が EGL ではサポートされない場合、マイグレーション・ツールは VAGen 値を使用します。その値がサポートされない場合は、「問題」ビューにメッセージが示されます。
  - EZESCRPT 特殊機能語はコメント化されます。対応する置換表現は EGL には存在しません。EZESCRPT は IF、WHILE、または TEST ステートメント内では使用できないので、関数のロジック構造は保持されます。マイグレーション・ツールはエラー・メッセージを出し、「問題」ビューにエラーは示されません。
- プログラムを使用せずにパーツ参照の解決を試みる際に、マイグレーション・ツールはマイグレーション・セット内のパーツを調べます。このため、プロジェク

トのグループに含めるマイグレーション・セットは、使用に適した組み合わせになるように定義する必要があります。例えば、次のようになります。

- 競合するパーツ名をもつ複数のサブシステムからのプロジェクトを組み込まない。
- サブシステムをマイグレーションする場合は、共通 Java プロジェクト、または共通 Smalltalk アプリケーションを組み込む。
- マイグレーション時にマイグレーション・ツールが行わないことがいくつかあります。
  - 次に示す状況では、マイグレーション・ツールは `import` ステートメントを追加しません。これは、VisualAge Generator 内ではこれらのものが関連パーツではないからです。
    - `CALL`、`DXFR`、または `XFER` のいずれかのステートメントを使用してプログラムに転送される関数。Java 用の生成を行っている場合は、その関数を含むファイル内でそのプログラムを含むパッケージの `import` ステートメントを追加するか、プログラム名をパッケージ名によって完全に修飾する必要があります。また、リンケージ・オプション・パーツ内の項目を使用して、プログラムがあるパッケージの名前を指定することもできます。
    - `.eglblld` ファイル内のビルド・パーツの場合。生成オプション・パーツなどの VAGen 制御パーツは関連パーツをリストしないので、マイグレーション・ツールは情報を入手できません。また、EGL がビルド記述子パーツを処理する方法が原因で、`nextBuildDescriptor` 値の再配列が必要になる可能性があります (VAGen /OPTIONS)。さらに、この再配列のためには、マイグレーション・ツールが行ったインポートを変更する必要が生じます。

**注:** ステージ 1 マイグレーション・ツールは、マイグレーション・セット内のパーツを分析して、各パーツの関連パーツを判別します。マイグレーション・セット内のパーツのみが分析対象に含まれるように、上位 PLP プロジェクトによって指定されたマイグレーション・セットをロードする前に、ステージ 1 マイグレーション・ツールはすべての Java プロジェクトをワークスペースから削除します。同様に、上位構成マップによって指定されたマイグレーション・セットをロードする前に、ステージ 1 マイグレーション・ツールはすべての Smalltalk 構成マップを削除します。関連パーツの分析対象はマイグレーション・セットに限定されるので、マイグレーション・ツールは、マイグレーション・セットに含まれない EGL プロジェクトを指定する EGL ビルド・パス・プロパティを設定しません。また、マイグレーション・ツールは、マイグレーション・セットに含まれない EGL パッケージの `import` ステートメントを組み込みません。

- EGL は、暗黙項目を許容しません。VisualAge Generator は暗黙項目を許容します (ただし推奨されません)。暗黙項目は、プログラム内で使用される項目ですが、そのプログラムによって使用されるレコード、テーブル、またはマップ内では明示的に定義されていないものです。非修飾のデータ項目をすべて評価して、暗黙項目かどうか判別する処理はパフォーマンスに影響を及ぼすため、マイグレーション・ツールは暗黙項目の定義を作成しません。マイグレーションの前に、ユーザーが暗黙項目の定義を提供する必要があります。マイグレーションの前に問題を解決するには、VisualAge Generator 内でプログラムを検証して、プログラムが実際に暗黙項目を使用しているかどうか判別してください。暗黙項目が使用されている場合、VAGen 検証機能はその項目の正しい定

義を示すメッセージを出します。マイグレーションの前に暗黙項目の定義を作成しない場合は、「問題」ビューにエラーが示され、EGL の中で問題を訂正できます。

- マイグレーション・ツールは、`containerContextDependent` プロパティの設定を試行しません。このプロパティは、サブシステムによって提供される他のパーツを参照する必要がある、特定の共通レコードまたは共通関数に対して後で追加できます。レコードまたは関数に対してこのプロパティを使用する方法について詳しくは、40 ページの『`containerContextDependent` プロパティ』を参照してください。
- マイグレーション・ツールは、VAGen 構文が有効であることと、マイグレーション・セットに含まれるパーツを使用するプログラムが VisualAge Generator 内で正常に検証できることを前提とします。マイグレーション・ツールは、無効な構文の修復を試行しません。例えば、次のようになります。
  - マップ配列の要素が異なる編集特性や属性を備えている場合は、配列の先頭要素の特性によって、EGL にマイグレーションされる内容が決まります。マイグレーション・ツールはメッセージを出しません。
  - レコード内の共用データ項目の長さが変更されたために、親項目の長さがその副構造内にある項目の長さの合計と一致しない場合、マイグレーション・ツールは項目の長さを変更せず、メッセージを出しません。「問題ビュー」には、子の長さの合計が親の長さとは一致しないことを示すエラーが出されます。
- マイグレーション・ツールは、EGL の構文エラーが発生する場合であっても、効率が良くないコードの改良を試みません。例えば、次のようになります。
  - プログラムのテーブルおよび追加レコードのリストに同じレコードが 2 回リストされている場合、マイグレーション・ツールはそのレコードを除去せず、メッセージも出しません。「問題」ビューにエラーが示されます。同様に、プログラムのテーブルおよび追加レコードのリストに同じテーブルが 2 回リストされている場合、マイグレーション・ツールは余分なテーブルを除去せず、メッセージも出しません。「問題」ビューにエラーが示されます。
  - レコードがプログラム内で使用されていないにもかかわらず、プログラムのテーブルおよび追加レコードのリストにある場合、マイグレーション・ツールはそのレコードを除去せず、メッセージも出しません。「問題」ビューにエラーは示されません。同様に、テーブルがプログラム内で使用されていないにもかかわらず、プログラムのテーブルおよび追加レコードのリストにある場合、マイグレーション・ツールはそのテーブルを除去せず、メッセージも出しません。「問題」ビューにエラーは示されません。
  - 作業用ストレージ・レコードがプログラムのテーブルおよび追加レコードのリストにあり、レコードがレベル 77 項目のみを含んでいる場合、マイグレーション・ツールはそのレコードを除去せず、メッセージも出しません。「問題」ビューには、レコードを検出できないことを示すエラーが出されます。これは、現時点で存在するレコードが、設定したレベル 77 接尾部をレコード名の一部として含むレコードのみであるからです。
  - VAGen プログラムがマップ・グループまたはヘルプ・マップ・グループを含んでいても、必ずしも実際にマップ・グループ・パーツが存在していたとは限りません。例えば、プログラムがマップの `DISPLAY` または `CONVERSE` を行ったことがない場合、またはプログラムがマップを呼び出

し先パラメーターとして使用したことがない場合は、マップ・グループ・パーツが実際には存在していなかった可能性があります。この状態では、マイグレーション・ツールはプログラムに `formGroup` の `use` ステートメントを組み込みますが、`import` ステートメントは組み込みません。これは、マップ・グループが `VisualAge Generator` 内でプログラムの関連パーツではなかったからです。マイグレーション・ツールはエラー・メッセージを出しません。EGL は実際の `formGroup` パーツを必要とします。`formGroup` パーツが存在しない場合、または `formGroup` パーツがプログラムと同じパッケージに入っていない場合は、プログラムの使用宣言に指定されている `formGroup` を検出できないことを示すエラーが「問題」ビューに出されます。

- 文書化されていなかった `VisualAge Generator` の機能の使用に関しては、同等な EGL サポートが存在しなくても、マイグレーション・ツールがそのことを検出したり、警告メッセージを出したりするとは限りません。例えば、次のようになります。
- `VAGen EZEBYTES` 関数の文書化されたサポートは、項目とレコードのサポートのみでしたが、マップに対して文書化されていないサポートが存在していました。EGL `sysLib.bytes` 関数は、項目とレコードのみをサポートします。マップに対して `EZEBYTES` を使用していた場合、マイグレーション・ツールは警告メッセージを出しません。「問題」ビューにエラーが示されます。
- `CALL` ステートメントが非修飾項目を引数として指定していて、プログラム内にこの項目名の定義が複数ある場合、`VAGen` はプログラムの 1 次作業用ストレージ・レコード内のレベル 77 項目を優先します。EGL は、項目が修飾されていることを必要とします。マイグレーション・ツールによって修飾が追加されることはなく、警告メッセージも出されません。「問題」ビューにエラーが示されます。

---

## EGL 5.1.2 以降の `VAGen` マイグレーション・ツールの新機能

次に示す機能は、EGL 5.1.2 の一般出荷開始日以降に新しく追加されたか、または改良されたものです。ご使用の `WebSphere Developer 5.1.2` 製品の保守レベルによっては、ステージ 1 の一部の機能がすでに強化されている場合があります。

- ステージ 1 は次のように変更されました。
  - マイグレーション・データベース用の DDL が更新されました。  
`SetupTables.bat` を再実行し、ステージ 1 を再実行して、マイグレーション・データベースにソース・コードを配置する必要があります。
  - EGL 予約語リストを外部化し、予約語リストのバージョンをログ情報に組み込みました。
  - エラーがないかどうかステージ 1 の結果を検査する、新しい `checkStage1.bat` ファイルを追加しました。
- ステージ 2 および 3 と単一ファイル・モードの両方のマイグレーションに、次の変更が組み込まれました。
  - 新しい EGL 構文のサポート。
  - オプションのユーザー出口を呼び出して、ステージ 2 マイグレーション時、または単一ファイル・マイグレーション時にパーツの名前を変更できます。例えば、ハイフン (-) を下線 (\_) に変更するユーザー出口を作成できます。



- 新しい VAGen 構文マイグレーション設定により、SQL レコード内のフィールドに対して次のことを指定できます。
  - SQL 列名が項目名と同じである場合に、マイグレーション・ツールが SQL 列名を省略する。
  - *isNullable* = yes プロパティをマイグレーション・ツールが常に省略する。
  - SQL レコードに対してただ 1 つの SQL 表が指定されていて、VAGen の「読み取り専用」プロパティが yes に設定されている場合に限り、マイグレーション・ツールが *isReadOnly* = yes プロパティを組み込む。
- 新しい VAGen 構文マイグレーション設定により、小数点位のコンマを小数点に変更するようにマイグレーション・ツールに指定できます (ご使用のワークステーションが、小数点をデフォルトに設定するロケールを使用している場合であっても)。
- ステージ 2 から 3 へのマイグレーションは、次の処理も行います。
  - *validatorFunction* プロパティまたは *validatorDataTable* プロパティを指定する *dataItem* パーツに対して、import ステートメントを追加します。
  - パーツ型の中で、EGL パーツ名を基準にファイル内のパーツをソートします。

次に示す EGL の変更により、マイグレーションされた VAGen プログラムのサポートが改良されます。

- スtring 連結に数値変数を使用できます。これにより、EGL の prepare ステートメントにマイグレーションされる、VAGen の SQL 入出力 Execution Time Statement Build がサポートされます。
- VisualAge Generator に備わっていた機能と同様に、Java ランタイム環境で EGL 製品メッセージを変更できます。

---

## EGL 6.0 iFix 以降の VAGen マイグレーション・ツールの新機能

次に示す機能は、EGL 6.0 iFix 以降に新しく追加されたか、または改良されたものです。

- ステージ 2 および 3 と単一ファイル・モードの両方のマイグレーションに、次の変更が組み込まれました。
  - ユーザー出口を呼び出して、ステージ 2 マイグレーション時、または単一ファイル・マイグレーション時にパーツの名前を変更できる機能をさらにサポートします。
  - VisualAge Generator 互換モードの使用を最小化することができる新規 VAGen マイグレーション設定。

---

### EGL 6.0.0.1 以降の VAGen マイグレーション・ツールの新機能

次に示す機能は、EGL 6.0.0.1 以降に新しく追加されたか、または改良されたものです。

- Web トランザクション・プログラムおよび UI レコードをマイグレーションするためのサポート。Web トランザクションに関連する生成オプションのマイグレーションもサポートします。

- DL/I レコード、DL/I ファンクション I/O、PSB パーツ、EZEDL\*、特殊機能語、CSPTDLI 特殊機能語をマイグレーションするためのサポート。生成オプション、リソース関連オプション、IMSVS/IMSBMP 環境に関連する関連リンケージ・テーブルのマイグレーションもサポートします (GSAM およびメッセージ・キューのサポートを含みます)。

注: 前バージョンのマイグレーション・ツールを使って、Web トランザクション、IMS、または DL/I に関連する言語要素を使用するパーツをマイグレーションしていた場合は、457 ページの『付録 I. 前のバージョンのマイグレーション・ツールを使用してマイグレーションを行った場合に必要な変更』を参照して、EGL 言語とマッチングさせるために手作業で何を変更する必要があるかを詳しく確認してください。

---

## マイグレーション・ツールに関する既知の制限事項

### 一般

EGL の制限事項、特に検証、デバッグ、または生成に関する制限事項について必ず検討してください。

### Java および Smalltalk のステージ 1

- マイグレーション・データベースを消去せずに、複数のマイグレーション・セット、またはマイグレーション・セットの複数バージョンをマイグレーションした場合は、いずれかのパーツ・エディションを含む最初のマイグレーション・セットのバージョンによって、EGL パーツのプロジェクト、パッケージ、およびファイル名が制御されます。次の条件に該当すれば、このことによって問題が生じることはありません。
  - パーツが Java パッケージ間で移動しておらず、同名の Java パッケージを含む異なる名前のマイグレーション・セットがない。
  - パーツが Smalltalk アプリケーション間で移動しておらず、同名の Smalltalk アプリケーションを含む異なる名前の構成マップがない。

前記に当てはまらない状況の場合、次善策としては、1 つのマイグレーション・セットをステージ 1 から 3 まで完全にマイグレーションしてから、マイグレーション・データベースを消去し、その後で次のマイグレーション・セット・バージョンをステージ 1 から 3 まで完全にマイグレーションしてください。

### ステージ 2 および 3

- VAGen マイグレーション・ウィザードには次の制限事項があります。
  - 進行状況ウィンドウの「取り消し」ボタンは機能しません。タスク・マネージャーを使用しない限り、ステージ 2 または 3 のマイグレーション・ツールを開始後に取り消すことはできません。
- マイグレーション・データベースおよびアプリケーション・データベースを切り換える場合は、切り換えるたびに EGL 開発環境をシャットダウンする必要があります。
- Linux 環境では、ステージ 2 と 3 はサポートされません。

## 構文のマイグレーション

- マイグレーション・ツールは、SQL レコード構造の中で列名として使用されている SQL キーワードを正しく変換します。ただしマイグレーション・ツールは、レコードの SQL defaultSelectCondition 内、または関数の SQL 文節内で列名として使用されている SQL キーワードを処理しません。次善策としては 244 ページの『特殊な処理を必要とする SQL 予約語』で説明されているとおりに、SQL defaultSelect Condition または SQL 文節を変更します。この項では、SQL キーワードのリストと、SQL defaultSelectCondition および SQL 入出力ステートメントに必要な構文の詳細を示しています。
- このプロパティの使用に関する制限について詳しくは、40 ページの『containerContextDependent プロパティ』を参照してください。



---

## 第 3 章 未確定状態の処理

対応する EGL ステートメントを作成するのに必要な VisualAge Generator からの一部の情報をマイグレーション・ツールが持っていない状態が多数あります。こうした状態を未確定状態と呼びます。これは、VisualAge Generator からの入力によって対応する EGL ステートメントが変化したり、異なる結果が得られたりするからです。このような未確定状態では、マイグレーション・ツールは VisualAge Generator で意図していたものと一致する EGL ステートメントにマイグレーションできないことがあります。未確定状態では、次に示すマイグレーション・プロセスのどれを行うかによって、作成される EGL ステートメントが異なる場合がよくあります。

- プログラムおよびその関連付けをマイグレーションするかどうか。そうである場合は、プログラムがマイグレーションされる順序。
- パーツ間マイグレーションを依然として実行できるように、プログラムをマイグレーションしないが必要な関連パーツをマイグレーションするかどうか。
- マイグレーション・ツールがインテリジェント選択を行うことができる情報に限定されるように、関連付けをマイグレーションしないでマイグレーションするかどうか。

プログラムおよびその関連パーツをマイグレーションすることは、最大の情報が保証されるので、関連付けをマイグレーションする推奨方法です。この後に示す表では、次のパーツ型について、関連パーツを使用したマイグレーションと使用しないマイグレーションの違いを説明します。

- 共用データ項目
- レコード
- テーブル
- マップ・グループおよびマップ
- プログラム
- I/O ステートメントを含む関数
- その他のステートメント
- EZE ワード

これらの表では、こうした未確定状態に関連して起こりうる問題と、これらの問題に対して考えられる解決策も示しています。それぞれの状態に最適な解決策は 1 つではありません。例えば、同じパーツ名のパーツが 2 つある場合には、使用頻度が低い方の名前を変更すれば、コードの他の部分に及ぶ影響が最小限になります。

---

### データ項目の未確定状態の処理

#### 偶数の長さの PACK データ項目

**VisualAge Generator:** PACK データ項目の長さは、最大 18 までの桁数として指定されます。共用データ項目定義の中、および非共用データ項目のレコード定義の中では、偶数の長さが記録されます。一方、ほとんどのエディター、およびテストと

生成に使用される長さは、次に大きな奇数の長さ (最大の場合は 18) です。SQL レコード・エディターのみが偶数の長さを表示します。SQL WHERE 文節、または Execution Time Statement Build を指定している SQL ステートメントの中で、偶数の長さの項目がホスト変数として使用されている場合、テストおよび生成機能は偶数の長さの一時変数を作成します。

**EGL:** decimal プリミティブ型が、VAGen の PACK 型の置換表現です。VisualAge Generator 互換モードでは、EGL のテストおよび生成機能は VisualAge Generator と同じサポートを提供します。偶数の精度をもつ 10 進項目の場合、テストおよび生成機能はすべてのレコードの精度を 1 ずつ増やし、SQL where 文節または prepare ステートメント内では偶数の精度の一時変数を使用します。 VisualAge Generator 互換モードが指定されていない場合、EGL はデータ項目に対して指定された精度を使用します。

**マイグレーションに必要な関連パーツ:** 適用外。

表 9. 偶数の長さの PACK データ項目

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>マイグレーション・ツールは、VAGen マイグレーション設定の「項目または変数の <code>evensql=y</code> を受け入れない (<i>Do not honor evensql=y for items or variables</i>)」に基づいてパック・データ項目をマイグレーションします。</p> <p>この設定が選択されていない場合、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"><li>項目が SQL 行レコードで使用されていたかどうかに関係なく、共用データ項目定義について VisualAge Generator で指定された偶数または奇数長を使用します。</li><li>すべてのレコード定義の非共用項目については、VisualAge Generator に指定された偶数または奇数長を使用します。これは、項目が SQL where 文節または prepare ステートメントでホスト変数として使用されることがあるからです。</li><li>テーブル、関数仮パラメーター、関数戻り値、および関数ローカル・ストレージの非共用項目については、偶数桁の数値を判別する情報がこの場合には VisualAge Generator に記録されていないので、奇数長 (項目が最大長の場合は 18) を使用します。</li></ul> <p>この設定が選択されている場合、マイグレーション・ツールは、変数のすべての項目について常に奇数長 (項目が最大長である場合は 18) を使用します。マイグレーション・ツールは <code>evensql=y</code> を指定しているあらゆるデータ項目に関して警告メッセージを発行します。</p>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄での説明と同じ処理を行いません。</p>

表 9. 偶数の長さの PACK データ項目 (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題 1:</b> 奇数長に <code>evensql = y</code> をマイグレーションする設定を選択すると、問題が発生します。この場合、SQL 列定義と一致しないホスト変数長を使うため、ランタイム・パフォーマンスに影響が出る可能性があります。</p> <p><b>解決策 1:</b> マイグレーション・ログ情報を参照し、<code>evensql = y</code> が指定されている <code>dataItem</code> パーツがないかどうか調べます。SQL テーブルおよびビューの定義を検証し、それらの定義が <code>dataItem</code> パーツの定義と一致しているかどうかを判断します。また、<code>dataItem</code> を型定義として指定する変数を使用している SQL の <code>where</code> 文節と <code>prepare</code> ステートメントを確認します。</p> <p><b>起こりうる問題 2:</b> マイグレーション中に <code>evensql=y</code> を受け入れる設定を選択せず、後で VisualAge 互換モードを使用しないと判断した場合にも、問題が発生します。この場合には、VisualAge Generator 互換モードよりも有効数字が少ないのでオーバーフローが発生することがあります。</p> <p><b>解決策 2:</b> EGL レコード内のすべての 10 進 <code>dataItem</code> パーツ定義とプリミティブ・フィールドを検討し、偶数長の項目がないかどうか調べます。これらのいずれかの項目にオーバーフローが発生するかどうか評価します。</p>	<p><b>起こりうる問題:</b> 『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があり、同じ解決策を適用できます。</p>

## 共用編集とメッセージ

**VisualAge Generator:** 共用 `dataItem` パーツ定義には、マップと UI レコードの両方のデフォルト編集とメッセージを指定できます。

**EGL:** `dataItem` パーツ定義の編集とメッセージのプロパティは、1 セットだけ存在します。マイグレーション・ツールは、データ項目のマップと UI プロパティをマージします。

**マイグレーションに必要な関連パーツ:** 適用外。

表 10. 共用編集とメッセージ

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>マイグレーション・ツールは、マップと UI 編集を次のようにマージします。</p> <ul style="list-style-type: none"> <li>各 編集またはメッセージについて、マイグレーション・ツールは次の内の最初に適用される処理を行います。 <ul style="list-style-type: none"> <li>UI 編集が指定されていると、マイグレーション・ツールは UI 編集とその関連メッセージ情報を対応する EGL プロパティに変換します。</li> <li>マップ編集が指定されている場合のみ、マイグレーション・ツールはマップ編集とその関連メッセージを対応する EGL プロパティに変換します。</li> <li>関連 UI 編集なしで UI メッセージが指定されていると、マイグレーション・ツールは UI メッセージを対応する EGL プロパティに変換します。</li> <li>関連マップ編集なしでマップ・メッセージが指定されていると、マイグレーション・ツールはマップ・メッセージを対応する EGL プロパティに変換します。</li> <li>UI およびマップ編集とメッセージ情報が指定されていない場合、マイグレーション・ツールは対応する EGL プロパティを設定しません。通常の EGL のデフォルトが適用されます。</li> </ul> </li> <li>VisualAge Generator では、マップ編集に行末そろえおよび 16 進数編集のみが指定されるので、対応する EGL プロパティの設定に常に使用されます。</li> </ul>	<p>63 ページの『共用データ項目のマップ編集ルーチン』で後述する場合を除き、マイグレーション・ツールはデフォルトの編集とメッセージを、関連パーツがあってもなくても同じようにマイグレーションします。</p>
<p><b>起こりうる問題 1:</b> 問題は、VisualAge Generator に矛盾するマップ編集と UI 編集が存在し、マップ・レコードと UI レコード間で編集が異なることを本当に意図している場合にのみ発生します。テキスト書式または印刷書式に項目が追加されるまで問題は発生しません。</p> <p><b>注:</b> VAGen Web トランザクションを使用していない場合は、マップ編集のみが VisualAge Generator に存在し、問題は発生しません。</p> <p><b>考えられる解決策:</b> VAGen マップ項目の編集とメッセージをリストするコメントを dataItem パーツ定義に追加するほかに、dataItem パーツ定義に対して実行できることはありません。項目をテキスト書式または印刷書式に追加すると、その特定の書式に対しては異なった指定が必要なプロパティを指定変更できます。</p> <p><b>起こりうる問題 2:</b> EGL VGUI レコードで項目を使用する場合にも問題が発生することがあります。その項目には、VAGen マップ編集からマイグレーションされた追加の編集やメッセージが存在する場合があります。</p> <p><b>解決策:</b> 編集を必ず検討し、VGUI レコード内の型定義を使用して定義されたフィールドがないかどうか調べてください。</p>	<p>『関連パーツを使用したマイグレーション』欄で説明した問題と同じことが起こる可能性があります。同じ解決策を使用できます。</p>

## 共用データ項目のマップ編集ルーチン

**VisualAge Generator:** 共用データ項目定義で、テーブル、関数、EZEC10 または EZEC11 であるマップ編集ルーチンを指定できます。編集メッセージは、編集ルーチンが EZEC10、EZEC11、またはテーブルである場合のみ使用されます。

**EGL:** データ項目には、`validatorDataTable` と `validatorFunction` の両方を指定できます。さらにデータ項目には、`validatorDataTableMsgKey` と `validatorFunctionMsgKey` の両方を指定できます。

**マイグレーションに必要な関連パーツ:** テーブルまたは演算部のいずれか。

表 11. 共用データ項目のマップ編集ルーチン

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>共用データ項目を初めてマイグレーションする場合、マイグレーション・ツールは次の内の最初に適用される処理を行います。</p> <ul style="list-style-type: none"><li><code>editRoutineName</code> が EZEC10 または EZEC11 ならば、マイグレーション・ツールは <code>validatorFunction</code> プロパティを同等な EGL システム・ライブラリー関数に設定します。さらにマイグレーション・ツールは、<code>validatorFunctionMsgKey</code> を編集メッセージ (ある場合) に設定します。</li><li><code>editRoutineName</code> が関数ならば、マイグレーション・ツールは <code>validatorFunction</code> プロパティを設定します。<code>validatorFunctionMsgKey</code> は VisualAge Generator 内では使用されないで、マイグレーション・ツールはこれを省略します。またマイグレーション・ツールは、<code>import</code> ステートメントがステージ 3 で作成されるように、データ項目パーツの関連としてこの関数を追加します。</li><li><code>editRoutineName</code> がテーブルならば、マイグレーション・ツールは <code>validatorDataTable</code> プロパティを設定します。さらにマイグレーション・ツールは、<code>validatorDataTableMsgKey</code> を編集メッセージ (ある場合) に設定します。またマイグレーション・ツールは、<code>import</code> ステートメントがステージ 3 で作成されるように、データ項目パーツの関連としてこのテーブルを追加します。</li><li>編集ルーチンが指定されておらず、編集メッセージが指定されている場合、マイグレーション・ツールは <code>validatorDataTableMsgKey</code> を編集メッセージに設定します。</li></ul> <p>注: プログラムのコンテキストでマイグレーションする場合でも、共用項目がマップで使用されていない、またはマップでの編集が共用項目定義で指定されたものと異なる場合には、<code>editRoutineName</code> が使用できないことがあります。</p>	<p><code>editRoutineName</code> と同じ名前の関数またはテーブルが使用できない場合、マイグレーション・ツールは次の内の最初に適用される処理を行います。</p> <ul style="list-style-type: none"><li><code>editRoutineName</code> が EZEC10 または EZEC11 ならば、マイグレーション・ツールは <code>validatorFunction</code> プロパティを同等な EGL システム・ライブラリー関数名に設定します。さらにマイグレーション・ツールは、<code>validatorFunctionMsgKey</code> を編集メッセージ (ある場合) に設定します。</li><li><code>editRoutineName</code> が 7 文字より長い場合、これは必ず関数名なので、マイグレーション・ツールは <code>validatorFunction</code> プロパティを設定します。<code>validatorFunctionMsgKey</code> は VisualAge Generator 内では使用されないで、マイグレーション・ツールはこれを省略します。</li><li>編集メッセージが指定されている場合、マイグレーション・ツールは <code>validatorDataTable</code> と <code>validatorDataTableMsgKey</code> を設定します。</li><li>そうでなければ、マイグレーション・ツールは <code>validatorFunction</code> プロパティを設定し、エラー・メッセージを出します。</li></ul> <p>編集ルーチンが指定されておらず、編集メッセージが指定されている場合、マイグレーション・ツールは <code>validatorDataTableMsgKey</code> を編集メッセージに設定します。</p>

表 11. 共用データ項目のマップ編集ルーチン (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題:</b> 関数と dataTable の名前が同じである (通常、異なるサブシステム内で) 場合に限り、問題が起こります。この場合、あるサブシステムは関数を使用し、別のサブシステムは dataTable を使用します。テキスト書式に項目が追加されるまで問題は発生しません。</p> <p><b>考えられる解決策:</b> 同名のパーツが 2 つ存在しないように、dataTable の名前を変更します。項目定義に、validatorFunction と validatorDataTable の両方のプロパティを指定します。項目をテキスト書式に追加する場合は、その特定の書式に必要な、validatorFunction または validatorDataTable のどちらかのプロパティを削除します。<b>欠点:</b> 新しい dataTable 名を使用するようにプログラムと書式を変更する必要があります。</p>	<p><b>起こりうる問題 1:</b> マイグレーション・ツールが予測を誤ると、問題が発生します。</p> <p><b>考えられる解決策:</b> データ項目定義を訂正します。</p> <p><b>起こりうる問題 2:</b> 関連パーツを使用したマイグレーション欄にリストされたのと同じ問題も発生することがあります。同じ解決策を使用できます。</p>

## 共用データ項目の充てん文字

**VisualAge Generator:** マップ編集のデフォルトの充てん文字は、文字、混合、または DBCS の場合は NULL で、数値の場合はブランク、16 進数の場合は 0 です。UI 編集のデフォルトの充てん文字は、文字、混合、DBCS、ユニコード、および数値の場合はブランクで、16 進数の場合は 0 です。Null は UI レコードでは無効な充てん文字です。マップ編集と UI 編集に別の充てん文字を指定できます。

**EGL:** dataItem パーツのデフォルト fillCharacter プロパティは 1 つだけ存在します。マイグレーション・ツールは、データ項目のマップと UI fillCharacter プロパティをマージします。

**マイグレーションに必要な関連パーツ:** 適用外。



表 12. 共用データ項目の充てん文字

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>共用データ項目を初めてマイグレーションする場合、マイグレーション・ツールは次の内の最初に適用される処理を行います。</p> <ul style="list-style-type: none"> <li>UI 編集で充てん文字を指定すると、マイグレーション・ツールは文字を EGL fillCharacter プロパティにマイグレーションします。NULL 充てんは VisualAge Generator 内の UI レコードでは無効であるため、マイグレーション・ツールは N to N 変換を行いません。</li> <li>マップ編集で充てん文字を指定すると、マイグレーション・ツールは文字を EGL fillCharacter プロパティにマイグレーションします。マイグレーション・ツールは N を NULL 充てんに変換します。</li> <li>UI 編集でもマップ編集でも充てん文字を指定しないと、マイグレーション・ツールは EGL fillCharacter プロパティを設定しません。</li> </ul>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄での説明と同じ処理を行います。</p>
<p><b>起こりうる問題:</b> あるタイプの編集を使用して別のタイプのユーザー・インターフェース（書式または VGUI レコード）にマイグレーションされた dataItem パーツに、型定義を使用してフィールドを追加した場合に限り、問題が起こります。</p> <p><b>考えられる解決策:</b> 書式または VGUI レコードに新規フィールドを追加する際には、必ず充てん文字を見直してください。</p>	<p>『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。</p>

## レコードの未確定状態の処理

### 再定義レコード

**VisualAge Generator:** 再定義レコード・タイプは、別のレコードに異なるデータ項目レイアウトを提供します。再定義レコードは、再定義するレコードの名前を指定します。例えば、RecordA は RecordB を再定義する再定義レコードであるとしします。VisualAge Generator は、プログラム内での RecordA の使用法に基づいて、RecordA が実際に RecordB の再定義であるかどうかを判別します。RecordA が呼び出し先パラメーターとして使用されている場合、RecordA は RecordB の再定義として扱われません。

**EGL:** RecordA は基本レコードです。再定義情報はプログラム定義内でのみ提供され、RecordA の定義内では提供されません。

**マイグレーションに必要な関連パーツ:**

- 再定義レコードをマイグレーションする場合: 適用外。
- プログラムをマイグレーションする場合: 再定義レコード (RecordA)。

表 13. 再定義レコード

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>再定義レコードをマイグレーションする際に、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>再定義レコード (RecordA) を基本レコードにマイグレーションします。</li> <li>RecordA が RecordB を再定義したことを示す VAGen 情報コメントを RecordA に組み込みます。</li> <li>RecordA が RecordB を再定義したことを示す情報メッセージを出します。</li> </ul>	<p>再定義レコードをマイグレーションする際に、マイグレーション・ツールは『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p>プログラムをマイグレーションする際に、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>RecordA が VisualAge Generator 内で RecordB の再定義として扱われている場合、マイグレーション・ツールは RecordA の宣言に redefines プロパティを組み込みます。</li> <li>RecordA が VisualAge Generator 内で RecordB の再定義として扱われていない場合、マイグレーション・ツールは RecordA の宣言に redefines プロパティを組み込みません。</li> </ul>	<p>プログラムをマイグレーションする際に、RecordA が使用不可ならば、マイグレーション・ツールは RecordA が再定義レコードであることを認識できません。マイグレーション・ツールは、RecordA の宣言に redefines プロパティを組み込みません。</p>
<p><b>新規使用の場合の考慮事項:</b> RecordA と RecordB を新規プログラム内で使用する必要がある場合に限り、問題が起こります。RecordA が RecordB の再定義として扱われるようにしたい場合は、必ず RecordA の redefines プロパティを組み込んでください。</p>	<p><b>起こりうる問題:</b> VAGen プログラムが RecordA を再定義として使用する場合には、問題が生じます。マイグレーションの直後は、RecordA の定義と import ステートメントが欠落しているので、プログラムは有効な EGL プログラムになりません。「問題」ビューにエラーが示されます。RecordA をマイグレーションしてプログラムに import ステートメントを追加すると、プログラムが有効な EGL プログラムに変換されます。ただし、データ域は 2 つ (RecordA 用に 1 つ、RecordB 用に 1 つ) 存在します。EGL は、検証または生成の際にはこの変更を検出しません。<b>プログラムは、VisualAge Generator の場合とは異なる動作をします。</b></p> <p><b>解決策:</b> 追加のレコードをマイグレーションする場合、またはプログラムに import ステートメントを追加する場合は、レコード定義にある VAGen 情報コメントを検討してください。RecordA が RecordB の再定義であることを示す VAGen 情報コメントがある場合は、RecordA の宣言に redefines プロパティを組み込んでプログラムを更新します。</p> <p><b>考慮事項:</b> 『関連パーツを使用したマイグレーション』欄にリストしたものと同じ、新規使用の場合の考慮事項が適用されます。</p>

## レコード内のレベル 77 項目

**VisualAge Generator:** 作業用ストレージ・レコードにはレベル 77 項目が存在する可能性があります。

**EGL:** レコードにレベル 77 項目は存在しません。

**マイグレーションに必要な関連パーツ:**

- 作業用ストレージ・レコードをマイグレーションする場合: 適用外。
- プログラムをマイグレーションする場合: 1 次作業用ストレージ・レコード。
- 関数をマイグレーションする場合: 作業用ストレージ・レコード。

表 14. レコード内のレベル 77 項目

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>レベル 77 項目を含む作業用ストレージ・レコードをマイグレーションする際に、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• レベル 77 項目を含む作業用ストレージ・レコードを 2 つの基本レコードに分割します (作業用ストレージ構造のために 1 つ、レベル 77 項目のために 1 つ)。新しいレベル 77 レコード名は、オリジナルの作業用ストレージ・レコード名にユーザー提供の接尾部を付けたものです。</li> <li>• オリジナルの作業用ストレージ・レコードと同じファイルに、新しいレベル 77 レコードを配置します。</li> <li>• レベル 77 レコードが作成されていることを示す情報メッセージを出します。</li> </ul>	<p>レベル 77 項目を含む作業用ストレージ・レコードをマイグレーションする際に、マイグレーション・ツールは『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p>プログラムをマイグレーションする際に、1 次作業用ストレージ・レコードが使用可能で、レベル 77 項目を含んでいる場合、マイグレーション・ツールは新しいレベル 77 レコードのレコード宣言をプログラムに追加します。</p>	<p>プログラムをマイグレーションする際に、1 次作業用ストレージ・レコードが使用不可ならば、マイグレーション・ツールは 1 次作業用ストレージ・レコードがレベル 77 項目を含んでいるかどうか判別できません。マイグレーション・ツールは、レベル 77 レコードのレコード宣言を組み込みません。</p>
<p>関数をマイグレーションする際に、作業用ストレージ・レコードが使用可能ならば、マイグレーション・ツールはその関数内のレベル 77 項目への修飾参照を変更して、新しいレベル 77 レコード名を使用するようにします。</p>	<p>関数をマイグレーションする際に、作業用ストレージ・レコードが使用不可ならば、マイグレーション・ツールは項目名の修飾を変更しません。</p>

表 14. レコード内のレベル 77 項目 (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題:</b> 同名のレコードが 2 つ存在し (異なるサブシステム内にあると考えられる)、一方のレコード内では項目がレベル 77 項目であり、他方ではそうでない場合に限り、レベル 77 項目に関する問題が起こります。</p> <p><b>考えられる解決策:</b> その項目を (新規) 共通レコードに移動し、すべての関数内でその項目の修飾を変更します。あるいは、関数内でその項目を修飾しません。</p> <p><b>新規使用の場合の考慮事項:</b> 新規プログラムの inputRecord プロパティとしてオリジナルの作業用ストレージ・レコードを指定した場合、問題が起こる可能性があります。新しいレベル 77 レコードの宣言をさらに追加する必要があるかどうか、必ず検討してください。</p>	<p><b>起こりうる問題 1:</b> 1 次作業用ストレージ・レコードがレベル 77 を含んでいて、プログラムがレベル 77 を使用していた場合には、問題が起こります。プログラムの検証は、項目定義の欠落のために失敗します。</p> <p><b>解決策:</b> プログラムにレベル 77 レコードを追加します。</p> <p><b>起こりうる問題 2:</b> 修飾データ項目が実際には VAGen レベル 77 項目である場合、関数に問題が起こります。</p> <p><b>解決策:</b> データ項目の正しい修飾子を指定するように関数を変更します。</p> <p><b>起こりうる問題 3:</b> 『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。</p> <p><b>考慮事項:</b> 『関連パーツを使用したマイグレーション』欄にリストしたものと同じ、新規使用の場合の考慮事項が適用されます。</p>

## 代替仕様レコード

**VisualAge Generator:** レコードは、別のレコードを代替仕様 (altspec) レコードとして指定できます。例えば、RecordA が altspec として RecordB を指定している場合、RecordB は RecordA の構造を提供します。SQL レコードの場合、RecordB は RecordA の SQL 表とキーのリストも提供します。RecordA がキー項目を指定している場合は、デフォルト選択条件を判別する際にその項目が RecordB のキーとマージされます。DL/I セグメント・レコードの場合、RecordB 内のフィールド名は DL/I PSB 内のフィールド名でもあります。

**EGL:** レコード構造を取得するために、レコードに別のレコードを組み込むことができます。組み込まれるものはレコード構造のみです。SQL レコードはそれぞれ、SQL 表とキー全体のセットを明示的に指定する必要があります。DL/I セグメント・レコードでは、RecordB が DL/I セグメントでもある場合、RecordB は DL/I PSB 内のフィールド名を指定する dliFieldName プロパティを提供します。また、RecordB が DL/I セグメントでない場合、RecordA は embed ステートメントをオーバーライドするものとして dliFieldName プロパティを提供することができます。

**マイグレーションに必要な関連パーツ:** RecordA が SQL レコードまたは DL/I セグメントである場合、altspec レコードとして指定されたレコード (RecordB) が必要です。

表 15. 代替仕様レコード

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>RecordA が代替仕様 RecordB を指定する SQL レコードであり、RecordB が使用可能ならば、マイグレーション・ツールは RecordA に対して次の処理を行います。</p> <ul style="list-style-type: none"> <li>RecordB 内で指定されている表のリストから、表名のリストを作成します。</li> <li>次のものをマージして、キーのリストを作成します。 <ul style="list-style-type: none"> <li>key=yes を指定した RecordB 内の項目 (存在する場合)。</li> <li>キー項目として指定された RecordA 内の項目 (存在する場合)。</li> </ul> </li> </ul> <p>キーの順序は、項目が RecordB の構造に出現する順序です。</p> <ul style="list-style-type: none"> <li>RecordB を指定する embed ステートメントを組み込みます。</li> <li>RecordA のデフォルト選択条件にある !itemColumnName 変数を、RecordB にある対応する SQL 列名にマイグレーションします。</li> </ul>	<p>RecordA が代替仕様 RecordB を指定する SQL レコードであり、RecordB が使用不可ならば、マイグレーション・ツールは RecordA に対して次の処理を行います。</p> <ul style="list-style-type: none"> <li>tableNames プロパティを <code>###TABLES_NOT_FOUND###</code> に設定します。</li> <li>keyItems プロパティを、<code>###KEYS_NOT_FOUND###</code> の後に RecordA 内でキー項目として指定されている項目 (存在する場合) を付けた内容に設定します。</li> <li>RecordB を指定する embed ステートメントを組み込みます。</li> <li>RecordA のデフォルト選択条件にある !itemColumnName 変数を、置換せずに !itemColumnName にマイグレーションします。</li> <li>表とキーを判別できないことを示すエラー・メッセージを出します。</li> <li>!itemColumnName 変数が存在する場合は、エラー・メッセージを出します。</li> </ul>
<p>RecordA が RecordB の代替仕様を指定する DL/I セグメント・レコードであり、RecordB が使用可能であって DL/I セグメント・レコードでない場合、マイグレーション・ツールは RecordA に関して次の処理を行います。</p> <ul style="list-style-type: none"> <li>RecordB を指定する embed ステートメントを組み込みます。</li> <li>RecordB 内の名前を変更しなければならない各フィールドごとに、オーバーライド・ステートメントを組み込みます。オーバーライド・ステートメントは、dliFieldName プロパティをオリジナルの VAGen フィールド名に設定して、DL/I フィールド名を EGL 内で使用可能にします。</li> </ul>	<p>RecordA が RecordB の代替仕様を指定する DL/I セグメント・レコードであり、RecordB が使用可能でない場合、マイグレーション・ツールは RecordA に関して次の処理を行います。</p> <ul style="list-style-type: none"> <li>RecordB を指定する embed ステートメントを組み込みます。</li> <li>dliFieldName 情報を保存するためにオーバーライド・ステートメントが必要かどうかをマイグレーション・ツールが決定できない、というエラー・メッセージを発行します。</li> </ul>

表 15. 代替仕様レコード (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題 1:</b> SQL については、RecordB の定義が異なる場合に限り (通常は、別々のサブシステム内で) 問題が生じます。</p> <p><b>解決策 1:</b> それぞれのサブシステムで RecordA を個別に定義されるように、RecordA の定義を複製します。あるいは、RecordA に対して <code>containerContextDependent=yes</code> を指定します。</p> <p><b>起こりうる問題 2:</b> DL/I についても、RecordB の定義が異なる場合に限り (通常は、別々のサブシステム内で) 問題が生じます。</p> <p><b>解決策 2:</b> 解決策は解決策 1 と同じです。</p>	<p><b>問題 1:</b> SQL レコード RecordA の EGL 検証の結果、「問題」ビューにメッセージが表示されます。</p> <p><b>解決策 1:</b> RecordA を編集し、RecordB に基づいて適切な表とキーの情報を組み込みます。また、デフォルト選択条件にある <code>!itemColumnName</code> 変数すべてを、RecordB からの対応する SQL 列名に置き換えます。</p> <p><b>起こりうる問題 2:</b> 予約語であるため名前を変更しなければならないフィールドが RecordB にある場合、またはそれらのフィールドが # 記号または @ 記号で始まっている場合、DL/I セグメント RecordA に関する問題が発生します。この場合、このフィールドが デフォルトの SSA で使用されていて、かつ変更後のフィールド名が 8 文字を超えていたり DL/I では無効な下線を含んでいたたりする場合、「問題」リストにメッセージが表示されます。</p> <p><b>解決策 2:</b> RecordA を編集し、DL/I フィールド名を指定するためのオーバーライド・ステートメントを組み込みます。</p> <p><b>起こりうる問題 3:</b> 名前変更されたフィールドがデフォルトの SSA 内で使用され、かつ有効な DL/I 名である場合も、DL/I セグメント RecordA について問題が発生します。この場合、プログラム実行時にランタイム・エラーが発生します。</p> <p><b>解決策 3:</b> RecordA を編集し、DL/I フィールド名を指定するオーバーライド・ステートメントを組み込みます。</p>

## 同じレコード名をもつ異なる定義

**VisualAge Generator:** レコードには、現在ワークスペース内にあるプロジェクトとパッケージに基づいた共用データ項目が含まれています。これにより、さまざまなサブシステムやプログラムが、同じレコード名に対して異なる定義を使用できます。

**EGL:** レコード定義用の `containerContextDependent=yes` プロパティを提供しています。このプロパティにより、プログラムのパーツのネーム・スペースに基づいた `dataItem` パーツを型定義に使用するよう指定できます。

**マイグレーションに必要な関連パーツ:** 適用外。このレコード・プロパティを設定するには、サブシステムすべての VAGen パーツ構造を完全に理解する必要があります。



表 16. 同じレコード名をもつ異なる定義

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
マイグレーション・ツールは、レコード定義に対して <code>containerContextDependent=yes</code> プロパティを設定しません。この機能が必要な場合は、この機能を必要とするそれぞれのレコードごとにこのプロパティを設定する必要があります。	マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。

## 予約語と UI レコード名

**VisualAge Generator:** VisualAge Generator に予約語はありません。VisualAge Generator では、UI レコード名または UI レコード内の項目の先頭文字として # 記号または @ 記号が使えます。

**EGL:** EGL には予約語があります。さらに EGL の場合は、レコード名の先頭文字として # 記号または @ 記号を使用することはできません。UI レコード名は予約語にすることはできず、また名前の先頭文字を # 記号や @ 記号にすることはできません。UI レコード内のフィールド名は予約語にすることはできず、また名前の先頭文字を # 記号や @ 記号にすることはできません。

**マイグレーションに必要な関連パーツ:** 適用外。

表 17. 予約語と UI レコード名

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>UI レコードをマイグレーションするときに、その UI レコード名が予約語であったり、先頭が # 記号または @ 記号であったりする場合、マイグレーション・ツールは以下の処理を行ないます。</p> <ul style="list-style-type: none"> <li>レコード名の冒頭に名前変更接頭部を組み込んで UI レコード名を変更します。これは、他の VAGen レコードに関してマイグレーション・ツールが行う名前変更処理と同じです。名前変更接頭部は、VAGen マイグレーション構文を設定することによって指定できます。</li> <li>alias プロパティを組み込み、UI レコードのオリジナルな VAGen 名に設定します。</li> <li>名前変更済みの UI レコードとマイグレーションのステージ 3 で一致するように、UI レコードの .egl ファイル名を変更します。</li> <li>警告メッセージを出します。</li> </ul>	マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。

表 17. 予約語と UI レコード名 (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>UI レコード内の項目をマイグレーションするとき、その項目名が予約語である場合または先頭が # 記号または @ 記号である場合、マイグレーション・ツールは以下の処理を行ないます。</p> <ul style="list-style-type: none"> <li>項目名の冒頭に名前変更接頭部を組み込んで項目を名前変更します。これは、他の VAGen レコード内のフィールドに関してマイグレーション・ツールが行う名前変更処理と同じです。</li> <li>フィールドの alias プロパティを組み込み、フィールドのオリジナルな VAGen 名に設定します。</li> </ul>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p>マイグレーション・ツールは、名前を変更するためには、他のレコードと同様に UI レコードを扱います。マイグレーション・ツールは、UI レコードに対する以下のようなすべての参照のために、EGL パーツ名を使用します。</p> <ul style="list-style-type: none"> <li>プログラム・パーツ: <ul style="list-style-type: none"> <li>先頭 UI レコード</li> <li>レコード宣言</li> </ul> </li> <li>UI レコード・パーツ - Link パラメーター内の先頭 UI レコード</li> <li>演算部 - ステートメントにおけるレコードの使用</li> </ul>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>起こりうる問題:</b> なし。 alias プロパティは VAGen と同じ JSP 名を設定します。</p>	<p><b>起こりうる問題:</b> なし。 alias プロパティは VAGen と同じ JSP 名を設定します。</p>

## テーブルの未確定状態の処理

### 予約語とテーブル名

**VisualAge Generator:** VisualAge Generator に予約語はありません。VAGen テーブル名では # 記号または @ 記号は無効です。

**EGL:** EGL には予約語があります。さらに EGL の場合は、パーツ名の先頭文字として # 記号または @ 記号を使用することはできません。dataTable 名が予約語であってはなりません。

**マイグレーションに必要な関連パーツ:** 適用外。

表 18. 予約語とテーブル名

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>マイグレーション・ツールは、テーブルの名前を自動的に変更しません。テーブル名が予約語リストと一致する場合、マイグレーションのステージ 1 で使用されるマイグレーション・ツールがエラー・メッセージを出します。テーブル名を変更しない場合は、マイグレーションのステージ 2 で使用されるマイグレーション・ツールもエラー・メッセージを出します。</p>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>起こりうる問題:</b> dataTable 名が予約語と一致している場合に限り、問題が起こります。EGL による検証の結果、「問題」ビューにメッセージが表示されます。</p> <p><b>解決策:</b> マイグレーションの前に VisualAge Generator のテーブルの名前を変更するか、マイグレーションの後に EGL の dataTable の名前を変更します。</p> <p>VisualAge Generator 内で名前を変更する場合は、プログラム、マップ、関数、UI レコード、およびデータ項目定義の中で、そのテーブルへのすべての参照を必ず変更してください。EGL で名前を変更する場合は、dataTable の名前とその参照すべてを変更する必要があります。この対象には、次の場所での参照が含まれます。</p> <ul style="list-style-type: none"> <li>プログラムの使用宣言ステートメント</li> <li>プログラムおよび関数のロジック・ステートメント</li> <li>データ項目の validatorDataTable プロパティ</li> <li>書式フィールドの validatorDataTable プロパティ</li> <li>VGUI レコードの validatorDataTable プロパティ</li> </ul> <p>生成される dataTable の名前としてオリジナルのテーブル名を保持する必要がある場合は、オリジナルの dataTable 名に <i>alias</i> プロパティを設定します。<i>alias</i> プロパティを指定しない場合は、CICS プログラム定義など、EGL 以外による dataTable 名の参照すべてを必ず変更してください。</p>	<p>『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。</p>

## マップ・グループとマップの未確定状態の処理

### 予約語と formGroup 名

- **VisualAge Generator:** VisualAge Generator に予約語はありません。VAGen マップ・グループ名に # 記号または @ 記号は使えません。
- **EGL:** EGL には予約語があります。さらに EGL の場合は、パーツ名の先頭文字として # 記号または @ 記号を使用することはできません。formGroup 名が予約語であってはなりません。
- **マイグレーションに必要な関連パーツ:** 適用外。

表 19. 予約語と formGroup 名

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>マイグレーション・ツールは formGroup の名前を自動的に変更しません。マップ・グループ名が予約語リストと一致する場合、マイグレーションのステージ 1 で使用されるマイグレーション・ツールがエラー・メッセージを出します。マップ・グループ名を変更しない場合は、マイグレーションのステージ 2 で使用されるマイグレーション・ツールもエラー・メッセージを出します。</p>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>起こりうる問題:</b> formGroup 名が予約語と一致している場合に限り、問題が起きます。EGL による検証の結果、「問題」ビューにメッセージが表示されます。</p> <p><b>解決策:</b> マイグレーションの前に VisualAge Generator のマップ・グループの名前を変更するか、マイグレーションの後に EGL の formGroup の名前を変更します。VisualAge Generator 内でマップ・グループの名前を変更する場合は、そのマップ・グループに属するマップすべての名前を必ず変更してください。また、すべてのプログラム定義にあるマップ・グループの参照をすべて変更してください。EGL で formGroup の名前を変更する場合は、プログラム使用宣言ステートメント内での参照など、formGroup の名前とその参照すべてを変更する必要があります。生成される formGroup の名前としてオリジナルのマップ・グループ名を保持する必要がある場合は、オリジナルのマップ・グループ (formGroup) 名に <i>alias</i> プロパティを設定します。<i>alias</i> プロパティを指定しない場合は、CICS プログラム定義など、EGL 以外による formGroup 名の参照すべてを必ず変更してください。</p>	<p>『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。</p>

## マップ・グループと formGroup の要件

**VisualAge Generator:** マップ・グループは、浮動域仕様が存在する場合のみ必要です。

**EGL:** formGroup は、フォームを格納するために常に必要です。

**マイグレーションに必要な関連パーツ:** マップ・グループと、マップ・グループ内のすべてのマップ。

表 20. マップ・グループと formGroup の要件

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>マップ・グループが存在しない場合、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• 同じマップ・グループ名をもつすべてのマップに対して、formGroup を作成します。</li> <li>• 同じ formGroup 内の書式を、すべて同じ EGL ファイルに書き込みます。</li> <li>• 単一ファイル・マイグレーションを使用したマイグレーションを行っていないければ、formGroup 定義の中で書式をネストします。</li> <li>• 単一ファイル・モードでマイグレーションを行っている場合は、書式をネストするために formGroup の編集が必要であることを示すエラー・メッセージが出されます。</li> </ul>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。ただし、マップ・グループとそのグループ内のマップすべてが同じマイグレーション・セットに入っていないければ、次に説明するような問題が生じる可能性があります。</p>
<p><b>起こりうる問題:</b> なし。マップ・グループのマップはすべて、同じマイグレーション・セットに含まれている必要があります。マイグレーション・セットは生成される内容の表現なので、マイグレーション・セットにはマップ・グループ内のマップをすべて組み込む必要があります。</p> <p>単一ファイル・モードでマイグレーションを行っている場合は、マップ・グループ内のマップすべてを必ず同じ外部ソース形式ファイルに組み込んでください。</p>	<p><b>起こりうる問題:</b> 同じマップ・グループ名のマップがすべて同じマイグレーション・セット (単一ファイル・モードのマイグレーションの場合は、外部ソース形式ファイル) に含まれていない場合は、formGroup にすべての書式が組み込まれません。</p> <p><b>考えられる解決策 1:</b> マイグレーション・セットが、同じマップ・グループ名をもつすべてのマップを含んでいることを確認します。</p> <p><b>考えられる解決策 2:</b> 欠落している書式を EGL ファイルに追加し、formGroup 定義の中でこれらのフォームをネストします。</p>

## 浮動域と開始位置

**VisualAge Generator:** VisualAge Generator は、同じ装置サイズをもつそれぞれの装置タイプに対して、異なる浮動域サイズと開始位置を指定することを許容します (ただし推奨されません)。

**EGL:** EGL formGroup と印刷書式は、装置サイズのみを指定します。EGL テキスト・フォームは、装置サイズとフォーム・サイズの両方を指定します。EGL の場合、1 つの装置サイズに対して指定できるマージン仕様は 1 セットだけです。

**マイグレーションに必要な関連パーツ:** 適用外。

表 21. 浮動域と開始位置

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>複数の装置が同じ装置サイズのもので、異なる浮動域サイズまたは開始位置を使用している場合は、エラー・メッセージが出されます。</li> <li>それぞれの VAGen 浮動域仕様ごとに、同等な EGL 装置サイズとマージン仕様を組み込みます。複数の VAGen 装置が変換によって同じ EGL 装置のサイズ仕様およびマージン仕様を持つようになった場合、マイグレーション・ツールは EGL に関して 1 つのエントリーのみを組み込みます。</li> </ul>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>起こりうる問題:</b> 同じ装置サイズをもつ複数の装置が、異なる浮動域サイズまたは開始位置を VisualAge Generator 内で指定している場合に限り、問題が起こります。EGL による検証の結果、「問題」ビューにメッセージが表示されます。</p> <p><b>考えられる解決策:</b> エラー・メッセージを検討します。formGroup 定義を編集して、この装置サイズに対して必要な浮動域サイズと開始位置を 1 つ指定します。</p>	<p>『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。</p>

## マップ・グループ、マップ、および装置サイズ

**VisualAge Generator:** VisualAge Generator は、表示マップの装置サイズとして、6x40、12x40、16x64、および 255x160 の縦の長さと幅をサポートします。

**EGL:** EGL は、テキスト書式に対しては共通の装置サイズをサポートしますが、COBOL 生成の場合は 6x40、12x40、16x64、および 255x160 の装置サイズを許容しません。これらの装置は、Java 生成の場合はサポートされます。

**マイグレーションに必要な関連パーツ:** 適用外。

表 22. マップ・グループ、マップ、および装置サイズ

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>マップ・グループをマイグレーションする際に、マイグレーション・ツールは、ScreenFloatingArea プロパティ内の screenSize プロパティにオリジナルの縦の長さおよび幅を組み込みます。また、ツールは警告メッセージを出します。</p>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p>表示マップをマイグレーションする際に、マイグレーション・ツールは、マイグレーションしたテキスト書式の screenSizes プロパティにオリジナルの縦の長さおよび幅を組み込みます。また、ツールは警告メッセージを出します。</p>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>



表 22. マップ・グループ、マップ、および装置サイズ (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題:</b> EGL COBOL 生成機能はこれらの装置をサポートしません。生成時にエラーが発生します。</p> <p><b>解決策:</b> COBOL の生成を行う場合は、EGL の <i>formGroup</i> とテキスト書式を編集し、サポートされない画面サイズを除去または変更します。</p>	<p>『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。</p>

## マップ名とヘルプ・マップ名

**VisualAge Generator:** マップ名は、マップ・グループとマップ名からなる 2 部構成の名前です。プログラムのメインのマップ・グループとヘルプ・マップ・グループは、両方とも同名のマップを含むことができます。例えば、プログラム X のメインのマップ・グループ GROUPA とヘルプ・マップ・グループ GROUPH は、それぞれ MAP1 という名前のマップを含むことができます。マップ名は 8 文字に制限されます。マップ名はプログラム名と同じでも構いません。VAGen マップ・グループ名の中では # 記号と @ 記号は無効ですが、マップ名のマップ名部分では # 記号も @ 記号も使用できます。

**EGL:** 書式名は *formGroup* 名を含みません。代わりに、*formGroup* パーツ内でテキスト書式と印刷書式が定義されます。また EGL の場合、メイン *formGroup* とヘルプ *formGroup* の書式名がすべて固有であることも必要です (1 つのプログラムに対して、重複する書式名が 2 つの *formGroup* に含まれていない)。EGL は、書式名がプログラム名と同じであることを許容しません。さらに、EGL では、フォーム名が予約語であること、またフォーム名の先頭文字として # 記号と @ 記号を使用することは許されません。EGL は、定義時に 8 文字を超える書式名を許します。生成時には、別名が指定されていれば、別名が書式名として使用されます。COBOL 生成では、*alias* のフォーム名は 8 文字に制限されます。生成されるコードのメイン *formGroup* とヘルプ *formGroup* 内では、名前の重複が許されます。

**マイグレーションに必要な関連パーツ:** マップ・グループをマイグレーションする際には、プログラムとそのマップ・グループ、ヘルプ・マップ・グループ、および両マップ・グループ内のマップすべてが必要です。

表 23. マップ名とヘルプ・マップ名

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>メインのマップ・グループ、またはヘルプ・マップ・グループのどちらかを最初にマイグレーションするプログラムに応じて、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>予約語が使用されている場合、または # 記号または @ 記号が名前のマップ名部分の先頭文字として使用されている場合に、マップ名の変更を実行します。プログラムのメインのマップ・グループ、およびヘルプ・マップ・グループにあるマップの名称が、必要に応じて変更されます。</li> <li>メインのマップ・グループと重複する名前がないかどうか、プログラムのヘルプ・マップ・グループにあるマップすべての名前を検査します。</li> <li>プログラム名と、プログラムのメインのマップ・グループおよびヘルプ・マップ・グループにあるマップすべての名前を比較します。</li> </ul> <p>ヘルプ・マップ・グループにあるマップの名称が、メインのマップ・グループにあるどのマップとも同じでなければ、マイグレーション・ツールはヘルプ・マップ名を変更しません。</p> <p>ヘルプ・マップ・グループにあるマップの名称が、プログラムのメインのマップ・グループにあるいずれかのマップと同じならば、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>ヘルプ・マップの内容が定数のみの場合、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>ヘルプ・マップの名称を、<code>helpMapName</code> にお客様指定の接尾部を付けた名称に変更します。</li> <li><code>alias</code> プロパティにオリジナルのヘルプ・マップ名を組み込みます。</li> <li>すべてのマップの <code>helpForm</code> プロパティを変更して、新しいヘルプ・マップ名を指定します。</li> </ul> </li> <li>ヘルプ・マップ・グループ内のマップが変数を含む場合、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>エラー・メッセージを出します。</li> <li>マップの名称を変更しません。</li> <li>マップをマイグレーションします。</li> </ul> </li> </ul> <p>これは、そのプログラムのメインのマップ・グループとしてそのヘルプ・マップ・グループを指定する他のプログラムが、そのマップを使用している可能性があるからです。</p> <p>ヘルプ・マップ・グループ内のマップの内容が定数フィールドのみであり、マップ名がプログラム名と同じである場合、マイグレーション・ツールはヘルプ・マップの名称を変更します。行われる処理は、ヘルプ・マップ・グループとメインのマップ・グループ内でマップ名が競合する場合に、名前変更を実行できるときの処理と同じです。</p> <p>ヘルプ・マップ・グループにあるマップが変数フィールドを含み、そのマップの名称がプログラム名と同じである場合、またはメインのマップ・グループにあるマップの名称がプログラムと同じである場合は、マイグレーション・ツールはマップの名称を変更しません。行われる処理は、ヘルプ・マップ・グループとメインのマップ・グループ内でマップ名が競合する場合に、名前変更を実行できないときの処理と同じです。</p>	<p>マップ・グループの名称を変更する際に、プログラムが使用できなければ、マイグレーション・ツールは 2 つのマップ・グループが関連しているかどうかを判別できず、マップ・グループがヘルプ・マップ・グループとして指定されているかどうかを判別できません。マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>予約語や # 記号、@ 記号が使用されている場合に、マップ名の変更を実行します。</li> <li>そのほかには、ヘルプ・マップの名称変更に関する検査を行いません。</li> </ul>

表 23. マップ名とヘルプ・マップ名 (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題 1:</b> formGroup が一方のプログラムではメインの formGroup として使用され、他方のプログラムではヘルプ formGroup として使用されている場合に、問題が起こる可能性があります。</p> <p><b>考えられる解決策:</b> ヘルプ formGroup を 2 つの formGroup に分割します。一方はヘルプ書式のみを含み、他方は変数フィールドのある書式を含みます。ヘルプ書式のみを含む formGroup を、オリジナル・プログラムのヘルプ formGroup として指定します。変数フィールドのある書式を含む formGroup をメインの formGroup として指定し、ヘルプ書式のみを含む formGroup を第 2 のプログラムのヘルプ formGroup として指定します。</p> <p><b>起こりうる問題 2:</b> ヘルプ formGroup 内のマップが変数フィールドを含んでいて、メイン formGroup 内のマップと同じ名前である場合に、問題が起こります。</p> <p><b>考えられる解決策:</b> 問題 1 の考えられる解決策と同じです。</p> <p><b>起こりうる問題 3:</b> 同じヘルプ formGroup が複数のプログラムによって共用されている場合に、問題が起こる可能性があります。この場合、マイグレーション・ツールは、さまざまなプログラムに対して、名前変更を必要とするヘルプ書式すべての名前を変更しないことがあります。</p> <p><b>考えられる解決策:</b> ヘルプ formGroup の中で、ヘルプ・マップ接尾部を名前に追加して、必要な書式すべての名前を変更します。alias プロパティを組み合わせ、生成時に使用するオリジナルのヘルプ・マップ名を指定します。すべての formGroup にある対応するテキスト書式すべてを変更して、新しいヘルプ書式名を指定します。</p>	<p><b>起こりうる問題:</b> formGroup がプログラム内で使用されていて、メイン formGroup とヘルプ formGroup 内の書式名の間に競合がある場合に限り、問題が起こります。</p> <p><b>考えられる解決策:</b> 『関連パーツを使用したマイグレーション』に示したものと同一解決策が適用されます。</p>

## 数値変数フィールド

**VisualAge Generator:** マップの数値フィールドの長さは 1 つです。すべての数字、小数点、符号、通貨記号、および数字分離記号が収まるだけの長さにする必要があります。ただし、実行時にフィールドの長さが不足している場合、VisualAge Generator は通貨記号と数字分離記号を省略します。VisualAge Generator は、数値が正の場合にも符号を省略します。使用可能なスペースに収めるために、VisualAge Generator は必要に応じて高位の桁を除去します。

**EGL:** 書式の変数フィールドは、桁数や小数部などの型定義と、データが書式に占めるスペースを指定する fieldLen プロパティを両方とも指定します。実行時に fieldLen の大きさが不足していて、すべての数字と書式制御文字が収まらない場合、EGL はランタイム・メッセージを出します。

**マイグレーションに必要な関連パーツ:** 適用外。

表 24. 数値変数フィールド

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>マップの数値フィールドをマイグレーションする際に、マイグレーション・ツールは長さで fieldLen を次のように設定します。</p> <ul style="list-style-type: none"> <li>マイグレーション・ツールは、VisualAge Generator の変数フィールドに対して指定されているものと同じ長さで fieldLen を常に設定します。</li> <li>マイグレーション・ツールは、型定義の長さで小数部を次のように設定します。 <ul style="list-style-type: none"> <li>変数フィールドが小数部を指定していない場合、マイグレーション・ツールは型定義の長さを fieldLen に設定します。</li> <li>変数フィールドが小数部を指定している場合は、小数点を入れるために、マイグレーション・ツールは型定義の長さを fieldLen から 1 だけ引いた値に設定します。この技法により、実行時に発生するオーバーフローの問題が回避されます。</li> </ul> </li> </ul>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>起こりうる問題:</b> 書式のフィールド長が不足していて、実行時に数字、小数点、符号、通貨記号、および数字分離記号がすべて収まらない場合、EGL はランタイム・エラー・メッセージを出します。</p> <p><b>解決策:</b> 実行時に発生する可能性がある最大の数値、および指定した書式制御文字すべてが収まるように、十分な大きさの fieldLen を指定して書式定義を変更します。</p>	<p>『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。</p>

## マップ変数フィールドと編集ルーチン

**VisualAge Generator:** マップ変数には編集ルーチンを指定でき、このルーチンはテーブル、関数、EZEC10、または EZEC11 です。編集メッセージは、編集ルーチンが EZEC10、EZEC11、またはテーブルである場合のみ使用されます。

**EGL:** 書式フィールドには、validatorDataTable と validatorFunction の両方を指定できます。さらに書式フィールドには、validatorDataTableMsgKey と validatorFunctionMsgKey の両方を指定できます。

**マイグレーションに必要な関連パーツ:** テーブルまたは演算部のどちらか。

表 25. 変数マップ・フィールドと編集ルーチン

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>マップが最初にマイグレーションされるときに、マイグレーション・ツールは次のうち最初に該当する処理を実行します。</p> <ul style="list-style-type: none"> <li>• <code>editRoutineName</code> が EZEC10 または EZEC11 ならば、マイグレーション・ツールは <code>validatorFunction</code> プロパティを同等な EGL システム・ライブラリー関数に設定します。さらにマイグレーション・ツールは、<code>validatorFunctionMsgKey</code> を編集メッセージ (ある場合) に設定します。</li> <li>• <code>editRoutineName</code> が関数ならば、マイグレーション・ツールは <code>validatorFunction</code> プロパティを設定します。<code>validatorFunctionMsgKey</code> は VisualAge Generator 内では使用されないため、マイグレーション・ツールはこれを省略します。</li> <li>• <code>editRoutineName</code> がテーブルならば、マイグレーション・ツールは <code>validatorDataTable</code> プロパティを設定します。さらにマイグレーション・ツールは、<code>validatorDataTableMsgKey</code> を編集メッセージ (ある場合) に設定します。</li> </ul>	<p><code>editRoutineName</code> と同じ名前の関数またはテーブルが使用できない場合、マイグレーション・ツールは次の内の最初に適用される処理を行います。</p> <ul style="list-style-type: none"> <li>• <code>editRoutineName</code> が EZEC10 または EZEC11 ならば、マイグレーション・ツールは <code>validatorFunction</code> プロパティを同等な EGL システム・ライブラリー関数名に設定します。さらにマイグレーション・ツールは、<code>validatorFunctionMsgKey</code> を編集メッセージ (ある場合) に設定します。</li> <li>• <code>editRoutineName</code> が 7 文字より長い場合、これは必ず関数名なので、マイグレーション・ツールは <code>validatorFunction</code> プロパティを設定します。<code>validatorFunctionMsgKey</code> は VisualAge Generator 内では使用されないため、マイグレーション・ツールはこれを省略します。</li> <li>• 編集メッセージが指定されている場合、マイグレーション・ツールは <code>validatorDataTable</code> と <code>validatorDataTableMsgKey</code> を設定します。</li> <li>• 編集メッセージが指定されていない場合、マイグレーション・ツールは <code>validatorFunction</code> プロパティを設定し、エラー・メッセージを出します。</li> </ul>
<p><b>起こりうる問題:</b> 関数と <code>dataTable</code> の名前が同じであり (通常は、異なるサブシステム内で)、2 つのプログラムが同じ <code>formGroup</code> を共用していて (通常は、同じサブシステム内で)、かつ一方のプログラムが関数の使用を予期し、他方のプログラムが <code>dataTable</code> の使用を予期している場合に限り、問題が起こります。</p> <p><b>考えられる解決策:</b> <code>formGroup</code> を共用するプログラムを検討します。この状態が生じる場合は、<code>validatorDataTable</code> を使用するための別個の <code>formGroup</code> を作成します。<b>欠点:</b> 保守する <code>formGroup</code> が 2 つになります。同一の書式を共通のファイルに移動し、それぞれの <code>formGroup</code> 内で共通書式を指す <code>use formName</code> ステートメントを指定することにより、この欠点を最小限にすることができます。</p>	<p><b>起こりうる問題:</b> マイグレーション・ツールが予測を誤った場合に限り、問題が起こります。マイグレーション・ツールが関数を指定したときに、この書式を使用するいずれかのプログラムが <code>dataTable</code> を予期している可能性があります。</p> <p><b>考えられる解決策:</b> エラー・メッセージのあるマップの使用方法を検討します。</p>

## マップ・フィールドと数値ハードウェア属性

**VisualAge Generator:** VisualAge Generator は、文字定数フィールド、文字変数フィールド、および数値変数フィールドに対して、数値ハードウェア属性をサポートします。数値ハードウェア属性により、エンド・ユーザーが変数フィールドに非数値データを入力することを防止できます。

**EGL:** EGL は、文字変数フィールドの `isDecimalDigit` 属性のみをサポートします。数値フィールドには、有効な数字と、符号や小数点などの書式制御文字のみがフィールドに入力されるように保証するソフト編集機能が備わっています。

## マイグレーションに必要な関連パーツ: 適用外。

表 26. マップ・フィールドと数値ハードウェア属性

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"><li>マップにある数値ハードウェア属性を指定した文字変数に対しては、ツールは <code>isDecimalDigit = yes</code> プロパティを組み込みます。</li><li>マップにある文字定数に対しては、ツールは常に <code>isDecimalDigit</code> プロパティを省略します。</li><li>マップにある数値変数フィールドに対しては、ツールは常に <code>isDecimalDigit</code> プロパティを省略します。</li></ul>	マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。
<p><b>起こりうる問題:</b> 数値フィールドに非数値データを入力できるようになるので、実行時にエンド・ユーザーが多少の違いを感じます。この入力が行われた場合、EGL はランタイム・エラー・メッセージを出します。</p> <p><b>考えられる解決策:</b> この変化は、VAGen 生成のコードから EGL 生成のコードに変更するときに予想される変化であることを、エンド・ユーザーに知らせてください。</p>	『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。

## マップ配列と属性

**VisualAge Generator:** VisualAge Generator は、配列の要素に対して異なる属性を使用することを許容します (ただし推奨されません)。例えば、VisualAge Generator の場合は、保護、入力必須 (input required)、入力時充てん必須 (require fill on input)、数値ハードウェア属性、変更データ・タグ、およびライト・ペン検出が、マップ配列のそれぞれの要素ごとに異なる可能性があります。

**EGL:** EGL の場合、配列項目のオーバーライド可能なプロパティは、フィールド表示プロパティ (色、強調表示、輝度、保護、変更、およびアウトライン) と、カーソル、位置、および値のみです。

## マイグレーションに必要な関連パーツ: 適用外。

表 27. マップ配列と属性フィールド

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
マイグレーション・ツールは、配列の先頭要素 (配列指標 1) に対して次のプロパティを使用して、同等な EGL プロパティ (入力必須 (input required)、入力時充てん必須 (require fill on input)、数値ハードウェア属性、およびライト・ペン検出) を設定します。EGL は、配列の先頭要素のプロパティを、配列のすべての要素に使用します。	マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。



表 27. マップ配列と属性フィールド (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題:</b> 配列の要素に異なる属性を使用していた場合に限り、問題が起こります。</p> <p><b>考えられる解決策:</b> 配列の先頭要素のプロパティを最も制約の小さい値に変更し、配列の各要素が必要な基準を満たしているかどうか検査するロジックを <code>validatorFunction</code> に追加します。また、実行時の書式の外観に違いが生じる場合は、エンド・ユーザーにそのことを知らせてください。</p>	<p>『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。</p>

## 名前なしマップ変数フィールド

**VisualAge Generator:** VisualAge Generator は、名前なし変数フィールドがマップに存在することを許容します (ただし推奨されません)。生成時に、名前なし変数フィールドは定数に変換されます。プログラムと関数は、名前なし変数フィールドを参照することはできません。

**EGL:** EGL は、名前なし変数フィールドが書式に存在することを許容しません。

**マイグレーションに必要な関連パーツ:** 適用外。

表 28. 名前なしマップ変数フィールド

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>マップにある名前なし変数フィールドに対して、マイグレーション・ツールは次のいずれかが指定されているかどうかを検査します。</p> <ul style="list-style-type: none"> <li>初期値</li> <li>Protect = yes</li> <li>Cursor = yes</li> <li>「アウトラインなし」以外のアウトライン</li> <li>「強調表示なし」以外の強調表示</li> </ul> <p>前記のいずれかが指定されている場合、マイグレーション・ツールは対応する EGL プロパティを使用して定数フィールドを作成し、警告メッセージを出します。</p>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p>名前なし変数フィールドに対してこれらのプロパティがどれも指定されていない場合 (またはデフォルト値のみが指定されている場合)、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>書式に定数フィールドを作成しません。</li> <li>警告メッセージを出します。</li> </ul>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>起こりうる問題:</b> なし。VisualAge Generator ではこのフィールドは参照できません。</p>	<p><b>起こりうる問題:</b> なし。</p>

## 無保護マップ定数

**VisualAge Generator:** VisualAge Generator は、マップ上で無保護定数の使用をサポートします。テストおよび生成時に、無保護定数は、保護が自動スキップに設定されている場合と同様に処理されます。

**EGL:** EGL は、書式上で無保護定数の使用をサポートしません。テキスト書式の定数の場合、EGL は protect=yes と protect=skip の両方をサポートします。印刷書式の場合、EGL は protect プロパティをサポートしません。

**マイグレーションに必要な関連パーツ:** 適用外。

表 29. 無保護マップ定数

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
書式をマイグレーションする際に、無保護定数フィールドに対してマイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"><li>書式がテキスト書式である場合、マイグレーション・ツールは EGL の protect プロパティを skip に設定して、エラー・メッセージを出します。</li><li>書式が印刷書式である場合、マイグレーション・ツールは protect プロパティを省略し、メッセージを出しません。protect プロパティは、EGL の印刷書式の中では使用されません。</li></ul>	マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。
起こりうる問題: なし。	起こりうる問題: なし。

## row=0, column=0 にあるフィールド

**VisualAge Generator:** VisualAge Generator 4.5 は、システム共通プロダクトまたは VisualAge Generator の旧リリースで row=0, column=0 に配置されたフィールドを許容します。ただし、VisualAge Generator 4.5 にはこの位置にフィールドを作成する手段はありません。row=0, column=0 に配置されたフィールドの属性設定情報は設定できません。

**EGL:** EGL は、row=0, column=0 に配置されたフィールドをサポートしません。すべてのフィールドに属性バイトが含まれている必要があります。

**マイグレーションに必要な関連パーツ:** 適用外。

表 30. row=0, column=0 にあるフィールド

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>書式をマイグレーションする際に、フィールドが row=0, column=0 に配置されている場合、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>フィールドが定数フィールドであり、値の先頭文字がブランクの場合、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>値から先頭文字を除去し、フィールド長を 1 だけ減らします。</li> <li>position プロパティを [1,1] に設定します。</li> <li>フィールドにデフォルト表示プロパティを組み込みます。</li> <li>警告メッセージを出します。</li> </ul> </li> <li>フィールドが定数フィールドであり、値の先頭文字がブランクでない場合、またはフィールドが変数フィールドである場合、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>値もフィールド長も変更しません。</li> <li>position プロパティを [0,0] に設定します。</li> <li>フィールドにデフォルト表示プロパティを組み込みます。</li> <li>エラー・メッセージを出します。</li> </ul> </li> </ul>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>起こりうる問題 1:</b> フィールドが変更できず、position=[0,0] にある場合は、「問題」ビューにエラーが表示されます。</p> <p><b>解決策 1:</b> 書式を修正して、フィールドの位置を変更します。フィールドの属性バイトが入る場所を確保するために、他のフィールドを移動したり、定数の位置を変更したりする必要が生じることがあります。また、デフォルト表示プロパティを検討して、正しい色、強調表示などが使用されていることを確認してください。</p> <p><b>起こりうる問題 2:</b> 定数フィールドが position=[1,1] に変更された場合に、デフォルト表示プロパティが原因で実行時の外観に相違点が生じる可能性があります。</p> <p><b>解決策 2:</b> マイグレーションの警告メッセージを検討し、マイグレーション・ツールがフィールドの位置を調整した箇所の書式を必ずテストしてください。</p>	<p><b>起こりうる問題:</b> 『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。</p>

## プログラムの未確定状態の処理

### プログラム名と予約語

**VisualAge Generator:** VisualAge Generator に予約語はありません。VAGen プログラム名で # 記号と @ 記号は使えません。

**EGL:** EGL には予約語があります。さらに EGL の場合は、パーツ名の先頭文字として # 記号または @ 記号を使用することはできません。プログラム名が予約語であってはなりません。

**マイグレーションに必要な関連パーツ:** 適用外。

表 31. プログラム名と予約語

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
マイグレーション・ツールはプログラムの名前を自動的に変更しません。プログラム名が予約語リストと一致する場合、マイグレーションのステージ 1 で使用されるマイグレーション・ツールがエラー・メッセージを出します。プログラム名を変更しない場合は、マイグレーションのステージ 2 で使用されるマイグレーション・ツールもエラー・メッセージを出します。	マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。
<p><b>起こりうる問題:</b> プログラム名が予約語と一致している場合に限り、問題が起こります。EGL による検証の結果、「問題」ビューにメッセージが表示されます。</p> <p><b>解決策:</b> プログラムの名前を変更します。この作業は、VisualAge Generator またはマイグレーション後の EGL のどちらでも行うこともできます。EGL でプログラム名を変更する場合は、call、transfer、show の各ステートメントでの参照およびリンクページ・オプション・パーツでの参照など、プログラム名とそれに対するすべての参照を変更する必要があります。また、このプログラムに対応するバインド制御パーツ、またはリンク・エディット・パーツの名前も変更します。生成されるプログラムの名前としてオリジナルのプログラム名を保持する必要がある場合は、<i>alias</i> プロパティをオリジナルのプログラム名に設定します。<i>alias</i> プロパティを指定しない場合は、CICS プログラム定義など、EGL 以外によるプログラム名の参照すべてを必ず変更してください。</p>	『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。

### プログラム内の暗黙データ項目

**VisualAge Generator:** VisualAge Generator は、暗黙データ項目（レコード、マップ、テーブル、呼び出し先パラメーター・リスト、関数仮パラメーター・リスト、または関数ローカル・ストレージ内で明示的に定義されていない項目）の使用を許容します（ただし推奨されません）。

**EGL:** EGL は、暗黙データ項目の使用を許容しません。

**マイグレーションに必要な関連パーツ:** 適用外。

表 32. プログラム内の暗黙データ項目

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
マイグレーション・ツールは、暗黙項目の定義を自動的に作成しません。プログラムが暗黙項目を許容している場合、マイグレーションのステージ 2 で使用されるマイグレーション・ツールが警告メッセージを出します。	マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。
<p><b>起こりうる問題:</b> プログラムが実際に暗黙項目を使用する場合に限り、問題が起こります。暗黙項目を許容するプログラムがないかどうか、「TO DO」リスト・ログを検討してください。プログラムが実際に暗黙項目を使用していた場合は、「問題」ビューにエラーが示されます。</p> <p><b>解決策:</b> VisualAge Generator または EGL のどちらでも、プログラムに暗黙項目の定義を追加できます。VAGen による検証を行うと、暗黙項目に必要な定義が表示されます。</p>	『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。

## 関連プログラム・パーツ

**VisualAge Generator:** VisualAge Java の場合は複数のプロジェクトとパッケージ、VisualAge Smalltalk の場合は複数の構成マップとアプリケーションに、プログラムの関連パーツが存在する可能性があります。

**EGL:** 複数のプロジェクト、フォルダー、パッケージ、およびファイルに、プログラムの関連パーツが存在する可能性があります。

**マイグレーションに必要な関連パーツ:** プログラムの場合: すべての関連パーツ。

**注:** import ステートメントについて詳しくは、38 ページの『EGL のビルド・パスと import ステートメント』を参照してください。

表 33. 関連プログラム・パーツ

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• EGL ファイルを配置する先のパッケージを指定する <code>package</code> ステートメントを組み込みます。</li> <li>• 現行ファイル内のパーツが必要とする、現行ファイルとは異なるパッケージにある関連パーツを含むパッケージの <code>import</code> ステートメントを、EGL ファイルに組み込みます。この <code>import</code> ステートメントは、ステージ 1 から 3 までを使用してマイグレーションを行った場合のみ組み込まれます。</li> <li>• プログラムに対して、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>– プログラムの 1 次作業用ストレージ・レコードの宣言を組み込みます。 VAGen の 1 次作業用ストレージ・レコードにレベル 77 項目が存在する場合、ツールは新しいレベル 77 項目レコードの宣言も組み込みます。</li> <li>– VAGen テーブルと追加レコード・リストにあるレコードすべての宣言を組み込み、該当する場合は、VAGen の再定義レコードに対する <code>redefines</code> プロパティを組み込みます。</li> <li>– 入出力レコードすべての宣言を組み込みます。</li> <li>– MQ API 呼び出しのパラメーターとして使用されたレコード (VisualAge Generator 内で MQ メッセージ・レコードの属性として指定されたレコード) の宣言を組み込みます。</li> <li>– 先頭 UI レコードとして使用される UI レコードの宣言を <code>CONVERSE</code> ステートメントまたは <code>XFER</code> ステートメント内に組み込みます。</li> <li>– プログラムの PSB 内で参照される DL/I セグメント・レコードの宣言を組み込みます。</li> <li>– VAGen のテーブルおよび追加レコードのリストにあるテーブルすべての使用宣言を組み込みます。</li> <li>– プログラムのマップ・グループとヘルプ・マップ・グループの使用宣言を組み込みます。</li> <li>– プログラムの PSB の変数宣言を組み込みます。</li> </ul> </li> </ul>	<p>マイグレーション・ツールは、使用可能なプログラム関連パーツをすべて使用します。</p>



表 33. 関連プログラム・パーツ (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>起こりうる問題: なし。</p>	<p><b>起こりうる問題 1:</b> 欠落しているパーツがある場合に限り、問題が起こります。マイグレーション・ツールは、欠落しているパーツを検出すると、欠落しているパーツを示す警告メッセージを出します。マイグレーション・ツールは、欠落しているパーツに関する推測は行いません。このため、次のように、マイグレーションしたプログラムにさまざまな問題が生じる可能性があります。</p> <ul style="list-style-type: none"> <li>• import ステートメントの欠落。</li> <li>• レベル 77 レコード宣言の欠落。</li> <li>• VAGen 再定義レコードの redefines プロパティの欠落。</li> <li>• 入出力レコード宣言の欠落。</li> <li>• MQ API 呼び出しのパラメーターとして使用されたレコードの宣言の欠落。</li> <li>• UI レコード宣言の欠落。</li> <li>• SSA 内で参照されるセグメントの DL/I セグメント・レコード宣言の欠落。</li> </ul> <p>redefines プロパティが欠落している場合を除いて、「問題」ビューにエラーが示され、問題の識別に役立ちます。  <b>注:</b> マイグレーション・ツールは、欠落しているパーツをすべて検出するとは限りません。</p> <p><b>考えられる解決策 1A:</b> マイグレーション・セットを変更して、VisualAge Generator 内でプログラムの検証に必要なパーツをすべて組み込みます。プログラムの関連パーツがすべて一緒にマイグレーションされるように、新しいマイグレーション・セットを使用してプログラムのマイグレーションを再度行います。</p> <p><b>考えられる解決策 1B:</b> EGL 内で欠落しているパーツを見つけて、EGL プログラムを修正します。</p> <p><b>起こりうる問題 2:</b> レベル 77 項目が欠落している場合は、66 ページの『レコード内のレベル 77 項目』を参照してください。</p> <p><b>起こりうる問題 3:</b> 再定義レコードが欠落している場合は、65 ページの『再定義レコード』を参照してください。</p>

## 呼び出しパラメーター・リストに EZEDLPCB が含まれるプログラム

**VisualAge Generator:** VisualAge Generator は EZEDLPCB[n] を使って、プログラムが PCB をパラメーターとして受け取ることを示します。n は数値リテラルでなければなりません。n の値は 0 (I/O PCB の場合) であるか、プログラムの PSB パーツ内で定義されている PCB のいずれかに対応する数値でなければなりません。

**EGL:** EGL は xxxx\_PCBRecord という型定義を持つ変数名を使って、プログラムが PCB をパラメーターとして受け取ることを示します。 xxxx は PCB の型に応じて IO、ALT、DB、または GSAM になります。 EGL にはまた、PCB 変数名をプログラムの PSB レコード内の対応する位置にマッピングするための pcbParms プロパティも必要です。

### マイグレーションに必要な関連パーツ: PSB パーツ

表 34. 呼び出しパラメーター・リストに EZEDLPCB が含まれるプログラム

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>プログラムをマイグレーションする際に、そのプログラムがパラメーターとして EZEDLPCB[n] を指定し、かつ PSB パーツが使用できる場合、マイグレーション・ツールは次の処理を行ないます。</p> <ul style="list-style-type: none"> <li>パラメーターとして変数 PCB <i>n</i> を組み込み、次のように PSB 内の対応する PCB に基づいて型定義を指定します。 <ul style="list-style-type: none"> <li>マイグレーション・ツールは、EZEDLPCB[0] の型定義を IO_PCBRecord に設定します。</li> <li><i>n</i> が PSB パーツ内の PCB に対応する場合、マイグレーション・ツールは その PSB 内の PCB 型を基に型定義を設定します。</li> <li><i>n</i> が PSB パーツ内の PCB 数より大きい場合、マイグレーション・ツールはメッセージを発行し、型定義を EZE_UNKNOWN_PCB_TYPE に設定します。</li> </ul> </li> <li>pcbParms プロパティ内の対応する場所に pcbn 変数をすべてリストします。 I/O PCB は pcb0 です。これは、パラメーターとして指定された場合、pcbParms プロパティの冒頭にリストされます。残りの pcbn 変数は、pcbParms リスト内の位置 <i>n</i>+1 にリストされます。</li> </ul>	<p>プログラムをマイグレーションする際に、そのプログラムがパラメーターとして EZEDLPCB[n] を指定し、かつ PSB パーツが使用できない場合、マイグレーション・ツールはメッセージを発行し、次の処理を行います。</p> <ul style="list-style-type: none"> <li>パラメーターとして変数 PCB <i>n</i> を組み込み、次のように型定義を指定します。 <ul style="list-style-type: none"> <li>マイグレーション・ツールは、EZEDLPCB[0] の型定義を IO_PCBRecord に設定します。</li> <li>マイグレーション・ツールはメッセージを発行し、その他すべての pcbn 変数の型定義を EZE_UNKNOWN_PCB_TYPE に設定します。</li> </ul> </li> <li>メッセージを発行し、pcbParms プロパティを EZE_UNKNOWN_PCB_MAPPING に設定します。</li> </ul>
<p><b>起こりうる問題:</b> PSB パーツに EZEDLPCB[n] パラメーターの <i>n</i> の最高値より少ない数の PCB しかない場合にのみ、問題が発生します。</p> <p><b>解決策:</b> VisualAge Generator ではプログラムは無効です。プログラム・ロジックを検証することで、プログラム・パラメーター・リストまたは プログラムの PSB パーツを変更するのかを判断します。</p>	<p><b>起こりうる問題 1:</b> 正しい PCB 型定義および pcbParms プロパティを指定する必要があります。</p> <p><b>解決策:</b> プログラムの PSB パーツを見つけます。プログラムを編集し、PCB 型定義を修正します。また、pcbParms プロパティの pcbn 変数を正しくマッピングします。</p>

## マイグレーションに必要な中間変数

**VisualAge Generator:** 一部の VAGen ステートメントは、EGL で同等なサポートを実現するために中間変数を必要とします。

**EGL:** EGL は、VAGen マイグレーションに必要な情報を提供するシステム・ライブラリー関数を備えています。このサポートは、VisualAge Generator 互換モードでのみ使用可能です。

マイグレーションに必要な関連パーツ: 適用外。

表 35. マイグレーションに必要な中間変数

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>プログラムをマイグレーションする際に、マイグレーション・ツールは常に次の宣言を組み込みます。</p> <ul style="list-style-type: none"> <li>• <code>custPrefixEZEREPLY</code></li> <li>• <code>custPrefixEZE_ITEMLEN</code></li> <li>• <code>custPrefixEZE_WAIT_TIME</code></li> </ul> <p>VAGen マイグレーション設定 「以前の <code>EZESYS</code> 値を初期化しない (<i>Do not initialize old EZESYS values</i>)」が選択されていない場合は、マイグレーション・ツールは以下も実行します。</p> <ul style="list-style-type: none"> <li>• <code>custPrefixEZESYS</code> の宣言を組み込みます。</li> <li>• <code>custPrefixEZESYS</code> の値を以前の VAGen <code>EZESYS</code> 値に設定する初期化ステートメントを組み込みます。</li> </ul> <p><code>custPrefix</code> は、予約語と競合するパーツ名の変更に使用されるものと同じ接頭部です。この値を設定するには、VAGen マイグレーション設定を使用します。</p>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p>これら 4 つの変数は、次のものをマイグレーションするために使用されます。</p> <ul style="list-style-type: none"> <li>• (REPLY オプションが指定されていない場合は、VAGen サービス・ルーチン。この場合は、<code>handleSysLibraryErrors</code> の現行値を保管してから復元する必要があります。</li> <li>• 直接相当するものが EGL にはない <code>TEST nnn</code>、<code>+nnn</code>、または <code>-nnn</code> ステートメント。ユーザーが入力したデータの長さの判別には、EGL システム・ライブラリー関数が使用されます。</li> <li>• <code>EZEWAIT</code> 関数。この場合、マイグレーション・ツールは時間を秒数に変換するためのロジックを追加します。</li> <li>• <code>IF</code>、<code>WHILE</code>、および <code>TEST</code> を除く、以前の VAGen 値を必要とするステートメント内での <code>EZESYS</code> の参照。</li> </ul>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>

表 35. マイグレーションに必要な中間変数 (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題:</b> マイグレーション中に、VAGen マイグレーション設定「以前の EZESYS 値を初期化しない (<i>Do not initialize old EZESYS values</i>)」を選択し、ステートメントに IF、WHILE、または TEST ではなく EZESYS を使用する場合に限り問題が起こります。この状態では、マイグレーション済みツールはステートメントで <i>custPrefixEZESYS</i> を使用しますが、プログラムには <i>custPrefixEZESYS</i> の宣言および初期化ステートメントがありません。「問題」ビューにエラーが示されます。</p> <p><b>考えられる解決策 1:</b> EGL ロジックを変更して、<i>sysVar.systemType</i> の新規の値を使用します。</p> <p><b>考えられる解決策 2:</b> <i>custPrefixEZESYS</i> の宣言および初期化ステートメントを、EZESYS の以前の VAGen 値の使用が必要なプログラムに追加します。</p>	<p>『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を使用できます。</p>

## 関数 (入出力ステートメントを含む) の未確定状態の処理

### マップの DISPLAY ステートメント

**VisualAge Generator:** 表示マップとプリンター・マップの両方に *DISPLAY* が使用されます。

**EGL:** 2 つの別個のステートメントを使用します。

- テキスト書式には *display form* を使用します。
- 印刷書式には *print form* を使用します。

VisualAge Generator 互換モードでは、書式が印刷書式である場合に *display form* が受け入れられます。

**マイグレーションに必要な関連パーツ:** 装置タイプを判別するためにマップが必要です。マイグレーション・ツールは、使用可能ないずれかのマップ・グループにある、このマップ名をもつ最初のマップを使用します。プログラムのコンテキスト内でマイグレーションを行う場合、マイグレーション・ツールはプログラムのメインのマップ・グループのみを検索します。

表 36. マップの Display ステートメント

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>この関数の最初のマイグレーションを基準として、この名前のマップが使用可能ならば、マイグレーション・ツールは次のように変換を行います。</p> <ul style="list-style-type: none"> <li>• マップが表示マップの場合は <i>display textForm</i></li> <li>• マップがプリンター・マップの場合は <i>print printForm</i></li> </ul>	<p>この名前のマップが使用不可ならば、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• <i>display form</i> に変換します。</li> <li>• マップ・タイプを判別できなかったことを示す警告メッセージを出します。</li> </ul>

表 36. マップの Display ステートメント (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題 1:</b> 最初にマイグレーションされたプログラムが印刷書式を使用していれば、マイグレーション・ツールは <code>print</code> ステートメントにマイグレーションします。別のプログラムが、同じ関数をテキスト書式に対して使用していると、問題が起こります。</p> <p><b>解決策 1:</b> VisualAge Generator 互換モードを使用します。関数を編集して、<code>print</code> ステートメントを <code>display</code> ステートメントに変更します。</p> <p><b>起こりうる問題 2:</b> VisualAge Generator 互換モードの使用を中止する場合に、2 つのプログラムがこの関数を使用していて、一方がテキスト書式、他方が印刷書式を指定している場合は、問題が起こります。</p> <p><b>考えられる解決策 2A:</b> 特定のターゲット環境が表示マップを常に使用し、その他の環境は印刷マップを常に使用する場合は、次のように EGL 関数を変更できます。</p> <pre>if (sysVar.systemType is zoscics)     DISPLAY_FUNCTION(); else     PRINT_FUNCTION(); end</pre> <p>ここで、<code>DISPLAY_FUNCTION</code> と <code>PRINT_FUNCTION</code> はそれぞれ、<code>display</code> ステートメントと <code>print</code> ステートメントを使用します。</p> <p><b>考えられる解決策 2B:</b> 関数が <code>display</code> ステートメントにマイグレーションされたとして、次の関数を変更します。</p> <pre>before-logic display textForm; after-logic</pre> <p>この関数を、次のように変更します。</p> <pre>before-logic-function(); display textForm; after-logic-function();</pre> <p><code>before-logic</code> と <code>after-logic</code> を別個の関数に配置すると、共通関数のロジックの大部分を保持できます。こうすれば、共通の <code>before-logic-function</code> と <code>after-logic-function</code> を引き続き使用しながら、変更した <code>display</code> 関数のコピーを作成して、印刷マップを使用するように変更できます。欠点: オリジナルの <code>DISPLAY</code> 関数を使用する関数に影響が波及する可能性があります。</p>	<p><b>起こりうる問題:</b> 『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があります。同じ解決策を適用できます。</p>

## 入出力エラー・ルーチン

**VisualAge Generator:** ファイルまたはデータベースの入出力を行う関数は、入出力エラー・ルーチンを指定できます。入出力エラー・ルーチンは、`main` 関数または非 `main` 関数のどちらであってもよく、構文は同じです。VisualAge Generator は、テスト

トまたは生成時に、入出力エラー・ルーチンがプログラムの main 関数または非 main 関数のどちらであるかを判別します。main 関数が入出力エラー・ルーチンとして使用されている場合、VisualAge Generator は関数スタックをスタックの先頭にポップし、スタックに (入出力エラー・ルーチンの) main 関数のみが存在する状態でスタックを再び開始してから、main 関数を呼び出します。非 main 関数が入出力エラー・ルーチンとして使用されている場合、VisualAge Generator は非 main 関数を現行関数スタックに追加してから、関数を呼び出します。

**EGL:** try ブロックと onException ステートメントがエラー処理に使用されます。onException ステートメントの構文は、次の機能をサポートします。

- `exit stack functionName;` を使用した、main 関数への再転送。
- `nonmainfunctionName();` を使用した、非 main 関数の呼び出し。
- `mainfunctionName();` を使用した、main 関数の呼び出し。VisualAge Generator はこの書式をサポートしません。EGL は、main 関数を現行関数スタックに追加してから、main 関数を呼び出します。

**マイグレーションに必要な関連パーツ:** main 関数のリストを指定したプログラム。

表 37. ファイルおよびデータベースの入出力エラー・ルーチン

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>この関数の最初のマイグレーションを基準として、使用可能なプログラムが存在すれば、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• プログラムの main 関数を指定する入出力エラー・ルーチンを次のように変更します。</li> </ul> <pre>try   I/O-Statement;   onException exit stack functionName; end</pre> <ul style="list-style-type: none"> <li>• 非 main 関数を指定する入出力エラー・ルーチンを次のように変更します。</li> </ul> <pre>try   I/O-Statement;   onException functionName(); end</pre>	<p>使用可能なプログラムが存在しなければ、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• 入出力エラー・ルーチン内で指定されている関数が非 main 関数であることを前提として、その関数を次のように変更します。</li> </ul> <pre>try   I/O-Statement;   onException functionName(); end</pre> <ul style="list-style-type: none"> <li>• 警告メッセージは出しません。これは、大量のメッセージが出される場合があり、これらのメッセージが無視されたり、他の重大なエラー・メッセージを隠したりする可能性があるからです。</li> </ul>



表 37. ファイルおよびデータベースの入出力エラー・ルーチン (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題:</b> この関数が使用されるプログラム内で、入出力エラー・ルーチンが main 関数または非 main 関数のどちらとして使用されるかがオリジナル・プログラムと異なる場合に、問題が起こります。</p> <p><b>注:</b> 「問題」ビューにメッセージは示されません。生成時にエラーは検出されません。ただし、プログラムは <i>VisualAge Generator</i> の場合と異なる動作をします。VisualAge Generator の場合のようにスタックをポップする代わりに、EGL は main 関数をスタックに追加します。</p> <p><b>考えられる解決策:</b> この状態が発生した場合は、main 関数への転送を行うための適切な構文を使用して、この入出力関数の新しいバージョンを作成します。<b>欠点:</b> この技法は、入出力関数を呼び出す他の関数に影響を与える可能性があります。</p>	<p><b>起こりうる問題:</b> 入出力エラー・ルーチンが main 関数であるプログラム内でこの関数が使用される場合に、問題が起こります。</p> <p><b>注:</b> 「問題」ビューにメッセージは示されません。生成時にエラーは検出されません。ただし、プログラムは <i>VisualAge Generator</i> の場合と異なる動作をします。VisualAge Generator の場合のようにスタックをポップする代わりに、EGL は main 関数をスタックに追加します。</p> <p><b>考えられる解決策:</b> 『関連パーツを使用したマイグレーション』にリストしたものと同一解決策が適用されます。</p>

## SQL 入出力ステートメント

**VisualAge Generator:** SQL 入出力の場合、レコード定義と Execution Time Statement Build の使用に基づいて、テストおよび生成機能が必要に応じて単一の入出力オプションを複数の SQL ステートメントに展開します。テストおよび生成機能は、SQL レコード定義にある入出力ステートメントに対応する tables 文節を常に作成します。

**EGL:** SQL ステートメントは、EGL プログラム内で明示的に指定されている必要があります。SQL ステートメントを変更する場合は、into 文節を除くすべての SQL 文節が必要です。Execution Time Statement Build は、prepare ステートメントと、それに続く open ステートメント、get ステートメント、または execute ステートメントに置き換えられます。

**マイグレーションに必要な関連パーツ:** SQL レコードと、代替仕様レコードとして指定されたレコード (存在する場合)。

表 38. SQL 入出力ステートメント

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>この関数の最初のマイグレーションを基準として、SQL レコードとその代替仕様レコードが使用可能ならば、マイグレーション・ツールはレコード定義、関数内の SQL ステートメント、および Execution Time Statement Build の使用に基づいて、対応する EGL ステートメントを作成します。関数内の SQL ステートメントが変更された場合、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• <code>into</code> 文節を含めたすべての文節を使用して、EGL SQL ステートメントを作成します。</li> <li>• SQL レコード内、またはその代替仕様レコード (該当する場合) 内の表名から、必要な <code>tables</code> 文節を作成します。</li> <li>• 入出力オブジェクトのレコード定義、または入出力オブジェクトの代替仕様レコードのレコード定義 (該当する場合) に基づいて、この SQL 入出力ステートメントに必要なその他の欠落している文節を作成します。</li> <li>• <code>!itemColumnName</code> を、項目名から対応する SQL 列名に変換します。</li> <li>• 特殊な処理を必要とする SQL 予約語について、SQL ステートメントの検討を行いません。予約語のリスト、およびこれらの予約語のいずれかを表名または列名として使用している場合に SQL ステートメントに対して必要な変更については、244 ページの『特殊な処理を必要とする SQL 予約語』を参照してください。</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• SQL 文節の欠落に関連した問題について詳しくは、97 ページの『SQL 入出力と必須 SQL 文節の欠落』を参照してください。</li> <li>• <code>!itemColumnName</code> の使用に関連した問題について詳しくは、99 ページの『SQL 入出力と <code>!itemColumnName</code>』を参照してください。</li> </ul>	<p>SQL レコード、またはその代替仕様レコードが使用不可ならば、マイグレーション・ツールは SQL ステートメントの変更情報と Execution Time Statement Build 情報のみを使用して、EGL SQL ステートメントを作成します。マイグレーション・ツールがレコード定義を使用できないので、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• <code>into</code> 文節を含めたすべての文節を使用して、EGL SQL ステートメントを作成します。</li> <li>• 表名として <code>EZE_UNKNOWN_SQLTABLE</code> を使用し、<code>tables</code> 文節の表ラベルとして <code>T1</code> を使用します。</li> <li>• 欠落している SQL 文節に対して、<code>EZE_UNKNOWN_SQL_clauseName</code> を使用します。ここで、<code>clauseName</code> は欠落している SQL 文節の外部ソース形式のキーワードです (例: <code>SELECT</code> または <code>VALUES</code>)。</li> <li>• 列名変数に対して <code>!itemColumnName</code> を使用します。</li> <li>• 関数の検討が必要であることを示すエラー・メッセージを出します。</li> <li>• 特殊な処理を必要とする SQL 予約語について、SQL ステートメントの検討を行いません。予約語のリスト、およびこれらの予約語のいずれかを表名または列名として使用している場合に SQL ステートメントに対して必要な変更については、244 ページの『特殊な処理を必要とする SQL 予約語』を参照してください。</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• SQL 文節の欠落に関連した問題について詳しくは、97 ページの『SQL 入出力と必須 SQL 文節の欠落』を参照してください。</li> <li>• <code>!itemColumnName</code> の使用に関連した問題について詳しくは、99 ページの『SQL 入出力と <code>!itemColumnName</code>』を参照してください。</li> </ul>

表 38. SQL 入出力ステートメント (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題 1:</b> 同名のレコードが 2 つあり、これらのレコードの SQL 表名または表ラベルが異なる場合に限り、問題が起こります。この問題は、複数の異なるサブシステムを使用する場合、またはテスト用と実動用に異なる表を使用して生成を行う場合に起こる可能性があります。</p> <p><b>考えられる解決策 1A:</b> テストから実動までの間に表名の修飾を変更したことが問題の原因である場合は、非修飾の表名を使用するように変更し、バインド時に修飾情報を指定します。</p> <p><b>考えられる解決策 1B:</b> 異なるサブシステム内で異なる表名を使用していることが問題の原因である場合は、レコードのコピーを作成し、コピーの名前を変更します。その後、入出力関数のコピーを作成して、新しいレコード名を使用します。新しい入出力関数の table 文節を適切に訂正します。<b>欠点:</b> この入出力関数を使用する関数に影響が波及する可能性があります。</p> <p><b>考えられる解決策 1C:</b> 異なるサブシステム内で異なる表名を使用していることが問題の原因である場合は、<code>tableNameVariables</code> プロパティを使用するようにレコードを変更し、また入出力関数を呼び出す前に表名変数を設定するように、このレコードの入出力を行う関数をすべて変更します (場合によっては、各プログラムの main 関数内で)。あるいは、VisualAge Generator 内で表名ホスト変数を変更して、プログラム、レコード、および関数を再度マイグレーションします。<b>欠点:</b> この処置により、静的 SQL が動的 SQL に変更されるので、パフォーマンスへの影響が生じる可能性があります。</p> <p><b>起こりうる問題 2:</b> SQL 表名または列名が、特殊な処理を必要とする SQL 予約語のいずれかである場合は、問題が起こります。マイグレーション・ツールは、これらの SQL 予約語を二重引用符で囲みません。「問題」ビューにエラーが示されます。</p> <p><b>解決策 2A:</b> 関数を編集して、SQL 表名または列名を二重引用符で囲みます。SQL 予約語のリストと、必要な構文の例については、244 ページの『特殊な処理を必要とする SQL 予約語』を参照してください。</p>	<p><b>起こりうる問題 1:</b> 変更された SQL ステートメント、または Execution Time Statement Build を使用する SQL ステートメントがある場合は、問題が起こります。レコードが欠落しているかどうか、および SQL ステートメントから具体的にどの SQL 文節が欠落しているかに応じて、「問題」ビューにエラーが示される場合があります。</p> <p><b>解決策:</b> 欠落している SQL 文節または表名に関連したメッセージがあるかどうか、マイグレーション・ログを検討してください。あるいは、EZE_UNKNOWN_SQL が発生しているかどうかワークスペースを検索します。レコード定義に基づいて、適切な tables 文節を判別します。EGL を使用して SQL 文節を再作成する方法については、97 ページの『SQL 入出力と必須 SQL 文節の欠落』を参照してください。<code>!itemColumnName</code> 変数の訂正については、99 ページの『SQL 入出力と <code>!itemColumnName</code>』を参照してください。</p> <p><b>起こりうるその他の問題:</b> 『関連パーツを使用したマイグレーション』に示した問題と同じことが起こる可能性があり、同じ解決策が適用されます。</p>

## SQL 入出力と必須 SQL 文節の欠落

**VisualAge Generator:** SQL 文節を変更した場合、VisualAge Generator 4.5 はすべての SQL 文節を保管します。一方、システム共通プロダクトと VisualAge Generator

の一部の旧バージョンは、変更した文節のみを保管します。旧バージョンの関数を VisualAge Generator 4.5 内でまったく変更しなかった場合、必須 SQL 文節の一部が欠落する可能性があります。

**EGL:** いずれかの SQL 文節を変更する場合は、SQL ステートメントの SQL 文節をすべて指定する必要があります。

**マイグレーションに必要な関連パーツ:** SQL レコードと、代替仕様レコードとして指定されたレコード (存在する場合)。

表 39. SQL 入出力と SQL 文節の欠落

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
SQL レコードとその代替仕様レコードが使用可能であり、いずれかの SQL 文節が存在し、ただし一部の文節が欠落している場合、マイグレーション・ツールはこの表の以降の行で説明するとおりに欠落している文節を作成します。この関数の最初のマイグレーションを基準として、マイグレーション・ツールは SQL レコードとその代替仕様レコード (存在する場合) を使用して、欠落している文節を作成します。	SQL レコードまたはその代替仕様レコードが使用できず、いずれかの SQL 文節が存在し、ただし一部の文節が欠落している場合、マイグレーション・ツールはこの表の以降の行で説明するとおりに欠落している文節を作成します。この関数の最初のマイグレーションを基準として、SQL レコードまたはその代替仕様レコードが使用できなければ、マイグレーション・ツールは意図的に無効な EGL 構文を作成します。
<b>tables 文節の欠落:</b> マイグレーション・ツールは、レコードにある SQL 表と表ラベルすべてをリストして、tables 文節を作成します。マイグレーション・ツールは、SQL 表名と表名ホスト変数の両方を、VAGen レコード定義内での出現順と同じ順序で組み込みます。	<b>tables 文節の欠落:</b> マイグレーション・ツールは、SQL 表名を EZE_UNKNOWN_SQLTABLE に設定して、表ラベルを T1 に設定し、エラー・メッセージを出します。
<b>SELECT 文節の欠落:</b> マイグレーション・ツールは、レコード内での項目の出現順と同じ順序でレコードにある SQL 列名すべてをリストして、select 文節を作成します。	<b>SELECT 文節の欠落:</b> マイグレーション・ツールは、select 文節の SQL 列名を EZE_UNKNOWN_SQL_SELECT に設定して、エラー・メッセージを出します。
<b>INTO 文節の欠落:</b> マイグレーション・ツールは、レコード内での項目の出現順と同じ順序でレコードにある項目名すべてをリストして、into 文節を作成します。	<b>INTO 文節の欠落:</b> マイグレーション・ツールは、into 文節の項目名を EZE_UNKNOWN_SQL_INT0 に設定して、エラー・メッセージを出します。
<b>INSERTCOLNAME 文節の欠落:</b> マイグレーション・ツールは、レコード内での項目の出現順と同じ順序でレコードにある SQL 列名をリストして、VAGen ADD 関数の挿入される列名のリストを作成します。マイグレーション・ツールは、読み取り専用として指定されている項目の SQL 列名を省略します。	<b>INSERTCOLNAME 文節の欠落:</b> マイグレーション・ツールは、リストの SQL 列名を EZE_UNKNOWN_SQL_INSERTCOLNAME に設定して、エラー・メッセージを出します。
<b>VALUES 文節の欠落:</b> マイグレーション・ツールは、レコード内での項目の出現順と同じ順序でレコードにある項目名すべてをリストして、VAGen ADD 関数の values 文節を作成します。マイグレーション・ツールは、読み取り専用として指定されている項目の項目名を省略します。	<b>VALUES 文節の欠落:</b> マイグレーション・ツールは、values 文節の項目名を EZE_UNKNOWN_SQL_VALUES に設定して、エラー・メッセージを出します。

表 39. SQL 入出力と SQL 文節の欠落 (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>FORUPDATEOF 文節の欠落:</b> マイグレーション・ツールは、レコード内での項目の出現順と同じ順序でレコードにある SQL 列名をリストして、for update of 文節を作成します。マイグレーション・ツールは、EGL keyItems プロパティに含まれている項目、および読み取り専用として指定されている項目の SQL 列名を省略します。</p>	<p><b>FORUPDATEOF 文節の欠落:</b> マイグレーション・ツールは、for update of 文節の SQL 列名を EZE_UNKNOWN_SQL_FORUPDATEOF に設定して、エラー・メッセージを出します。</p>
<p><b>SET 文節の欠落:</b> SET 文節は必要ありません。REPLACE 入出力ステートメントに SET 文節が欠落している場合、このステートメントはデフォルトの SQL 置換です。マイグレーション・ツールは SET 文節を作成しません。</p>	<p><b>SET 文節の欠落:</b> マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>WHERE 文節の欠落:</b> WHERE 文節は必要ありません。マイグレーション・ツールは where 文節を作成しません。</p>	<p><b>WHERE 文節の欠落:</b> マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>ORDERBY 文節の欠落:</b> ORDERBY 文節は必要ありません。マイグレーション・ツールは order by 文節を作成しません。</p>	<p><b>ORDERBY 文節の欠落:</b> マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>起こりうる問題:</b> 同名のレコードが 2 つあり (通常は、別のサブシステム内に)、これらのレコードの項目名または SQL 列名が異なる場合に限り、問題が起こります。</p> <p><b>考えられる解決策:</b> 別のサブシステム内で使用するための関数のコピーを作成し、正しい項目名と SQL 列名を使用するように新規関数を変更します。<b>欠点:</b> この入出力関数を使用する関数に影響が波及する可能性があります。</p>	<p><b>起こりうる問題 1:</b> 変更された SQL ステートメント、または Execution Time Statement Build を使用する SQL ステートメントがある場合は、問題が起こります。</p> <p><b>解決策 1A:</b> SQL 文節の欠落に関連したメッセージがあるかどうか、エラー・メッセージのリストを検討してください。SQL 入出力関数を変更して、欠落している文節を組み込みます。欠落している文節を作成するために必要な情報は、『関連パーツを使用したマイグレーション』欄の対応する行にあります。</p> <p><b>解決策 1B:</b> VisualAge Generator 内で関数を編集し、SQL エディターを使用して、行末へのブランクの追加など細かい変更を行います。SQL 文節を保管し、関数を再度マイグレーションします。マイグレーション・ツールが EGL 入出力ステートメントに SQL 表の情報を含めることができるように、レコード定義を必ず組み込んでください。</p> <p><b>起こりうる問題 2:</b> 『関連パーツを使用したマイグレーション』に示した問題と同じことが起こる可能性があり、同じ解決策が適用されます。</p>

## SQL 入出力と !itemColumnName

**VisualAge Generator:** SQL 入出力の場合、VisualAge Generator は SQL ステートメントの一部の文節で !itemColumnName の使用を許容します。テストおよび生成時に、SQL 行レコード内の項目名に対応する SQL 列名が決定されます。

**EGL:** !itemColumnName の使用はサポートされません。



**マイグレーションに必要な関連パーツ:** SQL レコードと、代替仕様レコードとして指定されたレコード (存在する場合)。

表 40. SQL 入出力と !itemColumnName

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
この関数の最初のマイグレーションを基準として、SQL レコードとその代替仕様レコードが使用可能ならば、マイグレーション・ツールは SQL レコードまたはその代替仕様レコード (存在する場合) に基づいて、!itemColumnNames を対応する SQL 列名に変換します。	SQL レコードまたはその代替仕様レコードが使用不可ならば、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>• 列名変数に対して !itemColumnNames を使用します。</li> <li>• 関数の検討が必要であることを示すエラー・メッセージを出します。</li> </ul>
<p><b>起こりうる問題:</b> 同名のレコードが 2 つあり (通常は、別のサブシステム内に)、これらのレコードの !itemColumnName に対応する SQL 列名が異なる場合に限り、問題が起こります。</p> <p><b>考えられる解決策:</b> 別のサブシステム内で使用するための関数のコピーを作成し、正しい SQL 列名を使用するように新規関数を変更します。<b>欠点:</b> この入出力関数を使用する関数に影響が波及する可能性があります。</p>	<p><b>起こりうる問題 1:</b> 変更された SQL ステートメント、または Execution Time Statement Build を使用する SQL ステートメントがある場合は、問題が起こります。</p> <p><b>解決策:</b> !itemColumnNames に関連したメッセージがあるかどうか、エラー・メッセージのリストを検討してください。SQL 入出力関数を変更して、SQL 行レコードに基づいた正しい列名を組み込みます。</p> <p><b>起こりうる問題 2:</b> 『関連パーツを使用したマイグレーション』に示した問題と同じことが起こる可能性があります、同じ解決策が適用されます。</p>

## 複数の更新を行う SQL 入出力

**VisualAge Generator:** SQL 入出力の場合、プログラム内に複数の UPDATE 関数または SETUPD 関数があれば、それぞれの SQL REPLACE 関数に、対応する UPDATE 関数または SETUPD 関数の名前を指定する必要があります。これは、非 SQL 入出力の場合は必要ありません。非 SQL 入出力の場合は、SETUPD はサポートされません。

**EGL:** SQL 入出力では、複数の get forUpdate ステートメントまたは open forUpdate ステートメントが存在する場合、それぞれの SQL replace ステートメントに、対応する get ステートメントまたは open ステートメントの名前を指定する必要があります。get ステートメントと open ステートメントはそれぞれ、resultSetID を指定します。replace ステートメントは、対応する get ステートメントまたは open ステートメントの resultSetID を指定します。resultSetID は、非 SQL 入出力の場合は適用されません。

**マイグレーションに必要な関連パーツ:** 入出力オブジェクトであるレコード。



表 41. 複数の更新を行う SQL 入出力

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>この関数の最初のマイグレーションを基準として、レコードが使用可能ならば、マイグレーション・ツールはレコード・タイプに基づいて対応する EGL ステートメントを作成します。</p> <p>SQL の場合、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>UPDATE 関数または SETUPD 関数をマイグレーションする場合は、常に resultSetID を組み込みます。resultSetID は、関数名とお客様指定の接尾部を使用して作成されます。</li> <li>対応する UPDATE または SETUPD の関数名を指定していた REPLACE 関数をマイグレーションする場合は、resultSetID を組み込みます。resultSetID は、対応する UPDATE または SETUPD の関数名と、お客様指定の接尾部を使用して作成されます。</li> </ul> <p>非 SQL の場合、マイグレーション・ツールは、UPDATE 関数または REPLACE 関数をマイグレーションする際に resultSetID を常に省略します。非 SQL 入出力の場合は、SETUPD 関数は存在しません。</p>	<p>UPDATE 関数をマイグレーションする際に、レコードが使用不可ならば、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>関数が SQL 入出力用かどうかの判別を試みます。このためには、関数に SQL 文節があるかどうか、または Execution Time Statement Build、single row select、cursor with hold など SQL 固有の情報があるかどうかを検査します。</li> <li>この UPDATE ステートメントが SQL レコード用であることをマイグレーション・ツールが判別できた場合、マイグレーション・ツールは get ステートメントに resultSetID を組み込みます。</li> <li>そうでなければ、マイグレーション・ツールは resultSetID を組み込みません。マイグレーション・ツールは警告メッセージを出します。</li> </ul> <p>SETUPD 関数をマイグレーションする場合は、SETUPD は SQL の場合のみ有効なので、マイグレーション・ツールは resultSetID を常に組み込みます。</p> <p>REPLACE 関数をマイグレーションする場合は、関数が対応する UPDATE または SETUPD の関数名を指定していれば、マイグレーション・ツールは resultSetID を組み込みます。</p>
<p>起こりうる問題: なし。</p>	<p>起こりうる問題: 未変更の UPDATE 関数が実際に SQL レコードを参照していて、複数の get または open forUpdate ステートメントが存在するプログラム内で使用されている場合に限り、問題が起こります。この場合、それぞれの replace ステートメントに resultSetID が組み込まれますが、VAGen UPDATE ステートメントに対応してマイグレーションされた get ステートメントには resultSetID が組み込まれません。プログラムの生成は失敗します。</p> <p>解決策: 関数を変更して、get ステートメントに resultSetID を組み込みます。</p>

## DL/I I/O および比較値項目

**VisualAge Generator:** DL/I I/O の場合、比較値項目が修飾されていない場合、VisualAge Generator は、現在のセグメント検索索引数 (SSA) に指定されたセグメントに対応するレコードを優先します。そのレコードから比較値項目が見つからない場合、VisualAge Generator は次に作業用ストレージ内、その次に I/O オブジェクトなどの他の DL/I セグメント・レコード内を検索します。関数のローカル・ストレージまたはパラメーター・リストにある項目は無視されます。関数のローカル・ストレージまたはパラメーター・リストにあるレコードが対象になりますが、プログラム内で一意な名前を持つ項目のみが考慮されます。

**EGL:** DL/I I/O の場合、比較値項目が修飾されていないと、EGL は標準の EGL 修飾ルールに従います。EGL はまず関数のローカル・ストレージ内またはパラメーター・リスト内の項目を検索し、次に関数のローカル・ストレージ内、パラメーター・リスト内、I/O オブジェクト内のレコードのフィールドを探し、最後にプログラム内のすべての変数を探します。

**マイグレーションに必要な関連パーツ:** DL/I セグメント・レコードと、代替仕様レコードとして指定されたレコード (存在する場合)。

表 42. DL/I I/O および比較値項目

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>この関数の最初のマイグレーションを基準として、比較値項目が修飾されていない場合には、マイグレーション・ツールは現在の SSA に関連する DL/I セグメント・レコードを検索します。DL/I セグメント・レコードとその代替仕様レコードが使用できる場合、マイグレーション・ツールは次のように比較値項目のレコードをチェックします。</p> <ul style="list-style-type: none"> <li>比較値項目が DL/I セグメント・レコード内にある場合、マイグレーション・ツールはその DL/I セグメント・レコード名を使って比較値項目を修飾します。</li> <li>比較値項目が DL/I セグメント・レコード内にない場合、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>レコード名として EZE_UNKNOWN_QUALIFIER を使います。</li> <li>マイグレーション・ツールが比較値項目の修飾を確定できないというメッセージを発行します。</li> </ul> </li> </ul>	<p>DL/I セグメント・レコードまたはその代替仕様レコードが使用不可な場合には、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>レコード名として EZE_UNKNOWN_QUALIFIER を使います。</li> <li>マイグレーション・ツールが比較値項目の修飾を確定できないというメッセージを発行します。</li> </ul>
<p><b>ありうる問題:</b> 比較値項目が修飾されていなくて、かつ関連する DL/I セグメント・レコードまたはその代替仕様レコード内にない場合にのみ、問題が発生します。</p> <p><b>考えられる解決策:</b> プログラム・ロジックを検討して、使用すべき適切な修飾を決定します。さらに、プログラムを生成した最後の時刻から、生成済み COBOL のソース・コードを検討できます。ある時期の VisualAge Generator で、比較値項目の修飾ルールが変化しました。したがって、最後のプログラム生成以降、VisualAge Generator の現行リリースが変更されていないという確信がない限り、現行のリリースを用いてプログラムを再生成することはしないでください。</p>	<p><b>起こりうる問題 1:</b> 修飾されていない比較値項目がある場合、問題が起こります。</p> <p><b>解決策:</b> DL/I I/O 関数を変更して、比較値項目を正しく修飾するようにします。まず、修飾ステートメントに関連する DL/I セグメント・レコードを必ずチェックしてください。</p> <p><b>起こりうる問題 2:</b> 『関連パーツを使用したマイグレーション』に示した問題と同じことが起こる可能性があり、同じ解決策が適用されます。</p>

## その他のステートメントの未確定状態の処理

### ステートメントの暗黙データ項目

**VisualAge Generator:** VisualAge Generator は、暗黙データ項目（レコード、マップ、テーブル、呼び出し先パラメーター・リスト、関数仮パラメーター・リスト、または関数ローカル・ストレージ内で明示的に定義されていない項目）の使用を許容します（ただし推奨されません）。

**EGL:** EGL は暗黙項目を許容しません。

**マイグレーションに必要な関連パーツ:** 適用外。

表 43. ステートメントの暗黙データ項目

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
86 ページの『プログラム内の暗黙データ項目』を参照	86 ページの『プログラム内の暗黙データ項目』を参照

### ステートメントのレベル 77 項目

**VisualAge Generator:** 作業用ストレージ・レコードのみがレベル 77 項目を含むことができます。プログラムは、1 次作業用ストレージ・レコード内のレベル 77 項目のみを参照できます。

**EGL:** レベル 77 項目は許容されません。

**マイグレーションに必要な関連パーツ:** 関数をマイグレーションする際には、作業用ストレージ・レコードが必要です。

表 44. ステートメントのレベル 77 項目

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
66 ページの『レコード内のレベル 77 項目』を参照	66 ページの『レコード内のレベル 77 項目』を参照

### 代入ステートメント

**VisualAge Generator:** 代入ステートメントは、レコードとマップに対して使用でき、「move corresponding」を行います。*MOVE* ステートメントは、項目に対して使用できます。

**EGL:** 代入ステートメントは、データ項目に対して使用するか、レコードのバイト単位での移動に使用することのみが可能です。代入ステートメントをマップに対して使用することはできません。レコードとマップの *move corresponding* には、*move byName* ステートメントが必要です。修飾子を指定しない *move* ステートメントを項目に対して使用できますが、代入ステートメントが推奨されます。

**マイグレーションに必要な関連パーツ:** 適用外。

表 45. 代入ステートメント

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>可能なかぎり多くの共通コードを保持するために、代入ステートメントまたは <code>move</code> ステートメントのソースとターゲットが両方とも非修飾で添え字なしの名前である場合、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>関数のパラメーター・リスト、ローカル・ストレージ、および入出力オブジェクトを検査して、代入ステートメントまたは <code>MOVE</code> ステートメントのソースまたはターゲットが項目、レコード、またはマップのどれであるか判別を試みます。マイグレーション・ツールによる判別が可能ならば、次のようにマイグレーションが行われます。 <ul style="list-style-type: none"> <li>ソースまたはターゲットが項目ならば、代入ステートメントへ。</li> <li>ソースまたはターゲットがレコードまたはマップならば、<code>move byName</code> ステートメントへ。</li> </ul> </li> <li>マイグレーション・ツールがパーツ型を判別できない場合、代入ステートメントと <code>MOVE</code> ステートメントは、修飾子なしの <code>move</code> ステートメントにマイグレーションされます。</li> </ul>	<p>この処理は、『関連パーツを使用したマイグレーション』欄の説明と同じように行われます。</p>
<p><b>起こりうる問題:</b> なし。テストおよび生成時に、修飾子のない <code>move</code> ステートメントは VAGen <code>MOVE</code> ステートメントに変換されます。移動の実際のソースとターゲットに応じて、項目間の移動、または <code>move byName</code> (<code>move corresponding</code>) が行われます。どのプログラムも、関数を変更せずに使用できます。</p>	<p><b>起こりうる問題:</b> なし。『関連パーツを使用したマイグレーション』欄の説明と同じ状況が当てはまります。</p>

## FIND ステートメント

**VisualAge Generator:** FIND ステートメントの検索列はオプションです。デフォルトは、VAGen テーブルの先頭列です。

**EGL:** FIND ステートメントは、`if` ステートメントに置き換わります。検索列は必須です。

**マイグレーションに必要な関連パーツ:** VAGen テーブル。

表 46. FIND ステートメント

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>この関数の最初のマイグレーションを基準として、検索列が明示的に指定されておらず、テーブルが使用可能ならば、マイグレーション・ツールはテーブルを展開して、テーブルの先頭列から検索列の名前を取得します。</p>	<p>検索列が明示的に指定されておらず、テーブルが使用できなければ、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>検索列を次のように設定します。 <code>EZE_UNKNOWN_SEARCH_COLUMN</code></li> <li>正しい列名を指定して関数を変更する必要があることを示すエラー・メッセージを出します。</li> </ul>

表 46. FIND ステートメント (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題:</b> 2 つの dataTable (異なるサブシステム内にあると考えられる) の dataTable 名が同じで、検索列名が異なる場合に限り、問題が起こります。</p> <p><b>解決策:</b> 第 2 のサブシステムに対して、dataTable の先頭列の副構造としてフィールドを追加します。この新規フィールドの名前は、第 1 のサブシステムの検索列と同じであることが必要です。この技法により、第 2 のサブシステム内でコードを変更せずに、共通関数を共用できます。</p>	<p><b>起こりうる問題 1:</b> 検索列名を指定する必要があります。「問題」ビューにエラーが示されます。</p> <p><b>解決策:</b> 関数を編集し、dataTable の正しい列名を指定します。</p> <p><b>起こりうる問題 2:</b> 『関連パーツを使用したマイグレーション』に示した問題と同じことが起こる可能性があり、同じ解決策が適用されます。</p>

## RETR ステートメント

**VisualAge Generator:** RETR ステートメントの検索列と戻す列はオプションです。検索列のデフォルトは、VAGen テーブルの先頭列です。戻す列のデフォルトは 2 列目です。

**EGL:** RETR ステートメントは、if ステートメントに置き換わります。検索列と戻す列は必須です。

**マイグレーションに必要な関連パーツ:** VAGen テーブル。

表 47. RETR ステートメント

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>この関数の最初のマイグレーションを基準として、検索列または戻す列が明示的に指定されておらず、テーブルが使用可能ならば、マイグレーション・ツールはテーブルを展開して、次の情報を取得します。</p> <ul style="list-style-type: none"> <li>検索列の名前をテーブルの先頭列から。</li> <li>戻す列の名前をテーブルの 2 列目から。</li> </ul>	<p>検索列または戻す列が明示的に指定されておらず、テーブルが使用できなければ、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>検索列を次のように設定します。 EZE_UNKNOWN_SEARCH_COLUMN</li> <li>戻す列を次のように設定します。 EZE_UNKNOWN_RETURN_COLUMN</li> <li>正しい列名を指定して関数を変更する必要があることを示すエラー・メッセージを出します。</li> </ul>
<p><b>起こりうる問題:</b> 2 つの dataTable (異なるサブシステム内にあると考えられる) の dataTable 名が同じで、検索列または戻す列の名前が異なる場合に限り、問題が起こります。</p> <p><b>解決策:</b> 第 2 のサブシステムに対して、dataTable の先頭列の副構造としてフィールドを追加します。この新規フィールドの名前は、第 1 のサブシステムの検索列と同じであることが必要です。dataTable の 2 列目の副構造として、第 1 のサブシステムにある戻す列の名前を追加します。この技法により、第 2 のサブシステム内でコードを変更せずに、共通関数を共用できます。</p>	<p><b>起こりうる問題 1:</b> 検索列と戻す列の名前を指定する必要があります。欠落している列ごとに、「問題」ビューにエラーが示されます。</p> <p><b>解決策:</b> 関数を編集し、dataTable の正しい列名を指定します。</p> <p><b>起こりうる問題 2:</b> 『関連パーツを使用したマイグレーション』に示した問題と同じことが起こる可能性があり、同じ解決策が適用されます。</p>



## SET map PAGE ステートメント

**VisualAge Generator:** *SET map PAGE* は、表示マップと印刷マップの両方に使用されます。

**EGL:** 2 つの別個のステートメントを使用します。マップ名は指定しません。

- テキスト (表示) 書式の場合は *clearScreen()*
- 印刷書式の場合は *pageEject()*

**マイグレーションに必要な関連パーツ:** 装置タイプを判別するためにマップが必要です。マイグレーション・ツールは、使用可能ないずれかのマップ・グループにある、このマップ名をもつ最初のマップを使用します。プログラムのコンテキスト内でマイグレーションを行う場合、マイグレーション・ツールはプログラムのメインのマップ・グループのみを検索します。

表 48. *SET map PAGE* ステートメント

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>この関数の最初のマイグレーションを基準として、マップが使用可能ならば、マイグレーション・ツールは <i>SET map PAGE</i> を次のように変換します。</p> <ul style="list-style-type: none"><li>• テキスト書式の場合は <i>clearScreen()</i></li><li>• 印刷書式の場合は <i>pageEject()</i></li></ul> <p>またマイグレーション・ツールは、オリジナルのマップ名を示すコメントを組み込みます。</p>	<p>マップが使用不可ならば、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"><li>• <i>SET map PAGE</i> を <i>EZE_SETPAGE()</i> に変換します。</li><li>• オリジナルのマップ名を示すコメントを組み込みます。</li><li>• マップのタイプを識別できないことを示すエラー・メッセージを出します。</li></ul>
<p><b>起こりうる問題:</b> 関数のマイグレーション時に決定されたものと異なるマップ・タイプを使用するプログラムは、実行時に異なる振る舞いをする可能性があります。これは、<i>clearScreen</i> がテキスト書式のみに適用され、<i>pageEject</i> が印刷書式のみに適用されるからです。「問題」ビューにエラーは示されません。プログラムの生成は失敗しません。</p> <p><b>考えられる解決策:</b> 特定のターゲット環境では印刷を行い、その他の環境では常に表示マップを使用する場合は、EGL 関数を次のようなものに変更します。</p> <pre>if (sysVar.systemType is zosbatch)   pageEject(); else   clearScreen(); end</pre> <p>ご使用のシステムの具体的な詳細に応じて、トランザクション・コード、ユーザー IDなどを基に同様なロジックを使用できます。</p>	<p><b>起こりうる問題 1:</b> ステートメントを含む関数がプログラムで使用される場合、EGL による検証の結果、「問題」ビューにエラーが表示されます。関数がプログラムで使用されていない場合、「問題」ビューにメッセージは表示されません。</p> <p><b>解決策:</b> 関数を編集し、<i>EZE_SETPAGE()</i> をマップ・タイプに応じて <i>clearScreen()</i> または <i>pageEject()</i> のどちらかに変更します。</p> <p><b>起こりうる問題 2:</b> 『関連パーツを使用したマイグレーション』に示した問題と同じことが起こる可能性があり、同じ解決策が適用されます。</p>

## SET mapItem 属性

**VisualAge Generator:** VisualAge Generator は、プリンター・マップ上の変数と定数に対して、保護、強調表示、色などの属性を許容します。

**EGL:** 下線を例外として、EGL は印刷書式の属性をサポートしません。



マイグレーションに必要な関連パーツ: 適用外。

表 49. SET mapItem 属性

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>プリンター・マップのマイグレーション時に、マイグレーション・ツールは EGL がサポートしない印刷書式の属性を省略します。</p> <p>関数をマイグレーションする場合、マイグレーション・ツールは、マップが表示マップかプリンター・マップかに関係なく SET ステートメントをマイグレーションします。</p>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>起こりうる問題:</b> テキスト書式の場合、問題はありません。印刷書式に対して、色、強調表示、保護などの属性を設定するロジックが関数に組み込まれている場合に限り、問題が起こります。「問題」ビューにエラーが示されます。</p> <p><b>解決策:</b> 関数が印刷書式のみに対して使用されている場合は、関数を変更して set ステートメントを除去します。関数がテキスト書式と印刷書式の両方に使用されている場合は、印刷書式に対して使用するための関数のコピーを作成します。新規関数を変更して set ステートメントを除去し、この新規関数を印刷書式に対して使用します。<b>欠点:</b> set ステートメントを含む関数を使用する関数に、影響が波及する可能性があります。</p>	<p><b>起こりうる問題:</b> 『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があり、同じ解決策が適用されます。</p>

## IN がリテラルかスカラーかの検査

**VisualAge Generator:** VisualAge Generator は、IF ステートメントまたは WHILE ステートメントのデータ項目 IN がリテラルかスカラーかの検査をサポートします。この場合、VisualAge Generator は EZETST の値を設定し、等価かどうかの比較を行います。

**EGL:** EGL は、IN のデータ項目がリテラルかスカラーかの検査をサポートしません。

マイグレーションに必要な関連パーツ: 適用外。

表 50. IN がリテラルかスカラーかの検査

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>データ項目 IN がリテラルかどうかを検査する IF ステートメントまたは WHILE ステートメントに対して、マイグレーション・ツールは VAGen の動作と一致させるために次の処理を行います。</p> <ul style="list-style-type: none"> <li>• sysVar.arrayIndex を 0 に初期化するステートメントを追加します。</li> <li>• if ステートメントまたは while ステートメントを等価比較 (例: if a == "b") に変更します。</li> <li>• if または while の直後に、sysVar.arrayIndex を 1 に設定するステートメントを追加します。</li> </ul> <p>データ項目が別のデータ項目内にあるかどうかを検査する IF ステートメントまたは WHILE ステートメントに対して、マイグレーション・ツールは 2 番目のデータ項目が配列かスカラーかの判別を試みません。マイグレーション・ツールは、EGL の in 比較にマイグレーションします。(例: if a in b)</p>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>起こりうる問題:</b> 比較の対象がリテラルである場合、問題はありません。2 番目のデータ項目が実際にはスカラーである場合に限り、問題が起きます。この場合は、「問題」ビューにエラーが示されます。</p> <p><b>解決策:</b> if ステートメントまたは while ステートメントの前に sysVar.arrayIndex を 0 に初期化し、if ステートメントまたは while ステートメントの直後に sysVar.arrayIndex を 1 に設定するように、関数を変更します。また、in ではなく == を使用して比較を行うように、if ステートメントまたは while ステートメントを変更します。</p>	<p><b>起こりうる問題:</b> 『関連パーツを使用したマイグレーション』欄にリストした問題と同じことが起こる可能性があり、同じ解決策が適用されます。</p>

## SQL 項目とマップ項目が NULL かどうかの検査

**VisualAge Generator:** IF、WHILE、および TEST は、SQL 項目またはマップ項目が NULL かどうかの検査をサポートします。

**EGL:** SQL 項目は NULL かどうかを検査できます。マップ項目はブランクかどうかを検査できます。

**マイグレーションに必要な関連パーツ:** レコードまたはマップ。項目が修飾されていない場合は、プログラムと関連パーツすべてが必要です。

表 51. SQL 項目とマップ項目が NULL かどうかの検査

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>この関数の最初のマイグレーションを基準として、項目が修飾されていれば、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>修飾子を検査して、レコードまたはマップのどちらであるかを判別します。</li> <li>修飾子が SQL レコードである場合は、<i>null</i> の検査に変換します。</li> <li>修飾子がマップである場合は、<i>blanks</i> の検査に変換します。</li> </ul>	<p>マイグレーション・ツールは、項目のタイプの判別を次のように試みます。</p> <ul style="list-style-type: none"> <li>項目が修飾されていて、修飾子を使用できなければ、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>修飾子が関数の入出力オブジェクトでもあるかどうかを検査します。そうである場合、CONVERSE オプションと DISPLAY 入出力オプションがあれば、入出力オブジェクトがマップであることが保証されます。CLOSE 入出力オプションは、レコードまたはマップのどちらにも有効です。その他の入出力オプションがあれば、入出力オブジェクトはレコードであることが保証されます。</li> <li>関数のパラメーター・リストとローカル・ストレージも検査します。修飾子が検出された場合、修飾子はレコードです。</li> </ul> </li> <li>項目が SQL レコードまたはマップにあることをマイグレーション・ツールが判別できた場合、ツールは次のようにマイグレーションを行います。 <ul style="list-style-type: none"> <li>SQL レコードの場合は <i>null</i></li> <li>マップ項目の場合は <i>blanks</i></li> </ul> </li> <li>項目が SQL レコードまたはマップにあることをマイグレーション・ツールが判別できない場合、ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>EZE_NULL に変換します。</li> <li>このステートメントの検討が必要であることを示すエラー・メッセージを出します。</li> </ul> </li> </ul>
<p>この関数の最初のマイグレーションを基準として、項目が修飾されていない場合は、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>関数のパラメーター・リストを検査して、項目が SQLITEM パラメーターまたは MAPITEM パラメーターとしてこのリストに指定されているかどうかを調べます。該当する場合、ツールはその結果を基礎としてマイグレーションを行います。</li> <li>プログラムとその関連パーツが使用可能ならば、マイグレーション・ツールは VAGen 修飾規則を使用して、その項目を含むレコードまたはマップを判別し、その結果を基礎としてマイグレーションを行います。</li> </ul>	<p>項目が修飾されていない場合、マイグレーション・ツールは関数のパラメーター・リストを検査して、項目が SQLITEM または MAPITEM のどちらとして指定されているかを調べます。</p> <p>項目が SQL レコードまたはマップにあることをマイグレーション・ツールが判別できた場合、ツールは次のようにマイグレーションを行います。</p> <ul style="list-style-type: none"> <li>SQL レコードの場合は <i>null</i></li> <li>マップ項目の場合は <i>blanks</i></li> </ul> <p>項目が SQL レコードまたはマップにあることをマイグレーション・ツールが判別できない場合、ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>EZE_NULL に変換します。</li> <li>このステートメントの検討が必要であることを示すエラー・メッセージを出します。</li> </ul>

表 51. SQL 項目とマップ項目が NULL かどうかの検査 (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
起こりうる問題: なし。	<p>起こりうる問題 1: マイグレーション・ツールは EZE_NULL を使用している場合は、問題が起こります。「問題」ビューにエラーが示されます。</p> <p>解決策: 関数を編集し、EZE_NULL を SQL 項目の場合は <i>null</i> に変更し、書式変数フィールドの場合は <i>blanks</i> に変更します。</p>

## 入出力エラー値 UNQ および DUP

**VisualAge Generator:** UNQ と DUP は、非 SQL の場合は常にソフト・エラーであり、SQL の場合はハード・エラーです。SQL の場合、UNQ と DUP は -803 SQL コードに基づいて常に設定されます。関数に対して入出力エラー・ルーチンが指定されていると、エラー・ルーチンは次の場合に制御を獲得します。

- すべてのソフト・エラー
- すべてのハード・エラー (EZEFECC = 1 の場合)
- DL/I I/O の場合、すべてのハード・エラー (EZEDLERR または EZEFECC = 1 のとき)

**EGL:** duplicate は常にソフト・エラーであり、入出力が成功したことを示します。unique は常にハード・エラーであり、入出力が失敗したことを示します。SQL の場合、duplicate はサポートされません。try ブロックと onException ステートメントがエラー処理に使用されます。onException ステートメントが入出力ステートメントに対して指定されていると、onException ステートメントは次の場合に制御を獲得します。

- すべてのソフト・エラー
- すべてのハード・エラー (handleHardIOErrors = 1 の場合)
- DL/I I/O の場合、すべてのハード・エラー (handleHardDLIErrors または handleHardIOErrors = 1 のとき)

**マイグレーションに必要な関連パーツ:** ステートメント内で使用されているレコード。

表 52. 入出力エラー値 UNQ および DUP

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>この関数の最初のマイグレーションを基準として、レコードが使用可能ならば、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>レコードが非 SQL であれば、マイグレーション・ツールは DUP を <i>duplicate</i> に変更し、UNQ を <i>unique</i> に変更します。</li> <li>レコードが SQL であれば、マイグレーション・ツールは DUP と UNQ を両方とも <i>unique</i> に変更します。</li> </ul>	<p>レコードが使用不可ならば、マイグレーション・ツールはレコードのタイプを次のように判別します。</p> <ul style="list-style-type: none"> <li>ステートメントが関数の入出力オブジェクトと同じレコードを指定していれば、マイグレーション・ツールは関数に SQL 文節があるかどうか、または Execution Time Statement Build、single row select、cursor with hold、UPDATE/SETUPD 関数など、SQL 固有の情報があるかどうかを検査します。該当する場合、マイグレーション・ツールはレコードが SQL であることを想定し、DUP と UNQ を <i>unique</i> に変換します。</li> <li>次のように、その他の状況ではマイグレーション・ツールはレコード・タイプを判別できません。 <ul style="list-style-type: none"> <li>レコードが関数の入出力オブジェクトとして使用されていて、関数に SQL 固有の情報がない場合。</li> <li>レコードが関数の入出力オブジェクトとして使用されていない場合。</li> </ul> </li> </ul> <p>前述の状況や、マイグレーション・ツールがレコード・タイプを判別できないその他の状況では、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>UNQ を <i>unique</i> に変換します。</li> <li>DUP を <i>EZE_DUPLICATE</i> に変換し、エラー・メッセージを出します。</li> </ul>

表 52. 入出力エラー値 UNQ および DUP (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題:</b> 同じレコード名に対して、SQL 用に 1 つ、非 SQL 用に 1 つの異なる定義が存在する (通常、異なるサブシステム内で) 場合に限り、問題が起こります。関数のマイグレーション時に非 SQL レコードが使用可能であれば、その関数を SQL レコードに対して使用して重複の検査を行う場合にエラーが発生します。関数のマイグレーション時に SQL レコードが使用可能であれば、重複検査によって得られる追加情報を非 SQL レコードに対して使用できません。</p> <p><b>考えられる解決策:</b> 関数をコピーして、オリジナルの関数を SQL に対して使用し、新規関数を非 SQL に対して使用します。<b>欠点:</b> UNQ または DUP の検査を行っていたオリジナルの関数を使用する関数に、影響が波及する可能性があります。</p> <p><b>SQL の場合に起こりうる問題:</b> なし。DUP と UNQ は常に同じ方法で設定され、<i>unique</i> は引き続きハード・エラーになります。</p> <p><b>非 SQL の場合に起こりうる問題 1:</b> プログラムに対して <code>handleHardIOErrors (EZEFEFC) = 1</code> を設定していない場合に問題が起こります。この場合、<i>unique</i> はハード・エラーになるので、<code>onException</code> ステートメントは制御を獲得せず、プログラムは終了します。</p> <p><b>解決策:</b> プログラムが <code>handleHardIOErrors = 1;</code> を指定するようにします。</p> <p><b>非 SQL の場合に起こりうる問題 2:</b> <code>hardIOError (HRD)</code> を明示的にテストしている場合にも、問題が起こります。この場合、<i>unique</i> はハード・エラーになるので、従来の VisualAge Generator では <code>hardIOError</code> のテスト結果が <code>true</code> にならなくても、EGL ではテスト結果が <code>true</code> になる場合があります。検証と生成時にエラーは検出されません。ただし、プログラムは <i>VisualAge Generator</i> の場合と異なる動作をする可能性があります。</p> <p><b>考えられる解決策:</b> プログラム・ロジックの中で入出力エラー値のテストを再配列する必要が生じる場合があります。</p>	<p><b>起こりうる問題 1:</b> EZE_DUPLICATE は EGL では無効です。</p> <p><b>解決策:</b> 関数を編集し、レコード・タイプに応じて EZE_DUPLICATE を <i>duplicate</i> または <i>unique</i> に変更します。</p> <p><b>起こりうるその他の問題:</b> 『関連パーツを使用したマイグレーション』に示した問題と同じことが起こる可能性があります。同じ解決策が適用されます。</p>

## 入出力エラー値 LOK

**VisualAge Generator:** LOK は、OS/400® の場合は常にソフト・エラーです。関数に対して入出力エラー・ルーチンが指定されていると、エラー・ルーチンは次の場合に制御を獲得します。

- すべてのソフト・エラー
- すべてのハード・エラー (EZEFEFC = 1 の場合)



**EGL:** *LOK* は *deadlock* に置き換えられますが、ハード・エラーです。*try* ブロックと *onException* ステートメントがエラー処理に使用されます。*onException* ステートメントが入出力ステートメントに対して指定されていると、*onException* ステートメントは次の場合に制御を獲得します。

- すべてのソフト・エラー
- すべてのハード・エラー (*handleHardIOErrors* = 1 の場合)

マイグレーションに必要な関連パーツ: 適用外。

表 53. 入出力エラー値 *LOK*

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
マイグレーション・ツールは、 <i>LOK</i> を常に <i>deadlock</i> に変更します。	マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。
<p><b>起こりうる問題 1:</b> プログラムに対して <i>handleHardIOErrors</i> (<i>EZEFECE</i>) = 1 を設定していない場合に問題が起こります。この場合、<i>deadlock</i> はハード・エラーなので、<i>onException</i> ステートメントは制御を獲得せず、プログラムは終了します。</p> <p><b>解決策:</b> プログラムが <i>handleHardIOErrors</i> = 1; を指定するようにします。</p> <p><b>起こりうる問題 2:</b> <i>hardIOError</i> (<i>HRD</i>) を明示的にテストしている場合にも、問題が起こります。この場合、<i>deadlock</i> はハード・エラーなので、従来の <i>VisualAge Generator</i> では <i>hardIOError</i> のテスト結果が <i>true</i> にならなくても、<i>EGL</i> ではテスト結果が <i>true</i> になることがあります。検証と生成時にエラーは検出されません。ただし、プログラムは <i>VisualAge Generator</i> の場合と異なる動作をする可能性があります。</p> <p><b>考えられる解決策:</b> プログラム・ロジックの中で入出力エラー値のテストを再配列する必要がある場合があります。</p>	『関連パーツを使用したマイグレーション』欄で説明した問題と同じことが起こる可能性があります。同じ解決策を使用できます。

## EZE ワードの未確定状態の処理

一部の EZE ワードを置換するためには、余分の項目変数を EGL プログラム内で宣言する必要があります。余分の項目変数は、関数内のローカル項目変数として宣言されることはありません。これは、ローカル・ストレージ、パラメーター、または戻り値をもつセグメント化対話に至るスタック内で関数が開いている場合に、セグメント化対話を行うことができないからです。余分の項目変数をプログラムに追加すれば、セグメント化対話の中断を回避できます。

## EZELTERM

**VisualAge Generator:** EZELTERM は Web トランザクション・プログラム内の会話 ID であり、他のすべてのプログラム・タイプでは端末 ID です。

**EGL:** *sysVar.conversationID* は *VGWebTransaction* プログラム内の会話 ID です。*sysVar.terminalID* はその他すべてのプログラム・タイプの端末 ID です。

`sysVar.conversationID` と `sysVar.terminalID` は同義語として扱われるので、いずれを使用してもプログラム・タイプに応じて適切な情報が提供されます。

マイグレーションに必要な関連パーツ: プログラム。

表 54. EZELTERM

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
この関数の最初のマイグレーションを基準として、プログラムが使用可能であれば、マイグレーション・ツールは次のようにプログラム・タイプに応じて EZELTERM を変換します。 <ul style="list-style-type: none"><li>プログラムが Web トランザクション・プログラムである場合、マイグレーション・ツールは <code>sysVar.conversationID</code> を使用します。</li><li>プログラムが Web トランザクション・プログラムでない場合、マイグレーション・ツールは <code>sysVar.terminalID</code> を使用します。</li></ul>	プログラムが使用不可であれば、マイグレーション・ツールは常に EZELTERM を次のように変換します。 <code>sysVar.terminalID</code>
<b>起こりうる問題:</b> なし。 <code>sysVar.conversationID</code> と <code>sysVar.terminalID</code> は同義語として扱われます。	<b>起こりうる問題:</b> なし。 <code>sysVar.conversationID</code> と <code>sysVar.terminalID</code> は同義語として扱われます。

## EZESYS

**VisualAge Generator:** EZESYS は、VisualAge Generator によって指定されるリテラル値を含む IF、WHILE、および TEST の各ステートメント内で一般に使用されます。ただし、EZESYS はその他のステートメント内でも許容されます。

**EGL:** EGL システム変数 `sysVar.systemType` の値は、VisualAge Generator とは異なります。IF、WHILE、および TEST 以外のステートメント内で EZESYS が使用されている場合、マイグレーション・ツールはプログラムが予期している値を判別できないので、オリジナルの VAGen 値を使用する必要があります。EGL システム・ライブラリー関数 `VGLib.getVGSystemType` は、以前の VAGen 値を提供します。

マイグレーションに必要な関連パーツ: 適用外。

表 55. EZESYS

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>プログラムをマイグレーションするとき、VAGen マイグレーション設定「以前の EZESYS 値を初期化しない (Do not initialize old EZESYS values)」が選択されていないとマイグレーション・ツールは以下を実行します。</p> <ul style="list-style-type: none"> <li>• <code>custPrefixEZESYS</code> の宣言を組み込みます。</li> <li>• <code>custPrefixEZESYS</code> の値を以前の VAGen EZESYS 値に設定する初期化ステートメントを組み込みます。</li> </ul> <p>この設定が選択されている場合、マイグレーション・ツールは宣言または初期化ステートメントを組み込みません。</p> <p><code>custPrefix</code> は、予約語と競合するパーツ名の変更に使用されるものと同じ接頭部です。この値を設定するには、VAGen マイグレーション設定を使用します。</p>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p>この関数の最初のマイグレーションを基準として、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• EZESYS が IF、WHILE、または TEST のいずれかのステートメント内で使用されている場合、マイグレーション・ツールは EZESYS を次のように変換します。</li> </ul> <p><code>sysVar.systemType</code></p> <p>マイグレーション・ツールは、EZESYS 値を等価な EGL 値に変換します。EZESYS 値に等価な EGL 値がない場合、マイグレーション・ツールはその値を「現状のまま」マイグレーションします。例えばマイグレーション・ツールは、MVS BATCH を EGL で同等の <code>zosbatch</code> に変換します。マイグレーション・ツールは、OS2 と NTCICS を VisualAge Generator の場合と同じ値にマイグレーションします。変換される値について詳しくは、335 ページの表 117 を参照してください。</p> <ul style="list-style-type: none"> <li>• EZESYS がその他のステートメント内で使用されている場合、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>– 以前の VAGen EZESYS 値が結果として使用されることを示す警告メッセージを出します。</li> <li>– 次の値を使用して、 <code>custPrefixEZESYS</code> ステートメント内の EZESYS を置き換えます。</li> </ul> </li> </ul>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>

表 55. EZESYS (続き)

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p><b>起こりうる問題 1:</b> マイグレーション中に、VAGen マイグレーション設定「以前の EZESYS 値を初期化しない (<i>Do not initialize old EZESYS values</i>)」を選択し、IF、WHILE、または TEST 以外のステートメントで EZESYS を使用する場合に問題が起こります。この状態では、マイグレーション済みツールはステートメントで <i>custPrefixEZESYS</i> を使用しますが、プログラムには <i>custPrefixEZESYS</i> の宣言および初期化ステートメントがありません。「問題」ビューにエラーが示されます。</p> <p><b>考えられる解決策 1A:</b> EGL ロジックを変更して、<i>sysVar.systemType</i> の新規の値を使用します。</p> <p><b>考えられる解決策 1B:</b> <i>custPrefixEZESYS</i> の宣言および初期化ステートメントを、EZESYS の以前の VAGen 値の使用が必要なプログラムに追加します。</p> <p><b>起こりうる問題 2:</b> 同等な EGL 値 (TSO、AIXCICS など) がない状態で現状のままマイグレーションされた EZESYS 値に関して、問題が起こります。「問題」ビューにエラーが示されます。</p> <p><b>考えられる解決策 2:</b> 関数を修正して、EGL では無効な値があるかどうか <i>sysVar.systemType</i> をチェックしないようにロジックを変更します。</p> <p><b>起こりうる問題 3:</b> if と while 以外のステートメント内で新規の EGL 値を使用すると、問題が起こります。</p> <p><b>考えられる解決策 3:</b> 関数を修正して、以下の代わりに <i>sysVar.systemType</i> を使用するようにロジックを変更します。</p> <p><i>custPrefixEZESYS</i></p> <p>比較に使用するすべての <i>dataTable</i> 内で、以前の VAGen 値を新規の EGL 値に必ず変更してください。</p>	<p>『関連パーツを使用したマイグレーション』欄で説明した問題と同じことが起こる可能性があります。同じ解決策を使用できます。</p>

## EZEWAIT

**VisualAge Generator:** EZEWAIT は、待機秒数を 100 分の 1 秒単位で指定します。

**EGL:** *sysLib.wait* が EZEWAIT の置換表現で、待機時間を秒数単位で指定します。

**マイグレーションに必要な関連パーツ:** 適用外。

表 56. EZEWAIT

関連パーツを使用したマイグレーション	関連パーツを使用しないマイグレーション
<p>プログラムをマイグレーションする際に、マイグレーション・ツールは常に次の宣言を組み込みます。</p> <pre>custPrefixEZE_WAIT_TIME.</pre>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p>関数をマイグレーションする際に、EZEWAIT が使用されていれば、マイグレーション・ツールは待機時間を秒数単位で計算し、結果を以下に格納するロジックを組み込みます。</p> <pre>custPrefixEZE_WAIT_TIME.</pre>	<p>マイグレーション・ツールは、『関連パーツを使用したマイグレーション』欄の説明と同じ処理を行います。</p>
<p><b>起こりうる問題:</b> なし。ただし、関数を新規プログラム内で使用する場合は、次の宣言をプログラムに必ず組み込んでください。</p> <pre>custPrefixEZE_WAIT_TIME</pre>	<p>『関連パーツを使用したマイグレーション』欄で説明した問題と同じことが起こる可能性があります。</p>





---

## 第 2 部 VisualAge Generator 4.5 on Java から EGL へのマイグレーション



---

## 第 4 章 ステージ 1 — Java からの抽出

ソース・コードを VisualAge Generator から抽出する前に、VisualAge for Java 上で稼働するステージ 1 マイグレーション・ツールをインストールする必要があります。また、VisualAge Generator 4.5 (VAGen 4.5) から EGL にマイグレーションするデータの保管に使用される、DB2 マイグレーション・データベースも作成する必要があります。

---

### VisualAge for Java へのステージ 1 マイグレーション・ツールのインストール

VisualAge Generator to EGL ステージ 1 マイグレーション・ツールは、VAGenMigJava.exe という名前の自己解凍ファイルとして提供されます。このファイルをインストールするには、次の手順で行います。

1. フィックスパック 4 を適用して VisualAge Generator 4.5 にアップグレードする。また、APAR PQ88461 をインストールして、VisualAge Generator のユーティリティー機能をアップグレードする必要があります。APAR の修正を入手するには、IBM サポートにご連絡いただくか、VAGen Web サイト (<http://www.ibm.com/software/awdtools/visgen/support/>) にアクセスして、「Download」セクションのリンクをたどってください。また、ユーザー個々の状況に応じてさらに必要になる可能性がある VisualAge Generator APAR については、439 ページの『付録 F. 外部ソース形式の誤りが原因で EGL の作成に問題が生じる状態』を参照してください。

注: フィックスパック 5 には、マイグレーション・ツールに必要なすべての APAR が含まれる予定です。

2. 使用しているシステム上で、VisualAge for Java がインストールされている場所を判別する。
3. VisualAge for Java をシャットダウンする。
4. 自己解凍ファイル VAGenMigJava.exe を実行する。このファイルは、Rational Developer のインストール・ディレクトリーの次に示すサブディレクトリーにあります。

¥bin

注: 現在ご使用の製品をインストールする前に、前のバージョンの開発者製品をインストールし、保持していた場合、対象のインストール・ディレクトリーは、以前のインストール時に使用されたディレクトリーである可能性があります。

5. GUI プロンプトが表示されたら、VisualAge for Java がインストールされているドライブとディレクトリーにナビゲートする (例えば、c:¥Program Files¥IBM¥VisualAge for Java)。次に、「抽出 (Extract)」をクリックする。

自己解凍型の実行可能ファイルが実行されると、次のファイルが VisualAge for Java インストール・ディレクトリーに抽出されます。

- %vgmigration%MigPreferences.xml
- %vgmigration%VGMigReserved.txt
- %vgmigration%checkStage1.sql
- %vgmigration%createdatabase.sql
- %vgmigration%createtables.sql
- %vgmigration%checkStage1.bat
- %vgmigration%SetupDatabase.bat
- %vgmigration%SetupTables.bat
- %vgmigration%deletemigsets.bat
- %features%com-ibm-vgj-mig%

この最後のディレクトリーは、Java 上でのステージ 1 マイグレーション・ツール用のフィーチャーを格納しています。また、Java 上でのステージ 1 マイグレーション・ツールに使用される .xml ファイル、および対応する .dtd ファイルも格納しています。

## マイグレーション・フィーチャーの追加

ステージ 1 マイグレーション・ツールを使用するには、IBM VisualAge Generator EGL マイグレーション・フィーチャーを追加する必要があります。このためには、次の手順で行います。

1. VisualAge Generator on Java を開始する。
2. 次のようにして、「IBM VisualAge Generator EGL マイグレーション (IBM VisualAge Generator EGL Migration)」フィーチャーを追加する。
  - a. 「ワークベンチ」ウィンドウで、F2 を押す。
  - b. 左側の列から「フィーチャー」を選択し、右側の列にある「フィーチャーの追加」を選択する。「OK」をクリックする。
  - c. 「IBM VisualAge Generator EGL マイグレーション - バージョン番号 (IBM VisualAge Generator EGL Migration - versionNumber)」を選択する。「OK」をクリックする。マイグレーション・フィーチャーがロードされます。
  - d. ワークベンチの「プロジェクト」タブをクリックする。「IBM VisualAge Generator EGL マイグレーション (IBM VisualAge Generator EGL Migration)」プロジェクトがワークスペースに表示されます。

## マイグレーション・データベースの作成

マイグレーション・データベースの作成については、441 ページの『DB2 マイグレーション・データベースの作成』を参照してください。VisualAge Java インストール・ディレクトリーの %vgmigration サブディレクトリーにある、SetupDatabase.bat ファイルと SetupTables.bat ファイルを使用する必要があります。

## ステージ 1 設定の実行

ステージ 1 マイグレーション・ツールを VisualAge for Java にインストールするときに、インストール・プロセスによってサンプル設定ファイル `MigPreferences.xml` がディレクトリ `VisualAge-Java-installation-directory\ide\vgmigration` に作成されます。設定を変更する前に、バックアップのために `MigPreferences.xml` ファイルのコピーを作成しておく必要があります。`MigPreferences.xml` を VisualAge for Java インストール・ディレクトリの外側のディレクトリにコピーして、そのコピーに対して変更を加えることもできます。これによって、新規バージョンのマイグレーション・ツールをインストールした場合に変更内容を誤って上書きしてしまうことを防止できます。

テキスト・エディター、またはステージ 1 マイグレーション・ツールに付属の GUI エディターを使用して、`MigPreferences.xml` ファイルを編集できます。GUI エディターを使用するには、次の手順で行います。

1. VisualAge Generator for Java を開始する。
2. 「ワークベンチ」ウィンドウの「プロジェクト」タブをクリックする。
3. 「IBM VisualAge Generator EGL マイグレーション (IBM VisualAge Generator EGL Migration)」プロジェクトにナビゲートする。
4. マイグレーション・プロジェクトを展開し、パッケージ `com.ibm.vgj.mig` を展開する。
5. このパッケージ内で、「**PreferencesUI**」クラスを選択する。
6. 「**PreferencesUI**」クラスを右クリックし、コンテキスト・メニューから「プロパティ」をクリックする。
7. 「プログラム」タブを選択する。
8. 「プログラム」ページの「コマンド行引数 (Command line arguments)」フィールドに次のように指定して、編集する `MigPreferences.xml` ファイルを指示する。

```
-p filename
```

ここで、`filename` は `MigPreferences.xml` ファイルのドライブ、ディレクトリ、およびファイル名です。

9. 「OK」をクリックしてプロパティを保管する。
10. 「**PreferencesUI**」を右クリックし、「実行」または「メインを実行... (Run main...)」をクリックする (または、ツールバーの走る人のアイコンをクリックすることもできます)。ステージ 1 GUI 設定エディターが開き、プログラムのプロパティ内で指定したファイルがロードされます。

注:

- 現在、Project List Parts (PLPs) を使用していない場合は、137 ページの『マイグレーション・プランと上位 PLP プロジェクト』を参照してください。
- ドライブとディレクトリを必要とする設定の場合は、次に示す 2 つの方法のどちらかで情報を指定できます。
  - 絶対パス。例: `d:\tempMig\MySystem\`
  - 相対パス。この場合、パスは作業ディレクトリに対する相対パスです。例えば、`..\tempMig\MySystem` は次のパスに相当します。

```
VisualAge-Java-installation-directory¥ide¥project_resources
¥IBM VisualAge Generator EGL Migration¥tempMig¥MySystem
```

- ログ・ファイル、デバッグ・ファイル、およびレポート・ファイル用のドライブとディレクトリーを指定しない場合、ファイルは次の作業ディレクトリーに書き込まれます。

```
VisualAge-Java-installation-directory¥ide¥project_resources
¥IBM VisualAge Generator EGL Migration
```

以降では、変更できる設定について、GUI 内でその設定が表示されるページ別に説明します。

- プランの作成 (Build Plans)
- マッピング (Mapping)
- 名前変更
- 実行 (Execution)

## 「プランの作成 (Build Plans)」 ページ

「プランの作成 (Build Plans)」 ページでは、ステージ 1 マイグレーション・ツールがマイグレーション・プラン・ファイルの読み取りまたは書き込みを行う場所、およびリポジトリからマイグレーションするプロジェクトとバージョンを指定します。

- **マイグレーション仕様 (Migration Specification)**。マイグレーション仕様は、ステージ 1 ツールがリポジトリ・フィルターに基づいて作成したマイグレーション・プラン・ファイルを、マイグレーション・ツールが書き込む場所を指定します。また、マイグレーション・プラン・ファイルをすでに作成してある場合、マイグレーション仕様はマイグレーション・ツールがマイグレーション・プラン・ファイルを読み取る場所を指定します。

注:

- マイグレーション・プラン・ファイルのファイル拡張子は、マイグレーション・データベースのロードに使用される前は *.pln*、正常に処理された後は *.done* になります。
- VAGenToEGLMigration クラス (実際のステージ 1 マイグレーション・ツール) に対する *-o* (オーバーライド) オプションの設定については、136 ページの『ステージ 1 ツールの実行』を参照してください。
- プラン・ディレクトリー (*Plan directory*)。これは、ステージ 1 マイグレーション・ツールがマイグレーション・プラン・ファイルを配置するターゲット・ディレクトリー、またはステージ 1 ツールが既存のマイグレーション・プラン・ファイルを検索するターゲット・ディレクトリーです。
- プラン・ファイル名 (*Plan file name*)。マイグレーション・データベースをロードするために作成または使用する、マイグレーション・プラン・ファイルのファイル名 (オプション)。ステージ 1 マイグレーション・ツールの実行時には、VAGenToEGLMigration クラスに対して指定した *-o* (オーバーライド) オプションとの組み合わせにより、このファイル名が次のように使用されます。
- VAGenToEGLMigration クラスのプロパティに *-o* オプションを組み込んだ場合、ステージ 1 マイグレーション・ツールは、マイグレーション仕様に指定したファイル名に基づいて次の処理を行います。



- プラン・ファイル名を指定しない場合は、プラン・ファイルを新規に作成する前に、指定したプラン・ディレクトリーにある**すべての** .pln ファイルがマイグレーション・ツールによって削除されます。マイグレーション・ツールは、それぞれのマイグレーション・セットごとにプラン・ファイルを 1 つずつ作成します。この場合、マイグレーション・プラン・ファイル名の形式は `migrationSetName_version.pln` です。
- プラン・ファイル名を指定した場合は、指定したプラン・ファイル名の .pln ファイルを新規に作成する前に、指定した .pln ファイルの**みが**、指定したプラン・ディレクトリーからマイグレーション・ツールによって削除されます。この場合は、単一のプラン・ファイルにマイグレーション・セットがすべてリストされます。

リポジトリ・フィルターと上位 PLP プロジェクトに基づいて、マイグレーション・プラン・ファイルをステージ 1 マイグレーション・ツールに作成させたい場合は、`-o` オプションを使用します。PLP プロジェクト作成のために支援が必要な場合は、138 ページの『上位 PLP プロジェクトの作成』を参照してください。

- VAGenToEGLMigration クラスのプロパティから `-o` オプションを省略した場合、ステージ 1 マイグレーション・ツールはマイグレーション・プラン・ファイルを新規に作成しません。代わりに、ステージ 1 マイグレーション・ツールは、マイグレーション仕様に指定したプラン・ディレクトリーとプラン・ファイル名に基づいて次のように実行されます。
  - プラン・ファイル名を指定しない場合は、指定したプラン・ディレクトリー内にある**すべての** .pln ファイルを使用して、マイグレーション・ツールが実行されます。
  - プラン・ファイル名を指定した場合は、指定したプラン・ディレクトリー内にあるその .pln ファイル 1 つだけを使用して、マイグレーション・ツールが実行されます。

マイグレーション・プラン・ファイルがすでに作成済みで、これらのファイルを使用してステージ 1 マイグレーション・ツールを実行し、マイグレーション・データベースをロードする場合は、`-o` オプションを省略します。ユーザー独自のマイグレーション・プラン・ファイルの作成について詳しくは、139 ページの『マイグレーション・プラン・ファイルの手動作成』を参照してください。

- **リポジトリ・フィルター (Repository filters)**。「リポジトリ・フィルター (Repository Filters)」セクションを使用して、ステージ 1 マイグレーション・ツールによって処理する Java リポジトリのプロジェクトとバージョンを制御できます。プロジェクトとバージョンを制限することにより、ステージ 1 マイグレーション・ツールのパフォーマンスを大幅に向上できます。複数のフィルターを指定できます。ステージ 1 マイグレーション・ツールは、プロジェクト・フィルターと、バージョン深さ フィルターまたはバージョン名 フィルターを次のように使用します。
  - マイグレーション・ツールは、リポジトリ内の各 VAGen プロジェクトを、プロジェクト・フィルターと突き合わせます。
    - プロジェクト名がプロジェクト・フィルターのどれともマッチングしない場合、プロジェクトは以後の処理対象になりません。

- プロジェクト名がプロジェクト・フィルターの少なくとも 1 つとマッチングする場合、プロジェクトのバージョンは次のように処理されます。
  - バージョン深さフィルターを選択した場合は、プロジェクトの最新バージョン (バージョン深さフィルターに指定した数まで) が以後の処理対象になります。デフォルトのバージョン深さフィルターは 1 です。
  - バージョン名フィルターを選択した場合は、プロジェクトの各バージョン名と、バージョン名フィルターのリストの突き合わせが行われます。バージョン名がバージョン名フィルターのいずれかとマッチングする場合、そのバージョンは以後の処理対象になります。

**注:** バージョン深さとバージョン名は相互に排他的です。デフォルトでは、バージョン名フィルターが *MigPreferences.xml* ファイルに組み込まれています。バージョン深さフィルターを使用する場合は、「バージョン深さ (Version depth)」ラジオ・ボタンを選択し、マイグレーションするバージョンの数を指定します。

- プロジェクト名とバージョン名の突き合わせの結果、プロジェクト・バージョンが以降の処理対象になった場合、ステージ 1 マイグレーション・ツールは次の処理を行います。
  - プロジェクト・バージョンが上位 PLP プロジェクトならば、ステージ 1 マイグレーション・ツールはそのプロジェクト・バージョンをマイグレーション・セットの作成の基礎として使用します。バージョン名がバージョン・フィルターとマッチングすることを前提とすると、上位 PLP プロジェクトのそれぞれのバージョンごとに、異なるマイグレーション・セットが作成されます。
  - プロジェクト・バージョンが上位 PLP プロジェクトでなければ、プロジェクト・バージョンは以後の処理対象になりません。そのプロジェクト・バージョンは、他のマイグレーション・セットにも組み込まれている可能性があります (マイグレーション・セットは、単にこのプロジェクト・バージョン専用ではありません)。

リポジトリ・フィルター情報は、次のように指定します。

- プロジェクト・フィルター。マイグレーション・ツールは、リポジトリ内のプロジェクト名と、ユーザーが指定したプロジェクト・フィルターを突き合わせます。プロジェクト・フィルターは複数指定できます。フィルターを追加または除去するには、「追加」または「除去」のプッシュボタンを使用します。フィルターを更新するには、テーブル内で上書き入力します。フィルターに大/小文字の区別はありません。次のようにワイルドカードを使用できます。
  - プロジェクト・フィルター *\*xyz\** は、リポジトリ内にある、ストリング「xyz」を名前のどこかに含むプロジェクト名すべてとマッチングします。
  - プロジェクト・フィルター *xyz\** は、リポジトリ内にある、「xyz」から始まるプロジェクト名すべてとマッチングします。
  - プロジェクト・フィルター *\*xyz* は、リポジトリ内にある、「xyz」で終わるプロジェクト名すべてとマッチングします。
- バージョン深さ フィルター。プロジェクト名がプロジェクト・フィルターのいずれかとマッチングする場合に、バージョン深さフィルターを選択していれば、ステージ 1 マイグレーション・ツールはバージョン深さに指定した数の

バージョンを処理します。デフォルトは 1 で、この場合ステージ 1 マイグレーション・ツールはプロジェクトの最新バージョンのみを処理します。

- バージョン名 フィルター。プロジェクト名がプロジェクト・フィルターのいずれかとマッチングする場合に、バージョン名フィルターを選択していると、ステージ 1 マイグレーション・ツールはバージョン名フィルターを使用して、マイグレーション対象になるプロジェクト・バージョン (存在する場合) を判別します。バージョン名フィルターは複数指定できます。フィルターを追加または除去するには、「追加」または「除去」のプッシュボタンを使用します。フィルターを更新するには、テーブル内で上書き入力します。フィルターに大/小文字の区別はありません。次のようにワイルドカードを使用できます。
  - バージョン名フィルター `*xyz*` は、ストリング「xyz」をバージョン名のどこかに含むプロジェクト・バージョン名すべてとマッチングします。
  - バージョン名フィルター `xyz*` は、「xyz」から始まるプロジェクト・バージョン名すべてとマッチングします。
  - バージョン名フィルター `*xyz` は、「xyz」で終わるプロジェクト・バージョン名とマッチングします。

## 「マッピング (Mapping)」 ページ

「マッピング (Mapping)」 ページを使用して、EGL ファイル内でのパーツの配置と、マイグレーション時に作成される EGL プロジェクト、パッケージ、およびファイルの一部の名前を制御できます。

- **ファイル名 (File names)**。「ファイル名 (File names)」セクションでは、マイグレーション時に作成される 2 つの EGL ファイルの名前を制御できます。
  - 「共通パーツ (Common Parts)」には、マイグレーション・セットの有効範囲内で複数の固有な生成可能パーツに共通のパーツを格納する、EGL ファイルの名前を指定できます。ファイル名は、拡張子またはパスを付けずに指定します。マイグレーション・ツールは、マイグレーション・セットにある複数の生成可能パーツによって使用されて (関連付けられて) いるパーツを含んでいるか、または共通プロジェクトまたはパッケージとして指定された VAGen プロジェクトまたはパッケージ内のパーツを含んでいる、それぞれの EGL パッケージ内に、1 つの共通パーツ・ファイルを作成します。パーツがプログラムと一緒に配置されるか、共通パーツ・ファイル内に配置されるかの判別について詳しくは、42 ページの『EGL ファイル内でのパーツの配置』を参照してください。
  - 「未使用パーツ (Unused Parts)」には、マイグレーション・セットの有効範囲内で使用されていないパーツを格納する EGL ファイルの名前を指定できます。ファイル名は、拡張子またはパスを付けずに指定します。対応する VAGen プロジェクトまたはパッケージが共通のプロジェクトまたはパッケージとして指定されていないことを条件として、マイグレーション・セットにある生成可能パーツによって使用されて (関連付けられて) いないパーツを含む、それぞれの EGL パッケージ内で未使用パーツ・ファイルがマイグレーション・ツールによって作成されます。
- **スパン・マップ (Spanning Maps)**。「スパン・マップ (Spanning Maps)」セクションでは、1 つのマップ・グループに複数のプロジェクトまたはパッケージからのマップが含まれている場合に使用される接尾部を指定できます。

- 「プロジェクト接尾部 (*Project suffix*)」には、新規 EGL プロジェクト名を作成するための接尾部を指定できます。ステージ 1 マイグレーション・ツールは、この接尾部をマイグレーション・セット名に連結します。マイグレーション・ツールは、マップ・グループとそのマップがマイグレーション・セット内で複数の VAGen プロジェクトに存在する場合に限り、この新規 EGL プロジェクトを作成します。新規プロジェクト名は、*migrationSetName\_ProjectSuffix* です。マイグレーション・ツールは、名前変更規則がすべて適用された後で、マイグレーション・セット名に接尾部を連結します。
- 「パッケージ接尾部 (*Package suffix*)」には、EGL プロジェクト内で新規 EGL パッケージ名を作成するための接尾部を指定できます。ステージ 1 マイグレーション・ツールは、この接尾部をプロジェクト名に連結します。マイグレーション・ツールは、マップ・グループとそのマップがプロジェクト内で複数の VAGen パッケージに存在する場合に限り、この新規 EGL パッケージを作成します。新規パッケージ名は、*projectName\_PackageSuffix* です。マイグレーション・ツールは、名前変更規則が適用された後で、プロジェクト名に接尾部を連結します。
- **共通 ID (Common Identifiers)**。「共通 ID (Common Identifiers)」セクションでは、共通 (共用) パーツを含む VAGen プロジェクトとパッケージを判別するためにマイグレーション・ツールが使用できるストリングのリストを、ワイルドカードとともに指定できます。
- 「プロジェクト (*Projects*)」リストには、共通パーツを含むプロジェクトを識別するストリングのリストを指定できます。マイグレーション・ツールは、このストリングのリストをマイグレーション・セット内の各プロジェクト名と突き合わせて、プロジェクトに共通パーツが含まれているかどうかを判別します。ストリングがプロジェクト名にマッチングする場合、そのプロジェクト内のパーツはすべて「使用済み」と見なされます。生成不可のパーツはそれぞれ、プログラム・ファイル内、または共通パーツ設定に指定したファイルに置かれます。パーツがマイグレーション・セット内の生成可能パーツによって使用されていない場合であっても、そのパーツは未使用パーツ・ファイルには置かれませんが、プロジェクト・フィルターは複数指定できます。フィルターを追加または除去するには、「追加」または「除去」のプッシュボタンを使用します。フィルターを更新するには、テーブル内で上書き入力します。フィルターに大/小文字の区別はありません。ストリングの先頭または末尾で、\* をワイルドカードとして使用することもできます。
- 「パッケージ (*Packages*)」リストには、共通パーツを含むパッケージを識別するストリングのリストを指定できます。マイグレーション・ツールは、このストリングのリストをマイグレーション・セット内の各パッケージ名と突き合わせて、パッケージに共通パーツが含まれているかどうかを判別します。ストリングがパッケージ名にマッチングする場合、そのパッケージ内のパーツはすべて「使用済み」と見なされます。生成不可のパーツはそれぞれ、プログラム・ファイル内、または共通パーツ設定に指定したファイルに置かれます。パーツがマイグレーション・セット内の生成可能パーツによって使用されていない場合であっても、そのパーツは未使用パーツ・ファイルには置かれませんが、複数のパッケージ・フィルターを指定できます。フィルターを追加または除去するには、「追加」または「除去」のプッシュボタンを使用します。フィルターを

更新するには、テーブル内で上書き入力します。フィルターに大/小文字の区別はありません。string の先頭または末尾で、\* をワイルドカードとして使用することもできます。

## 「名前変更」ページ

「名前変更」ページを使用して、プロジェクト名、パッケージ名、およびバージョン名を変更するための規則を指定できます。「名前変更規則 (Renaming Rules)」セクションでは、VAGen プロジェクトとパッケージの名前から得られる EGL プロジェクトとパッケージの名前を制御できます。「順序 (order)」列の番号は、ステージ 1 マイグレーション・ツールが名前変更規則を適用する順序を示し、最も小さい番号の規則が最初に適用されます。名前変更規則を追加または除去するには、「追加」または「除去」のプッシュボタンを使用します。名前変更規則を更新するには、テーブルのセルの内容に上書きして入力します。列見出しのいずれかをダブルクリックすると、その列を基準に規則をソートできます。規則を指定するには、次の情報を指定します。

- 「順序 (Order)」は、規則を適用する順序を指定します。
- 「変更する string (From String)」は、VAGen 名の中で変更したい文字を指定します。
- 「変更後の string (To String)」は、変更結果の EGL 名の中で使用したい文字を指定します。
- 「string・コンテキスト (String Context)」は、マイグレーション・ツールが名前変更を行うときに、VAGen 名の中で変更する string を検索する位置を指定します。値は次のとおりです。
  - 「前 (front)」は、変更する string がプロジェクト名、パッケージ名、またはバージョン名の先頭にある場合に規則を適用するように指定します。
  - 「後 (back)」は、変更する string がプロジェクト名、パッケージ名、またはバージョン名の末尾にある場合に規則を適用するように指定します。
  - 「任意 (any)」は、変更する string がプロジェクト名、パッケージ名、またはバージョン名の中でどの位置にあっても規則を適用するように指定します。
  - 「トークン (token)」は、変更する string がプロジェクト名、パッケージ名、またはバージョン名と完全に一致する場合に限って規則を適用するように指定します。
- 「マッピング・コンテキスト (Mapping Context)」は、マイグレーション・ツールがプロジェクト名、パッケージ名、またはバージョン名のどれに名前変更規則を適用するかを指示します。「マッピング・コンテキスト (Mapping Context)」の値は、次のとおりです。
  - 「プロジェクト (project)」は、名前変更規則を VAGen プロジェクト名のみに適用するように指定します。
  - 「パッケージ (package)」は、名前変更規則を VAGen パッケージ名のみに適用するように指定します。
  - 「両方 (both)」は、名前変更規則を VAGen プロジェクト名と VAGen パッケージ名の両方に適用するように指定します。
  - 「バージョン (version)」は、名前変更規則をすべてのプロジェクト名のバージョン名に適用するように指定します。ご使用のバージョン名が、ディレクトリ



一名やファイル名に使用できないセミコロン (;) などの特殊文字を含んでいる場合は、バージョン名変更規則を使用します。デフォルトの *MigPreferences.xml* ファイルには、バージョン名が無効なディレクトリー名やファイル名にならないようにするために役立つ、いくつかのバージョン名変更規則が組み込まれています。マイグレーション・ツールは、名前変更済みのバージョンを使用して、マイグレーションのステージ 1 でマイグレーション・プラン・ファイル名を作成し、ステージ 3 でディレクトリー名を作成します。

## 「実行 (Execution)」 ページ

- **実行オプション (Execution Options)**。「実行オプション (Execution Options)」セクションでは、ステージ 1 マイグレーション・ツールが行う処理を指定できます。
  - 「レポートの生成 (Generate report)」を指定すると、EGL プロジェクト、パッケージ、およびファイルの構造内でそれぞれのパーツが配置される場所を示すマイグレーション・レポートが作成されます。このレポートは、共通パーツと未使用パーツのファイル名、スパン・マップ接尾部、プロジェクトとパッケージの共通 ID、および名前変更規則に対して指定した設定の結果を検討するために役立ちます。「レポートの生成 (Generate report)」を選択すると、「検証 (Verification)」セクションの「レポート・ファイル名 (Report file name)」に指定したドライブ、ディレクトリー、およびファイルに、マイグレーション・ツールによるレポートが作成されます。
  - 「データベースの更新 (Update database)」を指定すると、ステージ 1 マイグレーション・ツールは、マイグレーション・プラン情報 (パーツの外部ソース形式など) をマイグレーション・データベースに格納します。

ステージ 1 マイグレーション・ツールは、次のようにいくつかのステップに分けて実行できます。

- ステップ 1 -- 「レポートの生成 (Generate report)」と「データベースの更新 (Update database)」を両方とも選択解除する。これにより、作成されたマイグレーション・プラン・ファイルを検討して、リポジトリ・フィルターが正しく設定されていることと、目的のプロジェクト・バージョンが処理されていることを確認できます。選択されるプロジェクト・バージョンが適切でない場合は、マイグレーション・ツールが正しいプロジェクト・バージョンを処理するようになるまで、リポジトリ・フィルターを改良してこのステップを再度実行できます。
- ステップ 2 -- 「レポートの生成 (Generate report)」のみを選択する。これにより、VAGen のプロジェクト、パッケージ、およびパーツが、マイグレーション時に EGL のプロジェクト、パッケージ、およびファイルにどのように割り当てられるかを検討できます。パーツの配置が適切でない場合は、パーツの配置が正しく行われるようになるまで、マッピング規則と名前変更規則を改良してレポートを再度実行できます。
- ステップ 3 -- 「レポートの生成 (Generate report)」と「データベースの更新 (Update database)」を両方とも選択する。これにより、マイグレーション・データベースに格納されている情報を記録する最終レポートが作成されます。

注:



- レポートの生成には多少時間がかかることがあります。このため、.pln ファイルを検討して、目的のプロジェクト・バージョンがマイグレーション・ツールの処理対象になるか確認することが最良策です。
  - レポートが生成されると、レポート・ファイルは上書きされます。前のレポート・ファイルを保管する必要がある場合は、レポート・ファイルを別のディレクトリに移動するか、新規レポート用のディレクトリを新しく指定する必要があります。レポート・ファイルは他のファイルにリンクしているので、レポート・ファイルの名前を変更するとリンクが失われ、ファイルが表示できなくなります。
- **データベース (Database)。**「データベース (Database)」セクションでは、マイグレーション・データベースに関する詳細を次のように指定できます。
    - データベース・ドライバ。この値は、常に **app.DB2Driver** にする必要があります。
    - データベース名。この値は、常に次のいずれかにする必要があります。
      - jdbc:DB2:databaseName (ローカル・データベースを使用する場合)
      - jdbc:nodeName:databaseName (リモート・データベースを使用する場合)
- 注:** どちらの場合にも、databaseName は、マイグレーション・ツールがマイグレーション・セット情報を書き込むマイグレーション・データベースの名前です。デフォルトでは、databaseName は VGMIG です。マイグレーション・データベースの作成時の VGMIG からデータベース名を変更した場合は、この設定に指定されているデータベース名を変更して、使用した名前と一致させる必要があります。
- 「スキーマ (Schema)」は、データベース・テーブル用の修飾子として使用される名前です。デフォルトでは、スキーマ名は MIGSCHEMA です。マイグレーション・データベースの作成時の MIGSCHEMA からスキーマ名を変更した場合は、この設定に指定されているスキーマ名を変更して、使用した名前と一致させる必要があります。
  - 「ユーザー ID (Userid)」は、マイグレーション・データベースへの接続に必要なユーザー ID です。ユーザー ID を指定しない場合、マイグレーション・ツールはログオン・ユーザー ID を使用して接続を試みます。この試行に失敗すると、マイグレーション・ツールは情報を尋ねるダイアログ・ウィンドウを表示します。
  - 「パスワード (Password)」は、マイグレーション・データベースへの接続に必要なパスワードです。パスワードを指定しない場合、マイグレーション・ツールはログオン・パスワードを使用して接続を試みます。この試行に失敗すると、マイグレーション・ツールは情報を尋ねるダイアログ・ウィンドウを表示します。
- 注:** パスワードは設定ファイル内で暗号化されません。このことが問題になる場合は、設定ファイルにパスワードを入力せず、プロンプトが出されるまで待ってください。
- **サービス (Service)。**「サービス (Service)」セクションでは、ステージ 1 の間に収集したいロギングとデバッグの情報に関する詳細を次のように指定できます。指定できる項目は次のとおりです。

- 「トレース・レベル (Trace level)」には、ログ・ファイルとデバッグ・ファイルに書き込む情報のレベルを指定できます。ドロップダウン・リストを使用して、次のいずれかの値を指定します。
  1. 重大 (Fatal)。致命エラー・メッセージがログに記録されます。重大メッセージが出された場合、マイグレーション・データベースは更新される可能性があります。マイグレーション・プラン・ファイル (.pln ファイル) は変更されず、.done ファイル拡張子は付きません。このため、.pln ファイルを再処理できます。
  2. 警告 (Warning)。警告メッセージが、致命エラー・メッセージとともにログに記録されます。
  3. 情報 (Informational)。情報メッセージが、警告メッセージおよび致命エラー・メッセージとともに記録されます。
  4. デバッグ (Debug)。デバッグ情報が、情報メッセージ、警告メッセージ、および致命エラー・メッセージとともに記録されます。トレース・レベルが「デバッグ (DEBUG)」の場合のみ、マイグレーション・ツールはデバッグ・ファイルに情報を書き込みます。

トレース・レベルは、ログ・ファイルとデバッグ・ファイルのみに影響を及ぼします。メッセージはすべてコンソール・ウィンドウに書き込まれます。

- 「ログ・ファイル名」には、ログ・ファイルのドライブ、ディレクトリー、およびファイル名を指定できます。任意のファイル拡張子を付けてログ・ファイルを作成できますが、このファイルは .xml ファイルとして最適に表示されます。ログ・ファイル名を省略した場合、マイグレーション・ツールは、「ログ・ファイル名」フィールドに指定したドライブとディレクトリーにある、*miglog.xml* という名前のファイルにログ情報を書き込みます。ログ・ファイルのドライブとディレクトリーを指定しない場合、マイグレーション・ツールはログ・ファイルを作業ディレクトリーに書き込みます。
- 「デバッグ・ファイル名 (Debug file name)」には、IBM サポートが必要とするデバッグ・ファイルのドライブ、ディレクトリー、およびファイル名を指定できます。任意のファイル拡張子を付けてデバッグ・ファイルを作成できますが、このファイルは .xml ファイルとして最適に表示されます。このファイルへの情報の書き込みは、トレース・レベルが「デバッグ (Debug)」に設定されている場合に限って行われます。デバッグ・ファイル名を省略した場合に、トレース・レベルとして「デバッグ (Debug)」を指定すると、マイグレーション・ツールは、「デバッグ・ファイル名 (Debug file name)」フィールドに指定したドライブとディレクトリーにある、ファイル *migdebug.xml* にデバッグ・ファイルの情報を書き込みます。デバッグ・ファイルのドライブとディレクトリーを指定しない場合、マイグレーション・ツールはデバッグ・ファイルを作業ディレクトリーに書き込みます。
- 検証 (Verification)。「検証 (Verification)」セクションでは、検証レポートのドライブ、ディレクトリー、およびファイル名を指定できます。このレポートは、「実行オプション (Execution Options)」セクションの「レポートの生成 (Generate report)」設定を選択した場合に作成されます。「レポートの生成 (Generate report)」を選択した場合は、レポート・ファイル名を入力する必要があります。必ず .htm 拡張子を指定してください。ドライブとディレクトリーを指定しない場合、マイグレーション・ツールはレポート・ファイルを作業ディレクトリーに書き込みます。

## MigPreferences.xml ファイルのサンプル

次に、MigPreferences.xml ファイルのサンプルを示します。

```
<preferences>
  <database>
    <driver>COM.ibm.db2.jdbc.app.DB2Driver</driver>
    <uri>jdbc:DB2:VGMIG</uri>
    <schema>MIGSCHEMA</schema>
    <userid></userid>
    <password></password>
  </database>
  <migrationSpec>
    <directory>d:¥tempMig¥MyMigSet</directory>
    <filename></filename>
  </migrationSpec>
  <repositoryFilters>
    <projectName>MyProject*</projectName>
    <versionName></versionName>
  </repositoryFilters>
  <service>
    <tracelevel>4</tracelevel>
    <debugfile>d:¥tempMig¥MyMigSet¥Stage1¥migdebug.xml</debugfile>
    <logfile>d:¥tempMig¥MyMigSet¥Stage1¥miglog.xml</logfile>
  </service>
  <eglMapping>
    <renameRule order = "1">
      <fromString> </fromString>
      <toString></toString>
      <stringContext>any</stringContext>
      <mappingContext>both</mappingContext>
    </renameRule>
    <renameRule order = "101">
      <fromString>Project</fromString>
      <toString></toString>
      <stringContext>any</stringContext>
      <mappingContext>project</mappingContext>
    </renameRule>
    <renameRule order = "301">
      <fromString>.pkg</fromString>
      <toString></toString>
      <stringContext>any</stringContext>
      <mappingContext>package</mappingContext>
    </renameRule>
    <renameRule order = "302">
      <fromString>.sql</fromString>
      <toString>sql</toString>
      <stringContext>any</stringContext>
      <mappingContext>package</mappingContext>
    </renameRule>
    <renameRule order = "501">
      <fromString>:</fromString>
      <toString>_</toString>
      <stringContext>any</stringContext>
      <mappingContext>version</mappingContext>
    </renameRule>
    <renameRule order = "502">
      <fromString>/</fromString>
      <toString>_</toString>
      <stringContext>any</stringContext>
      <mappingContext>version</mappingContext>
    </renameRule>
    <renameRule order = "503">
      <fromString>¥</fromString>
      <toString>_</toString>
      <stringContext>any</stringContext>
      <mappingContext>version</mappingContext>
    </renameRule>
  </eglMapping>
</preferences>
```

```

</renameRule>
<renameRule order = "504">
  <fromString>|</fromString>
  <toString>_</toString>
  <stringContext>any</stringContext>
  <mappingContext>version</mappingContext>
</renameRule>
<renameRule order = "505">
  <fromString>?</fromString>
  <toString>_</toString>
  <stringContext>any</stringContext>
  <mappingContext>version</mappingContext>
</renameRule>
<renameRule order = "506">
  <fromString>*</fromString>
  <toString>_</toString>
  <stringContext>any</stringContext>
  <mappingContext>version</mappingContext>
</renameRule>
<renameRule order = "507">
  <fromString>&lt;</fromString>
  <toString>_</toString>
  <stringContext>any</stringContext>
  <mappingContext>version</mappingContext>
</renameRule>
<renameRule order = "508">
  <fromString>&gt;</fromString>
  <toString>_</toString>
  <stringContext>any</stringContext>
  <mappingContext>version</mappingContext>
</renameRule>
<renameRule order = "509">
  <fromString>&quot;</fromString>
  <toString>_</toString>
  <stringContext>any</stringContext>
  <mappingContext>version</mappingContext>
</renameRule>
<renameRule order = "510">
  <fromString> </fromString>
  <toString>_</toString>
  <stringContext>any</stringContext>
  <mappingContext>version</mappingContext>
</renameRule>
<verification>
  <generateReport>true</generateReport>
  <reportName>d:¥tempMig¥MyMigSet¥report¥MyReport.htm</reportName>
</verification>
<dbUpdate>true</dbUpdate>
<spanningMapsProjectSuffix>MapsProject</spanningMapsProjectSuffix>
<spanningMapsPackageSuffix>mapspackage</spanningMapsPackageSuffix>
<commonPartsFileName>CommonParts</commonPartsFileName>
<unusedPartsFileName>UnusedParts</unusedPartsFileName>
<commonParts>
  <commonProject>*Common*</commonProject>
  <commonPackage>*common*</commonPackage>
</commonParts>
</eglMapping>
</preferences>

```

---

## ステージ 1 ツールを実行する前に — ヒント

ステージ 1 マイグレーション・ツールを実行する前に、いくつかの処置を行ってパフォーマンスを高めることができます。また、マイグレーションの完了後に使用するために、既存のワークスペースを保管することもできます。

## パフォーマンスの向上

パフォーマンス測定の結果、新規ワークスペースから作業を開始すると、ステージ 1 マイグレーション・ツールのパフォーマンスが大幅に向上することが判明しました。一連のテストでは、新規ワークスペースから開始すると、新規ワークスペースを使用しない場合に比べてステージ 1 の時間が 25% から 30% 短縮されました。既存のワークスペースが 20 メガバイトより大きい場合は、新規ワークスペースから開始するとステージ 1 ツールのパフォーマンスが向上する可能性があります。

新規ワークスペースから作業を開始するには、次の手順で行います。

1. VisualAge Generator をシャットダウンする。
2. マイグレーションの完了後に使用するために既存のワークスペースのバックアップ・コピーを保持したい場合は、135 ページの『ワークスペースの保管』を参照してください。
3. 次に示す VisualAge Generator ダウンロード・サイトから、新規ワークスペースのコピー (ファイル名 `ide.icx`) をダウンロードする。  
`ftp://ftp.software.ibm.com/ps/products/visualagegen/fixes/v4.5/FixPack4/windows`
4. `features.sav` ファイルと `projects.sav` ファイルを削除する。
5. VisualAge Generator を再始動する。
6. 必要な VisualAge Generator フィーチャーを追加する。
7. 「IBM VisualAge Generator EGL マイグレーション (IBM VisualAge Generator EGL Migration)」フィーチャーを追加する。
8. VisualAge Generator をシャットダウンする。

ステージ 1 マイグレーション・ツールがマイグレーション対象のプロジェクトとバージョンの分析に費やす時間を短縮するために、マイグレーションしたいプロジェクト・バージョンのみを含むリポジトリの作成を検討してください。マイグレーション中に VisualAge Generator の継続保守を行う場合は、別個のマイグレーション・リポジトリを使用すると次のような利点もあります。

- マイグレーションするプロジェクト・バージョンのセットが安定します。このことは、「バージョン深さ (Version Depth)」設定を使用してマイグレーション対象を制御する場合に特に重要です。
- 新規マイグレーション・リポジトリのバージョンを保守リポジトリと比較して、マイグレーションを必要とする他のプロジェクト・バージョンがまだあるかどうか判別できます。

特殊リポジトリを作成する場合は、ステージ 1 マイグレーションのパフォーマンスを高めるために、ローカル・リポジトリとして使用することを検討してください。

## ワークスペースの保管

ステージ 1 マイグレーション・ツールは、ステージ 1 処理の開始時と終了時に、VAGen パーツを含むプロジェクトをワークスペースからすべて削除します。これにより、ワークスペース内でパーツの重複を防止でき、ステージ 1 の実行中に、マイグレーション・セット内のパーツのみが関連パーツ・リストの対象と見なされるようになります。保管したいワークスペースがある場合は、ステージ 1 ツールを実行する前に次の手順で行う必要があります。

1. VisualAge Generator をシャットダウンする。
2. 次のファイルのバックアップ・コピーを、  
¥VisualAgeForJava-installation-directory¥ide¥program に保管する。
  - features.sav
  - projects.sav
  - ide.icx
  - ide.ini (ステージ 1 の実行中に設定を変更しない場合は保管不要)
  - hpt.ini (ステージ 1 の実行中に設定を変更しない場合は保管不要)
3. VisualAge Generator を開始する。

ステージ 1 ツールの実行を完了した後、ワークスペースを復元するには次の手順で行います。

1. VisualAge Generator をシャットダウンする。
2. ステージ 1 ツールを実行する前にバックアップしたファイルを復元する。
3. VisualAge Generator を開始する。

## ステージ 1 ツールの実行

設定の編集を完了した後、ステージ 1 マイグレーション・ツールを実行して、ソース・コードを Java リポジトリから抽出できます。このためには、次の手順で行います。

1. 「IBM VisualAge Generator EGL マイグレーション (IBM VisualAge Generator EGL Migration)」プロジェクトにナビゲートする。
2. マイグレーション・プロジェクトを展開し、パッケージ *com.ibm.vgj.mig* を展開する。
3. このパッケージ内で、「VAGenToEGLMigration」クラスを選択する。
4. 「VAGenToEGLMigration」クラスを右クリックし、コンテキスト・メニューから「プロパティ」をクリックする。
5. 「プログラム」タブを選択する。
6. 「プログラム」ページの「コマンド行引数 (Command line arguments)」フィールドに次のように指定して、編集する MigPreferences.xml ファイルを指示する。

表 57. VAGentoEGLMigration クラスの有効なコマンド行オプション

オプション	オプションの意味
-h	有効なオプションを示すヘルプ情報を表示します。
-p filename	設定ファイルの名前として「filename」を使用します。ファイル名は、ドライブとディレクトリーを含めて完全に修飾する必要があります。
-o	存在する場合はマイグレーション・ファイルを上書きし、再作成します。

7. ステージ 1 ツールを初めて実行する場合は、次の手順で行います。
  - a. 同じ「プロパティ (Properties)」ウィンドウで、「クラスパス」タブを選択する。



- b. 「クラスパス」 ページで、「追加のディレクトリー・パス (Extra directories path)」チェック・ボックスを選択し、追加のディレクトリーの「編集」 ボタンをクリックする。
  - c. 「Jar/Zip の追加 (Add Jar/Zip)」 ボタンを選択する。
  - d. 「ファイル選択」 ウィンドウで、db2java.zip ファイルにナビゲートしてこのファイルを選択する。
    - DB2 をインストールしたときにデフォルトのインストール・ディレクトリーを使用していた場合、ファイルは %SQLLIB%\java ディレクトリーにあります。
- db2java.zip ファイルを選択した後、ファイル名が「追加のディレクトリー (Extra directories)」 ウィンドウに表示されます。「追加のディレクトリー (Extra Directories)」 ウィンドウの「OK」をクリックします。
- e. 「クラスパス」 ページで、「計算 (Compute Now)」 ボタンをクリックしてから、プロンプトの「はい」をクリックする。
8. 「OK」をクリックしてプロパティを保管する。
9. 「VAGenToEGLMigration」を右クリックし、「実行」または「メインを実行... (Run main...)」をクリックする (または、ツールバーの走る人のアイコンを選択することもできます)。ステージ 1 マイグレーション・ツールが始動し、コンソール・ウィンドウが開いて進行状況とエラー・メッセージが報告されます。マイグレーション設定の中で指定したログ・ファイルにも、マイグレーション・ツールのメッセージが書き込まれます。

ステージ 1 マイグレーション・ツールの終了時に、「データベースの更新 (Update database)」設定を選択していれば、マイグレーション・プラン情報 (外部ソース形式の VAGen コードなど) がマイグレーション・データベースに格納されます。レポートとステージ 1 メッセージを検討した後、必要に応じて VisualAge Generator 内でコードを変更し、ステージ 1 を再度実行できます。ステージ 1 の結果に問題がなく、最終の外部ソース形式コードがマイグレーション・データベースに格納された後、マイグレーションのステージ 2 を実行できます。ステージ 2 マイグレーション・ツールを実行するには、EGL 開発環境を使用します。マイグレーション・プロセスの継続については、169 ページの『第 6 章 ステージ 2 — EGL 構文への変換』を参照してください。

---

## マイグレーション・プランと上位 PLP プロジェクト

マイグレーション・プラン・ファイルは単純な XML ファイルで、1 つ以上のマイグレーション・セットの名前を指定し、それぞれのマイグレーション・セットごとに、マイグレーション・セットを構成するプロジェクト名とバージョンのリストを指定します。ステージ 1 マイグレーション・ツールは、プロジェクトとバージョン名に関するリポジトリ・フィルターの設定に基づいて、マイグレーション・プラン・ファイルを自動的に作成するように設計されています。ステージ 1 ツールは、プロジェクト・バージョンが上位 PLP プロジェクトかどうかを判別するために、これらのフィルターを使用してプロジェクト・バージョンが検討対象かどうかを判別します。ステージ 1 ツールは、それぞれの上位 PLP プロジェクト・バージョンをマイグレーション・セットの基礎として使用します。

VAGen ソース・コードの生成時に PLP プロジェクトを使用した場合は、これらと同じ PLP プロジェクトをマイグレーションに使用します。これは、これらの PLP プロジェクトが生成時に一緒に使用されるパーツのグループを指定しているからで、したがってこのプロジェクトは一連のプログラムの関連パーツをすべて含んでいます。

PLP プロジェクトを現在使用していない場合は、次のいずれかを行うことができます。

- グループとしてマイグレーションするプロジェクト・バージョンのリストを指定する、上位 PLP プロジェクトを作成します。その後、ステージ 1 マイグレーション・ツールを使用して、マイグレーション・プランを自動的に作成できます。
- 上位 PLP プロジェクトを作成しない場合は、次のいずれかの手法を使用して、マイグレーション・プラン・ファイルをユーザーが独自に作成できます。
  - Java プロジェクトのバージョンに関連して、生成に必要なものを指定する情報がデータベースや他のシステム内にある場合は、データベースからマイグレーション・プラン・ファイル (複数可) を自動的に作成するツールをユーザーが開発できます。
  - マイグレーション・プラン・ファイル (複数可) を手作業で作成できます。

## 上位 PLP プロジェクトの作成

注: プロジェクト名またはバージョン名に DBCS 文字が含まれている場合、VisualAge Generator は PLP をサポートしません。プロジェクト名またはバージョン名に DBCS 文字が含まれている場合に、PLP を使用せずにマイグレーション・プラン・ファイルを作成する方法については、139 ページの『マイグレーション・プラン・ファイルの手動作成』を参照してください。

マイグレーションに使用する上位 PLP プロジェクトを作成するには、VisualAge Generator 内で次の手順で行います。

1. 「ワークベンチ」ウィンドウの「プロジェクト」タブを選択する。
2. プロジェクト・リスト・パーツを格納する Java プロジェクトを新規に作成する。プロジェクトには、必ず既存のプロジェクトと異なる名前を付けるようにしてください。例えば、MySubsystem1 という名前のプロジェクトを作成します。
3. 新規プロジェクトを選択し、右クリックして、コンテキスト・メニューから「管理 (Manage)」->「VAGen 必須プロジェクトの構成 (Configure VAGen Required Projects)」を選択する。
4. 「VAGen 必須プロジェクトの構成 (Configure VAGen Required Projects)」ウィンドウで、マイグレーション・セットに組み込む各プロジェクトを選択する。それぞれのプロジェクトごとに、マイグレーション・セットに組み込む特定のバージョンを選択できます。代わりに、「最新エディション (*Most recent edition*)」を選択することもできます。この場合、マイグレーション・ツールは、マイグレーション中にこのプロジェクトを使用するときは常に、現時点でリストの先頭にあるバージョンを自動的に組み込みます。
5. マイグレーション・セットに必要なプロジェクト・バージョンをすべて選択した後、「OK」をクリックする。
6. 上位 PLP プロジェクトのバージョンとリリースを指定する (例: MySubsystem1)。

7. その PLP プロジェクトによって、マイグレーション・セットに必要なプロジェクト・バージョンが正しくロードされることを、次のようにして確認する。
  - a. 上位 PLP プロジェクトと他のすべての VAGen プロジェクトをワークスペースから削除する。
  - b. 「選択」 -> 「追加」 -> 「プロジェクト」を選択する。
  - c. 「プロジェクトの追加」ウィンドウから、次の手順で行う。
    - 1) 「リポジトリからプロジェクトを追加 (Add projects from the repository)」を選択する。
    - 2) 直前に作成した上位 PLP プロジェクト、および作成したバージョンを選択する。
    - 3) 「VAGen 必須プロジェクトの追加 (Add VAGen required projects)」も選択する。
    - 4) 「完了」をクリックする。
  - d. 上位 PLP プロジェクトと、そのプロジェクトの指定するプロジェクト・バージョンすべてがワークスペースに追加されます。
  - e. VAGen パーツ・ブラウザーから、「ツール (Tools)」 -> 「重複パーツの表示 (Show Duplicate Parts)」を選択する。リストにパーツがあってはなりません。パーツがある場合は、重複がなくなるように上位 PLP プロジェクトを変更する必要があります。
  - f. また、プログラムとテーブルの検証を実行して、これらが VAGen 内で有効であることと、欠落しているパーツがないことを確認することもできます。

PLP プロジェクトはチェーニングできます。例えば、共通プロジェクトすべてのプロジェクト・バージョンをリストする PLP プロジェクトを作成します。次に、それぞれのサブシステムごとに、サブシステム固有のプロジェクト・バージョンすべてを含むそのサブシステム用の上位 PLP プロジェクトと、共通プロジェクト・バージョンすべてを指定する PLP プロジェクトを作成します。この方法により、サブシステムすべての上位 PLP プロジェクトに、それぞれの共通プロジェクト・バージョンをリストする必要がなくなります。

ステージ 1 マイグレーション・ツールの実行準備ができたなら、次の作業を行います。

- ステージ 1 設定を行うときに、「プランの作成 (Build plans)」ページの「リポジトリ・フィルター (Repository Filters)」セクションでプロジェクト・リストを設定して、リストにあるフィルターが、作成した上位 PLP プロジェクトにマッチングするようにします。
- 使用する設定ファイルをステージ 1 ツールに指定するときは、`-o` オプションも指定します。`-o` オプションを指定すると、ステージ 1 マイグレーション・ツールは上位 PLP プロジェクトに基づいてマイグレーション・プラン・ファイルを作成し、既存のマイグレーション・プラン・ファイルを上書きします。

## マイグレーション・プラン・ファイルの手動作成

VisualAge Generator での生成時にワークスペースに追加するプロジェクト・バージョンを決定する外部制御手段がすでに存在する場合は、マイグレーション・プラン・ファイルを手動で作成するか、外部情報からマイグレーション・プラン・ファ

イルを自動的に作成するツールを開発できます。マイグレーション・プラン・ファイルのファイル拡張子は *.pln* にする必要があり、フォーマットは次のとおりです。

```
<migrationDefinition>
  <migrationSet name="migrationSet1" version="migrationSet1Version1"
    vgName="migrationSet1" vgVersion="migrationSet1Version1">
    <project name="projectName1" version="projectName1Version1"></project>
    <project name="projectName2" version="projectName2Version1"></project>
    .
    .
    .
    <project name="projectNameN" version="projectNameNVersion1"></project>
  </migrationSet>
  <migrationSet name="migrationSet2" version="1.1"
    vgName="migrationSet2" vgVersion="1.1">
    <project name="projectNameA" version="projectNameAVersion1"></project>
    <project name="projectNameB" version="projectNameBVersion1"></project>
    .
    .
    .
    <project name="projectNameZ" version="projectNameZVersion1"></project>
  </migrationSet>
</migrationDefinition>
```

前述の例では、次のことが当てはまります。

- migrationSet1 は、一緒にマイグレーションする必要があるプロジェクトのグループを参照するために使用できる名前です。マイグレーション・セット名はマイグレーション・データベースに格納され、マイグレーションの以降のステージで次のように使用されます。
  - ステージ 1 マイグレーションでは、マップ・グループ内のマップが複数のプロジェクトにわたる場合に、接尾部を連結したマイグレーション・セット名を使用して、マイグレーション・セットとそのマップすべてを格納する新規 EGL プロジェクトの名前が作成されます。名前変更規則を変更する場合にも、マイグレーション・データベースから情報を除去するためにマイグレーション・セット名が使用されます。
  - ステージ 2 マイグレーションの場合、マイグレーション・セット名は、マイグレーション・データベース内で EGL に変換する対象のプロジェクト・グループを指定します。
  - ステージ 3 の場合、マイグレーション・セット名は、ワークスペースまたは一時ディレクトリーに EGL プロジェクト、パッケージ、およびファイルを作成するために使用する、マイグレーション・データベース内のプロジェクト・グループを指定します。ステージ 3 の出力を一時ディレクトリーに保管する場合、マイグレーション・セット名とマイグレーション・セット・バージョンは、上位ディレクトリー名の作成にも使用されます。

マイグレーション・セット名は、マイグレーション時にプロジェクト・グループを識別する手段としてのみ使用されます。マップが VisualAge Generator 内で複数のプロジェクトにわたっている場合を除き、マイグレーション・セット名はマイグレーション後には使用されません。

- projectName1、projectName2、...、projectNameN は、グループとしてマイグレーションするプロジェクトです。projectName は、マイグレーション・セット内で一度限りリストする必要があります。マイグレーション・ツールは、同じマイグレーション・セットの下にリストされているプロジェクト・バージョンすべてをワークスペースにロードし、グループとして処理します。

- projectName1Version1、projectName2Version1、...、projectNameNVersion1 は、これらのプロジェクトそれぞれのバージョンです。1 つのマイグレーション・セット内で、それぞれのプロジェクトごとにバージョンをただ 1 つ指定できます。
- 指定するプロジェクト名およびバージョン名は、リポジトリ内のプロジェクト名およびバージョン名と完全に一致する必要があります。名前には大/小文字の区別があります。この情報を使用して、ワークスペースにプロジェクト・バージョンが追加され、ステージ 1 マイグレーション・レポートの作成、およびデータベースのロードのためにパーツを分析できるようになります。

ただ 1 つのマイグレーション・セットを含むマイグレーション・プラン・ファイルを作成できます。また、それぞれのマイグレーション・セットごとに <migrationSet> タグと </migrationSet> タグの間の情報を繰り返すことによって、複数のマイグレーション・セットを含むマイグレーション・プラン・ファイルを作成することもできます。

ステージ 1 マイグレーション・ツールの実行準備ができれば、次の作業を行います。

- ステージ 1 設定を行うときに、「**プランの作成 (Build plans)**」タブで、「**プラン・ディレクトリー名 (Plan directory name)**」を、マイグレーション・プラン・ファイルの保管先のドライブとディレクトリーに設定します。作成したマイグレーション・プランを 1 つだけ使用してステージ 1 マイグレーション・ツールを実行する場合は、「**プラン・ファイル名 (Plan file name)**」を指定します。指定したプラン・ディレクトリーにあるすべてのマイグレーション・プラン・ファイルを使用してステージ 1 マイグレーション・ツールを実行する場合は、「**プラン・ファイル名 (Plan file name)**」をブランクのままにします。
- 使用する設定ファイルをステージ 1 ツールに指定するとき、**-o** オプションは必ず省略してください。**-o** オプションを省略すると、ステージ 1 ツールは既存のマイグレーション・プラン・ファイルを使用します。つまり、ツールはマイグレーション・プラン・ファイルを新規に作成しません。





---

## 第 3 部 VisualAge Generator 4.5 on Smalltalk から EGL へのマイグレーション



---

## 第 5 章 ステージ 1 — Smalltalk からの抽出

情報を VisualAge Generator から抽出する前に、VisualAge Smalltalk 上で稼働するステージ 1 マイグレーション・ツールをインストールする必要があります。また、VisualAge Generator 4.5 (VAGen 4.5) から EGL にマイグレーションするデータの保管に使用される、DB2 マイグレーション・データベースも作成する必要があります。

---

### VisualAge Smalltalk へのステージ 1 マイグレーション・ツールのインストール

VisualAge Generator to EGL ステージ 1 マイグレーション・ツールは、VAGenMigST.exe という名前の自己解凍ファイルとして提供されます。このファイルをインストールするには、次の手順で行います。

1. フィックスパック 4 を適用して VisualAge Generator 4.5 にアップグレードする。また、ユーザー個々の状況に応じてさらに必要になる可能性がある VisualAge Generator APAR については、439 ページの『付録 F. 外部ソース形式の誤りが原因で EGL の作成に問題が生じる状態』を参照してください。

注: フィックスパック 5 には、マイグレーション・ツールに必要なすべての APAR が含まれる予定です。

2. 使用しているシステム上で、VisualAge Smalltalk がインストールされている場所を判別する。
3. VisualAge Smalltalk をシャットダウンする。
4. 自己解凍型の VAGenMigST.exe ファイルを実行する。このファイルは、Rational Developer インストール・ディレクトリーの以下に示すサブディレクトリーにあります。

¥bin

注: 現在ご使用の製品をインストールする前に、前のバージョンの開発者製品をインストールし、保持していた場合、対象のインストール・ディレクトリーは、以前のインストール時に使用されたディレクトリーである可能性があります。

5. GUI プロンプトが表示されたら、VisualAge Smalltalk がインストールされているドライブとディレクトリーにナビゲートする。次に、「抽出 (Extract)」をクリックする。

自己解凍型の実行可能ファイルが実行されると、次のファイルが VisualAge Smalltalk インストール・ディレクトリーに抽出されます。

- import¥vgMigSt.dat
- feature¥vgMigSt.ctl
- image¥Messages.properties
- image¥MigPreferences.xml

- ¥image¥VGMigReserved.txt
- checkStage1.sql
- createdatabase.sql
- createtables.sql
- checkStage1.bat
- SetupDatabase.bat
- SetupTables.bat
- deletemigsets.bat

## マイグレーション・フィーチャーのロード

ステージ 1 マイグレーション・ツールを使用するには、VAGen EGL マイグレーション・フィーチャーをロードする必要があります。このためには、次の手順で行います。

1. VisualAge Generator on Smalltalk を開始する。
2. 次の手順で、VAGen EGL マイグレーション・フィーチャーをロードする。
  - a. システム・トランスクリプトから、「ツール (Tools)」 -> 「フィーチャーのロード/アンロード (Load/Unload Features)」を選択する。
  - b. 「必要な選択 (Selection Required)」ウィンドウで、次の操作を行う。
    - 1) 「他のフィーチャーを表示 (Show other features)」チェック・ボックスが選択されていることを確認する。
    - 2) 「使用可能なフィーチャー」ペインで、「その他: VAGen EGL マイグレーション - *versionName* (Other: VAGen EGL Migration - *versionName*)」を選択する。
    - 3) 「>>」ボタンを選択して、「その他: VAGen EGL マイグレーション - *versionName* (Other: VAGen EGL Migration - *versionName*)」を「ロードされるフィーチャー (Loaded features)」ペインに移動する。
    - 4) 「OK」をクリックする。VAGen EGL マイグレーション・フィーチャーがインポートされ、イメージにロードされます。
3. システム・トランスクリプトに、VAGen EGL マイグレーション・フィーチャーが正常にロードされたことを示すメッセージが表示されます。また、ツールバーに「EGL マイグレーション・ツール (EGL Migration Tools)」が表示されます。VisualAge Organizer 内では、「アプリケーション (Applications)」ペインに HptEglMigrationGuiApp が表示されます。
4. VAGen EGL マイグレーション・フィーチャーがロードされた後、イメージを保管するためのプロンプトが出されます。フィーチャーを再度ロードしなくて済むように、「はい」を選択します。

注: フィーチャーのロードに問題が生じた場合は、abt.ini ファイルを検査してください (VisualAge-Smalltalk-installation-directory¥image ディレクトリーに保管されています)。abt.ini ファイルの [EmLibraryInterface] 見出しの下に、次のフィールドが入力されていることを確認してください。

- ServerAddress=myserver.somecompay.somewhere.com。この値は、EMSRV を実行する社内のサーバーを指示する必要があります。ローカル・ライブラリーを使用している場合は、ServerAddress=127.0.0.1 を設定します。

- DefaultName=path-to-mgr50.dat¥mgr50.dat。この値は、Smalltalk ライブラリーの名前である必要があります。

---

## マイグレーション・データベースの作成

マイグレーション・データベースの作成については、441 ページの『DB2 マイグレーション・データベースの作成』を参照してください。自己解凍型の VAGenMigST.exe ファイルを実行したときに VisualAge Smalltalk インストール・ディレクトリーに置かれた、SetupDatabase.bat ファイルと SetupTables.bat ファイルを使用する必要があります。

---

## ステージ 1 設定の実行

ステージ 1 マイグレーション・ツールを VisualAge Smalltalk にインストールするときに、インストール・プロセスによってサンプル設定ファイル MigPreferences.xml がディレクトリー VisualAge-Smalltalk-installation-directory¥image に作成されます。設定を変更する前に、バックアップのために MigPreferences.xml ファイルのコピーを作成しておく必要があります。MigPreferences.xml を VisualAge for Smalltalk インストール・ディレクトリーの外側にあるディレクトリーにコピーして、そのコピーに対して変更を加えることもできます。これによって、新規バージョンのマイグレーション・ツールをインストールした場合に変更内容を誤って上書きしてしまうことを防止できます。

Smalltalk 上の VisualAge Generator to EGL マイグレーション・ツールには、ステージ 1 マイグレーションの設定値を指定するために役立つ GUI エディターが備わっています。ステージ 1 設定エディターは、次に示す 2 つの方法のどちらかで開始できます。

- システム・トランスクリプトから、「EGL マイグレーション・ツール (EGL Migration Tools)」->「設定エディター (Preferences Editor)」を選択する。EGL マイグレーション設定エディターが表示されます。設定エディターは、デフォルトで最後に変更した設定ファイル (また、設定をまだ一度も変更していない場合は、ステージ 1 ツールに付属の MigPreferences.xml ファイル) を開きます。別の設定ファイルを指定する必要がある場合は、「開く... (Open...)」ボタンをクリックします。
- システム・トランスクリプトから、「EGL マイグレーション・ツール (EGL Migration Tools)」->「マイグレーション・ドライバー (Migration Driver)」を選択する。「マイグレーション・ファイル設定 (Migration File Preference)」セクションで、設定ファイルのファイル名を指定して、「編集」をクリックする。EGL マイグレーション設定エディターが表示されます。この手法の利点は、設定ファイルの変更を完了した後、ステージ 1 マイグレーション・ツールを実行できる状態になることです。

どちらの手法を使用する場合も、EGL マイグレーション設定エディターによって、ステージ 1 マイグレーション・ツールを制御する設定を行うことができます。設定の編集が完了したら、「保管 (Save)」ボタンまたは「別名保管... (Save As...)」ボタンをクリックして、エディターを閉じます。

注:

- 現在、構成マップを使用していない場合には、162 ページの『マイグレーション・プランと上位構成マップ』を参照してください。
- ドライブとディレクトリーを必要とする設定の場合は、次に示す 2 つの方法のどちらかで情報を指定できます。
  - 絶対パス。例: d:\tempMig\MySystem\
  - 相対パス。この場合、パスは作業ディレクトリーに対する相対パスです。例えば、次のようになります。
    - .\tempMig\MySystem は、絶対パス  
VisualAge-Smalltalk-installation-directory\image\tempMig\MySystem に相当します。
    - ..\tempMig\MySystem は、絶対パス  
VisualAge-Smalltalk-installation-directory\tempMig\MySystem に相当します。

以降では、変更できる設定について、GUI 内でその設定が表示されるページ別に説明します。

- プランの作成 (Build Plans)
- マッピング (Mapping)
- 名前変更
- 実行 (Execution)

## 「プランの作成 (Build Plans)」ページ

「プランの作成 (Build Plans)」ページでは、マイグレーション・プランを配置する場所に関する情報を指定できます。また「プランの作成 (Build Plans)」ページでは、マイグレーション対象にするライブラリー内の構成マップとバージョンを指示することもできます。「プランの作成 (Build Plans)」ページは、次のセクションで編成されています。

- 「マイグレーション・プラン仕様 (Migration Plan Specification)」の情報は、ステージ 1 マイグレーション・ツールがマイグレーション・プラン・ファイルの読み取りまたは書き込みを行う場所を指定します。
  - プラン・ディレクトリー (Plan Directory)。これは、ステージ 1 マイグレーション・ツールがマイグレーション・プラン・ファイルを配置するターゲット・ディレクトリーです。
  - プラン・ファイル名 (Plan File Name)。マイグレーション・データベースをロードするために作成して使用する、マイグレーション・プラン・ファイルのファイル名 (オプション)。「プラン・ファイル名 (Plan File Name)」ボタンをクリックして、プラン・ディレクトリーにある既存のプラン・ファイルを表示できます。プラン・ファイル内の詳細を表示する必要がある場合は、「**プランの表示 (View Plans)**」ボタンをクリックしてプラン・ファイルを展開し、マイグレーション・セットを表示します。
  - プラン・ファイル名を指定しない場合は、プラン・ファイルを新規に作成する前に、指定したプラン・ディレクトリーにあるすべての .pln ファイルがマイグレーション・ツールによって削除されます。マイグレーション・ツールは、それぞれのマイグレーション・セットごとにプラン・ファイルを 1



つつつ作成します。この場合、マイグレーション・プラン・ファイル名の形式は `migrationSetName_version.pln` です。

- プラン・ファイル名を指定した場合は、指定したプラン・ファイル名の `.pln` ファイルを新規に作成する前に、指定した `.pln` ファイルの**み**が、指定したプラン・ディレクトリーからマイグレーション・ツールによって削除されます。この場合は、単一のプラン・ファイルにマイグレーション・セットがすべてリストされます。
- 「リポジトリ・フィルター (*Repository Filters*)」の情報をを使用して、ステージ 1 マイグレーション・ツールによって処理する Smalltalk ライブラリーの構成マップとバージョンを制御できます。構成マップとバージョンを制限することにより、ステージ 1 マイグレーション・ツールのパフォーマンスを大幅に向上できます。複数のフィルターを指定できます。ステージ 1 マイグレーション・ツールは、構成マップ・フィルターと、バージョン名フィルターまたはバージョン深さフィルターを次のように使用します。
  - マイグレーション・ツールは、ライブラリーにあるそれぞれの構成マップ名を、構成マップ・フィルターと突き合わせます。
    - 構成マップ名が構成マップ・フィルターのどれともマッチングしない場合、構成マップは以後の処理対象になりません。
    - 構成マップ名が構成マップ・フィルターの少なくとも 1 つとマッチングする場合、構成マップのバージョンは次のように処理されます。
      - バージョン名フィルターを指定した場合は、構成マップの各バージョン名と、バージョン名フィルターのリストの突き合わせが行われます。バージョン名がバージョン名フィルターのいずれかとマッチングする場合、そのバージョンは以後の処理対象になります。
      - バージョン深さフィルターを指定し、バージョン名フィルターを指定しなかった場合は、構成マップの最新バージョン (バージョン深さフィルターに指定した数まで) が以後の処理対象になります。デフォルトのバージョン深さフィルターは 1 です。

注: バージョン深さとバージョン名は相互に排他的です。デフォルトでは、バージョン深さフィルターが `MigPreferences.xml` ファイルに組み込まれています。

- 構成マップ名とバージョン名の突き合わせの結果、構成マップ・バージョンが以降の処理対象になった場合、ステージ 1 マイグレーション・ツールは次の処理を行います。
  - 構成マップ・バージョンが上位構成マップならば、マイグレーション・ツールはその構成マップ・バージョンをマイグレーション・セットの作成の基礎として使用します。バージョン名がバージョン・フィルターとマッチングすることを前提とすると、上位構成マップのそれぞれのバージョンごとに、異なるマイグレーション・セットが作成されます。
  - 構成マップ・バージョンが上位構成マップでなければ、構成マップ・バージョンは以後の処理対象になりません。その構成マップ・バージョンは、他のマイグレーション・セットにも組み込まれている可能性があります (マイグレーション・セットは、単にこの構成マップ・バージョン専用ではありません)。

リポジトリ・フィルター情報は、次のように指定します。

- 構成マップ・フィルター。マイグレーション・ツールは、ライブラリー内の構成マップ名と、ユーザー指定の構成マップ・フィルターを突き合わせます。構成マップ・フィルターは複数指定できます。フィルターを追加、変更、または除去するには、フィルターを右クリックして、コンテキスト・メニューのオプションを使用します。フィルターに大/小文字の区別はありません。フィルターには、次のようにワイルドカードを使用できます。
  - 構成マップ・フィルター `*xyz*` は、ライブラリー内にある、ストリング「xyz」を名前のどこかに含む構成マップ名すべてとマッチングします。
  - 構成マップ・フィルター `xyz*` は、ライブラリー内にある、「xyz」から始まる構成マップ名すべてとマッチングします。
  - 構成マップ・フィルター `*xyz` は、ライブラリー内にある、「xyz」で終わる構成マップ名すべてとマッチングします。
- バージョン名 フィルター。構成マップ名が構成マップ・フィルターとマッチングする場合、マイグレーション・ツールはバージョン名フィルターを使用して、マイグレーション対象になる構成マップ・バージョン (存在すれば) を判別します。バージョン名フィルターは複数指定できます。フィルターを追加、変更、または除去するには、フィルターを右クリックして、コンテキスト・メニューのオプションを使用します。フィルターに大/小文字の区別はありません。フィルターには、次のようにワイルドカードを使用できます。
  - バージョン名フィルター `*xyz*` は、ストリング「xyz」をバージョン名のどこかに含む構成マップ・バージョン名すべてとマッチングします。
  - バージョン名フィルター `xyz*` は、「xyz」から始まる構成マップ・バージョン名すべてとマッチングします。
  - バージョン名フィルター `*xyz` は、「xyz」で終わる構成マップ・バージョン名とマッチングします。
  - 「バージョン名フィルター (Version Name filters)」フィールドを空のままにした場合、マイグレーション・ツールはバージョン深さフィルターを使用します。
- バージョン深さ フィルター。マイグレーション対象にする、前のバージョンの数を指定できます。デフォルトは 1 で、この場合マイグレーション・ツールは構成マップの最新バージョンのみを処理します。バージョン名フィルターを指定した場合、バージョン深さフィルターは無視されます。

## 「マッピング (Mapping)」 ページ

「マッピング (Mapping)」 ページでは、次の項目を指定できます。

- 共通パーツと未使用パーツに対する EGL ファイル名。
- EGL のプロジェクト名とパッケージ名の作成に使用する接尾部。
- アプリケーション名を EGL パッケージ名に変換する方法を制御するオプション。
- 共通パーツを含む VAGen 構成マップとアプリケーションに関する情報。

次に、「マッピング (Mapping)」 ページの設定について詳しく説明します。

- ファイル名 (File Names)。「ファイル名 (File Names)」 セクションでは、マイグレーション時に作成される 2 つの EGL ファイルの名前を制御できます。

- 「共通パーツ (Common Parts)」には、マイグレーション・セットの有効範囲内で複数の固有な生成可能パーツに共通のパーツを格納する、EGL ファイルの名前を指定できます。ファイル名は、拡張子またはパスを付けずに指定します。マイグレーション・ツールは、マイグレーション・セットにある複数の生成可能パーツによって使用されて (関連付けられて) いるパーツを含んでいるか、または共通構成マップまたはアプリケーションとして指定された VAGen 構成マップまたはアプリケーション内のパーツを含んでいる、それぞれの EGL パッケージ内に 1 つの共通パーツ・ファイルを作成します。パーツがプログラムと一緒にファイルに配置されるか、共通パーツ・ファイル内に配置されるかの判別について詳しくは、42 ページの『EGL ファイル内でのパーツの配置』を参照してください。
- 「未使用パーツ (Unused Parts)」には、マイグレーション・セットの有効範囲内で使用されていないパーツを格納する EGL ファイルの名前を指定できます。ファイル名は、拡張子またはパスを付けずに指定します。マイグレーション・ツールは、対応する VAGen 構成マップおよびアプリケーションが共通の構成マップまたはアプリケーションとして指定されていない場合には、マイグレーション・セットにある生成可能パーツによって使用されて (関連付けられて) いないパーツを含んでいる、それぞれの EGL パッケージ内に 1 つの未使用パーツ・ファイルを作成します。
- スパン・マップ (Spanning Maps)。「スパン・マップ (Spanning Maps)」セクションでは、1 つのマップ・グループに複数の構成マップまたはアプリケーションからのマップが含まれている場合に使用される接尾部を指定できます。
  - 「プロジェクト接尾部 (Project Suffix)」には、新規 EGL プロジェクト名を作成するための接尾部を指定できます。ステージ 1 マイグレーション・ツールは、この接尾部をマイグレーション・セット名に連結します。マイグレーション・ツールは、マップ・グループとそのマップがマイグレーション・セット内で複数の VAGen 構成マップにわたっている場合に限り、この新規 EGL プロジェクトを作成します。マイグレーション・ツールは、名前変更規則がすべて適用された後で、マイグレーション・セット名に接尾部を連結します。
  - 「パッケージ接尾部 (Package Suffix)」には、EGL プロジェクト内で新規 EGL パッケージ名を作成するための接尾部を指定できます。ステージ 1 マイグレーション・ツールは、この接尾部をプロジェクト名に連結します。マイグレーション・ツールは、マップ・グループとそのマップが構成マップ内で複数の VAGen パッケージにわたっている場合に限り、この新規 EGL パッケージを作成します。マイグレーション・ツールは、EGL プロジェクト名を作成するための名前変更規則がすべて適用された後で、接尾部を連結します。
- EGL パッケージ命名オプション (EGL Package Naming Options)。「EGL パッケージ命名オプション (EGL Package Naming Options)」セクションでは、Smalltalk アプリケーション名を Java パッケージ名に変換するための一般規則を指定できます。
  - パッケージ名にドット表記を使用 (Use package naming dot notation)。このオプションを選択すると、マイグレーション・ツールはアプリケーション名の中で先頭より後にある大文字の前にドットを付けることによって、VAGen アプリケーション名を EGL パッケージ名に変換します。このオプションを選択すると、マイグレーション・ツールは例えば `MyOrderEntryApp` を `My.Order.Entry.App` に変更します。

- パッケージ名を小文字に変換 (*Convert package names to lowercase*)。このオプションを選択すると、マイグレーション・ツールは大文字を小文字に変換することによって、VAGen アプリケーション名を EGL パッケージ名に変換します。このオプションを選択すると、マイグレーション・ツールは例えば `MyOrderEntryApp` を `myorderentryapp` に変更します。

通常は、両方のオプションを選択する必要があります。両方とも選択すると、マイグレーション・ツールは `MyOrderEntryApp` を `my.order.entry.app` に変更します。EGL パッケージ命名オプションは、すべての名前変更規則の**後**で適用されます。

- 共通 ID (*Common Identifiers*)。このセクションでは、共通 (共用) パーツを含む構成マップとアプリケーションを判別するためにマイグレーション・ツールが使用できるストリングのリストを、ワイルドカードとともに指定できます。共通 ID を追加または削除するには、フィールドを右クリックして、コンテキスト・メニューのオプションを使用します。ID を追加するときには、次の情報を入力するようにエディターからプロンプトが出されます。
  - 「コンテキスト (*context*)」は、ストリングを突き合わせる対象を、構成マップ名またはアプリケーション名、あるいはその両方に指定します。
  - 「構成マップ (*ConfigMap*)」を使用すると、共通パーツを含む構成マップを識別するストリングを指定できます。マイグレーション・ツールは、このストリングをマイグレーション・セット内の各構成マップ名と突き合わせて、構成マップに共通パーツが含まれているかどうかを判別します。構成マップ名がストリングのいずれかとマッチングする場合、その構成マップ内のパーツすべてが「使用済み」と見なされます。生成不可のパーツはそれぞれ、プログラム・ファイル内、または共通パーツ・ファイル名の設定に指定されたファイルに置かれます。パーツがマイグレーション・セット内の生成可能パーツによって使用されていない場合であっても、そのパーツは未使用パーツ・ファイルには置かれませんが、「構成マップ (*ConfigMap*)」のストリングは複数入力できます。
  - 「アプリケーション (*Application*)」を使用すると、共通パーツを含むアプリケーションを識別するストリングを指定できます。マイグレーション・ツールは、このストリングをマイグレーション・セット内の各アプリケーション名と突き合わせて、アプリケーションに共通パーツが含まれているかどうかを判別します。ストリングがアプリケーション名にマッチングする場合、そのアプリケーション内のパーツはすべて「使用済み」と見なされます。生成不可のパーツはそれぞれ、プログラム・ファイル内、または共通パーツ・ファイル名の設定に指定されたファイルに置かれます。パーツがマイグレーション・セット内の生成可能パーツによって使用されていない場合であっても、そのパーツは未使用パーツ・ファイルには置かれませんが、「アプリケーション (*Application*)」のストリングは複数入力できます。
  - 「両方 (*Both*)」を使用すると、マイグレーション・ツールがマイグレーション・セット内の構成マップ名とアプリケーション名の両方に突き合わせるストリングを指定できます。「両方 (*Both*)」は、「構成マップ (*ConfigMap*)」のコンテキストと「アプリケーション (*Application*)」のコンテキストに同じストリングを指定した場合に相当します。
  - 「共通コードを識別するためのパターン (*Pattern to identify the common code*)」には、指定したコンテキストに基づいてマイグレーション・ツールが突



き合わせるストリングを指定できます。ストリングの先頭または末尾で、\* をワイルドカードとして使用できます。フィルターに大/小文字の区別はありません。

## 「名前変更」ページ

「名前変更 (Renaming)」ページを使用して、VAGen 構成マップ名、アプリケーション名、およびバージョン名から得られる EGL プロジェクト、パッケージ、およびバージョンの名前を制御できます。「順序 (Order)」列の番号は、マイグレーション・ツールが名前変更規則を適用する順序を示し、最も小さい番号の規則が最初に適用されます。名前変更規則を追加または削除するには、規則をクリックして、コンテキスト・メニューのオプションを使用します。規則を追加すると、新規の規則は常にリストの末尾に置かれます。規則を追加するときには、次の情報を入力するようにエディターからプロンプトが出されます。

- 「変更するストリング (*from string*)」は、VAGen 名の中で変更したい文字を指定します。
- 「変更後のストリング (*to string*)」は、変更結果の EGL 名の中で使用したい文字を指定します。
- 「ストリング・コンテキスト (*string context*)」は、マイグレーション・ツールが名前変更を行うときに、VAGen 名の中で変更するストリングを検索する位置を指定します。値は次のとおりです。
  - 「前 (*front*)」は、変更するストリングが構成マップ名、アプリケーション名、またはバージョン名の先頭にある場合に規則を適用するように指定します。
  - 「後 (*back*)」は、変更するストリングが構成マップ名、アプリケーション名、またはバージョン名の末尾にある場合に規則を適用するように指定します。
  - 「任意 (*any*)」は、変更するストリングが構成マップ名、アプリケーション名、またはバージョン名の中でどの位置にあっても規則を適用するように指定します。
  - 「トークン (*token*)」は、変更するストリングが構成マップ名、アプリケーション名、またはバージョン名と完全に一致する場合に限って規則を適用するように指定します。
- 「マッピング・コンテキスト (*mapping context*)」は、マイグレーション・ツールが構成マップ名、アプリケーション名、またはバージョン名のどれに名前変更規則を適用するかを指示します。「マッピング・コンテキスト (*mapping context*)」の値は、次のとおりです。
  - 「構成マップ (*configMap*)」は、名前変更規則を VAGen 構成マップ名のみに適用するように指定します。
  - 「アプリケーション (*application*)」は、名前変更規則を VAGen アプリケーション名のみに適用するように指定します。
  - 「両方 (*both*)」は、名前変更規則を VAGen 構成マップ名と VAGen アプリケーション名の両方に適用するように指定します。
  - 「バージョン (*version*)」は、名前変更規則をすべての構成マップのバージョン名に適用するように指定します。ご使用のバージョン名が、ディレクトリー名やファイル名に使用できないセミコロン (;) などの特殊文字を含んでいる場合は、バージョン名変更規則を使用します。デフォルトの `MigPreferences.xml` ファイルには、バージョン名が無効なディレクトリー名やファイル名にならない

ようにするために役立つ、いくつかのバージョン名変更規則が組み込まれています。マイグレーション・ツールは、名前変更済みのバージョンを使用して、マイグレーションのステージ 1 でマイグレーション・プラン・ファイル名を作成し、ステージ 3 でディレクトリー名を作成します。

## 「実行 (Execution)」 ページ

「実行 (Execution)」 ページでは、マイグレーション・データベースの場所に関する情報、およびステージ 1 の実行中に収集するロギング、デバッグ、およびレポートの情報を指定できます。次に、「実行 (Execution)」 ページで指定できる設定について詳しく説明します。

- 「データベース (Database)」 情報。このセクションでは、マイグレーション・データベースに関する詳細を次のように指定できます。
  - 「DB」 は、マイグレーション・ツールがマイグレーション・セット情報を書き込むマイグレーション・データベースの名前です。マイグレーション・データベースの作成時の VGMIG からデータベース名を変更した場合は、この設定に指定されているデータベース名を変更して、使用した名前と一致させる必要があります。
  - 「スキーマ (Schema)」 は、データベース・テーブル用の修飾子として使用される名前です。スキーマを指定しない場合、マイグレーション・ツールはデフォルトとして MIGSCHEMA を使用します。マイグレーション・データベースの作成時の MIGSCHEMA からスキーマ名を変更した場合は、この設定に指定されているスキーマ名を変更して、使用した名前と一致させる必要があります。
  - 「ユーザー ID (Userid)」 は、マイグレーション・データベースへの接続に必要なユーザー ID です。ユーザー ID を指定しない場合、マイグレーション・ツールはデフォルトとして、VAGen SQL 設定に指定されているユーザー ID を使用して接続を試みます。接続に失敗した場合、マイグレーション・ツールはログオン・ユーザー ID の使用を試みます。どちらの試行も失敗した場合、マイグレーション・ツールは情報を尋ねるダイアログ・ウィンドウを表示します。
  - 「パスワード (Password)」 は、マイグレーション・データベースへの接続に必要なパスワードです。パスワードを指定しない場合、マイグレーション・ツールはデフォルトとして、VAGen SQL 設定に指定されているパスワードを使用して接続を試みます。接続に失敗した場合、マイグレーション・ツールはログオン・パスワードの使用を試みます。どちらの試行も失敗した場合、マイグレーション・ツールは情報を尋ねるダイアログ・ウィンドウを表示します。

**注:** パスワードは設定ファイル内で暗号化されません。このことが問題になる場合は、設定ファイルにパスワードを入力せず、プロンプトが出されるまで待ってください。

- 「サービス (Service)」 情報。このセクションでは、ステージ 1 の間に収集したいロギングとデバッグの情報に関する詳細を次のように指定できます。指定できる項目は次のとおりです。
  - 「トレース・レベル (Trace level)」 には、ログ・ファイルとデバッグ・ファイルに書き込む情報のレベルを指定できます。次の値から 1 つを指定できます。
    - 重大 (レベル 1) -- エラー・メッセージがログに記録されます。



- 警告 (レベル 2) -- 警告メッセージとエラー・メッセージがログに記録されます。
- 情報 (INFO) (レベル 3) -- 情報メッセージ、警告メッセージ、およびエラー・メッセージがログに記録されます。
- デバッグ (DEBUG) (レベル 4) -- デバッグ情報が、情報メッセージ、警告メッセージ、およびエラー・メッセージとともに記録されます。トレース・レベルが「デバッグ (DEBUG)」の場合のみ、マイグレーション・ツールはデバッグ・ファイルに情報を書き込みます。
- 「ログ・ファイル名」には、ログ・ファイルのドライブ、ディレクトリー、およびファイル名を指定できます。任意のファイル拡張子を付けてログ・ファイルを作成できますが、このファイルは .xml ファイルとして最適に表示されます。ログ・ファイル名を省略した場合は、「ログ・ファイル名」フィールドに指定したドライブとディレクトリーに、*migLog.xml* という名前のファイルが書き込まれます。ドライブとディレクトリーを指定しない場合、マイグレーション・ツールはログ・ファイルをマイグレーション・プラン・ディレクトリーに書き込みます。
- 「デバッグ・ファイル名 (Debug File Name)」には、IBM サポートが必要とするデバッグ・ファイルのドライブ、ディレクトリー、およびファイル名を指定できます。任意のファイル拡張子を付けてデバッグ・ファイルを作成できますが、このファイルは .xml ファイルとして最適に表示されます。このファイルへの情報の書き込みは、トレース・レベルが「デバッグ (DEBUG)」に設定されている場合に限って行われます。デバッグ・ファイル名を省略した場合に、トレース・レベルとして「デバッグ (DEBUG)」を指定すると、「デバッグ・ファイル名 (Debug File Name)」フィールドに指定したドライブとディレクトリーに *migDebug.xml* が書き込まれます。ドライブとディレクトリーを指定しない場合、マイグレーション・ツールはデバッグ・ファイルをマイグレーション・プラン・ディレクトリーに書き込みます。
- 「検証 (Verification)」情報。このセクションでは、ステージ 1 マイグレーション・ツールから出力できるレポート・ファイルに関する情報を指定できます。指定できる項目は次のとおりです。
  - 「レポート・ファイル名 (Report File Name)」には、レポート・ファイルに使用するドライブ、ディレクトリー、およびファイル名を指定できます。このレポートには、VAGen ファイルのマイグレーション方法に関する情報があります。必ず .htm 拡張子を指定してください。レポート・ファイル名を省略した場合は、「レポート・ファイル名 (Report File Name)」フィールドに指定したドライブとディレクトリーに、*report%MigrationReport.htm* という名前のファイルが書き込まれます。ドライブとディレクトリーを指定しない場合、マイグレーション・ツールはレポート・ファイルをマイグレーション・プラン・ディレクトリーに書き込みます。

## MigPreferences.xml ファイルのサンプル

次に、MigPreferences.xml ファイルのサンプルを示します。

```
<preferences>
  <database>
    <uri>VGMIG</uri>
    <schema>MIGSCHEMA</schema>
    <userid></userid>
    <password></password>
```

```

</database>
<migrationSpec>
  <directory>d:¥TempMig¥Stage1</directory>
  <filename></filename>
</migrationSpec>
<service>
  <traceLevel>4</traceLevel>
  <logfile>d:¥TempMig¥stage1¥migLog.xml</logfile>
  <debugfile>d:¥TempMig¥stage1¥migDebug.xml</debugfile>
</service>
<repositoryFilters>
  <projectName>MyConfigMap*</projectName>
  <versionNumber>1</versionNumber>
</repositoryFilters>
<verification>
  <reportName>d:¥TempMig¥report¥MigrationReport.htm</reportName>
</verification>
<eglMapping>
  <commonPartsFileName>CommonParts</commonPartsFileName>
  <unusedPartsFileName>UnusedParts</unusedPartsFileName>
  <spanningMapsProjectSuffix>MapsProject</spanningMapsProjectSuffix>
  <spanningMapsPackageSuffix>mapspackage</spanningMapsPackageSuffix>
  <packageDotNotation>true</packageDotNotation>
  <packageLowercase>true</packageLowercase>
  <commonParts>
    <commonConfigMap>*Common*</commonConfigMap>
    <commonApplication>*Common*</commonApplication>
  </commonParts>
  <renameRule order="1">
    <fromString></fromString>
    <toString></toString>
    <stringContext>any</stringContext>
    <mappingContext>both</mappingContext>
  </renameRule>
  <renameRule order="101">
    <fromString>CM</fromString>
    <toString></toString>
    <stringContext>back</stringContext>
    <mappingContext>configMap</mappingContext>
  </renameRule>
  <renameRule order="301">
    <fromString>App</fromString>
    <toString></toString>
    <stringContext>back</stringContext>
    <mappingContext>application</mappingContext>
  </renameRule>
  <renameRule order="501">
    <fromString>:</fromString>
    <toString>_</toString>
    <stringContext>any</stringContext>
    <mappingContext>version</mappingContext>
  </renameRule>
  <renameRule order="502">
    <fromString></fromString>
    <toString>_</toString>
    <stringContext>any</stringContext>
    <mappingContext>version</mappingContext>
  </renameRule>
  <renameRule order="503">
    <fromString>¥</fromString>
    <toString>_</toString>
    <stringContext>any</stringContext>
    <mappingContext>version</mappingContext>
  </renameRule>
  <renameRule order="504">
    <fromString>|</fromString>
    <toString>_</toString>

```

```

        <stringContext>any</stringContext>
        <mappingContext>version</mappingContext>
    </renameRule>
    <renameRule order="505">
        <fromString>?</fromString>
        <toString>_</toString>
        <stringContext>any</stringContext>
        <mappingContext>version</mappingContext>
    </renameRule>
    <renameRule order="506">
        <fromString>*</fromString>
        <toString>_</toString>
        <stringContext>any</stringContext>
        <mappingContext>version</mappingContext>
    </renameRule>
    <renameRule order="507">
        <fromString>&lt;</fromString>
        <toString>_</toString>
        <stringContext>any</stringContext>
        <mappingContext>version</mappingContext>
    </renameRule>
    <renameRule order="508">
        <fromString>&gt;</fromString>
        <toString>_</toString>
        <stringContext>any</stringContext>
        <mappingContext>version</mappingContext>
    </renameRule>
    <renameRule order="509">
        <fromString>&quot;</fromString>
        <toString>_</toString>
        <stringContext>any</stringContext>
        <mappingContext>version</mappingContext>
    </renameRule>
    <renameRule order="510">
        <fromString></fromString>
        <toString>_</toString>
        <stringContext>any</stringContext>
        <mappingContext>version</mappingContext>
    </renameRule>
</eglMapping>
</preferences>

```

## 設定からのファイル名の取得

ステージ 1 マイグレーション・ツールは、ログ・ファイル、デバッグ・ファイル、およびレポート・ファイルの名前に使用するファイル名を同じ方法で取得します。次の表に、設定の中で指定できる名前と、その結果としてマイグレーション・ツールが使用するドライブ、ディレクトリー、およびパス名を示します。この例では、マイグレーション・プラン・ディレクトリーは `d:¥myVAGenMig` です。

表 58. 設定から得られるファイル名

ログ・ファイル名の設定	ステージ 1 マイグレーション・ツールが使用するファイル名
設定はブランクのまま。	<code>d:¥myVAGenMig¥migLog.xml</code> <b>注:</b> <ul style="list-style-type: none"> <li>デバッグ・ファイルのデフォルトのファイル名は <code>migDebug.xml</code> です。</li> <li>レポート・ファイルのデフォルトのファイル名は <code>¥report¥MigrationReport.xml</code> です。</li> </ul>
<code>mine.xml</code>	<code>d:¥myVAGenMig¥mine.xml</code>
<code>logs¥mine.xml</code>	<code>d:¥myVAGenMig¥logs¥mine.xml</code>

表 58. 設定から得られるファイル名 (続き)

ログ・ファイル名の設定	ステージ 1 マイグレーション・ツールが使用するファイル名
.mine.xml	VisualAge-Generator-installation-directory¥image¥mine.xml

## ステージ 1 ツールを実行する前に — ヒント

ステージ 1 マイグレーション・ツールを実行する前に、いくつかの処置を行ってパフォーマンスを高めることができます。また、マイグレーションが完了した後で使用するために、既存のイメージを保管することもできます。

### パフォーマンスの向上

メモリーの使用量を最小限にするために、ステージ 1 マイグレーション・ツールを実行する前にイメージをクリーンアップ (または「消し込み」) することが最良策です。イメージをクリーンアップ (または「消し込み」) するには、次の手順で行います。

1. システム・トランスクリプトから、「ツール (Tools)」→「VAGen ツール・ワークスペースを開く (Open VAGen Tools Workspace)」を選択する。
2. 「イメージ管理 (Image Management)」セクションで、次の項目を選択する。

**System abtScrubImage**

3. 右クリックし、「実行 (Execute)」を選択して System abtScrubImage を実行する。
4. イメージの消し込みを行うと、VAGen EGL マイグレーション・フィーチャーの再ロードが必要になる場合があります。フィーチャーをロードする方法については、146 ページの『マイグレーション・フィーチャーのロード』を参照してください。また、「システム・トランスクリプト」ツールバーに「EGL マイグレーション・ツール (EGL Migration Tools)」オプションを再度追加するには、次の手順で行います。
  - a. 「システム・トランスクリプト」ウィンドウに次のように入力する。  
HptEglMigrationGuiApp loaded
  - b. 入力した行を選択し、右クリックして「実行 (Execute)」を選択する。

ステージ 1 マイグレーション・ツールがマイグレーション対象の構成マップとバージョンの分析に費やす時間を短縮するために、マイグレーションしたい構成マップ・バージョンのみを含むライブラリーの作成を検討してください。マイグレーション中に VisualAge Generator の継続保守を行う場合は、別個のマイグレーション・ライブラリーを使用すると次のような利点もあります。

- マイグレーションする構成マップ・バージョンのセットが安定します。このことは、「バージョン深さ (Version Depth)」設定を使用してマイグレーション対象を制御する場合に特に重要です。
- 新規マイグレーション・ライブラリーのバージョンを保守ライブラリーと比較して、マイグレーションを必要とする他の構成マップ・バージョンがまだあるかどうか判別できます。

特殊ライブラリーを作成する場合は、ステージ 1 マイグレーションのパフォーマンスを高めるために、ローカル・ライブラリーとして使用することを検討してください。

## イメージの保管

ステージ 1 マイグレーション・ツールは、ステージ 1 処理の開始時に、イメージからの VAGen パーツを含むアプリケーションをワークスペースからすべてアンロードします。ステージ 1 ツールの終了時には、最後に処理されるマイグレーション・セットのみがイメージに残ります。すべてのアプリケーションをアンロードすることにより、ステージ 1 の実行中に、マイグレーション・セット内のパーツのみが関連パーツ・リストの対象と見なされるようになります。保管したいイメージがある場合は、ステージ 1 ツールを実行する前に次の手順で行う必要があります。

1. VisualAge Generator をシャットダウンする。
2. 次のファイルのバックアップ・コピーを、  
¥VisualAgeForSmalltalk-installation-directory¥image に保管する。
  - abt.icx
  - abt.ini (ステージ 1 の実行中に設定を変更しない場合は保管不要)
  - hpt.ini (ステージ 1 の実行中に設定を変更しない場合は保管不要)
3. VisualAge Generator を開始する。

ステージ 1 ツールの実行を完了した後、ワークスペースを復元するには次の手順で行います。

1. VisualAge Generator をシャットダウンする。
2. ステージ 1 ツールを実行する前にバックアップしたファイルを復元する。
3. VisualAge Generator を開始する。

---

## ステージ 1 マイグレーション・ツールの実行

設定の編集を完了した後、ステージ 1 マイグレーション・ツールを実行して、ソース・コードを Smalltalk ライブラリーから抽出できます。このためには、次の手順で行います。

1. 次のいずれかの手法を使用して、「EGL マイグレーション・ドライバ (EGL Migration Driver)」ビューを開始する。
  - a. 設定エディターを開始して設定を変更した場合は、システム・トランスクリプトから「EGL マイグレーション・ツール (EGL Migration Tools)」->「マイグレーション・ドライバ (Migration Driver)」を選択して、「EGL マイグレーション・ドライバ (EGL Migration Driver)」ビューを開始する。
  - b. EGL マイグレーション・ドライバを開始して設定を変更した場合は、設定ファイルを保管したときに、「EGL マイグレーション・ドライバ (EGL Migration Driver)」ビューが再度表示されます。
2. マイグレーション設定ファイルのファイル名が、設定を保管したファイルを指していることを確認する。別の設定ファイルを指定するには、「参照...」ボタンを使用します。設定の検討または最終的な変更を行うには、「編集...」ボタンを使用します。

3. 設定に問題がなければ、使用する実行オプションを選択する。実行オプションは、ステージ 1 マイグレーション・ツールの出力を次のように制御します。
- 「*PLN の上書き (Overwrite PLN)*」は、マイグレーション・プラン・ファイルを次のように制御します。
    - 「*PLN の上書き (Overwrite PLN)*」を選択すると、設定の中で指定したプラン・ディレクトリー内のファイルに対して、ステージ 1 マイグレーション・ツールによって次の処理が行われます。
      - 設定ファイルに .pln ファイルのファイル名が指定されていない場合、マイグレーション・ツールは指定のプラン・ディレクトリーにある .pln ファイルをすべて削除し、ファイルを新規に作成します。
      - 設定ファイルに .pln ファイルのファイル名が指定されている場合、マイグレーション・ツールは .pln ファイルを新規に作成する前に、指定のプラン・ディレクトリーから同名のファイルのみを削除します。

リポジトリ・フィルターと上位構成マップに基づいて、ステージ 1 マイグレーション・ツールにマイグレーション・プラン・ファイルを作成させたい場合は、「*PLN の上書き (Overwrite PLN)*」オプションを使用します。マイグレーションに使用する構成マップを作成するために支援が必要な場合は、163 ページの『上位構成マップの作成』を参照してください。

- 「*PLN の上書き (Overwrite PLN)*」を選択しない場合、ステージ 1 マイグレーション・ツールはマイグレーション・プラン・ファイルを新規に作成しません。代わりに、ステージ 1 マイグレーション・ツールは、設定の中で指定したプラン・ディレクトリーとプラン・ファイル名に基づいて次のように実行されます。
  - 設定ファイルに .pln ファイルのファイル名が指定されていない場合、マイグレーション・ツールは指定のプラン・ディレクトリーにあるプランすべてを使用して実行されます。
  - 設定ファイルに .pln ファイルのファイル名が指定されている場合、マイグレーション・ツールはその .pln ファイルのみを使用して実行されます。

マイグレーション・プラン・ファイルがすでに作成済みであって、これらのファイルを使用してステージ 1 マイグレーション・ツールを実行してマイグレーション・データベースをロードしたい場合は、「*PLN の上書き (Overwrite PLN)*」オプションをオフにします。ユーザー独自のマイグレーション・プラン・ファイルの作成について詳しくは、164 ページの『マイグレーション・プラン・ファイルの手動作成』を参照してください。

- 「*レポートの生成 (Generate Report)*」は、マイグレーション・ツールが設定ファイルの「*検証 (Verification)*」セクションに指定されたレポートを作成するかどうかを制御します。このオプションを選択しない場合、レポートは作成されません。このオプションを選択すると、設定の中で指定したレポート・ファイル名を使用して、マイグレーション・ツールによるレポートが作成されます。このレポートは、構成マップ、アプリケーション、および VAGen パーツが、マイグレーション中に EGL のプロジェクト、パッケージ、およびファイルにどのように割り当てられるかを示します。最初はこのオプションを選択解除しておくことにより、.pln ファイルを検討して、マイグレーション・ツールが目的の構成マップ・バージョンを処理するかどうか確認できます。選択される構



成マップ・バージョンが適切でない場合は、設定を改良して「PLN の上書き (Overwrite PLN)」を再度実行できます。処理される構成マップ・バージョンに問題がなければ、「レポートの生成 (Generate Report)」オプションを選択してステージ 1 を再度実行します。

**注:**

- レポートの生成には時間がかかる可能性があるので、.pln ファイルを検討して、目的の構成マップ・バージョンがマイグレーション・ツールの処理対象になるか確認することが最良策です。
- 「レポートの生成 (Generate Report)」オプションを選択すると、ステージ 1 マイグレーション・ツールは既存のレポート・ファイルをレポート・ディレクトリーから自動的に削除します。前のレポート・ファイルを保管する必要がある場合は、レポート・ファイルを別のディレクトリーに移動するか、新規レポート用のディレクトリーを新しく指定する必要があります。レポート・ファイルは他のファイルとリンクしているので、レポート・ファイルの名前を変更するとリンクが失われ、ファイルが表示できなくなります。
- 「データベースの更新 (Update Database)」は、マイグレーション・ツールがマイグレーション・プラン情報を反映してマイグレーション・データベースを更新するかどうかを制御します。このオプションを選択しない場合、マイグレーション・データベースは更新されません。このオプションを選択すると、設定の中で指定したマイグレーション・データベースが、マイグレーション・プランの情報 (マイグレーション・プランに含まれるすべての VAGen パーツの外部ソース形式など) を反映して更新されます。最初はこのオプションを選択解除しておくことにより、レポートを検討して、構成マップ、アプリケーション、および VAGen パーツが、EGL のプロジェクト、パッケージ、およびファイルにどのように配置されるか確認できます。配置が適切でない場合は、設定を改良してレポートを再度実行できます。配置に問題がなければ、「データベースの更新 (Update Database)」を選択してステージ 1 を最終的に実行することにより、結果がマイグレーション・ツールによってマイグレーション・データベースに書き込まれます。

**注:**

- 「データベースの更新 (Update Database)」オプションを選択した場合に、マイグレーション・セットがデータベースにすでに存在していると、ステージ 1 マイグレーション・ツールは、マイグレーション・セットに関する以前の情報をデータベースから自動的に削除し、マイグレーション・セットに関する新しい情報を追加します。マイグレーション・セットをデータベースからクリーンアップする必要はありません。
  - マイグレーション・ツールは、マイグレーション・プラン全体を自動的にクリーンアップしません。
4. 実行オプションを選択した後、「OK」をクリックして、マイグレーション・ツールのステージ 1 を実行します。マイグレーション設定の中で指定したログ・ファイルに、マイグレーション・ツールのメッセージが書き込まれます。また、システム・トランスクリプトにもツールの同じメッセージが書き込まれます。

「データベースの更新 (Update database)」オプションを選択した場合、ステージ 1 マイグレーション・ツールの終了時に、マイグレーション・プラン情報 (外部ソー

ス形式の VAGen コードなど) がマイグレーション・データベースに格納されます。レポートとステージ 1 メッセージを検討した後、必要に応じて VisualAge Generator 内でコードを変更し、ステージ 1 を再度実行できます。ステージ 1 の結果に問題がなく、最終の外部ソース形式コードがマイグレーション・データベースに格納された後、マイグレーションのステージ 2 を実行できます。ステージ 2 マイグレーション・ツールを実行するには、EGL 開発環境を使用します。マイグレーション・プロセスの継続については、169 ページの『第 6 章 ステージ 2 — EGL 構文への変換』を参照してください。

---

## マイグレーション・プランと上位構成マップ

マイグレーション・プラン・ファイルは単純な XML ファイルで、1 つ以上のマイグレーション・セットの名前を指定し、それぞれのマイグレーション・セットごとに 1 つの上位構成マップとバージョンを指定して、そのマイグレーション・セットのアプリケーションとそのバージョンを指定します。上位構成マップは、他の構成マップとそのバージョンを必須マップとして指定することもできます。ただし、マイグレーション・セットに対して指定できる上位構成マップ・バージョンはただ 1 つです。ステージ 1 マイグレーション・ツールは、構成マップとバージョン名に関するリポジトリ・フィルターの設定に基づいて、マイグレーション・プラン・ファイルを自動的に作成するように設計されています。ステージ 1 ツールは、どの構成マップ・バージョンが上位構成マップ・バージョンであるかを判別するために、これらのフィルターを使用して検討対象になる構成マップ・バージョンを選択します。ステージ 1 ツールは、それぞれの上位構成マップ・バージョンをマイグレーション・セットの基礎として使用します。

VAGen ソース・コードの生成時に上位構成マップを使用した場合は、これらと同じ上位構成マップをマイグレーションに使用します。これは、それぞれの上位構成マップが生成時に一緒に使用されるパーツのグループを指定しているからで、したがって上位構成マップは一連のプログラムの関連パーツをすべて含んでいます。

構成マップを現在まったく使用していない場合は、マイグレーションに使用する構成マップを作成する必要があります。この場合、最も簡単な手法としては、グループとしてマイグレーションしたいアプリケーション・バージョンすべて (共通アプリケーション・バージョンを含む) を組み込んだ構成マップ・バージョンを 1 つ作成します。詳しくは、163 ページの『上位構成マップの作成』を参照してください。構成マップを作成した後、ステージ 1 マイグレーション・ツールを使用して、マイグレーション・プランを自動的に作成できます。

構成マップを現在使用している場合は、上位構成マップが存在しない可能性があります。例えば、共通アプリケーション用の構成マップと、サブシステム用の別の構成マップがあるとします。生成時に、それぞれの構成マップのどのバージョンをイメージにロードするか決定します。この場合は、次のいずれかを行って、グループとしてマイグレーションする対象を指定できます。

- ・ マイグレーション中に使用する上位構成マップを作成する。この上位構成マップには、アプリケーション・バージョンのリスト、必要な構成マップ・バージョンのリスト、またはアプリケーション・バージョンと必要な構成マップ・バージョンの組み合わせを指定できます。例えば、上位構成マップには共通構成マップとサブシステム構成マップをリストして、両方の構成マップがマイグレーション時にグループとして扱われるようにすることができます。

詳しくは、163 ページの『上位構成マップの作成』を参照してください。上位構成マップを作成した後、ステージ 1 マイグレーション・ツールを使用して、マイグレーション・プランを自動的に作成できます。

- 上位構成マップを作成しない場合は、次のいずれかの手法を使用して、マイグレーション・プラン・ファイルをユーザーが独自に作成できます。
  - Smalltalk 構成マップ・バージョンに関連して、生成に必要なものを指定する情報がデータベースや他のシステム内にある場合は、データベースからマイグレーション・プラン・ファイル (複数可) を自動的に作成するツールをユーザーが開発できます。
  - マイグレーション・プラン・ファイル (複数可) を手作業で作成できます。

## 上位構成マップの作成

マイグレーションに使用する上位構成マップを作成するには、VisualAge Generator 内で次の手順で行います。

1. VisualAge Organizer から「オプション」を選択して、「全メニュー (Full Menus)」が選択されていることを確認する。
2. VisualAge Organizer から「ツール (Tools)」 -> 「構成マップ (Configuration Maps)」を選択する。
3. 構成マップ・ブラウザーから「名前 (Names)」 -> 「作成 (Create)」を選択する。
4. 「必須情報 (Information Required)」ウィンドウに構成マップの名前を入力して、「OK」をクリックする。構成マップの新規エディションが自動的に作成され、選択されます。
5. 「アプリケーション (Applications)」 -> 「追加」を選択する。続いて、マイグレーションするそれぞれのアプリケーションと、そのアプリケーションのバージョンを選択する。それぞれのアプリケーションごとに、バージョンをただ 1 つ指定できます。組み込む必要があるそれぞれのアプリケーションごとにバージョンを選択したら、「OK」をクリックします。指定したアプリケーション・バージョンが、グループとしてマイグレーションされます。このグループは、マイグレーション・ツールが未確定状態の解決に使用するパーツのセットを決定します。
6. 「エディション (Editions)」 -> 「バージョン (Version)」を選択して、構成マップのバージョンを設定する。
7. 「エディション (Editions)」 -> 「ロード (Load)」を選択して、構成マップ・バージョンを選択し、イメージにロードする。
8. 新しい上位構成マップをロードした後、プログラムとテーブルの検証を実行して、これらが VAGen 内で有効であることと、欠落しているパーツがないことを確認することもできます。プログラムを検証する際には、/GENMAPS、/GENHELPMAPS、および /NOGENTABLES を組み込みます。これら 2 つのマップ・オプションを指定すると、マップが使用されるプログラム内で有効であることを確認できます。/NOGENTABLES を指定すると、テーブルが使用されるすべてのプログラムを対象にテーブルを再検証せずに、テーブルを一度限り検証できます。

## 構成マップのチェーニング

構成マップはチェーニングできます。例えば、それぞれの共通アプリケーションのバージョンをリストする構成マップを作成できます。その後、サブシステムごとに、サブシステム固有のアプリケーションそれぞれの必要なバージョンを組み込んで、そのサブシステム用の上位構成マップを作成します。次のようにして、サブシステム構成マップ内の共通アプリケーションに構成マップを組み込むことができます。

1. 構成マップ・ブラウザーから、サブシステム構成マップのオープン・エディションを選択する。
2. 「式」 -> 「追加」を選択する。
3. 「必須情報 (Information Required)」ウィンドウで「OK」をクリックして、式として「真 (*true*)」を受け入れる。
4. 「式の構成 (Config. Expressions)」ペインで「真 (*true*)」を選択する。その後、「必須マップ (Required Maps)」 -> 「追加」 -> 「先頭として (As First)」を選択する。続いて、共通アプリケーションを含む構成マップ・バージョンを選択する。
5. 「エディション (Editions)」 -> 「バージョン (Version)」を選択して、構成マップのバージョンを設定する。
6. 「エディション (Editions)」 -> 「必須マップとともにロード (Load With Required Maps)」を選択して、構成マップ・バージョンを選択し、イメージにロードする。

必須マップを使用すると、共通アプリケーション・バージョンを簡単に指定でき、すべてのサブシステムの上位構成マップに共通アプリケーション・バージョンを明示的にリストする必要がなくなります。

## ステージ 1 ツールに対する構成マップの使用

ステージ 1 マイグレーション・ツールの実行準備ができれば、次の作業を行います。

- ステージ 1 設定を行うときに、「プランの作成 (Build Plans)」ページの「リポジトリ・フィルター (Repository Filters)」セクションで構成マップ・リストを設定して、リストにあるフィルターが、作成した上位構成マップにマッチングするようにします。
- 使用する設定ファイルをステージ 1 ツールに指定するときに、「PLN の上書き (Overwrite PLN)」オプションも選択します。このオプションを指定すると、ステージ 1 マイグレーション・ツールは上位構成マップに基づいてマイグレーション・プラン・ファイルを作成し、既存のマイグレーション・プラン・ファイルを上書きします。

## マイグレーション・プラン・ファイルの手動作成

VisualAge Generator での生成時にイメージにロードする構成マップ・バージョンを決定する外部制御手段がすでに存在する場合は、マイグレーション・プラン・ファイルを手動で作成するか、外部情報からマイグレーション・プラン・ファイルを自動的に作成するツールを開発できます。マイグレーション・プラン・ファイルのファイル拡張子は *.pln* にする必要があり、フォーマットは次のとおりです。

```

<migrationDefinition>
  <migrationSet
    name="migrationSet1"
    version="migrationSet1Version1"
    vgName="migrationSet1"
    vgVersion="migrationSet1Version1">
    <configMap
      name="configurationMap1"
      version="configurationMap1Version1">
    </configMap>
    <configMap
      name="configurationMap2"
      version="configurationMap2Version1">
    </configMap>
    .
    .
    .
    <configMap
      name="configurationMapN"
      version="configurationMapNVersion1">
    </configMap>
  </migrationSet>
</migrationDefinition>

```

前述の例では、次のことが当てはまります。

- migrationSet1 は、一緒にマイグレーションする必要がある構成マップのグループを参照するために使用できる名前です。マイグレーション・セット名はマイグレーション・データベースに格納され、マイグレーションの以降のステージで次のように使用されます。
  - ステージ 1 マイグレーションでは、マップ・グループ内のマップが複数の構成マップにわたる場合に、接尾部を連結したマイグレーション・セット名を使用して、マイグレーション・セットとそのマップすべてを格納する新規 EGL プロジェクトの名前が作成されます。名前変更規則を変更する場合にも、マイグレーション・データベースから情報を除去するためにマイグレーション・セット名が使用されます。
  - ステージ 2 マイグレーションの場合、マイグレーション・セット名は、マイグレーション・データベース内で EGL に変換する対象の構成マップ・グループを指定します。
  - ステージ 3 の場合、マイグレーション・セット名は、ワークスペースまたは一時ディレクトリーに EGL プロジェクト、パッケージ、およびファイルを作成するために使用する、マイグレーション・データベース内の構成マップ・グループを指定します。ステージ 3 の出力を一時ディレクトリーに保管する場合、マイグレーション・セット名とマイグレーション・セット・バージョンは、上位ディレクトリー名の作成にも使用されます。

マイグレーション・セット名は、マイグレーション時に構成マップ・グループを識別する手段としてのみ使用されます。マップが VisualAge Generator 内で複数の構成マップにわたっている場合を除き、マイグレーション・セット名はマイグレーション後には使用されません。

- configurationMap1、configurationMap2、... configurationMapN は、グループとしてマイグレーションする構成マップです。configurationMap は、マイグレーション・セット内で一度限りリストする必要があります。



- configurationMap1Version1、configurationMap2Version1、...  
configurationMapNVersion1 は、これらの構成マップそれぞれのバージョンです。1つのマイグレーション・セット内で、それぞれの構成マップごとにバージョンをただ 1 つ指定できます。
- 指定する構成マップ名およびバージョン名は、ライブラリー内の構成マップ名およびバージョン名と完全に一致している必要があります。名前には大/小文字の区別があります。この情報を使用して、イメージに構成マップ・バージョンが追加され、ステージ 1 マイグレーション・レポートの作成、およびマイグレーション・データベースのロードのためにパーツを分析できるようになります。

ステージ 1 マイグレーション・ツールの実行準備ができれば、次の作業を行います。

- ステージ 1 設定を行うときに、「**プランの作成 (Build plans)**」タブで、「**プラン・ディレクトリー名 (Plan directory name)**」を、マイグレーション・プラン・ファイルの保管先のドライブとディレクトリーに設定します。作成したマイグレーション・プランを 1 つだけ使用してステージ 1 マイグレーション・ツールを実行する場合は、「**プラン・ファイル名 (Plan file name)**」を指定します。指定したプラン・ディレクトリーにあるすべてのマイグレーション・プラン・ファイルを使用してステージ 1 マイグレーション・ツールを実行する場合は、「**プラン・ファイル名 (Plan file name)**」をブランクのままにします。
- 作成する出力をステージ 1 ツールに指示する際には、「**PLN の上書き (Overwrite PLN)**」を必ず選択解除してください。これにより、「**プラン・ファイル名 (Plan file name)**」の設定を基に、前に作成した .pln ファイルを使用してマイグレーション・ツールが実行されます。



---

## 第 4 部 ステージ 2 および 3 — 共通のマイグレーション・ステップ

以降のマイグレーション手順は、VisualAge Generator on Java、または VisualAge Generator on Smalltalk のどちらからマイグレーションする場合も同じです。



---

## 第 6 章 ステージ 2 — EGL 構文への変換

マイグレーションのステージ 2 は、Java または Smalltalk のどちらからマイグレーションする場合も同じです。

別のマイグレーション・ツールを実行して、ソースを外部ソース形式の構文から EGL 構文に変換する必要があります。このマイグレーション・ツールは EGL のインストール後に使用可能になるプラグインです。ツールは、バッチ・モードまたは対話モードで実行できます。オプションで、ステージ 2 の完了後にステージ 3 を自動的に実行するように指定できます。

---

### ワークベンチの設定

マイグレーションを開始する前に、ワークベンチの設定を行う必要があります。次のような設定を行います。

- 始動パラメーター
- 必要な EGL 設定
- 推奨設定
- VAGen マイグレーションの設定
- その他の推奨設定

### 始動パラメーター

EGL 開発環境のパフォーマンスを向上させるには、初期設定ファイルにいくつかの始動パラメーターを設定する必要があります。使用する製品に関わらず、パラメーターは同じです。初期設定ファイルは常に製品のインストール・ディレクトリにあります。ファイルの名前はご使用の製品によって異なります。例えば、Rational Application Developer を使用している場合、初期設定ファイルの名前は `rationalsdp.ini` です。始動パラメーターを設定するには、次の手順で行います。

1. テキスト・エディターを使用して、初期設定ファイルを編集する。
2. 次の行を見付ける。

```
VMArgs=-Xj9
```

3. この行を次の 3 行に変更する。

```
VMArgs=-Xj9  
VMArgs=-Xmx700m  
VMArgs=-Xms256m
```

-Xmx を設定することにより使用可能メモリーが増加します。通常、これを実メモリー値未満に設定します。例えば、実メモリーが 1000K であれば、-Xmx を 700 に設定します。これが仮想メモリーの使用の回避に役立ちます。

-Xms の設定により、製品を始動するときに使用されるメモリーの量が増加します。この値を、-Xmx の設定値より小さいか等しい値にします。

4. 初期設定ファイルを保管する。

5. EGL 開発環境を開始する。例えば、Rational Application Developer を使用している場合は、その製品を開始してください。

## 必要な EGL 設定

マイグレーション・ツールを使用する前に、EGL とマイグレーション・ツールの機能を両方とも使用可能にする必要があります。これらの機能を使用可能にするには、次の手順で行います。

- 「ワークベンチ」ウィンドウから、「**ウィンドウ**」 -> 「**設定**」を選択する。
- 「ワークベンチ」設定の横にある「+」を選択して設定を展開し、「**機能 (Capabilities)**」を選択する。
- 「EGL デベロッパー」を展開する。次に、「**EGL 開発**」と「**VisualAge Generator から EGL へのマイグレーション**」の両方を選択する。その他のマイグレーション・ツールを選択する必要はありません。
- 「**OK**」を選択する。

マイグレーションの前に、VisualAge Generator 互換モードも設定する必要があります。VisualAge Generator 互換モードの設定により、「問題」ビューに膨大な数のメッセージが表示されなくなります。この設定を行うには、次の手順で行います。

- 「ワークベンチ」ウィンドウから、「**ウィンドウ**」 -> 「**設定**」を選択する。
- 「**EGL**」を選択する。
- 「*VisualAge Generator との互換性*」設定を選択する。
- 「**OK**」を選択する。
- ワークスペースの完全再ビルドのプロンプトが出されたら、「**OK**」を選択する。

VAGen Web トランザクション・プログラムをマイグレーションする予定の場合は、EGL プロジェクトのタイプを設定しておくべきです。この設定を行うには、次の手順で行います。

- 「ワークベンチ」ウィンドウから、「**ウィンドウ**」 -> 「**設定**」を選択する。
- 「**EGL**」を選択する。
- 「デフォルト EGL Web プロジェクト機能の選択」で以下のことを行なう。
  - 「*EGL の JSF サポート*」を選択解除する。
  - 「*EGL のレガシー Web トランザクションのサポート*」を選択する。
- 「**OK**」を選択する。

## 推奨される設定

「問題」ビューの EGL 検証メッセージの解決に役立つように、次の設定をお勧めします。「ワークベンチ」ウィンドウから「**ウィンドウ**」 -> 「**設定**」を選択して、次に示す設定ページを選択します。

- 「**EGL**」 -> 「**エディター (Editor)**」。「**行番号の表示**」を選択します。
- 「ワークベンチ」 -> 「**エディター (Editors)**」 -> 「**テキスト・エディター (Text Editor)**」。「**行番号の表示**」を選択します。
- 「ワークベンチ」ページ。「**自動的にビルド**」を選択するか、選択解除するかを決めてください。このオプションを選択すると、ファイルを保管するたびに、EGL によってワークスペース内のものがすべて再ビルドされ、検証が実行されま

す。このオプションを選択する場合の利点は、行った変更のフィードバックが即時に得られることです。欠点は、ワークスペース内のパーツの数によっては再ビルドに時間がかかる可能性があることです。このオプションを選択しないと、ファイルの保管時に、EGL による再ビルドは行われません。このオプションを選択解除する場合の利点は、複数のファイルを変更するときに再ビルドが複数回行われないことです。ただし、メッセージに対する変更の結果を「問題」ビューに表示するには、プロジェクトをそのたびに再作成する必要があります（「プロジェクト」 -> 「プロジェクトのビルド」または「プロジェクト」 -> 「すべてビルド」）。マイグレーション・ログのメッセージのリストを処理するときには、「自動的にビルド」を選択解除できます。これにより、再ビルドが行われる時期を制御できます。通常のコード開発を行うときには、このオプションを選択できます。ステージ 3 マイグレーション・ツールは、「自動的にビルド」を常にオフにします。

- 「ワークベンチ」 -> 「ローカル・ヒストリー」。マイグレーション・ツールによって作成される .egl ファイルは、非常に大きくなる可能性があります。このため、「最大ファイル・サイズ (MB)」を大きな値に変更する必要があります (例: 5)。また、バックアップ要件に応じて、「ファイルを保持する日数」と「ファイル当たりのエントリー数」を変更する必要があることもあります。
- 「ワークベンチ」 -> 「パースペクティブ」。EGL パースペクティブ、または Web パースペクティブをデフォルト・パースペクティブとして設定できます。Web アプリケーションを開発しない場合は、EGL パースペクティブを使用します。Web トランザクションをマイグレーションしたり Web アプリケーションを開発したりする予定の場合は、Web パースペクティブを使用します。デフォルト・パースペクティブを設定するには、使用するパースペクティブを選択して、「デフォルトにする」を選択します。
- 「ワークベンチ」 -> 「機能」 -> 「テスター」。「プロファイルおよびロギング」機能を選び、次に「OK」を選択します。「設定」ウィンドウを再オープンします。「ロギング」セクションで「一般 (General)」タブを選択し、「デフォルトのロギング・レベル」を「重大」に変更します。オプションで、「デフォルトのロギング・レベル」を「情報 (INFO)」または「警告」に変更できますが、「なし (NONE)」に設定されたままにしないでください。

## VAGen マイグレーションの設定

次にリストする設定は、マイグレーション・プロセス全体を制御します。特に注記がない限り、これらの設定は、ステージ 1 から 3 のマイグレーション中のステージ 2 と、単一ファイル・マイグレーションの両方に使用されます。これらの設定を行うには、「ワークベンチ」ウィンドウから、「Windows」 -> 「設定」 -> 「VAGen マイグレーション」を選択します。

### シングル・ファイル・モード設定:

- パーツを EGL ファイルに分離。この設定は、単一ファイル・マイグレーション時にのみ使用されます。この設定を選択すると、それぞれのプログラム、マップ・グループ、およびテーブルが個別のファイルに配置されます。その他のパーツは、「VAGen 外部ソース形式ファイルのインポート (Import VAGen External Source Format File)」ウィザードに指定したファイルに配置されます。これは、ファイルごとに 1 つの生成可能パーツを配置するという EGL の要件に準拠しています。この設定を選択しない場合は、「VAGen 外部ソース形式ファイルのインポ

ート (Import VAGen External Source Format File)」ウィザードに指定した EGL ファイルに、すべてのパーツが配置されます。単一ファイル・モードのパーツ配置アルゴリズムについて詳しくは、26 ページの『単一ファイル・マイグレーションの概要』を参照してください。

#### 名前変更ユーザー出口情報:

- **名前変更ユーザー出口 (Rename user exit)**。VAGen マイグレーション・ツールは、パーツ名またはパーツ参照に接頭部を追加することによって、データ項目、レコード、関数、およびマップの単純な名前変更を行います。オプションで、ユーザー出口ルーチンを作成して、より複雑な名前変更を行うことができます。例えば、マイグレーション時にハイフン (-) を下線 (\_) に変更できます。パーツの名前を変更するためのユーザー出口ルーチンを提供する場合は、「名前変更ユーザー出口 (Rename user exit)」の設定を選択します。このオプションを選択する場合は、名前変更ユーザー出口ルーチンに関する追加情報を指定する必要があります。名前変更ユーザー出口の作成に関する詳細、およびサンプル・コードについては、ホワイト・ペーパー「Using the Rename User Exit in the VisualAge Generator to EGL Migration Tool」を参照してください。
- **JAR ファイルのロケーション (JAR file location)**。名前変更ユーザー出口ルーチンを含む、ご使用のシステム上にある .jar ファイルのロケーションを指定します。
- **パッケージ名 (Package name)**。名前変更ユーザー出口ルーチンを含む、.jar ファイル内のパッケージの名前を指定します。
- **クラス名 (Class name)**。名前変更ロジックを含む、パッケージ内のクラスの名前を指定します。このクラスには、メソッド `renameUserExit(String s, Connection c)` が含まれている必要があります。
- **データベースを使用 (Use a database)**。ご使用の名前変更ユーザー出口が、データベースを使用して前のパーツ名と新規パーツ名の間の関係を取得する場合は、このオプションを選択します。

**VisualAge Generator 互換モードを最小化する:** この設定グループによって、VisualAge Generator 互換性モードを必要とするマイグレーション技法が自動的に使用されるのを最小化することができます。これらの設定を選択すると、結果として EGL 「問題」リストのエラー・メッセージやランタイムの振る舞いの変更点が増加する可能性がありますので注意してください。

- **以前の EZESYS 値を初期化しない (Do not initialize old EZESYS values)**。この設定を選択すると、マイグレーション・ツールは、EZESYS の以前の VAGen 値を組みこむための宣言および初期化ステートメントを、追加の変数でそれぞれのプログラム中に組み込みません。マイグレーション・ツールは、IF、WHILE、または TEST 以外の EZESYS を含むステートメントを変換するときには、追加の変数を使用します。この設定を選択した場合に、IF、WHILE、または TEST 以外の EZESYS を含むステートメントが存在すると「問題」リストにエラーが出されます。すべてのステートメントを変換して新規の EGL `sysVar.systemType` 値を使用する計画である場合や、EZESYS を使用していないことがわかっている場合にこの設定の選択を検討します。詳しくは、114 ページの『EZESYS』を参照してください。
- **テーブルに deleteAfterUse を組み込まない (Do not include deleteAfterUse for tables)**。この設定を選択すると、マイグレーション・ツールは、それぞれのプログラムにおいて、テーブルの使用宣言ステートメントの `deleteAfterUse` プロパテ



イーを常時省略します。マイグレーション・ツールは、deleteAfterUse プロパティが省略される状態で、警告メッセージを出します。VisualAge Generator バージョン 4.5 フィックスパック 4 から直接マイグレーションしており、かつすべての製品プログラムが、そのバージョンおよびフィックスパックで生成される場合にこの設定を検討します。システム共通プロダクトまたは VisualAge Generator の前のバージョンからのマイグレーション中にこの設定を選択する場合は、必ず、テーブルの使用宣言から deleteAfterUse プロパティが省略されるすべてのプログラムを十分にテストしてください。

- 項目または変数の evensql=y を受け入れない (*Do not honor evensql=y for items or variables*)。この設定を選択すると、マイグレーション・ツールは、PACK データ項目部分またはレコードの非共用項目をマイグレーションするときに、常に奇数精度 (項目が最大長である場合は 18) を使用します。マイグレーション・ツールは、偶数精度 (evensql=y) を指定された VAGen 項目がある状態で、警告メッセージを出します。ご使用の SQL 表で、SQL の where 文節または EGL の prepare ステートメントで参照する可能性のある列で偶数精度を使用していないことが確実である場合に、この設定を検討します。あるいは、この設定を選択してマイグレーションした後で、マイグレーション・ツールが警告メッセージを出した項目定義を検査することができます。SQL 表定義に指定されていない精度を使用すると、データベース・アクセスのローパフォーマンスの原因になります。
- 互換モードを設定しない (*Do not set compatibility mode*)。この設定を選択すると、マイグレーション・ツールは、生成オプションの部分を変換するときはいつも、vagCompatibility="YES" ビルド記述子を省略します。以下のすべてを行なう場合のみ、この設定を選択します。
  - 『VisualAge Generator 互換モードの最小化』セクションの他の 3 つの設定のすべてを選択します。
  - VisualAge Generator 互換モードが使用不可にされたときに、マイグレーションされたすべてのパーツ名が EGL パーツの命名規則に従っていることを確認します。このために「ユーザー出口の名前変更 (Rename User Exit)」を指定し、出口をコーディングしてマイグレーション中に VAGen パーツを名前変更することができます。
  - マイグレーションを終了した後に、VisualAge Generator 互換モードの EGL 設定を使用不可にします。注: 設定内容に関わらず、マイグレーション・ツールはワークスペースをリフレッシュするときに、VisualAge Generator 互換モードを常にオンにします。

## VAGen マイグレーション・データベースの I/O 設定

以下にリストする設定によって、SQL および DL/I データベース I/O に関して VAGen 構文から EGL 構文へのマイグレーションを制御します。特に注記がない限り、これらの設定は、ステージ 1 から 3 のマイグレーション中のステージ 2 と、単一ファイル・マイグレーションの両方に使用されます。これらを設定するには、「ワークベンチ」ウィンドウから「ウィンドウ」->「Preferences (設定)」->「VAGen マイグレーション」->「VAGen マイグレーション・データベースの I/O 設定」を選択します。

### SQL 設定:

- 結果セット接尾部。VisualAge Generator の場合、SQL REPLACE 関数が対応する UPDATE 関数または SETUPD 関数を参照する必要がある場合に、REPLACE 関

数は関数名を指定する必要があります。EGL の場合は、単一の関数内で複数の入出力がサポートされます。get ステートメントまたは open ステートメントを一意的に識別するために、結果セット ID が使用されます。マイグレーション・ツールは、関数名に結果セット接尾部を連結して結果セット ID を作成します。例えば、関数の名前が MY-SETUPD で、デフォルトの結果セット接尾部 \_RSI01 を使用する場合、関数の open ステートメントに組み込まれる結果セット ID は MY-SETUPD\_RSI01 になります。結果セット接尾部は、ブランク以外の任意の値に設定できます。ただし、どのパーツ名とも競合を起こさないものを必ず選択してください。

- **Prepare 接尾部。** VisualAge Generator では、SQL 入出力ステートメントを実行時に準備する必要がある場合に、実行時ステートメント・ビルド (Execution Time Statement Build) を選択します。対応する EGL ステートメントは、prepare ステートメントです。close、get、execute、および open など、他の入出力ステートメントがどの prepare ステートメントと関連しているか指定できるように、prepare ステートメントには prepare ステートメント ID が組み込まれます。マイグレーション・ツールは、関数名に Prepare 接尾部を連結して prepare ステートメント ID を作成します。例えば、MY-EXEC-TIME-BUILD という名前の関数が存在し、デフォルトの Prepare 接尾部 \_PREP01 を使用する場合、prepare ステートメントに組み込まれる prepare ステートメント ID は MY-EXEC-TIME-BUILD\_PREP01 です。Prepare 接尾部は、ブランク以外の任意の値に設定できます。ただし、どのパーツ名とも競合を起こさないものを必ず選択してください。
- **列名を省略。** VisualAge Generator 内では、SQL レコード内のフィールドに対して SQL 列名が常に指定されます。EGL の場合は、フィールド名と同じならば SQL 列名を省略できます。「列名を省略」を選択すると、SQL 列名がフィールド名と同じである場合に、マイグレーション・ツールは EGL の列プロパティを省略します。列名を省略すると、EGL ソース・コードを読みやすくすることができます。
- **isNullable プロパティを省略。** VisualAge Generator 内では、SQL レコードのすべてのフィールドに NULL 標識変数が常に組み込まれます。VisualAge Generator の振る舞いを保持するために、マイグレーション・ツールは SQL レコードのすべてのフィールドに isNullable プロパティを常に組み込みます。ただし、EGL の場合、列が SQL で非 NULL として定義されていれば、NULL 標識変数を省略できます。「isNullable プロパティを省略」を選択すると、マイグレーション・ツールは SQL レコード内のすべてのフィールドから isNullable プロパティを省略します。isNullable プロパティを省略すると、実行時のパフォーマンスを高めることができます。ただし、「isNullable プロパティを省略」の選択は、SQL 列がすべて非 NULL として定義されていることが確実である場合に限り必要があります。いずれかの SQL 列が NULL 可能である場合は、マイグレーション時に「isNullable プロパティを省略」を選択してはなりません。マイグレーション後に、パフォーマンスを高めたい場合には、SQL レコード内で選択したフィールドの isNullable プロパティを除去できます。
- **isReadOnly プロパティを省略。** VisualAge Generator の場合は、SQL レコード内のそれぞれのフィールドごとに、「読み取り専用」プロパティを明示的に設定する必要があります。SQL レコードに対して複数の表が指定されている場合、「読み取り専用」は常に yes にする必要があります。デフォルトでは、マイグレーション・ツールは SQL レコード内の、複数の SQL 表を参照するすべてのフ

フィールドに関して isReadOnly プロパティを組み込みます。ただし、SQL 表がただ 1 つ存在する場合、EGL isReadOnly プロパティはデフォルトで no に設定され、SQL レコードに対して複数の SQL 表が指定されている場合は yes に設定されます。「isReadOnly プロパティを省略」を選択した場合は、SQL レコードに対して単一の表が指定されていて、かつ VisualAge Generator の「読み取り専用」プロパティが yes に設定されている場合に限り、マイグレーション・ツールによって isReadOnly が組み込まれます。isReadOnly プロパティを省略すると、EGL ソース・コードを読みやすくすることができます。

#### DL/I 設定:

- データベース PCB 接尾部。VisualAge Generator では、同じデータベース名を PSB 内で複数回使用できます。EGL では、PSB はレコードです。データベース名はフィールド名になるものであり、固有でなければなりません。マイグレーション・ツールは、PSB 内のフィールド名をデータベース名、数値 (固有にするため必要なとき)、およびデータベース PCB 接尾部から作成します。これによって、データベース名とプログラム内の他の名前が競合しないようにします。例えば、2 つの異なる PCB に対してデータベース名 COURSE が使用され、かつ COURSE が DL/I セグメント・レコード名としても使用されている場合があります。データベース PCB のデフォルト接尾部 \_dbPCB を使用すると、最初の COURSE PCB のフィールド名は COURSE\_dbPCB になります。2 番目の COURSE PCB のフィールド名は COURSE\_n\_dbPCB になります。ここで *n* は VAGen PSB 内の PCB 数です。DL/I セグメント名は COURSE のままです。データベース PCB 接尾部は、ブランク以外の任意の値に設定できます。ただし、どのパーツ名とも競合を起こさないものを必ず選択してください。
- GSAM PCB 接尾部 データベース PCB 接尾部がデータベース PCB に対して使われるのと同じ方法で、GSAM PCB 接尾部が GSAM PCB に対して使用されます。GSAM PCB 接尾部は、ブランク以外の任意の値に設定できます。ただし、どのパーツ名とも競合を起こさないものを必ず選択してください。

#### VAGen マイグレーション構文の設定

以下にリストする設定によって、VAGen 構文から EGL 構文へのマイグレーションが制御されます。特に注記がない限り、これらの設定は、ステージ 1 から 3 のマイグレーション中のステージ 2 と、単一ファイル・マイグレーションの両方に使用されます。これらの設定を行うには、「ワークベンチ」ウィンドウから、「ウィンドウ」->「設定」->「VAGen マイグレーション」->「VAGen マイグレーション構文の設定」を選択します。

#### 名前変更設定:

- 名前変更接頭部 (*Renaming prefix*)。マイグレーション・ツールは、VAGen データ項目名、レコード名、関数名、またはマップ名が EGL または SQL の予約語である場合、あるいは # 記号または @ 記号から始まっている場合に、この接頭部を使用します。このツールは、名前変更接頭部を VAGen パーツ名に追加して、有効な EGL パーツ名を作成します。例えば、*date* は EGL の予約語です。DATE という名前の関数がある場合に、デフォルトの名前変更接頭部 VAGen\_ を使用すると、マイグレーション・ツールは関数 DATE を VAGen\_DATE に変更します。また、ツールはこの関数の参照もすべて DATE から VAGen\_DATE に変更します。名前変更接頭部は、ブランクまたは EZE 以外の任意の値に設定

できます。また、名前変更接頭部を # 記号または @ 記号で始めることはできません。必ず、どのパーツ名とも競合を起こさないものを選択してください。

- レベル 77 接尾部。VAGen 作業用ストレージ・レコードにレベル 77 項目が含まれている場合に、マイグレーション・ツールはこの接尾部を使用します。EGL はレベル 77 項目をサポートしません。マイグレーション・ツールは、VAGen 作業用ストレージ・レコードを 2 つの EGL レコードに分割します。1 つ目のレコードは、オリジナルの VAGen 作業用ストレージ・レコードと同じ名前で、レベル 77 以外の項目をすべて含みます。2 つ目のレコードは、レベル 77 項目をすべて含みます。マイグレーション・ツールは、オリジナルの作業用ストレージ・レコード名を基に、レベル 77 接尾部を連結してこの 2 つ目のレコードの名前を付けます。例えば、オリジナルの作業用ストレージ・レコードの名前が MYRECORD で、デフォルトのレベル 77 接尾部 \_Level77Items を使用する場合、レベル 77 項目を含む EGL レコードの名前は MYRECORD\_Level77Items になります。レベル 77 接尾部は、ブランク以外の任意の値に設定できます。ただし、どのパーツ名とも競合を起こさないものを必ず選択してください。
- ヘルプ・マップ接尾部。プログラムのメインのマップ・グループと、ヘルプ・マップ・グループに同名のマップが含まれている場合、マイグレーション・ツールはこの接尾部を使用します。EGL の場合、プログラムの両方の書式グループにある formGroup すべてに固有の名前が付いている必要があります。マイグレーション・ツールは、メインのマップ・グループのマップ名と競合するヘルプ・グループのマップ名を変更します。次の例を考えてみましょう。プログラムのメインのマップ・グループは MAPGP1 で、MAP1 を含んでいます。同じプログラムのヘルプ・マップ・グループは MAPGP2 で、MAP1 を含んでいます。デフォルトのヘルプ・マップ接尾部 \_helpmap を使用した場合、マイグレーション・ツールは MAPGP2 内の MAP1 の名前を MAP1\_helpmap に変更します。ヘルプ・マップ接尾部は、ブランク以外の任意の値に設定できます。ただし、どのパーツ名とも競合を起こさないものを必ず選択してください。

#### 他の変換オプション:

- 共用データ項目をプリミティブ項目定義に変換。この設定を選択した場合は、共用データ項目がレコード、テーブル、関数ローカル・ストレージ、関数仮パラメーター・リスト、またはプログラム・パラメーター・リスト内で使用されているときに、マイグレーション・ツールは共用項目をプリミティブ定義に変換します。現在の組織の標準では、新規アプリケーションに共用データ項目を使用することを推奨していない場合は、このオプションを使用すれば、マイグレーション時に既存アプリケーションが共用データ項目を使用しないようにすることができます。
- 小数点位のコンマを小数点に変更 (*Change decimal comma to decimal point*)。ご使用のワークステーションが、小数点を示すために通常はコンマを使用するロケールを指定している場合、マイグレーション・ツールは小数点を使用するように数値リテラルを自動的に変換します。例えば、フランス語およびドイツ語のロケールの場合、マイグレーション・ツールはコンマを小数点に自動的に変換します。ただし、お客様によっては、ロケールとして英語を指定し、通常的小数点設定をコンマに指定変更している場合もあります。この手法を使用している場合は、「小数点位のコンマを小数点に変更 (*Change decimal comma to decimal point*)」の設定を選択する必要があります。これにより、マイグレーション・ツールは数値リテラルのコンマを点に変換します。EGL は、小数点位の標識として点のみをサポートします。



## その他の推奨設定

次に示す追加設定が推奨されます。

- (オプション) 「ようこそ」ビューを閉じる。
- まだ EGL パースペクティブを使用していない場合は、「ウィンドウ」->「パースペクティブを開く」->「その他 (Other)」->「EGL」->「OK」を選択して、このパースペクティブに切り替える。また、VAGen Web トランザクション・プログラムをマイグレーションするか、または新規の EGL PageHandler を開発する場合は、Web パースペクティブに切り替える必要があります。ウィンドウの右上隅のタブにあるパースペクティブのアイコンを右クリックし、「閉じる (Close)」を選択すると、他のパースペクティブを閉じることができます。
- 使用しているパースペクティブに「ナビゲーター」ビューがまだ組み込まれていない場合は、このビューを追加できます。「ナビゲーター」ビューを追加するには、「ウィンドウ」->「ビューの表示」->「その他 (Other)」を選択します。「ビューの表示」ウィンドウから、「基本」を展開し、「ナビゲーター」を選択します。

注: EGL パッケージの削除など、一部のアクティビティは「ナビゲーター」ビューで完全にはサポートされません。プロジェクトまたはパッケージの再構築を行う場合は、「プロジェクト・エクスプローラー」ビューを使用してください。

- 「問題」ビューには、検証エラーが表示されます。このため、「問題」ビューは前景に配置して設定することをお勧めします。「問題」ビューを前景に配置するには、「ワークベンチ」ウィンドウの右下のセクションで「問題」タブを選択します。
- 「問題」ビューで、右上隅の「フィルター...」アイコンを選択します。「フィルター」ウィンドウで、次の設定を行います。
  - 「選択されたリソースのみ」ラジオ・ボタンを選択する。これにより、「問題」ビューのエラー・メッセージが、現在選択されているファイルに関するメッセージのみに制限されます。このようにすると、エラーが発生したときに、1 つずつのファイルに注意を集中することができます。「問題」ビューのタイトル・バーには、フィルターにマッチングしたメッセージの数と、ワークスペース内のプロジェクトすべてに関するメッセージの合計数が表示されます。
  - 「表示する項目の上限」を選択解除する。これにより、メッセージがすべて表示されます。
- 編集のために複数のファイルを開いているとき、「ナビゲーター」ビューでオープン・ファイルを選択するたびに、そのオープン・ファイルが自動的に前景に配置される (そのエディター・セッションをアクティブ・エディターにする) ように「ナビゲーター」ビューを構成できます。このためには、「ナビゲーター」ビューのツールバーにある「エディターにリンク」アイコンを選択します。また、「プロジェクト・エクスプローラー」ビューのツールバーにある「エディターにリンク」アイコンを選択することもできます。

## ステージ 2 VAGen マイグレーション・ファイルの設定

マイグレーションのステージ 2 を実行するツールは、ウィザードから呼び出すことができます。ステージ 2 の準備をするために、ステージ 2 設定を後で使用できるように保管する (オプション) ためのプロジェクトを作成します。ステージ 2 マイグレーション設定を作成するには、次の手順で行います。

1. EGL 開発環境を開始する。必ず、169 ページの『ワークベンチの設定』セクションの説明どおりにワークベンチを設定してください。
2. ステージ 2 ウィザードから、データベース・ドライバのロケーションを尋ねられます。この値がウィザードに自動的に取り込まれるように、この値を格納するクラスパス変数を設定できます。この設定を行う最も簡単な方法は次のとおりです。
  - a. 「ウィンドウ」->「設定」から、「Java」->「ビルド・パス」->「クラスパス変数」を選択する。
  - b. 「新規」ボタンをクリックする。
  - c. 「名前」に対して、`DB2_DRIVER_PATH` と入力する。
  - d. 「パス」に対しては、「ファイル」ボタンをクリックし、`db2java.zip` ファイルにナビゲートする。(これは、ステージ 1 で使用した `db2java.zip` ファイルと同じです。デフォルトでは、ファイルは `¥SQLLIB¥java¥db2java.zip` にあります。)
  - e. 「新規変数エントリ」ウィンドウの「OK」をクリックし、「Preferences (設定)」ウィンドウで「OK」をクリックする。
3. ステージ 2 設定ファイルを保管する場合にファイルを格納できる、シンプルなプロジェクトを作成する。これは、ステージ 2 をバッチ・モードで実行する場合に役立ちます。「ファイル」->「新規」->「プロジェクト」を選択して、このためのウィザードを開始します。「シンプル」を展開し、「プロジェクト」を選択して、「次へ」を選択します。
4. プロジェクトに名前を付ける。例えば `VAGENMIG` です。その後「完了」をクリックする。
5. 「プロジェクト・エクスプローラー」ビューの「その他のプロジェクト」フォルダーに、シンプル・プロジェクトがリストされます。「ナビゲーター」ビュー内では、シンプル・プロジェクトはその他のすべてのプロジェクトとともにアルファベット順にリストされます。
6. プロジェクトを右クリックし、コンテキスト・メニューから「新規」->「その他 (Other)」を選択する。
7. 「EGL への VAGen マイグレーション」を展開し、「VAGen マイグレーション・ファイル (VAGen Migration File)」を選択する。「次へ」をクリックします。
8. 以下の適切なステージ 2 設定を入力する。
  - a. ウィザードの先頭ページで、次の表に説明するように設定を編集する。Tab キーを押してこのフィールドから出るまで、マイグレーション・ツールによる「データベース情報」フィールドの検証は行われません。これにより、情報の入力中にデータベースへの接続が繰り返し試行されることを防止できます。



表 59. ウィザードの先頭ページに入力する設定

設定	意味	値
既存ファイルをロード	この設定により、前に保管したステージ 2 設定ファイルを選択できます。「ファイルを選択」ボタンをクリックして、既存の .vgmig ファイルを選択します。「ロード・ファイル」ボタンをクリックすると、そのファイルから設定が取り込まれ、ウィザードに表示されます。	(オプション) 既存の .vgmig ファイルを選択してロードします。
データベース・ドライバのロケーション	これは、DB2 ドライバのロケーションです。	<code>path_to_db2java.zip¥db2java.zip</code>
データベース・ドライバ	これは、DB2 ドライバの名前です。	この値は、常に <code>COM.ibm.db2.jdbc.app.DB2Driver</code> にする必要があります。この値は、ローカル・データベースと、ローカル側でカタログされたりリモート・データベースの両方に有効です。
データベース名	これは、マイグレーションのステージ 1 で使用した DB2 データベースの名前です。	この値は、次のフォーマットで指定する必要があります。 <ul style="list-style-type: none"> <li><code>jdbc:DB2:databaseName</code></li> </ul> <code>databaseName</code> は、マイグレーションのステージ 1 で使用した DB2 データベースの名前です。ステージ 1 のデフォルト値は <code>VGMIG</code> です。
データベース・スキーマ (Database schema)	これは、マイグレーションのステージ 1 で使用した DB2 データベース・スキーマの名前です。	この値は、マイグレーションのステージ 1 で使用した DB2 スキーマの名前です。ステージ 1 のデフォルト値は <code>MIGSCHEMA</code> です。
データベース・ユーザー ID	これは、マイグレーションのステージ 1 で使用したデータベース・ユーザー ID です。	ステージ 1 に使用したものと同じデータベース・ユーザー ID を使用します (デフォルト値は、ご使用のログオン ID)。
データベース・パスワード	これは、マイグレーションのステージ 1 で使用したデータベース・パスワードです。	ステージ 1 に使用したものと同じデータベース・パスワードを使用します (デフォルト値は、ご使用のログオン・パスワード)。
ログ・ファイルのロケーション	これは、ログ・ファイルが書き込まれるロケーションです。	ファイル・システム内の有効なロケーション (ドライブとディレクトリ) を入力します。
ログ・ファイル名	これは、ステージ 2 メッセージを書き込むログ・ファイルの名前です。	有効なファイル名を入力します。

b. ウィザードの 2 ページ目で、次の表の説明のとおり、設定を編集する。

表 60. ウィザードの 2 ページ目に入力する設定

設定	意味	値
「Java」または「COBOL」ラジオ・ボタン	この選択により、マイグレーション・ツールが Java ソース・フォルダーを含むプロジェクトを作成するかどうかが決まります。	Rational Web Developer または Rational Application Developer を使用している場合は、必ず「Java」ラジオ・ボタンを選択する必要があります。  その他の製品においては、COBOL のみを生成する予定ならば COBOL を選択できます。

表 60. ウィザードの 2 ページ目に入力する設定 (続き)

設定	意味	値
残りの VAGen パーツをマイグレーションする	この設定により、マイグレーション・セット内の生成可能パーツによって参照されないパーツを EGL に変換するかどうかが決まります。 <b>注:</b> 残りの VAGen パーツをマイグレーションする 設定については、UI レコードは他のレコードと同じように扱われます。この設定では、UI レコードを生成可能パーツとはみなされません。	参照されないパーツを EGL に変換する場合は、ボックスを選択します。通常は、「残りのパーツをマイグレーションする」を選択する必要があります。 「残りのパーツをマイグレーションする」を選択しない場合、マイグレーション・セット内で使用されない制御パーツやその他のパーツは、EGL ソースにマイグレーションされません。
ワークスペースにインポート	この設定により、ステージ 2 の完了後に、ステージ 3 (現行ワークスペース内のファイルに EGL をインポートする) を自動的に開始するかどうかが決まります。 <b>注:</b> このチェック・ボックスを選択する場合は、ボックスの下にあるラジオ・ボタンの 1 つを選択して、インポートするバージョン (最新または最も古いバージョン。次の説明を参照) を指定する必要があります。これは、一度にワークスペースに存在できるプロジェクトのバージョンがただ 1 つであるからです。	パーツを EGL に変換した直後に EGL ファイルをインポートする場合は、このボックスを選択します。ファイルを後で (ステップ 3 で) インポートする場合は、このボックスを選択解除されたままにします。 <b>注:</b> <ul style="list-style-type: none"> <li>このオプションを選択した場合、ステージ 3 を別個に実行する必要はありません。マイグレーション・ツールは、ステージ 2 (変換) の直後にステージ 3 (インポート) を自動的に開始し、マイグレーション・プロセスを完了します。</li> </ul>
最新バージョン	この設定は、対象のマイグレーション・セットの最新バージョンをインポートするように指定します。	このオプションでは、「ワークスペースにインポート」チェック・ボックスを選択した場合のみ選択できます。
最も古いバージョン	この設定は、対象のマイグレーション・セットの最も古いバージョンをインポートするように指定します。	このオプションでは、「ワークスペースにインポート」チェック・ボックスを選択した場合のみ選択できます。
既存ファイルを上書き	ステージ 3 (インポート・プロセス) では、ステージ 2 で作成した EGL を使用して、ステージ 1 レポートの中で指定された EGL ファイルを作成し、インポートします。ステージ 3 でインポートしようとしている EGL ファイルと同名の EGL ファイルがワークスペースにすでに存在する場合は、このオプションによってこれらのファイルを上書きするかどうかが決まります。	このオプションでは、「ワークスペースにインポート」チェック・ボックスを選択した場合のみ選択できます。「既存ファイルを上書き」を選択すると、現在マイグレーション中のマイグレーション・セットに、ワークスペース内の既存ファイルに配置する必要があるパーツが含まれている場合に、ステージ 3 マイグレーション・ツールがその状況をどのように処理するかを指定できます。「既存ファイルを上書き」を選択すると、ステージ 3 マイグレーション・ツールは既存のファイルを書き換え、 <b>現行</b> マイグレーション・セット内にあるパーツのみを組み込みます。「既存ファイルを上書き」を選択しない場合、ステージ 3 マイグレーション・ツールは新規パーツすべてを既存ファイルにマージします。新規パーツは、パーツ型別にアルファベット順に配置されます。このオプションの影響について詳しくは、49 ページの『ファイルの上書きとマージ』を参照してください。

表 60. ウィザードの 2 ページ目に入力する設定 (続き)

設定	意味	値
マイグレーションしたファイルを一時ディレクトリーに保管	このオプションを選択すると、EGL ファイルをファイル・システム内のロケーションに保管できます。これにより、プロジェクトの複数バージョン用の EGL ファイルに同時にアクセスできます (ただし、ワークスペース内で一度に表示できるバージョンはただ 1 つ)。このロケーションから、リポジトリに EGL ファイルを直接移動できます。	<p>マイグレーション・セットの複数バージョンをマイグレーションする予定の場合は、次のように設定します。</p> <ul style="list-style-type: none"> <li>それぞれのバージョンを別々のサブディレクトリーに書き込むことができるように、このボックスを選択する。</li> <li>これらのバージョンを配置するサブディレクトリーを格納する「フォルダー (Folder)」を指定する。</li> <li>「今すぐマイグレーションを実行」は選択しない。リソース所要量が大きいので、一時ディレクトリーへのマイグレーションはバッチ・モードでのみ行う必要があります。「今すぐマイグレーションを実行」を選択した場合は、本当にオンライン・モードで実行するかどうか、マイグレーション・ツールから確認を求められます。</li> <li>「現行構成をファイルに保管」を選択する。また、現行構成を .vgmig ファイルとして保管する先のプロジェクトとファイル名も指定する必要があります。オンラインまたはバッチのどちらのモードでマイグレーションを行うかに関係なく、「マイグレーションしたファイルを一時ディレクトリーに保管」を選択した場合は .vgmig ファイルが必要です。バッチ・モードでステージ 2 を実行する場合は、マイグレーション設定を指定するために、保管した .vgmig ファイルを指示します。</li> </ul>
フォルダー (Folder)	これは、EGL ファイルを保管するディレクトリーです。「フォルダー (Folder)」に指定したディレクトリーの下に、それぞれのマイグレーション・セット・バージョンのサブディレクトリーが作成されます。	ファイル・システム内の既存ディレクトリーを指定します。
今すぐマイグレーションを実行	後でマイグレーションを行うために設定ファイルをセットアップするだけでなく、この時点でステージ 2 を実行するように指定します。	「今すぐマイグレーションを実行」を選択してステージ 2 をオンライン・モードで実行するか、または後でステージ 2 をバッチ・モードで実行できるように、「現行構成をファイルに保管」を選択して設定を保管する必要があります。後で参照できるように設定のコピーを保持したい場合は、両方のオプションを選択できます。「マイグレーションしたファイルを一時ディレクトリーに保管」をすでに選択した場合は、「今すぐマイグレーションを実行」を選択しないでください。一時ディレクトリーへの保管は、バッチ・モードでのみ実行できます。「今すぐマイグレーションを実行」の選択は、オンライン・モードでマイグレーションを行うように指示します。

表 60. ウィザードの 2 ページ目に入力する設定 (続き)

設定	意味	値
現行構成をファイルに保管	<p>この設定により、指定中の設定をファイルに保管できます。後で、次のどちらかの方法でステージ 2 を実行できます。</p> <ul style="list-style-type: none"> <li>オンライン・モードの場合は、保管した .vgmig ファイルを右クリックし、コンテキスト・メニューから「マイグレーションの開始 (Start Migration)」を選択する。</li> <li>バッチ・モードの場合は、-importFile オプションを使用して、保管した .vgmig ファイルを指定する。詳しくは、184 ページの『バッチ・モードでのステージ 2 の実行』を参照してください。</li> </ul>	<p>「今すぐマイグレーションを実行」を選択してステージ 2 をオンライン・モードで実行するか、または後でステージ 2 を実行できるように、「現行構成をファイルに保管」を選択して設定を保管する必要があります。後で参照できるように設定のコピーを保持したい場合は、両方のオプションを選択できます。</p> <p>このオプションを選択した場合は、現行構成を .vgmig ファイルとして保管する先のパスとファイル名も指定する必要があります。後でステージ 2 を実行する場合は、マイグレーション設定を指定するために、保管した .vgmig ファイルを指示します。</p>
パス	ファイルを保管する先のプロジェクトを指定します。	<code>¥projectName</code> 。ここで、projectName は保管するファイルを格納するプロジェクトの名前です。
ファイル名	設定を保管する先のファイルの名前を指定します。	<code>fileName</code> 。ここで、fileName は目的のファイルの名前で、ファイル拡張子を付けずに指定します。拡張子 .vgmig が自動的に付加されます。

- c. ウィザードの 3 ページ目では、指定したデータベース内にあるマイグレーション・セットがウィザードによってリストされます。マイグレーションするマイグレーション・セットを選択します。マイグレーション・セットを選択しない場合、マイグレーション・ツールはデータベース内にあるすべてのマイグレーション・セットをマイグレーションします。「完了」をクリックします。

選択したチェック・ボックスの組み合わせによって、ウィザードが実行するアクションが決まります。

- 「現行構成をファイルに保管」を選択した場合は、「完了」ボタンをクリックした後、すべてのオプションが指定したファイルに保管されます。
- 「今すぐマイグレーションを実行」を選択した場合は、「完了」ボタンをクリックした後、ステージ 2 マイグレーションが実行されます。
- さらに、「ワークスペースにインポート」または「マイグレーションしたファイルを一時ディレクトリに保管」のどちらかまたは両方を選択した場合は、ステージ 2 の完了後にステージ 3 が自動的に開始されます。

次に、ステージ 2 設定ファイル stage2.vgmig の例を示します。

```
databaseDriverLocation=d:¥SQLLIB¥java¥java¥db2java.zip
databaseDriver=COM.ibm.db2.jdbc.app.DB2Driver
databaseName=jdbc:DB2:VGMIG
databaseSchema=MIGSCHEMA
databaseUserId=myuserID
databasePassword=encoded:AAEDAwQFBwYKCwo+Pw==
configurationPlan=MyMigrationSetA,1.1
migrateRemainingParts=yes
```

```
workspaceImport=latest
overrideExistingFiles=yes
tempDirectory=
logFileLocation=D:\tempmig\MyMigrationSetA\stage2\MyMigrationSetA.log
migrateNow=yes
projectType=COBOL
```

---

## ステージ 2 の実行

ステージ 2 マイグレーション・ツールは、バッチ・モードで実行することも、EGL 開発環境のユーザー・インターフェースから実行することもできます。

### ユーザー・インターフェースからのステージ 2 の実行

178 ページの『ステージ 2 VAGen マイグレーション・ファイルの設定』で説明したウィザードには、.vgmig ファイルに設定を保管するオプションがあります。ウィザードの「今すぐマイグレーションを実行」ボックスを選択すると、ウィザードの「完了」ボタンをクリックしたときにステージ 2 マイグレーションが開始されます。ウィザードの「今すぐマイグレーションを実行」ボックスを選択しなかった場合は、保管した .vgmig ファイルを使用してステージ 2 マイグレーションを実行する準備ができたなら、次の手順で行います。

1. 「ナビゲーター」ビューで、プロジェクト名の隣にある「+」記号を右クリックして、ステージ 2 設定ファイルを含むプロジェクトを展開する。
2. .vgmig 設定ファイルを右クリックして、コンテキスト・メニューを表示する。
3. 「マイグレーションの開始 (Start Migration)」をクリックする。

ステージ 2 マイグレーションが開始され、指定したマイグレーション・セットの外部ソース形式が EGL ソースに変換されて、EGL ソースがマイグレーション・データベースに格納されます。「ワークスペースにインポート」または「マイグレーションしたファイルを一時的ディレクトリに保管」のどちらかを選択した場合は、ステージ 2 の完了後にステージ 3 が自動的に開始されます。ステージ 3 の後、マイグレーション・ツールはワークスペースのリフレッシュを自動的に開始します。特にパーツ数が多い場合は、リフレッシュ・ステップに時間がかかる可能性があります。リフレッシュ・ステップが完了すると、マイグレーションが完了したことを通知するポップアップ・ウィンドウが表示されます。必ず、ポップアップ・ウィンドウが表示されるまで待ってください。

マイグレーションとリフレッシュ・ステップが完了すると、次の出力が使用可能になります。

- ステージ 2 マイグレーション・ログ・ファイル。このログ・ファイルは、「ログ・ファイルのロケーション」として指定したディレクトリにあります。このログ・ファイルには、マイグレーションされたパーツに関する情報と、ステージ 2 マイグレーション中に出された情報メッセージ、警告メッセージ、またはエラー・メッセージが保管されています。
- マイグレーション・セットの「to do」リスト・ログ・ファイル。この「to do」リスト・ファイルは、ステージ 3 の開始時に作成される、ステージ 2 で生成されたメッセージを統合したリストで、マイグレーションを完了するために必要になる可能性がある追加作業を示しています。それぞれの生成可能パーツとその関連パーツに関するメッセージが、グループとしてリストされているという点では、この「to do」リストは VisualAge Generator 生成メッセージと似ています。ある



パーツに対してメッセージが存在し、そのパーツが複数のプログラムに関連している場合、これらのメッセージはそれぞれのプログラムごとに 1 回リストされます。未使用の生成不可パーツに関するメッセージは、プロジェクト、パッケージ、およびファイル名別に、「to do」リストの末尾にリストされます。この点では、この「to do」リストは VisualAge Generator 生成メッセージと異なっています。「to do」リストは、ステージ 2 マイグレーション・ログ・ファイルと同じディレクトリに置かれます。

- 「ワークスペースにインポート」を選択した場合、ステージ 3 が自動的に開始され、マイグレーション・セットに必要な EGL プロジェクト、ソース・フォルダー、パッケージ、および EGL ファイルが作成されます。またステージ 3 ツールは、プロジェクトをワークスペースにインポートし、EGL 検証を実行するためにプロジェクトを再ビルドします。
- 「マイグレーションしたファイルを一時ディレクトリに保管」を選択した場合、ステージ 3 が自動的に開始され、マイグレーション・セットに必要な EGL プロジェクト、ソース・フォルダー、パッケージ、および EGL ファイルが作成されます。ステージ 3 ツールは、これらのプロジェクトを指定の一時ディレクトリに配置します。ステージ 3 マイグレーション・ツールは、一時ディレクトリ内でプロジェクトを作成するときに、VAGen バージョン名をプロジェクト名に追加します。これにより、プロジェクトの複数のバージョンを一度にマイグレーションして、後でワークスペースにインポートできます。

## バッチ・モードでのステージ 2 の実行

ステージ 2 ウィザードを使用して、1 つ以上のマイグレーション・セットを選択し、即時にマイグレーションできます。また、後でバッチ・モードでのマイグレーションを行うために、ファイルに情報を保管することもできます。後でマイグレーションを行うためのファイルを作成するには、次の手順で行います。

1. 178 ページの『ステージ 2 VAGen マイグレーション・ファイルの設定』で説明した手順を行う。ただし、それ以外に次のことを行います。
  - 「現行構成をファイルに保管」を選択し、パスとファイル名を指定する。自動的に作成されるファイルには接尾部 `.vgmig` が付けられ、バッチ・モードでステージ 2 またはステージ 3 を実行するときには、このファイルを `-importFile` として指定する必要があります。
  - 「今すぐマイグレーションを実行」を必ず選択解除する。このオプションを選択解除することにより、後でマイグレーションを行うための情報を保管するように指示します。
  - 複数の `.vgmig` ファイルを定義して、後で単一のバッチとしてマイグレーションできます。
2. `.bat` ファイル拡張子の付いたファイルを作成する。

Windows 環境の場合、`.bat` ファイルの内容は次のようにする必要があります。

```
set path=InstallDirectory\%eclipse%\jre\bin;%path%
set classpath=InstallDirectory\%eclipse%\startup.jar;
InstallDirectory\%egl%\eclipse\plugins\
com.ibm.etools.egl.vagenmigration_version\runtime\eglMigration.jar;
cd InstallDirectory
java com.ibm.etools.egl.internal.vagenmigration.batch.VGMIG
    -importFile Path\vgmigFileName.vgmig
    -data Path\workspace
    >Path\LogName.log
```



注:

- 次のステートメントは、ステートメント全体が 1 行になるように入力する必要があります。
  - Windows 環境の場合は、**set classpath** ステートメント。
  - **java** ステートメント。
- バッチ・モードでマイグレーションするそれぞれの .vgmig ファイルごとに、**java** ステートメントを 1 つずつ繰り返します。ただし、.vgmig ファイルを作成したときに「ワークスペースにインポート」を選択した場合は、.vgmig ファイルから得られる EGL プロジェクト名が同じにならないようにしてください。同じ .bat ファイル内で同じ EGL プロジェクトに対する複数の .vgmig ファイルをマイグレーションしようとする、最後にマイグレーションされる .vgmig ファイルのみが EGL プロジェクトに反映されます。
- *InstallDirectory* は、Rational Developer 製品をインストールしたドライブとディレクトリーです。path ステートメント、classpath ステートメント、および cd (ディレクトリーの変更) ステートメントに、*InstallDirectory* を組み込む必要があります。

注: 現在ご使用の製品をインストールする前に、前のバージョンの開発者製品をインストールし、保持していた場合、対象のインストール・ディレクトリーは、以前のインストール時に使用されたディレクトリーである可能性があります。

- *version* は、プラグインのバージョン番号です (例: 6.0 または 6.0.1)。通常は、com.ibm.etools.egl.vagenmigration\_version プラグインに存在する最も高いバージョン番号を使用する必要があります。
- *Path¥vgmigFileName.vgmig* は、マイグレーション・データベースからマイグレーションするマイグレーション・セットを指定する、.vgmig ファイルのドライブ、ディレクトリー、およびファイル名を示します。このディレクトリーにはワークスペース名が含まれている必要があります。これは、ステップ 1 で保管した .vgmig ファイルです。(例:  
d:¥myworkspace¥mySimpleProject¥myMigrationInformation.vgmig)
- *Path¥workspace* は、EGL ファイルを配置するドライブ、ディレクトリー、およびワークスペース名です (例えば、ワークスペースは d:¥myworkspace)。マイグレーション・セットに使用される EGL プロジェクトとパッケージは、マイグレーション・ツールによって自動的に作成されます。*-data* パラメーターはオプションです。*-data* は、デフォルト値以外の「VAGen マイグレーションの設定」を指定する必要がある場合にのみ必須です。「VAGen マイグレーションの設定」を設定する場合は、.bat ファイルを実行する前に、*-data* パラメーターに指定したワークスペース内でこれらの設定を行う必要があります。設定方法については、171 ページの『VAGen マイグレーションの設定』を参照してください。
- *Path¥LogName.log* は、java コマンドに対して作成するログ・ファイルのドライブ、ディレクトリー、およびファイル名を指定します。このログ・ファイルは、java コマンド自体の問題をリストします。ステージ 2 またはステージ 3 で生成されるログ・メッセージは、ウィザードの先頭ページで指定したログ・ファイルに格納され、その後 vgmig ファイルに保管

されます。同じ .bat ファイルに複数の java コマンドがある場合は、それぞれの java コマンドごとに異なるログ・ファイル名を必ず指定してください。

Windows 環境の場合、java コマンドの例は次のようなものです (ただし、コマンド全体を 1 つの行に入力する必要があります)。

```
java com.ibm.etools.egl.internal.vagenmigration.batch.VGMIG
  -importFile d:\myworkspace\mySimpleProject\myMigrationInformation.vgmig
  -data d:\myworkspace\
  >d:\migrationLogs\myMigrationInformationPiped.log
```

3. EGL 開発環境をシャットダウンする。
4. コマンド・プロンプト・ウィンドウを開き、.bat ファイルがあるディレクトリーにナビゲートして、.bat ファイルを実行する。

注: 「*PolicyClassLoader* は、ポリシー *com.ibm.jxesupport.JxeClassLoaderPolicy* を見付けることができませんでした (*PolicyClassLoader could not find policy com.ibm.jxesupport.JxeClassLoaderPolicy*)」 というメッセージは、無視して構いません。

5. 処理が完了すると、EGL プロジェクト、ソース・フォルダー、パッケージ、およびファイルが、それぞれ指定したディレクトリーに保管されます。それぞれの java コマンドに対応するログ・ファイルには、マイグレーション済みパーツのリストと、エラー・メッセージが保管されています。メッセージは、ユーザー・インターフェースを使用してステージ 2 を実行した場合にログ・ファイルに書き込まれるメッセージと同じです。同様に、「to do」リスト・ファイルに保管されるメッセージは、ユーザー・インターフェースを使用してステージ 2 を実行した場合にこのファイルに書き込まれるメッセージと同じです。
6. EGL 開発環境を開始する。
7. EGL プロジェクト、ソース・フォルダー、パッケージ、およびファイルが、ワークスペースに表示されます。

## 第 7 章 ステージ 3 — インポート

マイグレーションのステージ 3 も、EGL に付属のプラグインを使用して実行します。このステージでは、別のマイグレーション・ツールを実行して、ステージ 2 でマイグレーション・データベースに格納された EGL 構文から EGL ファイルを作成します。

### ステージ 3 ツールの実行

EGL 開発環境のワークベンチ・ウィンドウで以下を実行します。

1. 「ファイル」メニューの「インポート」を選択する。
2. 「データベースからの VAGen マイグレーション」を選択し、「次へ」をクリックする。
3. このマイグレーション・ステージの設定を次のように指定する。
  - a. ウィザードの先頭ページで、次の表に説明するように設定を編集する。Tab キーを押してこのフィールドから出るまで、マイグレーション・ツールによる「データベース情報」フィールドの検証は行われません。これにより、情報の入力中にデータベースへの接続が繰り返し試行されることを防止できます。

表 61. ウィザードの先頭ページに入力する設定

設定	意味	値
既存ファイルをロード	この設定により、前に保管したステージ 3 設定ファイルを選択できます。「ファイルを選択」ボタンをクリックして、既存の .vgmig ファイルを選択します。「ロード・ファイル」ボタンをクリックすると、そのファイルから設定が取り込まれ、ウィザードに表示されます。	(オプション) 既存の .vgmig ファイルを選択してロードします。
データベース・ドライバのロケーション	これは、DB2 ドライバのロケーションです。	<code>path_to_db2java.zip¥db2java.zip</code>
データベース・ドライバ	これは、DB2 ドライバの名前です。	この値は、常に <code>COM.ibm.db2.jdbc.app.DB2Driver</code> にする必要があります。この値は、ローカル・データベースと、ローカル側でカタログされたりモート・データベースの両方に有効です。
データベース名	これは、マイグレーションのステージ 1 で使用した DB2 データベースの名前です。	この値は、次のフォーマットで指定する必要があります。 • <code>jdbc:DB2:databaseName</code> <code>databaseName</code> は、ステージ 1 で使用した DB2 データベースの名前です。ステージ 1 のデフォルト値は <code>VGMIG</code> です。

表 61. ウィザードの先頭ページに入力する設定 (続き)

設定	意味	値
データベース・スキーマ (Database schema)	これは、マイグレーションのステージ 1 で使用した DB2 データベース・スキーマの名前です。	この値は、ステージ 1 で使用した DB2 スキーマの名前です。ステージ 1 のデフォルト値は MIGSCHEMA です。
データベース・ユーザー ID	これは、マイグレーションのステージ 1 で使用したデータベース・ユーザー ID です。	ステージ 1 に使用したものと同じデータベース・ユーザー ID を使用します (デフォルト値は、ご使用のログオン ID)。
データベース・パスワード	これは、マイグレーションのステージ 1 で使用したデータベース・パスワードです。	ステージ 1 に使用したものと同じデータベース・パスワードを使用します (デフォルト値は、ご使用のログオン・パスワード)。
ログ・ファイルのロケーション	これは、ログ・ファイルが書き込まれるロケーションです。	ファイル・システム内の有効なロケーション (ドライブとディレクトリー) を入力します。
ログ・ファイル名	これは、ステージ 3 メッセージを書き込むログ・ファイルの名前です。	有効なファイル名を入力します。

b. ウィザードの 2 ページ目で、次の表の説明のとおり、設定を編集する。

表 62. ウィザードの 2 ページ目に入力する設定

設定	意味	値
「Java」または「COBOL」ラジオ・ボタン	この選択により、マイグレーション・ツールが Java ソース・フォルダーを含むプロジェクトを作成するかどうかが決まります。	Rational Web Developer または Rational Application Developer を使用している場合は、必ず「Java」ラジオ・ボタンを選択する必要があります。  その他の製品においては、COBOL のみを生成する予定ならば COBOL を選択できます。
最新バージョン	この設定は、対象のマイグレーション・セットの最新バージョンをインポートするように指定します。	ラジオ・ボタンの 1 つを選択します。
最も古いバージョン	この設定は、対象のマイグレーション・セットの最も古いバージョンをインポートするように指定します。	ラジオ・ボタンの 1 つを選択します。
既存ファイルを上書き	ステージ 3 (インポート・プロセス) では、ステージ 2 で作成した EGL を使用して、ステージ 1 レポートの中で指定された EGL ファイルを作成し、インポートします。ステージ 3 でインポートしようとしている EGL ファイルと同名の EGL ファイルがワークスペースにすでに存在する場合は、このオプションによってこれらのファイルを上書きするかどうかが決まります。	「既存ファイルを上書き」を選択すると、現在マイグレーション中のマイグレーション・セットに、ワークスペース内の既存ファイルに配置する必要があるパーツが含まれている場合に、ステージ 3 マイグレーション・ツールがその状況をどのように処理するかを指定できます。「既存ファイルを上書き」を選択すると、ステージ 3 マイグレーション・ツールは既存のファイルを置き換え、 <b>現行</b> マイグレーション・セット内にあるパーツのみを組み込みます。 「既存ファイルを上書き」を選択しない場合、ステージ 3 マイグレーション・ツールは新規パーツすべてを既存ファイルにマージします。新規パーツは、パーツ型別にアルファベット順に配置されます。このオプションの影響について詳しくは、49 ページの『ファイルの上書きとマージ』を参照してください。

表 62. ウィザードの 2 ページ目に入力する設定 (続き)

設定	意味	値
マイグレーションしたファイルを一時ディレクトリーに保管	このオプションを選択すると、EGL ファイルをファイル・システム内のロケーションに保管できます。これにより、プロジェクトの複数のバージョンの EGL ファイルに同時にアクセスできるようになります (ワークスペースで一度に表示できるバージョンは 1 つのみです)。このロケーションから、リポジトリに EGL ファイルを直接移動できます。	<p>マイグレーション・セットの複数バージョンをマイグレーションする予定の場合は、次のように設定します。</p> <ul style="list-style-type: none"> <li>それぞれのバージョンを別々のサブディレクトリーに書き込むことができるように、このボックスを選択する。</li> <li>これらのバージョンを配置するサブディレクトリーを格納する「フォルダー (Folder)」を指定する。</li> <li>「今すぐマイグレーションを実行」は選択しない。リソース所要量が大きいのので、一時ディレクトリーへのマイグレーションはバッチ・モードでのみ行う必要があります。「今すぐマイグレーションを実行」を選択した場合は、本当にオンライン・モードで実行するかどうか、マイグレーション・ツールから確認を求められます。</li> <li>「現行構成をファイルに保管」を選択する。また、現行構成を .vgmig ファイルとして保管する先のプロジェクトとファイル名も指定する必要があります。オンラインまたはバッチのどちらのモードでマイグレーションを行うかに関係なく、「マイグレーションしたファイルを一時ディレクトリーに保管」を選択した場合は .vgmig ファイルが必要です。バッチ・モードでステージ 3 を実行する場合は、マイグレーション設定を指定するために、保管した .vgmig ファイルを指示します。</li> </ul>
フォルダー (Folder)	これは、EGL ファイルを保管するディレクトリーです。「フォルダー (Folder)」に指定したディレクトリーの下に、それぞれのマイグレーション・セット・バージョンのサブディレクトリーが作成されます。	ファイル・システム内の既存ディレクトリーを指定します。
今すぐマイグレーションを実行	後でマイグレーションを行うために設定ファイルをセットアップするだけでなく、この時点でステージ 3 を実行するように指定します。	「今すぐマイグレーションを実行」を選択してステージ 3 をオンライン・モードで実行するか、または後でステージ 3 をバッチ・モードで実行できるように、「現行構成をファイルに保管」を選択して設定を保管する必要があります。後で参照できるように設定のコピーを保持したい場合は、両方のオプションを選択できます。「マイグレーションしたファイルを一時ディレクトリーに保管」をすでに選択した場合は、「今すぐマイグレーションを実行」を選択しないでください。一時ディレクトリーへの保管は、バッチ・モードでのみ実行できます。「今すぐマイグレーションを実行」の選択は、オンライン・モードでマイグレーションを行うように指示します。

表 62. ウィザードの 2 ページ目に入力する設定 (続き)

設定	意味	値
現行構成をファイルに保管	<p>この設定により、指定中の設定をファイルに保管できます。後で、次のどちらかの方法でステージ 3 を実行できます。</p> <ul style="list-style-type: none"> <li>オンライン・モードの場合は、保管した .vgmig ファイルを右クリックし、コンテキスト・メニューから「マイグレーションの開始 (Start Migration)」を選択する。</li> <li>バッチ・モードの場合は、-importFile オプションを使用して、保管した .vgmig ファイルを指定する。詳しくは、184 ページの『バッチ・モードでのステージ 2 の実行』を参照してください。</li> </ul>	<p>「今すぐマイグレーションを実行」を選択してステージ 3 をオンライン・モードで実行するか、または後でステージ 3 を実行できるように、「現行構成をファイルに保管」を選択して設定を保管する必要があります。後で参照できるように設定のコピーを保持したい場合は、両方のオプションを選択できます。</p> <p>このオプションを選択した場合は、現行構成を .vgmig ファイルとして保管する先のパスとファイル名も指定する必要があります。後でステージ 3 を実行する場合は、マイグレーション設定を指定するために、保管した .vgmig ファイルを指示します。</p>
パス	ファイルを保管する先のプロジェクトを指定します。	¥ <i>projectName</i> 。ここで、 <i>projectName</i> は保管するファイルに格納するプロジェクトの名前です。
ファイル名	設定を保管する先のファイルの名前を指定します。	<i>fileName</i> 。ここで、 <i>fileName</i> は目的のファイルの名前で、ファイル拡張子を付けずに指定します。拡張子 .vgmig が自動的に付加されます。

- c. ウィザードの 3 ページ目で、インポートするマイグレーション・セットを選択する。

注: 「データベースからの VAGen マイグレーション・インポート」ウィザードには、EGL にマイグレーション済みのマイグレーション・セットのみがリストされます。このため、ステージ 3 を実行する前に、必ずステージ 2 を実行してマイグレーション・セットを EGL ソースに変換し、EGL をマイグレーション・データベースに格納してください。

- 「完了」をクリックする。
- マイグレーション・ツールは、選択したマイグレーション・セットに基づいて、EGL プロジェクト、EGL ソース・フォルダー、および EGL パッケージを作成します。このツールは、EGL ソースをマイグレーション・データベースから抽出し、マイグレーション・セットに基づいて EGL ファイルを作成します。マイグレーション・ツールはまた、import ステートメントを組み込み、パーツ参照を解決できるようにプロジェクトの EGL ビルド・パスを更新します。

## バッチ・モードでのステージ 3 の実行

ステージ 2 とステージ 3 をバッチ・モードで実行する場合の違いは、.vgmig ファイルの作成に使用するウィザードのみです。ステージ 3 のみを実行するために .vgmig をセットアップする方法について詳しくは、187 ページの『ステージ 3 ツールの実行』を参照してください。.bat ファイルに組み込むコマンドの詳細、およびバッチ・モードのために指定できるオプションについては、184 ページの『バッチ・モードでのステージ 2 の実行』を参照してください。



## 一時ディレクトリーに書き込まれたマイグレーション・セットの使用

ステージ 3 の出力を一時ディレクトリーに送った場合は、マイグレーション・ツールによって、それぞれのマイグレーション・セット・バージョンごとにサブディレクトリーが 1 つ作成されます。サブディレクトリー名の形式は *migrationSetName\_versionName* です。次のいずれかの手法を使用して、プロジェクトをワークスペースに取り込むことができます。

- サブディレクトリーに保管されているプロジェクトが少数の場合は、既存のワークスペースからそれぞれのプロジェクトを指定できます。この手法では、プロジェクトをワークスペースにコピーせずに、単にワークスペース上でプロジェクトを使用可能にします。プロジェクト内のファイルを変更または削除すると、ワークスペースが指示しているファイル・システムのディレクトリー内で変更が行われます。
  1. 既存のワークスペースから、「ウィンドウ」 -> 「設定」 -> 「ワークベンチ」を選択し、「自動的にビルド」を選択解除する。これにより、それぞれのプロジェクトを取り込むときに再ビルドが複数回行われなくなります。
  2. 「ナビゲーター」ビューまたは「プロジェクト・エクスプローラー」ビューのコンテキスト・メニューから、「インポート」 -> 「既存プロジェクトをワークスペースへ」を選択する。
  3. 「プロジェクト・コンテンツ」の「参照」ボタンを選択し、最初のプロジェクトをポイントする。「完了」を選択します。
  4. マイグレーション・セットのサブディレクトリー内で、それぞれのプロジェクトごとにステップ 2 と 3 を繰り返す。
  5. 「プロジェクト」 -> 「すべてをビルド」を選択して、新規プロジェクトを使用してワークスペースを再ビルドする。
- サブディレクトリーに多数のプロジェクトがある場合は、次のように、サブディレクトリーに対してワークスペースを起動した方が使いやすことがあります。
  1. EGL 開発環境を開始する。
  2. ワークスペース名を指定するためのプロンプトが出されたら、マイグレーション・セット・バージョンを含むサブディレクトリーを指定して、「OK」を選択する。
  3. ワークベンチ設定を変更して、次のことを行う。
    - 「EGL 開発」と「VisualAge Generation から EGL へのマイグレーション」のワークベンチ機能をオンにする。
    - EGL 設定が VisualAge Generator 互換モードに設定されていることを確認する。
    - 新規ワークスペースに対して、その他の通常の設定を行う。
  4. プロジェクトがビルドできないことを示すエラーが「問題」ビューに表示された場合は、「ナビゲーター」ビューを使用して、閉じたプロジェクトを見付ける。閉じたプロジェクトは、プロジェクト名の左側にプラス (+) 記号が付いていません。コンテキスト・メニューから「プロジェクトを開く」を選択して、閉じたプロジェクトを選択します。これで、「プロジェクト・ナビゲーター (Project Navigator)」ビューにプロジェクトが表示されます。



## 第 8 章 単一ファイル・モードでのマイグレーションの実行

ステージ 1 から 3 を使用してマイグレーションを実行する代わりに、単一ファイル・モードでマイグレーションを実行できます。このプロセスにより、1 つの外部ソース形式ファイルを直接 EGL ファイルにマイグレーションできます。このモードでマイグレーションを実行するには、まず VisualAge Generator パーツを外部ソース形式ファイルにエクスポートしてから、外部ソース形式ファイルを EGL にインポートする必要があります。インポート・プロセス中に、外部ソース形式ファイルは、設定に応じて 1 つ以上の EGL ファイルにマイグレーションされます。

パーツを VisualAge Generator からエクスポートするには、次の手順で行います。

1. VisualAge Generator on Java (または VisualAge Generator on Smalltalk) を開始し、VAGen パーツ・ブラウザを開く。
2. エクスポートするパーツを選択し、選択項目を右クリックする。
3. コンテキスト・メニューから、「インポート/エクスポート」→「VAGen エクスポート (VAGen Export)」(または「関連パーツを含む VAGen エクスポート (VAGen Export with Associates)」) を選択する。
4. 外部ソース形式ファイルの名前をボックスに入力し、「保管 (Save)」ボタンをクリックする。(既存ファイルの名前を入力した場合は、そのファイルにパーツを追加するか、ファイルを上書きするかを尋ねられます。どちらか適当な方法を選択してください。)

単一ファイル・モード用に準備するには、次の手順で行います。

1. EGL 開発環境を開始し、ワークスペースを指定する。  
(例: d:\workspaces\myworkspace)
2. マイグレーションの設定を行う。これを行うための方法については、171 ページの『VAGen マイグレーションの設定』を参照してください。
3. 「ワークベンチ」ウィンドウから、「ウィンドウ」->「設定」->「VAGen マイグレーション」を選択する。通常は、「パーツを EGL ファイルに分離」の設定が選択されていることを必ず確認してください。この設定を選択すると、それぞれのプログラム、マップ・グループ、テーブル、および UI レコードが独自のファイルに配置されます。これは、ファイルごとに 1 つの生成可能パーツを配置するという EGL の要件に準拠しています。「パーツを EGL ファイルに分離」を選択しない場合は、UI レコード以外のすべてのパーツが同じ EGL ファイルに配置されます。単一ファイル・モードのパーツ配置アルゴリズムについて詳しくは、26 ページの『単一ファイル・マイグレーションの概要』を参照してください。
4. 新規 EGL プロジェクトを作成する。(例: MyProject)。また、VAGen Web トランザクションをマイグレーションするか、または新規の EGL PageHandlers を開発する予定の場合は、EGL Web プロジェクトを作成してください。
5. EGL プロジェクトの EGLSource ディレクトリーの下に、新規 EGL パッケージを作成する。(例: my.pkg)
6. オンライン・モードでの実行について詳しくは、194 ページの『ユーザー・インターフェースを使用した単一ファイル・マイグレーションの実行』を参照してください。

ださい。また、バッチ・コマンド・ファイルの作成により、単一のコマンド・ファイルを使用して複数の外部ソース形式ファイル进行处理する方法について詳しくは、195 ページの『バッチ・モードを使用した単一ファイル・マイグレーションの実行』を参照してください。

## ユーザー・インターフェースを使用した単一ファイル・マイグレーションの実行

外部ソース形式ファイルを EGL にインポートするには、次の手順で行ないます。

1. 「プロジェクト・エクスプローラー」ビューまたは「ナビゲーター」ビューから、作成される EGL ファイルを取り込む EGL パッケージを選択する。
2. パッケージを右クリックする。コンテキスト・メニューから「インポート」を選択する。
3. 「VAGen 外部ソース形式ファイル」を選択し、「次へ」を選択する。
4. 「入力ファイル名」フィールドに、インポートする外部ソース形式ファイルの名前を入力する。
5. 「ソース・フォルダー」フィールドに、EGL ファイルを配置するプロジェクトとソース・フォルダーの名前を入力する。(例: MyProject\EGLSource)
6. 「パッケージ名」フィールドに、EGL ファイルを配置するパッケージの名前を入力する。マイグレーション・ツールは、EGL ファイル内の package ステートメントに指定したパッケージ名も使用します。(例: my.pkg)
7. 「EGL ファイル名」フィールドに、外部ソース形式ファイルから作成される EGL ファイルの名前を入力する。デフォルトでは、EGL ファイル名は外部ソース形式ファイルと同じですが、.egl ファイル拡張子が付けられます。マイグレーション・ツールが、「パーツを EGL ファイルに分離」設定、および外部ソース形式ファイル内のパーツ型を使用して、単一ファイル・モードでのマイグレーション中に作成するファイルを決定する方法については、26 ページの『単一ファイル・マイグレーションの概要』を参照してください。
8. 「ログ・ファイルのロケーション」フィールドに、マイグレーション・ログ・ファイルを配置するドライブとディレクトリーを入力する。「ログ・ファイル名」フィールドに、マイグレーション・ログ・ファイルの名前を入力する。ログ・ファイル名のデフォルトは、指定した外部ソース形式ファイルの名前と一致します。マイグレーション・ログ・ファイルには、マイグレーション中に書き込まれたメッセージがすべて保管されます。
9. 「完了」をクリックします。「EGL ファイル名」フィールドに指定したファイル名が、「コンテナ (Container)」フィールドに指定したコンテナにすでに存在する場合は、そのファイルに追加するか上書きするかを尋ねるプロンプトが出されます。上書きプロンプトへお客様の応答をに基づいて、マイグレーション・ツールは次の処理を行います。
  - 既存の targetFile への上書きを希望しないことを指定した場合、2 番目のインポートのデータ項目、関数、PSB、および非 VGUI レコードは、targetFile に追加されます。2 番目のインポートにある共通パーツは、targetFile 内で重複パーツになります。
  - 既存の targetFile への上書きを希望することを指定した場合、2 番目のインポートのデータ項目、関数、PSB、および非 VGUI レコードは、既存の

targetFile を完全に置き換えます。このため、最初のインポートに含まれていて、2 番目のインポートに含まれていないパーツはすべて失われます。

- マイグレーション設定として「パーツを EGL ファイルに分離」を選択した場合、マイグレーション・ツールはプログラム、formGroup、および dataTable 用に作成されたファイルを上書きします。マイグレーション設定を選択しなかった場合、これらのパーツは targetFile に配置され、上書きプロンプトに対する応答に従って追加または上書きされます。
  - マイグレーション・ツールは、VGUI レコードと .eglbld ファイル用のファイルを常に上書きします。
10. マイグレーションが完了したときには、次の情報に注意する必要があります。
- 指定したプロジェクト、ソース・フォルダー、およびパッケージに、1 つ以上の EGL ファイルがリストされます。マイグレーション・ツールが、「パーツを EGL ファイルに分離」設定、および外部ソース形式ファイル内のパーツ型を使用して、単一ファイル・モードでのマイグレーション中に作成するファイルを決定する方法については、26 ページの『単一ファイル・マイグレーションの概要』を参照してください。
  - エラー・メッセージがあれば、ポップアップ・ウィンドウに表示されます。ログ・ファイルのロケーションを指定しなかった場合は、「ファイルに保管 (Save to File)」ボタンを使用して、メッセージをファイルに保管できます。必ず、ポップアップ・ウィンドウを閉じてください。
11. プロジェクトを選択し、「プロジェクト」→「プロジェクトのビルド」を選択する。これにより、検証が実行されるので、プロジェクト内のファイルすべてに関する最新メッセージが「問題」ビューに反映されます。

---

## バッチ・モードを使用した単一ファイル・マイグレーションの実行

ユーザー・インターフェースを使用する場合は、外部ソース形式ファイルを一度に 1 つずつマイグレーションできます。バッチ・モードでは、単一のコマンド・ファイルによって複数の外部ソース形式ファイルをマイグレーションできます。バッチ・モードを使用するには、次の手順で行います。

1. .bat ファイル拡張子の付いたファイルを作成する。Windows 環境の場合、.bat ファイルの内容は次のようにする必要があります。

```
set path=InstallDirectory\eclipse\jre\bin;%path%
set classpath=InstallDirectory\eclipse\startup.jar;
               InstallDirectory\egl\eclipse\plugins\
               com.ibm.etools.egl.vagenmigration_version\runtime\eglMigration.jar;
cd InstallDirectory
java com.ibm.etools.egl.internal.vagenmigration.batch.VGMIG
    -importFile Path\ExternalSourceFormatFile.esf
    -eglFile Path\EGLFile.egl
    -data Path\workspace
    -package packageName
    -overwrite
    >Path\LogName.log
```

注:

- 次のステートメントは、ステートメント全体が 1 行になるように入力する必要があります。
  - Windows 環境の場合は、**set classpath** ステートメント。

- **java** ステートメント。
- マイグレーションするそれぞれの外部ソース形式ファイルごとに、**java** ステートメントを 1 つずつ繰り返します。
- *InstallDirectory* は、Rational Developer 製品をインストールしたドライブとディレクトリーです。path ステートメント、classpath ステートメント、および cd (ディレクトリーの変更) ステートメントに、*InstallDirectory* を組み込む必要があります。

注: 現在ご使用の製品をインストールする前に、前のバージョンの開発者製品をインストールし、保持していた場合、対象のインストール・ディレクトリーは、以前のインストール時に使用されたディレクトリーである可能性があります。

- *version* は、プラグイン・バージョン番号です (例: 6.0 または 6.0.1)。通常は、com.ibm.etools.egl.vagenmigration\_version プラグインに存在する最も高いバージョン番号を使用する必要があります。
- *Path¥ExternalSourceFormatFile.esf* は、マイグレーションする外部ソース形式ファイルのドライブ、ディレクトリー、およびファイル名を指します。(例: d:¥temp¥VAGenFiles¥PROG1.esf)
- *Path¥EGLFile.egl* は、作成する EGL ファイルのドライブ、ディレクトリー、およびファイル名を指します。ディレクトリーには EGL ソース・ファイルを入れるワークスペース、EGL ソース・フォルダー、およびパッケージが含まれている必要があります。(例: d:¥myworkspace¥MyProject¥EGLSource¥my¥pkg¥prog1.egl) *EGLFile.egl* は、「VAGen 外部ソース形式ファイルのインポート (Import VAGen External Source Format File)」ウィザードを使用するときに指定する「EGL ファイル名」フィールドと同じように使用されます。マイグレーション・ツールが、「パーツを EGL ファイルに分離」設定、および外部ソース形式ファイル内のパーツ型を使用して、単一ファイル・モードでのマイグレーション中に作成するファイルを決定する方法については、26 ページの『単一ファイル・マイグレーションの概要』を参照してください。
- *Path¥workspace* は、ワークスペースのドライブとディレクトリーです。(例:d:¥workspaces¥myworkspace) *-data* オプションを指定しない場合、「VAGen マイグレーション設定」に指定した内容は無視され、マイグレーション・ツールはデフォルトの VAGen マイグレーション設定を使用します。VAGen マイグレーション設定を指定する場合は、*-data* オプションを指定し、指定対象のワークスペースをポイントする必要があります。
- *packageName* は、EGL ファイルを関連付けるパッケージの名前です。(例: my.pkg) パッケージ名は、マイグレーション・ツールが作成する .egl ファイルの package ステートメントにも使用されます。
- *-overwrite* パラメーターはオプションです。このパラメーターは、指定されたディレクトリーにある、指定された名前の既存 EGL ファイルを上書きするかどうかマイグレーション・ツールに指示します。
- *Path¥LogName* は、対応する外部ソース形式ファイルのマイグレーションに関して作成するログ・ファイルのロケーションとファイル名を指定します。マイグレーション・メッセージをログ・ファイルに送る設定もオプシ



ョンですが、強くお勧めします。同じ .bat ファイルに複数の java コマンドがある場合は、それぞれの java コマンドごとに異なるログ・ファイル名を必ず指定してください。

Windows 環境の場合、java コマンドの例は次のようなものです (ただし、コマンド全体を 1 つの行に入力する必要があります)。

```
java com.ibm.etools.egl.internal.vagenmigration.batch.VGMIG
  -importFile d:%temp%VAGenFiles%prog1.esf
  -eglFile d:%workspaces%myworkspace%MyEGLProject%EGLSource%my%pkg%prog1.egl
  -data d:%workspaces%myworkspace
  -package my.pkg -overwrite >d:%temp%EGLLogs%prog1.log
```

2. EGL 開発環境をシャットダウンする。
3. コマンド・プロンプト・ウィンドウを開き、.bat ファイルがあるディレクトリーにナビゲートして、.bat ファイルを実行する。

注: 「*PolicyClassLoader* は、ポリシー *com.ibm.jxesupport.JxeClassLoaderPolicy* を見付けることができませんでした (*PolicyClassLoader could not find policy com.ibm.jxesupport.JxeClassLoaderPolicy*)」というメッセージは、無視して構いません。

4. 処理が完了すると、EGL ファイルとログ・ファイルが、それぞれ指定したディレクトリーに保管されます。ログ・ファイルには、マイグレーション済みパーツのリストと、エラー・メッセージが保管されています。メッセージは、オンライン・モードで「インポート」ウィザードを使用するときにポップアップ・ウィンドウにリストされるメッセージと同じです。
5. EGL 開発環境を開始する。
6. 外部ソース形式ファイルをインポートした先のプロジェクトを選択し、右クリックして「更新」を選択する。この最新表示によってファイル・システムのプロジェクトが最新表示になり、バッチ・モードでのマイグレーション中に作成、追加、または上書きされた EGL ファイルが EGL に認識されます。続いて検証が実行されるので、プロジェクト内のファイルすべてに関する最新メッセージが「問題」ビューに反映されます。次に、作成したパッケージを展開して EGL ファイルを表示できます。



---

## 第 5 部 マイグレーションの完了



---

## 第 9 章 マイグレーションの完了

ステージ 1 から 3 のマイグレーション、または単一ファイル・マイグレーションを使用してソース・コードをマイグレーションした後、いくつかの追加作業を行う必要があります。次のような作業を行います。

- 設定のエクスポート。
- ソース・コード・リポジトリ内の EGL プロジェクトとパッケージに関するベースラインの保管。
- 単一ファイル・マイグレーションを完了するための予備手順。
- EGL ソース・コードの検討。
- EGL ビルド記述子パーツの検討。
- EGL リンケージ・オプション・パーツの検討。
- EGL リソース関連パーツの検討。
- テンプレートとして使用するバインド制御パーツの確立
- プログラム固有のバインド制御パーツの確立
- linkedit コマンドの検討
- VGWebTransactions の検討
- デバッグの準備。
- COBOL 生成を行う場合の生成とテスト。
- Java 生成を行う場合の生成とテスト。
- 標準の検討。
- VisualAge Generator Compatibility モードの使用を除去するかどうかの考慮

---

### 設定のエクスポート

パイロット・プロジェクト中に EGL に関する処理を行った後で、170 ページの『必要な EGL 設定』、170 ページの『推奨される設定』、および 171 ページの『VAGen マイグレーションの設定』で説明した必須または推奨の設定以外に、追加設定を行った可能性があります。例えば、ご使用のソース・コード・リポジトリと、選択したライブラリ管理プロセスに関連して、別途設定を行う場合があります。その設定をファイルにエクスポートすれば、他の開発者がその設定をインポートして、個別の設定を開始するための基礎として使用できます。このエクスポート手法は、設定をワークスペース間で簡単に移動するための手段にもなります。設定をエクスポートするには、次の手順で行います。

- 「ウィンドウ」 -> 「設定」を選択する。
- 「設定」ウィンドウの左下隅にある「エクスポート (Export)」を選択する。
- 設定を保管するファイルを指定する。

他の開発者は、次の手順で設定をワークスペースにインポートできます。

- 「ウィンドウ」 -> 「設定」を選択する。
- 170 ページの『必要な EGL 設定』で説明したとおりに EGL 機能を選択する。

- ・「ウィンドウ」 -> 「設定」を選択する。
- ・「設定」ウィンドウの左下隅にある「インポート (Import)」を選択する。
- ・設定が保管されているファイルを指定する。

注: この手法は、パースペクティブとビューの設定値には影響を及ぼしません。設定のみが変更されます。

---

## EGL プロジェクトとパッケージに関するベースラインの保管

「問題」ビューのメッセージを解決したり、マイグレーション済みの EGL コードを変更したりする前に、ソース・コード・リポジトリ内で EGL プロジェクトとパッケージのバージョンを作成できます。マイグレーションの直後に EGL のプロジェクトとパッケージを保管し、バージョン管理を行うと、これがベースラインとなるので、マイグレーション・ツールによって作成されたソース・コードが正確に分かります。このベースラインは、手作業で行う必要があるコード変更を追跡する手段にもなります。特に、パイロット・プロジェクト中の変更内容をすべて収集して、必要な変更のタイプを文書化するための手段として有用です。この文書は、他のサブシステムをマイグレーションするための補助になります。

---

## 単一ファイル・マイグレーションを完了するための予備手順

単一ファイル・マイグレーションは、ステージ 1 から 3 のマイグレーションが行うことの一部を行いません。次の手順は手動で行う必要があります。

- ・対応する formGroup 内で書式をネストする。この作業は、2 つの formGroup を同じパッケージにマイグレーションする場合に、これら 2 つの formGroup が同じ書式名を含んでいると必要になります。(例: MAPI)
- ・同じ EGL パッケージ内で重複するパーツを解決する。この作業は、2 つのプログラムを関連パーツとともに同じ EGL パッケージにマイグレーションする場合に、これら 2 つのプログラムが共通のパーツを共用していると必要になることがあります。共通のパーツ定義を中央の共通ファイルに分離することもでき、どちらかのファイルから重複パーツを除去することもできます。すべてのファイルが同じパッケージにある場合は、EGL ビルド・パス・プロパティを変更したり、import パッケージを追加したりする必要はありません。
- ・現行プロジェクトの EGL ビルド・パス・プロパティを更新して、現行プロジェクトがパーツ名を解決するために参照する必要があるプロジェクトをすべてリストする。EGL ビルド・パスを更新するには、「プロジェクト・ナビゲーター (Project Navigator)」ビュー内で現行プロジェクトを選択し、選択項目を右クリックして、「プロパティ」を選択します。「EGL ビルド・パス」ページの「プロジェクト」タブで、現行プロジェクトが参照する必要がある追加プロジェクトを選択します。EGL ビルド・パスには、現行プロジェクト内のファイルがインポートする必要があるパッケージを含むプロジェクトをすべて組み込んでください。例えば、FileA が ProjectB 内にあり、FileA が packageC をインポートする必要がある場合は、packageC が置かれているプロジェクトを ProjectB の EGL ビルド・パスに必ず組み込みます。
- ・import ステートメントを EGL ファイルに追加して、そのファイルが参照する必要がある共通パッケージを指定する。import ステートメントに指定するパッケージは、現行 EGL ファイルが置かれている EGL ビルド・パスに対して指定され



たプロジェクトに存在している必要があります。例えば、FileA が ProjectB にある場合、FileA 内の import ステートメントは、ProjectB の EGL ビルド・パスに指定されたプロジェクトにあるパッケージのみを参照できます。

## ステージ 1 から 3 のマイグレーションと単一ファイル・マイグレーションに共通する手順

### EGL ソース・コードの検討

ステージ 1 から 3 のマイグレーション、または単一ファイル・マイグレーションのどちらを使用した場合にも、次の手順を実行する必要があります。

- マイグレーション・ログ、または「TODO」リスト・ログにあるエラーを検討し、解決します。これらのエラーは、マイグレーション・ツールが解決できなかった未確定状態を反映しています。EGL ソース・コードを変更して、これらのエラーを解決します。例えば、VAGen RETR ステートメントを使用し、検索列を明示的に指定しなかった場合に、テーブルがマイグレーション中に使用できなければ、EGL 構文には EZE\_UNKNOWN\_SEARCH\_COLUMN が含まれています。dataTable 定義に基づいた正しい検索列名を指定して、EGL ソース・コードを更新する必要があります。マイグレーション・ログまたは「TODO」リスト・ログのメッセージを解決するためのヘルプについては、397 ページの『付録 C. マイグレーション・ツールからのメッセージ』を参照してください。マイグレーション・ツールが意図的に無効な EGL 構文を作成するときに使用する具体的なストリングの解決法については、427 ページの『付録 D. 「問題」ビューのメッセージ』を参照してください。
- 予約語であるプログラム名、dataTable 名、または formGroup 名が存在する場合は、パーツ名を変更する必要があります。COBOL を生成する場合は、パーツの *alias* プロパティを設定して、オリジナルのパーツ名を指定できます。これにより、プログラム、dataTable、または formGroup の外部参照を変更する必要がなくなります。
- 「問題」ビューにあるその他のエラーを検討し、解決します。マイグレーション・プロセスの結果として「問題」ビューに表示される共通メッセージの解決法については、433 ページの『付録 E. 「問題」ビューの IWN.xxx メッセージ』を参照してください。
- レコードまたは関数に対して *containerContextDependent* プロパティを設定する必要があるかどうか判別します。詳しくは、40 ページの『containerContextDependent プロパティ』を参照してください。

### EGL ビルド記述子パーツの検討

マイグレーション・ツールは、VAGen 生成オプション・パーツを EGL ビルド記述子パーツに変換します。ただし、一部の VAGen オプションには同等な EGL オプションがありません。また、EGL には新しいビルド記述子オプションがいくつかあり、これらの設定が必要になることがあります。これらの変更が原因で、「問題」ビューにエラーが表示される場合があります。マイグレーション・プロセスの結果として「問題」ビューに表示される共通メッセージの解決法については、433 ページの『付録 E. 「問題」ビューの IWN.xxx メッセージ』を参照してください。問題を解決するには、テキスト・エディターが必要になることがあります。ステージ 1

から 3 のマイグレーション、または単一ファイル・マイグレーションのどちらを使用した場合にも、次の手順を実行する必要があります。

- 汎用ビルド記述子オプションの検討。
- COBOL 生成ビルド記述子オプションの検討。
- Java 生成ビルド記述子オプションの検討。
- デバッグ・ビルド記述子パーツの設定。

## 汎用ビルド記述子オプションの検討

COBOL または Java のどちらの生成を予定している場合にも、次のビルド記述子オプションを検討する必要があります。

- 小数点がコンマ記号である各国語を対象に COBOL を生成する場合は、*decimalSymbol* ビルド記述子オプションをインクルードする必要があります。Java を生成する場合は、ランタイム・パフォーマンスを改善するために、*decimalSymbol* ビルド記述子オプションをインクルードすることができます。
- VAGen EZESYS 特殊機能語を使用してランタイム環境を決定する場合は、EGL ビルド記述子オプション *eliminateSystemDependentCode* を設定する必要があることがあります。このオプションについて詳しくは、オンライン・ヘルプを参照してください。
- マスター・ビルド記述子については、オンライン・ヘルプを参照してください。この技法は、VAGen のデフォルト生成オプション・パーツ設定の代替になります。マイグレーション・ツールは、*NOOVERRIDE* 属性を含む生成オプション・パーツを 2 つのビルド記述子パーツに自動的に分割します。ビルド記述子パーツの 1 つは *xxxxx*、もう 1 つは *xxxxx\_NOOVERRIDE* という名前になります。ここで、*xxxxx* はオリジナルの VAGen 生成オプション・パーツ名です。*xxxxx* という名前のパーツには、*NOOVERRIDE* 属性を指定していない VAGen 生成オプションすべての EGL 置換表現が含まれています。パーツ名 *xxxxx\_NOOVERRIDE* には、*NOOVERRIDE* 属性を指定したすべての VAGen 生成オプションの EGL 置換表現が含まれています。この 2 つのパーツへの分割は、マスター・ビルド記述子を使用する場合に必要です。
- VAGen の */OPTIONS* 生成オプションを使用して生成オプション・パーツをチェーンニングしていた場合は、*nextBuildDescriptor* オプションを使用した EGL ビルド記述子パーツのチェーンニングを検討してください。ビルド記述子オプションのセットを VisualAge Generator 内で使用していたものと同じにするには、このチェーンニングを変更する必要があることがあります。
- COBOL 環境の Web トランザクション・パーツを生成する場合は、オンライン・ヘルプで、VGUI レコードに関連付けされた Java パーツの生成に使用される *secondaryTargetBuildDescriptor* に関する情報を参照してください。マイグレーション・オプション・パーツが、次のオプションのいずれか 1 つでも含む場合、マイグレーション・ツールは生成オプション・パーツを 2 つのビルド記述子パーツに自動的に分割します。対象となるオプションは、*/javadestdir*、*/javadesthost*、*/javadestpassword*、*/javadestuserid*、または */javasystem* です。ビルド記述子パーツの 1 つは、*xxxxx* という名前で、もう 1 つは *xxxxx\_TARGET2* という名前です。ここで、*xxxxx* はオリジナルの VAGen 生成オプション・パーツ名です。*xxxxx* という名前のパーツには、1 次 (COBOL) ランタイム環境用の EGL VGWebTransaction プログラムを生成する時に使用するすべての VAGen 生成オプションに関する、EGL 置換表現が含まれています。

xxxxx\_TARGET2 という名前のパーツには、2 次 (Java) ランタイム環境用の EGL VGUI レコードを生成するときに使用するすべての VAGen 生成オプションに関する、EGL 置換表現が含まれています。マイグレーション・ツールは、2 次ビルド記述子パーツにある次の生成オプションに関して EGL 同等物を配置します。対象の生成オプション

は、/javadestdir、/javadesthost、/javadestpassword、/javadestuserid、および /javasystem です。マイグレーション・ツールは、1 次および 2 次ビルド記述子パーツにある次のオプションに関して、EGL 同等物を配置します。対象のオプションは、/genout、/genresourcebundle、/msgtableprefix、/resourcebundlelocale、および /targnls です。マイグレーション・ツールは、1 次ビルド記述子パーツに secondaryTargetBuildDescriptor オプションをインクルードし、オプションの値を 2 次ビルド記述子パーツの名前にセットします。

- Web トランザクション・パーツを生成し、メッセージ・テーブルを使用する場合、msgTablePrefix ビルド記述子オプションを変更しなければならないことがあります。メッセージ・テーブルは、VGUI レコードを使用するプログラムによって指定されます。メッセージ・テーブルと VGUI レコードが別のパッケージにある場合、2 次ビルド記述子パーツを変更し、パッケージ名 (例えば msgTablePrefix = "packageName.prefixID") をインクルードする必要があります。
- マイグレーション・ツールは、EGL 非 Web プロジェクトを作成するときにデフォルトのビルド記述子を作成しません。これによりユーザーは、マイグレーション済みビルド記述子パーツの 1 つを、デフォルトのビルド記述子に指定することができます。ファイル、パッケージ、EGL ソース・フォルダー、プロジェクト、またはワークベンチのレベルで、デフォルトのビルド記述子を設定できます。生成可能パーツに最も近いデフォルトのビルド記述子を使用されます。例えば、ただ 1 つのファイルに対してデフォルトのビルド記述子を指定し、ワークベンチに対しては別のデフォルトのビルド記述子を指定できます。この場合、そのファイルに含まれているプログラムを生成すると、ファイルのデフォルトのビルド記述子を使用されます。その他のプログラムを生成すると、ワークベンチのデフォルトのビルド記述子を使用されます。
- 特定のファイル、パッケージ、EGL ソース・フォルダー、プロジェクトの設定を行うには、リソース (ファイル、パッケージ、フォルダー、またはプロジェクト) を選択し、右クリックしてコンテキスト・メニューから「プロパティ」を選択します。左側のペインにある「EGL デフォルト・ビルド記述子」を選択します。このリソース内の生成可能パーツすべてのデフォルトとして使用するビルド記述子を選択します。近い EGL デフォルト・ビルド記述子が存在しないとすると、このリソース内で生成を行うときには、「ターゲット・システムのビルド記述子」がデフォルトで使用されます。デバッグ・ツールを使用するときには、「デバッグ・ビルド記述子」がデフォルトで使用されます。
- ビルド記述子パーツのワークベンチ設定を行うには、「ウィンドウ」->「設定」->「EGL」->「デフォルトのビルド記述子」を選択します。この設定は、明示的にオーバーライドしなければ、すべてのプロジェクト、パッケージ、ソース・フォルダー、およびファイルに適用されます。生成に使用する「ターゲット・システムのビルド記述子」と、デバッグ・ツールに対して使用する「デバッグ・ビルド記述子」の両方を設定できます。
- マイグレーション・ツールは、マイグレーション・セットの最初の EGL Web プロジェクトを作成するときに自動的にデフォルトのビルド記述子を作成します。これにより、ワークスペースがステージ 3 の終了で更新されるときに、VGUI

レコードが JSP に生成されるようになります。プロジェクト、EGL ソース・フォルダー、あるいはプロジェクトが含むパッケージまたはファイルのための、デフォルトのビルド記述子を変更することができます。

- 制御パーツ (ビルド記述子、リンケージ・オプション、リソース関連、バインド制御、およびリンク・エディット・パーツ) がすべて同じファイルにない場合は、現行ファイルを変更して、現行ファイルから参照したい他のビルド・パーツを含むファイルの `import` ステートメントを組み込む必要があります。例えば、`buildDescriptorPartA` が別のファイルにある `linkageTableB` を参照している場合は、`buildDescriptorPartA` を含むファイルに、`linkageTableB` を含むファイルの `import` ステートメントを組み込む必要があります。`import` ステートメントを追加するには、EGL ビルド・パーツ・エディターを使用します。

## COBOL 生成ビルド記述子オプションの検討

COBOL の生成を予定している場合は、次のビルド記述子オプションを検討または設定する必要があります。

- VisualAge Generator の場合は、COBOL 生成出力がホストに転送されて、準備ステップが実行されます。EGL は、ビルド・サーバーを使用して準備ステップを処理します。EGL z/OS および iSeries Build Server の場合、生成の出力を転送する `destPort` ビルド記述子オプションのために指定しなければならないポート番号があります。リモート・ビルド・サーバーがビルド要求を `listen` するポート番号を判別するには、Build Server のインストールと構成を行った担当者に問い合わせてください。
- z/OS 環境で COBOL を生成する場合は、次のことを行う必要もあります。
  - テンプレートとして使用するバインド制御パーツの確立。(209 ページの『テンプレートとして使用するバインド制御パーツの設定』を参照してください。)
  - プログラム固有のバインド制御パーツの設定。(211 ページの『プログラム固有のバインド制御パーツの設定』を参照してください。)
  - リンク・エディット・コマンドの検討。(212 ページの『リンク・エディット・コマンドの検討』を参照してください。)
- VSE 環境で COBOL を生成する場合は、次のことを行う必要もあります。
  - `linkedit` コマンドを検討する(212 ページの『リンク・エディット・コマンドの検討』を参照してください)。
  - 「*VisualAge Generator EGL Plug-in for VSE Reference Manual*」を読む。

## Java 生成ビルド記述子オプションの検討

Java の生成を予定している場合は、次のビルド記述子オプションを検討または設定する必要があります。

- `genProject` ビルド記述子オプションを追加して、Java 生成の出力を配置する場所を指定します。EGL の `genProject` ビルド記述子オプションにマイグレーションされる VAGen 生成オプションはありません。`genProject` オプションは次の場合に必要です。
  - HP-UX または SOLARIS 向けに生成する場合
  - VGWebTransactions または VGUI レコード、または `dataTable` のようなそれらの関連パーツを生成する場合。この場合には、必ず `genProject` オプションで EGL Web プロジェクトを指定してください。



- 一部の EGL ビルド記述子オプションの振る舞いは、対応する VAGen 生成オプションとは異なります。オンライン・ヘルプにある、次のビルド記述子オプションに関する情報を参照して、これらのオプションをご使用の環境に応じて設定または変更する必要があるかどうか判断してください。
  - genProperties (VAGen の /genproperties オプションに基づいて、マイグレーション・ツールによって設定される)
  - enableJavaWrapperGen (VAGen の /system=JAVAWRAPPER オプションに基づいて、マイグレーション・ツールによって設定される)
- 一部の新しい EGL ビルド記述子オプションには、対応する VAGen 生成オプションがありません。オンライン・ヘルプにある、次のビルド記述子オプションに関する情報を参照して、これらのオプションをご使用の環境に応じて設定する必要があるかどうか判断してください。
  - dateMask
  - sessionBeanID
  - sqlJDBCClass
  - sqlValidationConnectionURL
  - tempDirectory (VGUI レコードの場合のみ)

## デバッグ・ビルド記述子パーツの設定

デバッグ時に使用するビルド記述子オプションを含むビルド記述子パーツを作成します。デバッグ・ビルド記述子パーツを作成するためのガイドについては、オンライン・ヘルプを参照してください。

## EGL リンケージ・オプション・パーツの検討

マイグレーション・ツールは、VAGen リンケージ・テーブル・パーツを EGL リンケージ・オプション・パーツに変換します。ただし、一部の VAGen オプションには同等な EGL オプションがありません。また、EGL には新しいリンケージ・オプションがいくつかあり、これらの設定が必要になることがあります。これらの変更が原因で、「問題」ビューにエラーが表示される場合があります。マイグレーション・プロセスの結果として「問題」ビューに表示される共通メッセージの解決法については、433 ページの『付録 E. 「問題」ビューの IWN.xxx メッセージ』を参照してください。また、ご使用の環境に応じてサポートされるリンケージ・オプションについて詳しくは、オンライン・ヘルプを参照してください。ステージ 1 から 3 のマイグレーション、または単一ファイル・マイグレーションのどちらを使用した場合にも、次の手順を実行する必要があります。

- マイグレーション・ログと「問題」ビューにあるメッセージを検討し、解決します。問題を解決するには、テキスト・エディターが必要になることがあります。
- callLink の場合は、次のことを検討します。
  - VisualAge Generator の linktype の一部は、EGL でサポートされません。例えば、CSOCALL はサポートされなくなりました。マイグレーション・ツールは、CSOCALL を remoteCall に変換します。ただし、EGL remoteCall に対して指定する必要がある属性は、CSOCALL のものとは異なります。
  - VisualAge Generator の remoteComType 値の一部は、EGL ではサポートされません。例えば、DCE、DCESECURE、および APPCIMS はサポートされなくなりました。マイグレーション・ツールは、これらのサポートされない値を

そのまま変換するので、無効な EGL リンケージ・オプション・パーツが生じます。このため、「問題」ビューにエラーが表示されます。このエラーを覚え書として、リンケージ・オプション・パーツを変更し、EGL に対して使用するオプションを指定する必要があります。

- remoteComType=CICSECI を使用するように変更する場合は、*ctgPort* 属性と *ctgLocation* 属性を追加する必要があります。このためには、リモート CICS トランザクションを呼び出すために、CICS Transaction Gateway Server の構成とセットアップを行う必要があります。
- remoteComType=CICSSSL を使用するように変更する場合は、*ctgKeyStore* 属性と *ctgKeyStorePassword* 属性を追加する必要があります。また、VAGen リンケージ・テーブルに *ctgPort* 属性と *ctgLocation* 属性をまだ組み込んでいない場合は、EGL の remoteComType=CICSSSL に対してこれらの属性を組み込む必要があります。
- CICSJ2C を使用する場合は、*pgmName*、*conversionTable*、*remotePgmType*、*luwControl*、*remoteBind*、*location*、および *parmForm* 属性を追加する必要があります。
- APPCIMS の代わりに remoteComType=IMSTCP を使用するように変更する場合、必要な追加属性の設定支援については、オンライン・ヘルプを参照してください。指定が必要な値が、IMSTCP では違う意味を持つため、既存の属性についてもオンライン・ヘルプを検討してください。
- *conversionTable*=BINARY は、EGL ではサポートされません。マイグレーション・ツールは、この値を完全に現状のまま変換するので、EGL リンケージ・パーツ内にプレースホルダーができますが、この値は変更する必要があります。
- *calllink* エントリーの追加が必要な場合があります。EGL は、次の状態で *calllink* エントリーを必要とします。
  - 生成された Java プログラムが、ネイティブ C++ または VAGen 生成プログラムを呼び出す場合、プログラムが同じワークステーション上で実行中であっても、常にリモート呼び出しになります。 *calllink* エントリーが必要です。
  - 呼び出し先プログラムに関して VAGen 生成オプション */system=JAWRAPPER* を使用した場合、*javaWrapper="YES"* という属性の EGL *calllink* エントリーを作成する必要があります。 エントリーがない場合、EGL は Java ラッパーを生成しません。
  - Java を生成し、呼び出し先プログラムが EGL 予約語と競合する場合、EGL *calllink* エントリーを作成して、*alias* 属性を呼び出し先プログラムの実際の名前にセットする必要があります。
- *fileLink* の場合、*conversionTable*=BINARY は EGL ではサポートされません。マイグレーション・ツールは、この値を完全に現状のまま変換するので、EGL リンケージ・パーツ内にプレースホルダーができますが、この値は変更する必要があります。
- *asynchLink* (VAGen *crtxlink*) の場合、*conversionTable*=BINARY は EGL ではサポートされません。マイグレーション・ツールは、この値を完全に現状のまま変換するので、EGL リンケージ・パーツ内にプレースホルダーができますが、この値は変更する必要があります。



- EGL の Transfer To Program のリンケージ情報は、VAGen の dxfrlink エントリーと同等です。Java の生成を行い、VAGen XFER ステートメントを使用している場合は、EGL の Transfer to Transaction エントリーを追加する必要が生じることがあります。この新しいリンケージ・エントリーについては、オンライン・ヘルプを参照してください。

## EGL リソース関連パーツの検討

マイグレーション・ツールは、VAGen リソース関連パーツを EGL リソース関連パーツに変換します。ただし、一部の VAGen オプションには同等な EGL オプションがありません。また、EGL には新しいリソース関連オプションがいくつかあり、これらの設定が必要になることがあります。これらの変更が原因で、「問題」ビューにエラーが表示される場合があります。マイグレーション・プロセスの結果として「問題」ビューに表示される共通メッセージの解決法については、433 ページの『付録 E. 「問題」ビューの IWN.xxx メッセージ』を参照してください。また、ご使用の環境に応じてサポートされるリソース関連オプションについて詳しくは、オンライン・ヘルプを参照してください。ステージ 1 から 3 のマイグレーション、または単一ファイル・マイグレーションのどちらを使用した場合にも、次の手順を実行する必要があります。

- マイグレーション・ログと「問題」ビューにあるメッセージを検討し、解決します。問題を解決するには、テキスト・エディターが必要になることがあります。
- VisualAge Generator のファイル・タイプの一部は、EGL ではサポートされません。例えば、BTRIEVE と MFCOBOL はサポートされなくなりました。マイグレーション・ツールは、これらのサポートされないオプションを完全に現状のまま変換するので、リソース関連パーツ内にプレースホルダーができます。このため、「問題」ビューにエラーが表示されます。このエラーを覚え書として、リソース関連パーツを変更し、EGL に対して使用するオプションを指定する必要があります。選択した EGL ファイル・タイプ・オプションによっては、リソース関連エントリーに対してその他の属性を設定する必要が生じることがあります。
- ご使用の環境に応じて、これらの値を設定する必要があるかどうか判断するには、*FormFeedOnClose* 属性と *text* 属性に関するオンライン・ヘルプを検討してください。VisualAge Generator の場合、同等なオプション (それぞれ /noff と /text) は、ワークステーション環境用のランタイム・リソース関連ファイル内でのみ指定できます。このため、これらのオプションはマイグレーション・ツールによって設定されません。これは、マイグレーション・ツールがリソース関連パーツのみを処理するからです。

## テンプレートとして使用するバインド制御パーツの設定

VisualAge Generator は、バインド制御テンプレートを使用してデフォルトのバインド制御コマンドを作成します。デフォルトの VAGen テンプレートは DB2 プランをバインドしますが、ユーザーがテンプレートを変更してパッケージをバインドするようにしたり、組織の標準に準拠するための変更を加えたりする場合があります。VAGen テンプレートは、ワークスペースの外部で EFK2MBDx.tpl という名前のファイルに保管されます。x は A から D までの文字です。バインド制御パーツは、特定プログラムに対して特殊バインドを行う必要がある場合のみ必須です。

EGL は、バインド制御テンプレートを使用しません。代わりに、EGL はバインド制御パーツを必要とします。パッケージをバインドする場合は、すべてのバインド

に使用するテンプレートを含む EGL バインド制御パーツを作成し、このパーツをワークスペースに保管することによって、VisualAge Generator テンプレートと同様な効果を得ることができます。

**注:** ここで説明する手法は、プランをバインドする場合には使用できません。プランをバインドする場合は、211 ページの『プログラム固有のバインド制御パーツの設定』を参照してください。

それぞれのプログラムごとにパッケージをバインドするように VAGen バインド制御テンプレートを変更した場合は、そのテンプレートを調整して EGL バインド制御パーツとして使用できます。このバインド制御パーツは、他の制御パーツと同じファイルに格納することをお勧めします。例えば、次のような VAGen バインド制御テンプレートがあるとしています。

```
DSN SYSTEM(%MYDB2SUBSYSTEM%)
BIND PACKAGE(%MYCOLLECTIONNAME%) -
    MEMBER(%EZEMBR%) -
    .
    .
    .
```

前の例の中で、MYDB2SUBSYSTEM と MYCOLLECTIONNAME は VAGen 生成オプションに設定したシンボリック・パラメーターで、EZEMBR は現在生成しているプログラムの名前に自動的に設定されます。

パッケージをバインドする場合、作成する必要がある EGL バインド制御パーツは、VAGen テンプレートとよく似ていますが、さらに 3 つの行が必要で、また EZEMBR シンボリック・パラメーターを変更する必要があります。対応する EGL バインド制御パーツは、次のようになります。

```
TSOLIB ACTIVATE DA('%DSNLOAD%')
ALLOC FI(DBRMLIB) SHR DA('%EZEPID%.SYSTEM%.DBRMLIB' +
'%ELA%.SELADBRM')
DSN SYSTEM(%MYDB2SUBSYSTEM%)
BIND PACKAGE(%MYCOLLECTIONNAME%) -
    MEMBER(%EZEALIAS%) -
    .
    .
    .
```

DSNLOAD、EZEPID、および ELA の意味は、すべて VisualAge Generator 内での意味と同じです。バインド制御パーツ内で生成するプログラムの名前が必要な場合、EGL では EZEMBR が EZEALIAS に置き換わります。EGL では、SYSTEM は EZEENV に置き換わります。ご使用の EGL ビルド・サーバー上では異なるデータ・セット命名規則を使用している場合は、バインド制御パーツの先頭 3 行を変更する必要があります。組織の命名規則に基づいてどのような 3 行を追加する必要があるか判別するには、Enterprise Developer ビルド・サーバーのインストールと構成を行った担当者に問い合わせてください。また、VisualAge Generator 内で該当する値を設定していなかった場合は、EGL ビルド記述子オプションを変更して、projectID ビルド記述子オプションと DSNLOAD および ELA の両シンボリック・パラメーターを設定する必要があります。シンボリック・パラメーターの名前の変更については、393 ページの『シンボリック・パラメーター』を参照してください。また、EGL バインド制御パーツに対するテンプレートの使用と、EGL シンボリック・パラメーターの値の設定について詳しくは、オンライン・ヘルプを参照してください。

テンプレートとして使用する EGL バインド制御パーツを作成するほかに、ビルド記述子パーツを変更して、バインド制御パーツを指定する *bind* ビルド記述子オプションを組み込む必要があります。変更が必要なビルド記述子パーツの数を最小限にするために、*bind* ビルド記述子オプションは、既存の共通ビルド記述子パーツの 1 つに追加することをお勧めします。

注: 任意のターゲット環境について、必ず VAGen バインド制御テンプレートの比較を行ってください。テンプレートが違う場合、相違点をサポートするために追加シンボリック・パラメーターを追加することができます。そのようにしない場合、異なる ターゲット環境用のテンプレートとして必要な、EGL バインド制御パーツをポイントするように、プログラム・ベースで異なるビルド記述子オプションをセットしなければならないことがあります。

## プログラム固有のバインド制御パーツの設定

VisualAge Generator 内でプランをバインドする場合、通常はそれぞれのプログラムに異なるバインド・コマンドを使用する必要があります。この場合は、プログラム固有のバインド・コマンドを使用して、そのプログラムのプランを同じ実行単位内にある他のプログラムすべてにバインドする必要があります。このための標準的な方法としては、*xxxxx.BND* という名前のバインド制御パーツを作成します (ここで、*xxxxx* はプログラムの名前)。その後、VAGen 生成オプション */BIND=BND* を設定して、プログラム固有のバインド・コマンドを検索する際に VisualAge Generator が使用する接尾部を指定します。まれな状況で、プログラムの 1 つが必要としているものがテンプレートの指定内容と異なるときにパッケージをバインドする場合にも、*.BND* 接尾部を使用できます。

EGL *bind* ビルド記述子オプションには、接尾部を指定できません。代わりに、*bind* ビルド記述子オプションには、特定のバインド制御パーツの名前を指定する必要があります。EGL の場合は、*bind* ビルド記述子オプションを指定しなければ、EGL はプログラムと同名のバインド制御パーツを検索します。一般に、最も簡単な手法としては、パッケージをバインドして、209 ページの『テンプレートとして使用するバインド制御パーツの設定』に説明する手順のとおりに行います。ただし、プランをバインドする場合、またはプログラムの 1 つがバインド制御パーツ・テンプレートの指定内容以外のものを必要としている場合には、プログラム固有のバインド制御パーツを作成できます。

デフォルトの *.BND* 接尾部を使用した VAGen プログラム固有のバインド制御パーツが存在する場合、マイグレーション・ツールは *.BND* 接尾部を自動的に除去し、EGL バインド制御パーツに必要なステートメントを追加します。命名規則が *programName.BND* で、常にプログラム固有のバインド・コマンド・パーツを使用していた場合、このプログラムに対して EGL の *bind* ビルド記述子オプションを指定する必要はありません。ただし、EGL の *bind* ビルド記述子を使用して、大部分のプログラムに対してテンプレートとして使用するバインド制御パーツを指定し、1 つのプログラムに対してはプログラム固有のバインド制御パーツを提供する必要がある場合は、この特定プログラム用にビルド記述子パーツを作成し、プログラム固有のバインド制御パーツを指定して *bind* ビルド記述子オプションを設定する必要があります。このようにしなければ、EGL は通常の *bind* ビルド記述子オプションの指定に従って、テンプレートであるバインド制御パーツを選択します。

## リンク・エディット・コマンドの検討

VisualAge Generator は、ターゲット環境とデータベース・アクセスに基づいてデフォルトのリンク・エディット・コマンドを提供します。ただし、ユーザーによっては、特定のリンク・エディット・コマンドを使用している場合があります。(例えば、MVS バッチ環境用に PL/I プログラムをリンクインする目的で。) このための標準的な方法としては、`xxxxx.LKG` という名前のリンク・エディット・パーツを作成します (ここで、`xxxxx` はプログラムの名前)。その後、VAGen 生成オプション `/LINKEDIT=LKG` を設定して、プログラム固有のリンク・エディット・コマンドを検索する際に VisualAge Generator が使用する接尾部を指定します。

EGL の `linkEdit` ビルド記述子オプションには、接尾部を指定できません。代わりに、`linkEdit` ビルド記述子オプションには、特定のリンク・エディット・パーツの名前を指定する必要があります。EGL の場合は、`linkEdit` ビルド記述子オプションを指定しなければ、EGL はプログラムと同名のリンク・エディット・パーツを検索します。EGL がプログラムと同名のリンク・エディット・パーツを検出できない場合は、VisualAge Generator と同様の方法で、ターゲット環境とデータベース・アクセスに基づいて EGL がデフォルトのリンク・エディット・コマンドを作成します。このため、`linkEdit` ビルド記述子オプションを指定する必要がある状況は、プログラムとは異なる名前を指定してリンク・エディット・パーツを作成する場合に限ります。幾つかの COBOL 環境で同じプログラムを生成する場合に、これを行わなければならない可能性があります。

デフォルトの `.LKG` 接尾部を使用する VAGen プログラム固有のリンク・エディット・パーツが存在する場合、マイグレーション・ツールは自動的に `.LKG` 接尾部を除去します。命名規則が `programName.LKG` である場合、このプログラムに対して EGL の `linkedit` ビルド記述子オプションを指定する必要はありません。EGL は、デフォルトのリンク・エディット・コマンドの作成を試みる前に、まずプログラム固有のパーツを検索します。

## ご使用の VGWebTransactions の検討

マイグレーション済み VGWebTransaction プログラムを検討するときには、次のことを考慮してください。

- EGL パッケージ名が VAGen パッケージ名と異なる場合、JSP およびプロパティ・ファイルを更新する必要があります。以下のいずれかの理由により、パッケージ名が変更されている可能性があります。
  - パッケージが、EGL 予約語と競合するため、ステージ 1 のマイグレーション・ツール名前変更規則を使用した。
  - パッケージを EGL に統合するため、ステージ 1 マイグレーション・ツール名前変更規則を使用したか、ステージ 1 ツールを変更した。
  - VAGen ソース・コードのパッケージ名と異なる可能性があるランタイム・パッケージ名を指定するために、VAGen `/packagename` 生成オプションを使用した。EGL では、ランタイム・パッケージ名は常に EGL ソースのパッケージ名と同じです。
- VisualAge Generator から変更済み JSP を使用できます。しかし、次の状態ではなんらかの変更が必要になる場合があります。
  - ランタイム・パッケージ名が変更された場合には、JSP を変更して新規のパッケージ名を指定する必要があります。



- EGL Web トランザクションをデプロイするには、オンライン・ヘルプを参照して支援を受けてください。以下のことを必ず行ってください。
  - プロジェクトの `JavaSource` フォルダのデフォルトの `gw.properties` ファイルの検討および変更を行ってください。 `hptEntryPage` 値および `hptEntryApp` 値を必ず設定してください。ご使用の `VisualAge Generator` システム内の対応する `gw.properties` ファイルからこの情報をコピーできる可能性があります。  
`VisualAge Generator` の `gw.properties` ファイルに加えた変更によっては、追加オプションを設定しなければならない可能性があります。  
`hptDisableRMIDManager` オプションを `VisualAge Generator` フィックスパックに追加されました。このオプションを初めて使用する場合は、EGL オンライン・ヘルプを参照し、値の設定の参考にしてください。
  - プロジェクトの `JavaSource` フォルダのデフォルトの `csogw.properties` ファイルの検討および変更を行ってください。このとき、どのアプリケーションがどのサーバーにあるのかを指定する情報を必ず組み込んでください。ご使用の `VisualAge Generator` システム内の対応する `csogw.properties` ファイルからこの情報をコピーできる可能性があります。  
`VisualAge Generator` の `csogw.properties` ファイルに加えた変更により、追加オプションを設定しなければならない可能性があります。
  - プロジェクトの `WebContent` フォルダのデフォルトの `VAGen1EntryPage.jsp` の検討および変更を行ってください。 `hptAppId` の `OPTION` 情報を更新し、リストの各プログラム用に表示する `VGWebTransaction` プログラムの名前および関連するテキストを必ず含むようにしてください。ご使用の `VisualAge Generator` システム内の対応する `JSP` ファイルからこの情報をコピーできる可能性があります。
  - プロジェクトを生成します。
  - エンタープライズ・アプリケーション・リソース (EAR) プロジェクトを次のように作成します。
    - 「ナビゲーター」ビューまたは「プロジェクト・エクスプローラー」ビューのワークベンチ・ウィンドウから、「新規」->「その他」->「J2EE」->「エンタープライズ・アプリケーション・プロジェクト」を選択する。
    - EAR プロジェクトの「名前」を入力する。
    - 「次へ」を選択する。
    - EAR プロジェクトにインクルードするプロジェクトを選択する。
    - 「完了」を選択する。
  - Web アプリケーション・サーバーを定義する。
  - EAR プロジェクトをサーバーに追加する。
  - プロジェクトの `WebContent` フォルダの `EGLWebStartup.jsp` を選択してアプリケーションを実行し、コンテキスト・メニューから「実行」->「サーバーで実行 (Run on Server)」を選択する。

## デバッグの準備

デバッグの準備のために、次のことを行う必要があります。

- ワークベンチ・ウィンドウから、「ウィンドウ」->「設定」-> **EGL** -> 「デバッグ (Debug)」を選択します。ご使用の環境に応じて行う必要がある設定 (あれば) を判別するには、オンライン・ヘルプを参照してください。

- さらに、「EGL」->「デフォルトのビルド記述子」の設定を検討します。ワークスペース全体に対して、デフォルトのデバッグ・ビルド記述子を設定できます。また、プロジェクト、EGL ソース・フォルダー、パッケージ、またはファイルに対して、デフォルトのデバッグ・ビルド記述子を設定することもできます。
- 生成した EGL プログラムまたは非 EGL プログラムを、リモート CICS システム上でデバッガーから呼び出す場合は、CICS Transaction Gateway Server を構成して使用する必要があります。CICS Client または CICS Transaction Gateway を使用した、CICS の直接呼び出しはサポートされなくなりました。

## COBOL 生成を行う場合の生成とテスト

COBOL 生成の準備のために、次のことを行う必要があります。

- この時点で、ソース・コード・リポジトリにある EGL プロジェクトとパッケージの別のバージョンを作成できます。これは、生成の前に手動で行った変更を反映した、コードのもう 1 つのベースラインになります。
- VSE 用に生成している場合、「*VisualAge Generator EGL Plug-in for VSE Reference Manual*」に記載された指示に従ってください。
- z/OS 用に生成を行う場合は、最新 PTF をすべて適用した Enterprise Developer Server for z/OS バージョン 5.0 がインストールされていることを確認してください。
- iSeries 用の生成を行う場合は、最新 PTF をすべて適用した iSeries 用の EGL ランタイムが、ご使用のホスト環境にインストールされていることを確認してください。
- ご使用のホスト環境に EGL ビルド・サーバーがインストールされていて、構成済みであることを確認してください。VisualAge Generator の場合、z/OS と iSeries の準備プロセスに対するカスタマイズは、ワークステーション上で準備テンプレートを変更することによって行われていました。EGL の場合、このカスタマイズはホスト・マシン上で行われます。カスタマイズが引き続き必要かどうか、およびそれぞれのターゲット・ホスト環境に対してカスタマイズを行う方法について詳しくは、オンライン・ヘルプを参照してください。
- EGL ビルド・サーバーのインストールと構成を行った担当者にお問い合わせください。生成と準備の出力を格納するホスト・データ・セットの命名規則に変更があるかどうか確認してください。例えば、VisualAge Generator では、MVS バッチ用に生成する場合、データ・セットのデフォルト名は `xxxx.MVSBATCH.yyyy` です。ここで、`xxxx` は `/projectid` 生成オプションに指定した高位修飾子であり、`yyyy` はコードのタイプです。(例: COBOL ソースの場合は `EZESRC`。) EGL の場合は、ターゲット環境名が変更されているため、対応するデフォルト・データ・セット名は `xxxx.ZOSBATCH.yyyy` です。このため、ホスト上でデータ・セットのグループを新規に定義する必要が生じることがあります。
- プログラムと `dataTable` を生成します。プログラムを生成する際には、次のビルド記述子オプションを使用します。
  - `genFormGroup="YES"`
  - `genHelpFormGroup="YES"`
  - `genDataTables="NO"`



これにより、formGroup を使用するプログラムと一緒に formGroup を生成でき、また dataTable を使用するプログラムの数に関係なく、dataTable の生成は一度だけにすることができます。生成中に検出された検証エラーは、すべて解決してください。

- 生成されたコードをテストします。
- この時点で、ソース・コード・リポジトリにある EGL プロジェクトとパッケージの別のバージョンを作成できます。これは、生成およびテスト中に検出された問題に対処するために行った変更を反映した、コードのもう 1 つのベースラインになります。

## Java 生成を行う場合の生成とテスト

Java 生成の準備のために、次のことを行う必要があります。

- この時点で、ソース・コード・リポジトリにある EGL プロジェクトとパッケージの別のバージョンを作成できます。これは、生成の前に手動で行った変更を反映した、コードのもう 1 つのベースラインになります。
- プログラムと dataTable を生成します。プログラムを生成する際には、次のビルド記述子オプションを使用します。
  - genFormGroup="YES"
  - genHelpFormGroup="YES"
  - genDataTables="NO"

これにより、formGroup を使用するプログラムと一緒に formGroup を生成でき、また dataTable を使用するプログラムの数に関係なく、dataTable の生成は一度だけにすることができます。生成中に検出された検証エラーは、すべて解決してください。

- VAGen 製品のメッセージ・テキストを変更した場合は、EGL メッセージ・テキストに対して同様な変更を行うことができます。
- 生成されたコードをテストします。
- この時点で、ソース・コード・リポジトリにある EGL プロジェクトとパッケージの別のバージョンを作成できます。これは、生成およびテスト中に検出された問題に対処するために行った変更を反映した、コードのもう 1 つのベースラインになります。

## 標準の検討

現行のコーディング標準を検討して、作成される新規コードに使用する新しい標準を設定する必要があることがあります。例えば、COBOL の生成を行う場合は、次のような標準を検討してください。

- パーツ名の中で、ハイフンの代わりに下線を使用する。COBOL プログラムは、名前に下線を使用することを許容しません。ただし、COBOL の生成によって下線が自動的にハイフンに変更されるので、生成される COBOL は引き続き読み取り可能です。下線をハイフンに変更した後、重複するパーツ名が存在する場合に限り、別名が割り当てられます。
- 生成される COBOL コードを読みやすくするために、生成によって別名が割り当てられるような名前の使用を避けます。このためには、次の命名規則を使用します。

- レコード名と関数名は 18 文字以下にする必要があります。これは、VisualAge Generator 内での制限と同じです。
- データ項目名は 27 文字以下にする必要があります。これは、VisualAge Generator の推奨限度である 30 文字、および最大の 32 文字よりわずかに少ない文字数です。
- プログラム名と dataTable 名は、8 文字にすることができます。

## VisualAge Generator 互換モードの使用を中止する場合

VAGen 互換モードは、VAGen プログラムの振る舞いを容易に保持できるように、さまざまな振る舞いをサポートしています。VAGen 互換モードをオフにする必要はありません。ただし、特に互換性のための振る舞いをあまり使用する必要がない場合は、VAGen 互換モードの使用を中止できます。次のリストでは、VAGen 互換モードをオンにした場合の EGL の振る舞いについて説明し、この振る舞いを使用する必要がなくなるようにするために役立つ情報を示します。

- EGL は、パーツ名の中でハイフン (-) と各国語文字 @ および # の使用を許容します。VAGen 互換モードの使用を中止する予定の場合は、ステージ 2 マイグレーション時に使用する名前変更ユーザー出口を作成する必要があります。ハイフン、@、および # の使用箇所を除去するための名前変更ユーザー出口を作成します。例えば、VAGen パーツ名の中で下線を使用していない場合は、ハイフンを下線に変更してパーツ名を変更する名前変更ユーザー出口を作成できます。ステージ 2 マイグレーションに対して名前変更ユーザー出口を指定する方法について詳しくは、171 ページの『VAGen マイグレーションの設定』を参照してください。VAGen 互換モードをオフにすると、ハイフン、@、または # を含むパーツ名または変数名に対して、「問題」リストにエラーが示されます。名前を変更することによって、この問題を訂正できます。
- EGL は、プリミティブ・データ型 *numc* と *pacf* の使用を許容します。*numc* と *num* は同様ですが、*numc* は正符号標識として C を使用するという点が異なります。*pacf* と *decimal* (VAGen の PACK) は同様ですが、*pacf* は正符号標識として F を使用するという点が異なります。VAGen 互換モードをオフにすると、プリミティブ型 *numc* または *pacf* を指定する *dataItem* 定義または変数宣言に対して、「問題」リストにエラーが示されます。プリミティブ型をそれぞれ *num* または *decimal* に変更して、新しい正符号標識に関連して必要なプログラムの変更を行うことにより、この問題を訂正できます。
- EGL は、1 次元の構造化フィールド配列の添え字をデフォルトで 1 に設定します。1 次元の構造化フィールド配列は、VAGen の配列、あるいはレコード、マップ、またはテーブル内で複数回出現する項目の EGL 置換表現です。VAGen 互換モードをオフにすると、添え字を必要とするようになった構造化フィールド配列を指定しているステートメントに対して、「問題」リストにエラーが示されます。この問題を訂正するには、添え字 1 を明示的に指定するようにステートメントを変更する必要があります。
- EGL は、dataTable の使用宣言の *deleteAfterUse* プロパティを許容します。*deleteAfterUse* プロパティは、VAGen の *Keep After Use* プロパティの置換表現です。VAGen 互換モードをオフにすると、*deleteAfterUse* を指定する使用宣言を含むそれぞれのプログラムごとに、「問題」リストにエラーが示されます。

`deleteAfterUse` プロパティを除去することによって、この問題を訂正できます。この場合、EGL によるテーブルの処理は、`Keep After Use = yes` を指定した VAGen テーブルと同様になります。

VisualAge Generator バージョン 4.5 フィックスパック 4 を使用して、すでに実動プログラムが生成済みの場合は、EGL `deleteAfterUse` プロパティを除去することによる影響はありません。システム共通プロダクト、または前のバージョンの VisualAge Generator を使用してマイグレーションしている場合は、`deleteAfterUse` プロパティを除去する変更を行なうすべてのプログラムについて十分テストする必要があります。

VAGen マイグレーション設定「テーブルに `deleteAfterUse` を組み込まない (*Do not include deleteAfterUse for tables*)」を指定すると、マイグレーション・ツールは自動的に `deleteAfterUse` プロパティを省略し、影響のあるプログラムおよびテーブルを対象に警告メッセージを出します。

- EGL は、SQL レコードの `sqlDataCode` プロパティを許容します。マイグレーション・ツールは、16 進項目に対して SQL 型 (例: SQL タイム・スタンプ) を指定する場合、または共用データ項目がマイグレーション・セットに含まれていないためにプリミティブ型が不明な場合に、VAGen の SQL Data Code プロパティを保持します。データ項目パーツをマイグレーション・セットに常に組み込むことによって、EGL `sqlDataCode` プロパティへのマイグレーションを最小限にすることができます。これにより、マイグレーション・ツールはプリミティブ型が不明であるために `sqlDataCode` を組み込まなくなります。ただし、16 進項目に対しては、マイグレーション・ツールは `sqlDataCode` を引き続き組み込む必要があります。VAGen 互換モードをオフにすると、`sqlDataCode` プロパティを指定する SQL フィールドに対して、「問題」リストにエラーが示されます。新しい EGL データ型のいずれかを使用するようにフィールドを変更し、この新しいデータ型を使用するために関連したプログラムの変更を行うことによって、この問題を訂正できます。可変長フィールドの場合は、`sqlVariableLen = yes` プロパティを組み込みます。
- EGL は、call ステートメントのオプション `externallyDefined` と `noRefresh` をサポートします。これらは、VAGen の call ステートメントに対する NONCSP オプションと NOMAPS オプションの置換表現です。VAGen 互換モードをオフにすると、`externallyDefined` または `noRefresh` を指定する call ステートメントに対して、「問題」リストにエラーが示されます。これらのオプションを call ステートメントから除去し、呼び出し先プログラムに対するリンケージ・オプション・エントリーの中で、対応する EGL 置換表現のオプションを指定することにより、この問題を訂正できます。ビルド記述子オプションの中で、このリンケージ・オプション・パーツを必ず指定してください。
- EGL は、transfer ステートメントと show ステートメントの `externallyDefined` オプションをサポートします。これは、VAGen の DXFR ステートメントと XFER ステートメントの NONCSP オプションに対する置換表現です。VAGen 互換モードをオフにすると、`externallyDefined` を指定する transfer ステートメントまたは show ステートメントに対して、「問題」リストにエラーが示されます。このオプションを transfer ステートメントまたは show ステートメントから除去し、転送元プログラムおよび転送先プログラムに対するリンケージ・オプション・エントリーの中で、対応する EGL 置換表現のオプションを指定することにより、この問題を訂正できます。ビルド記述子オプションの中で、このリンケージ・オプシ

ョン・パーツを必ず指定してください。リンケージ・オプション・パーツ内で、`transfer` ステートメントのタイプに応じて、転送先プログラムまたは転送先トランザクションの形式の転送リンク・エントリーを使用します。`show` ステートメントの場合、転送リンク・エントリーは、転送先トランザクション・エントリーの形式である必要があります。

- EGL は、`print printForm` ステートメントと同じように `display printForm` ステートメントをインプリメントします。マイグレーション・セットにマップを組み込むことによって、`display printForm` の使用を最小限にすることができます。これにより、マイグレーション・ツールは、テキスト・マップの場合は `display` ステートメント、プリンター・マップの場合は `print` ステートメントに正しくマイグレーションできます。VAGen 互換モードをオフにすると、それぞれの `display printForm` ステートメントごとに、「問題」リストにエラーが示されます。このステートメントを `print` ステートメントに変更することによって、この問題を訂正できます。
- EGL は、値が割り当てられていないフィールドを画面に表示するときのみ、書式フィールドの `value` プロパティを使用します。`value` プロパティは、ストレージ内の書式フィールドの初期値を設定しません。マイグレーション・ツールは、マップをマイグレーションする際に `value` プロパティを組み込みます。VAGen 互換モードをオフにすると、EGL は `value` プロパティを使用してストレージ内の書式フィールドの初期値を設定します。「問題」リストには、エラーが示される場合も示されない場合もあります。例えば、VisualAge Generator では、数値マップ変数フィールドの初期値を「MM/DD/YYYY」にすることにより、マップ・エディターのプレビュー・モードの値を指定でき、また `DISPLAY` または `CONVERSE` 入出力オプションの前にプログラムがこのフィールドにデータを移動しない場合に、エンド・ユーザーに出力を提供できました。この例では、VAGen 互換モードをオフにすると、値が `num` プリミティブ型と互換でないので、「問題」リストにメッセージが示されます。ただし、初期値が 5 のような数値ならば、「問題」リストにメッセージは示されません。ただし、プログラムは VisualAge Generator とは異なる振る舞いをする可能性があります。VAGen 互換モードをオフにする場合は、「問題」リストにあるメッセージを訂正するほかに、EGL 書式の `value` プロパティを検索して、振る舞いの変化しないようにするために必要なプログラムの変更を判別する必要があります。この変更は、例えば書式フィールドの `value` プロパティの除去や変更などです。
- EGL は、`VGVar.handleSysLibraryErrors` システム変数の使用を許容します。これは、`call` ステートメントまたは `EZE` 関数呼び出しステートメントに対する `EZERT8` (EGL `sysVar.errorCode`) の設定を制御する、VAGen `EZEREPLY` の置換表現です。VAGen 互換モードをオフにすると、`VGVar.handleSysLibraryErrors` を指定するそれぞれのステートメントごとに、「問題」リストにエラーが示されます。この問題は、`VGVar.handleSysLibraryErrors` の使用箇所を除去することによって訂正できます。`VGVar.handleSysLibraryErrors` が 1 にセットされている場合は、呼び出しまたは関数呼び出しを `try ... onError ... end` ブロック内でネストします。`VGVar.handleSysLibraryErrors` が 0 に設定されている場合は、`try ... onError ... end` ブロックを追加しないでください。
- ご使用の EGL のリリースが `DL/I` をサポートしている場合、EGL は `dliVar.handleHardDLIErrors` システム変数の使用を許可します。これは、VAGen `EZEDLERR` の置換表現です。VAGen では、`EZEDLERR` と `EZEFEC` (EGL `sysVar.handleHardIOErrors`) を組み合わせて使用してエラー処理を制御します。



EZEDLERR と EZEFECE の両方が 0 であるか、エラー・ルーチンが指定されていない場合は、ハード入出力エラーが発生するとプログラムは終了します。EZEDLERR または EZEFECE のどちらかが 1 に設定されていて、エラー・ルーチンが指定されている場合は、ハード入出力エラーが発生すると、エラー・ルーチンに制御が渡されます。一般に、EZEDLERR と EZEFECE はプログラムの開始時に設定され、以後変更されることはありません。VAGen 互換モードをオフにすると、`dliVar.handleHardDLIErrors` を指定するそれぞれのステートメントごとに、「問題」リストにエラーが示されます。プログラム・ロジックを検討して、`dliVar.handleHardDLIErrors` を `sysVar.handleHardIOErrors` と組み合わせて問題がないかどうか判断する必要があります。`sysVar.handleHardIOErrors` がプログラム内で使用されていない場合、またはプログラムの開始時にのみセットされている場合は、`dliVar.handleHardDLIErrors` と `sysVar.handleHardIOErrors` を比較的単純に組み合わせることができます。

- EGL は、`VGLib.getVAGSysType` システム関数の使用を許容します。マイグレーション・ツールは、`customerPrefixEZESYS` という名前の変数を宣言し、それぞれのプログラム内でこの変数を `VGLib.getVAGSysType` の結果に初期化します。`customerPrefix` は、ステージ 2 マイグレーション時に指定した名前変更接頭部です。`VGLib.getVAGSysType` は、IF、WHILE、または TEST 以外のステートメントをマイグレーションする際に使用する、オリジナルの VAGen EZESYS システム値 (例: MVSCICS または OS400) を提供します。VAGen 互換モードをオフにすると、`customerPrefixEZESYS` を初期化するステートメントに対して、それぞれのプログラムの「問題」リストにエラーが示されます。プログラムを変更して、`customerPrefixEZESYS` 宣言と初期化ステートメントを除去することによって、この問題を訂正できます。プログラムを保管するとき、`customerPrefixEZESYS` を使用するそれぞれのステートメントごとに、「問題」リストにさらにエラーが示されます。このエラーを訂正するには、新しい EGL システム値を提供する EGL `sysVar.systemType` システム変数を使用するようにステートメントを変更できます。`customerPrefixEZESYS` の使用方法によっては、データベース、ファイル、または `dataTable` の値を、以前の VAGen システム値から新しい EGL システム値に変更する必要があることもあります。114 ページの『EZESYS』 および 339 ページの EZESYS の状態条件を参照してください。

VAGen マイグレーション設定「以前の EZESYS 値を初期化しない (*Do not initialize old EZESYS values*)」を指定すると、マイグレーション・ツールは自動的にそれぞれのプログラムの変数宣言と初期化ステートメントを省略します。`customerPrefix EZESYS` を使用するステートメントに関して、問題ビューにエラーが示されます。

- EGL は、`VGLib.connectionService` システム関数の使用を許容します。これは、指定した引数とランタイム環境に応じてさまざまな SQL 接続サービスを提供する、VAGen EZECONCT システム関数の置換表現です。VAGen 互換モードをオフにすると、`VGLib.connectionService` を使用するそれぞれのステートメントごとに、「問題」リストにエラーが示されます。新しい EGL 特殊システム関数 (例: `sysLib.connect`、`sysLib.disconnect`、`sysLib.disconnectAll`、または `sysLib.queryDatabase`) のいずれかを使用するように変更することによって、エラーを訂正できます。使用する EGL システム関数は、VAGen EZECONCT システム関数に対して指定した引数、およびランタイム環境によって異なります。また、

VGVar.sqlIsolationLevel は、EGL システム関数の選択、またはこのシステム関数に対して指定する必要がある引数の選択に影響する可能性があるため、この使用についても調べる必要があります。

- EGL は、*ConverseVar.segmentedMode* システム変数の使用を許可します。これは、メインの CICS トランザクション・プログラム内で、実行時にセグメント化モードと非セグメント化モードの切り替えを可能にする、VAGen EZESEGM の置換表現です。VAGen 互換モードをオフにすると、*ConverseVar.segmentedMode* を使用するそれぞれのステートメントごとに、「問題」リストにエラーが示されます。EZESEGM が使用されることはまれなので、多くの場合、「問題」リストにエラーは示されません。エラーがある場合、*ConverseVar.segmentedMode* を使用しないようにするために、セグメント化モードと非セグメント化モードの切り替えが必要なくなるように、プログラムの再構築が必要になる可能性があります。
- EGL は、*VGVar.sqlIsolationLevel* システム変数の使用を許容します。これは VAGen EZESQISL の置換表現です。この変数は、システム共通プロダクトの旧リリースと VSE ランタイム環境用の VisualAge Generator で SQL 分離レベルを制御するために使用され、また VisualAge Generator 4.5 では ODBC データベースへのアクセスに使用されます。VAGen 互換モードをオフにすると、*VGVar.sqlIsolationLevel* を使用するそれぞれのステートメントごとに、「問題」リストにエラーが示されます。EZESQISL が使用されることはまれなので、多くの場合、「問題」リストにエラーは示されません。エラーがある場合は、*VGVar.sqlIsolationLevel* がプログラム・ロジックの制御に使用されていないければ、この変数を完全に除去できます。また、*VGVar.sqlIsolationLevel* が一部のプログラム・ロジックの制御に使用されている場合は、プログラム内で宣言した新規変数によって *VGVar.sqlIsolationLevel* を置き換えることができます。システム関数 *VGLib.connectionService* の振る舞いは *VGVar.sqlIsolationLevel* の値によって決まる場合があるので、このシステム関数の使用箇所を必ず調べてください。
- SQL レコードにない SQL ホスト変数は、データベースから検索された値が NULL ならば、プリミティブ型に応じて、EGL によってブランクまたはゼロに初期化されます。SQL レコード内のホスト変数は、フィールドに対して *isNullable=yes* プロパティが設定されている場合にのみ、EGL によって初期化されます。VAGen 互換モードをオフにした場合、「問題」リストにエラーは示されません。ただし、データベースから検索された値が NULL である場合にフィールドが初期化されないため、プログラムは VisualAge Generator の場合とは異なる動作をする可能性があります。一般に、VAGen 互換モードを問題なくオフにできるかどうかを判断するには、それぞれのプログラムを慎重に検討する必要があります。すべてのデータベース列が SQL に対して NOT NULL として定義されていることが分かっているならば、VAGen 互換モードを問題なくオフにできます。また、SQL データベースからデータを検索する前にホスト変数をブランクまたはゼロに初期化する必要がある場合は、NULL データを含むすべての列がデータベース・アクセスの前にすでに初期化されているため、VAGen 互換モードを問題なくオフにできます。SQL レコードにないホスト変数は、必ず初期化してください。
- EGL は、SQL WHERE 文節と EGL prepare ステートメント内のホスト変数参照の場合を除き、精度を 1 つ増やすことによって 10 進フィールド (VAGen PACK フィールド) の偶数精度をサポートします。VAGen 互換モードをオフにした場合、「問題」リストにエラーは示されません。ただし、プログラムは VisualAge Generator の場合と異なる動作をする可能性があります。具体的には、偶数精度を



もつ 10 進フィールドが、データベースに格納されているデータに対して小さすぎる可能性があります。一般に、VAGen 互換モードを問題なくオフにできるかどうかを判断するには、それぞれのデータ項目を慎重に検討する必要があります。VisualAge Generator では、「参照」ツールを使用してテキスト・ストリング `evensql = Y` (= 記号の前後にブランクを 1 つ入れる) を検索することにより、すべてのデータ項目とレコード・パーツを検索できます。この検索により、偶数精度を指定した項目またはレコードがあるかどうか判別できます。EGL では、`decimal(2, decimal(4, .... decimal(18` を検索することによって、偶数精度の 10 進フィールドを検索できます。偶数精度の 10 進フィールドを使用していない場合は、VAGen 互換モードを問題なくオフにできます。偶数精度の 10 進フィールドを使用していた場合は、次に大きな奇数精度への変更が SQL アクセスのパフォーマンスに及ぼす影響を検討する必要があります。EGL ホスト変数が SQL 列定義の精度と正確に一致している場合に、SQL は 10 進フィールドに対して良好なパフォーマンスを発揮します。

VAGen マイグレーション設定「項目または変数の `evensql=y` を受け入れない (*Do not honor evensql=y for items or variables*)」を指定すると、マイグレーション・ツールは自動的に奇数精度 (項目が最大長である場合は 18) を使用し、影響のあるデータ項目パーツや非共用レコード項目を対象に警告メッセージを出します。

VAGen 互換モード設定をオフにする場合は、それぞれのビルド記述子パーツから `vagCompatibility="YES"` を必ず除去してください。VAGen マイグレーション設定「互換モードを設定しない (*Do not set compatibility mode*)」を指定すると、マイグレーション・ツールは自動的にそれぞれのビルド記述子パーツから `vagCompatibility="YES"` を省略します。



---

## 第 6 部 言語と実行時に関する違い

VisualAge Generator と EGL の間には、さまざまな言語の違い、および実行時の違いがあります。



---

## 第 10 章 言語と実行時に関する違い

---

### 言語に関する違い

EGL が VisualAge Generator Developer を完全に置き換えない部分については、 8 ページの『EGL にマイグレーションできるかどうかの判別』を参照してください。

EGL 言語に正確にマイグレーションできない VAGen 言語エレメント、またはマイグレーション方式については、 59 ページの『第 3 章 未確定状態の処理』を参照してください。

それぞれの VAGen 言語エレメントを EGL にマイグレーションする方法の詳細については、 247 ページの『付録 B. VisualAge Generator と EGL の言語エレメントの関係』を参照してください。

---

### 実行時の違い

ソース・コードを EGL にマイグレーションした後、コードを生成し、十分なテストを行って、コードが VisualAge Generator の場合と同じように実行されることを確認する必要があります。具体的な実行時の違いは、ターゲット環境によって次のように異なります。

- 一般的な違い
- デバッグに関する違い
- 生成される COBOL に関する違い
- 生成される Java に関する違い
- ホスト環境とワークステーション環境の違い
- 分散 CICS 環境とネイティブ・ワークステーション環境の違い
- 生成される C++ と生成される Java の違い

#### 一般的な違い

次に示す実行時の振る舞いの違いは、発生してもマイグレーション・ログや「問題」ビューにメッセージが示されない可能性があります。デバッグ時、または生成された Java コードまたは COBOL コードの実行時に、問題が起こることがあります。

- テキスト・プログラムと印刷フォームには、次のことが当てはまります。
  - 書式フィールドの長さが不足していて、数字とフォーマット設定情報 (符号、小数点、通貨記号、および数字分離記号) がすべて収まらない場合は、ランタイム・エラーが発生します。
  - デフォルト以外の充てん文字は、プログラムが SET formItem FULL ステートメントを発行しなくても、常に受け入れられます。
  - 書式にある配列は、配列の先頭エレメントの検証プロパティとフォーマット設定プロパティを常に使用します。このため、配列の要素によってこれらの

プロパティが異なることを許容していた VisualAge Generator とは少し異なる振る舞いが生じる場合があります。詳しくは、82 ページの『マップ配列と属性』を参照してください。

- row=0, column=0 のフィールドがマップに含まれていた場合は、対応する書式を使用するプログラムをテストして、外観や振る舞いの違いを必ず確認してください。詳しくは、84 ページの『row=0, column=0 にあるフィールド』を参照してください。
- VAGen REDEFINED レコードであるレコードが、プログラムのマイグレーション時に使用不可ならば、マイグレーション・ツールはデータ宣言に *EGL redefines* プロパティを組み込みません。この結果、VisualAge Generator の場合は単一の領域に 2 つの定義がありましたが、その代わりにレコード域が 2 つに分割されます。このため、未初期化のデータや無効データが原因で、異常終了などのエラーが発生する可能性があります。詳しくは、65 ページの『再定義レコード』を参照してください。
- EGL では、VisualAge Generator より多くの状況でハード入出力エラーが起こります。
  - VisualAge Generator の場合、非 SQL レコードの UNQ はソフト・エラーなので、HRD 入出力エラー状態は設定されません。EGL の場合、*unique* はハード・エラーなので、*hardIOError* のテスト結果は true になります。詳しくは、110 ページの『入出力エラー値 UNQ および DUP』を参照してください。
  - iSeries の場合、VAGen 入出力エラー値 LOK は EGL *deadlock* 入出力エラー状態にマイグレーションされます。VisualAge Generator の場合、LOK はソフト・エラーなので、HRD 入出力エラー状態は設定されません。EGL の場合、*deadlock* はハード・エラーなので、*hardIOError* のテスト結果は true になります。詳しくは、112 ページの『入出力エラー値 LOK』を参照してください。
- 関数のマイグレーション時に入出力エラー・ルーチンが使用不可ならば、マイグレーション・ツールは入出力エラー・ルーチンが main 関数でないと想定し、関数呼び出しに変換します。VisualAge Generator では、入出力エラー・ルーチンが main 関数である場合に、エラーが実行時に発生すると、VisualAge Generator は関数の現行の実行スタックを消去し、入出力エラー・ルーチンとして指定された main 関数のみを含む新規スタックを開始します。これにより、実行スタック用のストレージも消去されます。EGL では、マイグレーション・ツールが関数呼び出しへの変換を行うので、エラーが実行時に発生すると、EGL は main 関数を現行の実行スタックに追加します。現行の実行スタックを消去して、main 関数のみの新規スタックを開始することはありません。このため、関数にローカル・ストレージやパラメーター・リストがある場合は、無限ループが発生したり、大量のリソースが使用されたりする可能性があります。詳しくは、93 ページの『入出力エラー・ルーチン』を参照してください。

## デバッグに関する違い

デバッグに関する違いが、テストに影響を及ぼす可能性があります。COBOL 環境で生成を行う場合、デバッグが提供する次の領域のサポートは、生成される COBOL のサポートと異なるので、特にこれらの違いに注意する必要があります。

- マップに関する違いは、次のとおりです。
  - テキスト書式では明滅はサポートされません。



- *isDecimalDigit* プロパティは、文字フィールドのみに対してサポートされます。ハードウェア属性としてではなく、ソフトウェア編集としてインプリメントされます。数値フィールドもソフトウェア編集を使用します。詳しくは、81ページの『マップ・フィールドと数値ハードウェア属性』を参照してください。
- 代替索引レコードが定義されている索引レコードの場合、DUP 入出力エラー値の設定は VisualAge Generator とは異なります。VisualAge Generator の場合、SET record SCAN に続いて SCAN または SCANBACK 入出力オプションを指定した場合、SET record SCAN ステートメントに対して DUP 入出力エラー値は設定されません。DUP 入出力エラー値は、最後に検索された重複キー・レコードを除く、それぞれの重複キー・レコードに対して設定されます。Java 生成の場合、*set record position* の後に *get next* ステートメントまたは *get previous* ステートメントが続いていると、最初に検索された重複キー・レコードに対してではなく、*set record position* に対して *duplicate* 入出力エラー状態が設定されます。その他の重複キー・レコードに対しては、VisualAge Generator の場合と同じように、*duplicate* 入出力エラー状態が設定されます。EGL の *duplicate* 状態は、最初と最後の重複キー・レコードを除くすべてのレコードに対して設定されます。索引レコードと代替索引レコードの詳細、および *set record position*、*get next*、*get previous* に対してこれらのレコードを使用する方法については、オンライン・ヘルプを参照してください。
- SQL に関する違いは、次のとおりです。
  - ODBC は EGL ではサポートされません。DB2 以外の SQL データベース・マネージャーを使用する場合は、ご使用のデータベース・マネージャー用の JDBC ドライバーを入手する必要があります。
  - JDBC は 2 フェーズ・コミットをサポートしません。このため、次の違いがあります。
    - コミットとロールバックに関して、SQL マネージャーと MQ series マネージャーの呼び出しは別個に行われます。このため、問題が発生した場合に、一方のリソースのコミットとロールバックが行われ、対応する他方のリソースのコミットとロールバックが行われない可能性があります。
    - EZECONCT (EGL VGLib.connectionService)。VisualAge Generator の場合は、*unit of work* 引数の R オプションにより、現行接続を終了せずに接続を別のデータベースに切り替えることができます。これにより、同じ作業単位内で複数のデータベースを更新できます。EGL では、この R オプションは、*unit of work* 引数の D1C、D2A、D2C、および D2E の各オプションと同様に、D1E を指定した場合と同じように扱われます。D1E は 1 フェーズ・コミットですが、データベース接続を自動的に解放しません。データベースを切断するには、DISC、DCURRENT、または DALL オプションを明示的に要求する必要があります。詳しくは、VGLib.connectionService のオンライン・ヘルプを参照してください。
    - EZESQISL (EGL VGVar.sqlIsolationLevel)。VisualAge Generator では、値 1 はカーソル固定の要求を表します。EGL では、値 1 は順序付け可能トランザクションの要求を表します。
    - EZESQRRM (EGL VGVar.sqlerrmc) はサポートされません。
    - EZESQWN6 (EGL VGVar.sqlwarn[7]) はサポートされません。

- EZESQLCA (EGL sysVar.sqlca) フィールドには制限があります。これらのフィールドには、EZESQRRM と EZESQWN6 の値を含めることはできません。

## 生成される COBOL に関する違い

生成される COBOL コードには、次の違いが生じます。

- EGL によって生成される COBOL テキスト・プログラムと基本プログラムは、VisualAge Generator プログラムと完全に互換です。次のどちらの場合も、VAGen プログラムの再生成または再コンパイルは必要ありません。
  - VAGen プログラムが、EGL への転送手段として CALL、DXFR、または XFER を使用しています。
  - EGL プログラムが、VAGen プログラムへの転送手段として *call*、*transfer to program*、*transfer to transaction*、または *show* を使用している。

EGL プログラムと VAGen プログラムの間で行われる呼び出しまたは転送に関する制限は、2 つの VAGen プログラム間で呼び出しまたは転送を行う場合の制限と同様です。例えば、VAGen の呼び出し先プログラムは、他のプログラムへの転送に DXFR ステートメントまたは XFER ステートメントを使用できません。同様に、EGL の呼び出し先プログラムは、他のプログラムへの転送に *transfer to program*、*transfer to transaction*、または *show* を使用できません。

- マップに関する違いは、次のとおりです。
  - *isDecimalDigit* プロパティは、文字フィールドのみに対してサポートされます。このプロパティは、数値ハードウェア属性としてインプリメントされます。数値フィールドはソフトウェア編集を使用します。詳しくは、81 ページの『マップ・フィールドと数値ハードウェア属性』を参照してください。
  - 6x40、12x40、16x64、および 255x60 の装置サイズはサポートされなくなりました。詳しくは、76 ページの『マップ・グループ、マップ、および装置サイズ』を参照してください。

## 生成される Java に関する違い

生成される Java コードには、次の違いが生じます。

- EGL 生成 Java プログラムは、VisualAge Generator プログラムと完全互換ではありません。

次のことがサポートされています。

- EGL プログラムは、リモート呼び出しを使用して VAGen 生成 Java または C++ プログラムを呼び出すことができます。同様に、VAGen 生成 Java または C++ プログラムは、リモート呼び出しを使用して EGL プログラムを呼び出すことができます。
- EGL VGWebTransaction プログラムは、*show VGUI record* ステートメントを使用して、VAGen Web Transaction プログラムに間接的に転送することができます。同様に、VAGen Web トランザクション・プログラムは、UI record ステートメントを含む XFER を使用して、EGL VGWebTransaction プログラムに間接的に転送することができます。

次のことはサポートされていません。

- EGL プログラムは、ローカル呼び出しを使用して VAGen 生成 Java または C++ プログラムを呼び出すことはできません。同様に、VAGen 生成 Java または C++ プログラムは、ローカル呼び出しを使用して EGL プログラムを呼び出すことはできません。
- EGL プログラムは、*transfer to program*、*transfer to transaction*、または *show textForm* ステートメントを使用して VAGen プログラムに転送することはできません。同様に、VAGen プログラムは、*DXFR* または *XFER* ステートメントを使用して EGL プログラムに転送することはできません。

## ホスト環境とワークステーション環境の違い

ホスト環境 (CICS など) への生成からワークステーション環境 (ネイティブ AIX など) への生成に変更する場合は、以下を考慮する必要があります。

- ホスト環境の照合シーケンスは EBCDIC です。Java 生成の照合シーケンスは UNICODE です。したがって、以下を検討する必要があります。
  - 適合テーブルの範囲
  - キーの high-value または low-value としてコード化された特定の値
  - 2 つのファイルが一致するキーの比較
- エンド・ユーザーは、以下を知っておく必要があります。
  - ファンクション・キーのマッピング。通常、ワークステーション・キーボードには、PF13 - PF24 および PA1 - PA3 のキーがないためです。
  - DBCS モードから、または DBCS モードへの切り替えを行なうときに、シフトイン文字およびシフトアウト文字を入力する必要がないこと。
- CALL で AIX ネイティブから CICS へ渡されるすべてのレコードの全長は 32567 を超えることはできません。これは、CICS の制限事項です。以前に CICS の生成を行い、デフォルトの parmform=COMMPTR を使用した場合は、すべてのレコードの全長がこの制限を超える可能性があります。以前に parmform=COMMDATA を使用した場合は、すべてのレコードの全長はこの制限内になります。
- 以下のセクションの情報も検討してください。
  - 『分散 CICS 環境とネイティブ・ワークステーション環境の違い』
  - 233 ページの『生成される C++ と生成される Java の違い』

## 分散 CICS 環境とネイティブ・ワークステーション環境の違い

生成された EGL コードをワークステーション環境で実行するには、Transaction Series (TX Series または CICS) の環境で実行する代わりに、ネイティブ・プロセスとして実行するように変更する必要があります。次に、CICS 環境からネイティブ環境への移行に伴う相違点または変更点を示します。このリストは VAGen 用語を使用していますが、対応する EGL 言語エレメント内では変更を加える必要があります。対応する EGL 言語エレメントを判別するには、247 ページの『付録 B. VisualAge Generator と EGL の言語エレメントの関係』を参照してください。

- 一般的な違いは、次のとおりです。
  - 複数のユーザーが 1 つのサーバー上の同じアドレス・スペースで実行することはできません。ユーザーはそれぞれのワークステーションで実行します。
  - クライアントの作業単位はサポートされません。

- C++ の生成から Java の生成に変更されました。必ず 233 ページの『生成される C++ と生成される Java の違い』に関するセクションを参照してください。
- CICS からネイティブ環境にマイグレーションする際には、パフォーマンスとスケーラビリティを必ずテストしてください。
- CICS とネイティブ環境の間では、通信プロトコルが異なります。使用するプロトコルを決定してから、それに応じて EGL リンケージ・オプション・パーツとリソース関連パーツを変更する必要があります。
- 次の CICS 固有の特殊機能語とサービス・ルーチンは、ネイティブ環境ではサポートされません。
  - CICS ジャーナル・エントリーを書き込む場合の AUDIT (EGL sysLib.audit)。AUDIT という名前の非 EGL プログラムをユーザーが独自に作成して、ネイティブ環境用の同様な情報をファイルに書き込むことができます。
  - 一時ストレージ・キューを削除するための EZEPURGE (EGL sysLib.purge)。sysLib.purge の参照は除去する必要があります。代わりに、sysVar.systemType を検査して、z/OS 環境の CICS で実行するときのみ sysLib.purge を使用することもできます。この手法を使用する場合は、EGL ビルド記述子オプション eliminateSystemDependentCode="YES" を必ず組み込んでください。
  - リモート・ファイル、リモート・プログラムのロケーション、または CREATX (EGL sysLib.startTransaction) を使用してリモート・トランザクションを開始するロケーションを設定するための EZELOC (EGL sysLib.remoteSystemID)。
- CICS 固有のリソース関連は、ネイティブ環境ではサポートされません。EGL ネイティブ環境でサポートされるオプションを使用するように、リソース関連パーツを変更する必要があります。ネイティブ環境で生成を行う場合にサポートされない、CICS 固有のリソース関連は次のとおりです。
  - CICS スプール・ファイル。
  - 一時データ・キュー (トリガー・レベル 1 の一時データ・キューを含む)。
  - 一時ストレージ・キュー。
  - ローカル VSAM ファイル (AIX 環境の場合を除く)。
- 次に示す CICS 提供の機能は、ネイティブ環境ではサポートされません。
  - セキュリティー・サービス。
  - データベース接続と保存。
  - CICS ファイル管理 (リカバリー可能ファイルの使用など)。
  - 実際のセグメンテーションのサポート。
  - プログラム管理。
- プログラム間で転送を行う場合の違いは、次のとおりです。
  - Web トランザクション以外のメインプログラムの場合、CICS の XFER ステートメントは、次のトランザクション ID に転送されます。ネイティブ環境の場合、XFER ステートメントはプログラム名に転送されます。このため、すべての EGL *transfer to transaction* ステートメントと *show* ステートメントを変更して、プログラム名を指定する必要があります。

- CICS 環境では、非 VAGen プログラムへの XFER または DXFR がサポートされます。ネイティブ環境の場合、非 EGL プログラムへの *transfer to program*、*transfer to transaction* および *show* はサポートされません。
- コミットとロールバックに関する違いは、次のとおりです。
  - CICS は 2 フェーズ・コミットをサポートします。ネイティブ環境は、単一フェーズ・コミットのみをサポートします。
  - CICS に対しては、ファイルをリカバリー可能ファイルとして定義できます。ネイティブ環境の場合は、このように定義することはできません。
  - メッセージ・キューは、CICS 環境では他のリソースと同時にコミットまたはロールバックされます。ネイティブ環境では、単一フェーズ・コミットのみがサポートされるので、コミットとロールバック中に問題が発生した場合は、メッセージ・キューが SQL リソースと同時にコミットまたはロールバックされない可能性があります。
- CALL CREATX (EGL sysLib.startTransaction) に関する違いは、次のとおりです。
  - CALL CREATX は、CICS 環境で別のトランザクションを開始し、パラメーター *prid* と *recip* を受け入れます。EGL sysLib.startTransaction は、ネイティブ環境で別のプログラムを開始し、パラメーター *prid* と *recip* を無視します。ネイティブ環境用の生成を行う場合は、少なくとも sysLib.startTransaction を変更してプログラム名を指定する必要があります。
  - CICS は、ローカルとリモートの両方の CALL CREATX をサポートします。EGL sysLib.startTransaction は、ローカル・プログラムの開始のみをサポートします。
- EZECONCT (EGL VGLib.connectionService) を使用した SQL 接続サービス。  
VisualAge Generator の CICS 環境の場合、EZECONCT はパスワードを無視します。EGL のネイティブ環境の場合、VGLib.connectionService はパスワードを使用します。Java 生成の変更に伴うその他の違いについては、233 ページの『生成される C++ と生成される Java の違い』を参照してください。
- EZE 特殊データ・ワードに関する違いは、次のとおりです。
  - EZEAPP (EGL sysVar.transferName)。VisualAge Generator の CICS 環境の場合に、XFER ステートメントに EZEAPP を使用するときは、EZEAPP の内容は開始される新規トランザクションの名前です。EGL のネイティブ環境の場合に、transfer to transaction ステートメントまたは show ステートメントに sysVar.transferName を使用するときは、sysVar.transferName の内容は開始される新規プログラムの名前です。
  - EZEDEST (EGL sysVar.resourceAssociation)。VisualAge Generator の CICS 環境の場合、EZEDEST の内容はプログラムの実行中にレコードに関連付けられるシステム・リソース名です。EGL のネイティブ環境の場合も、sysVar.resourceAssociation の内容は、レコードに関連付けられるシステム・リソース名です。ただし、情報のフォーマットはランタイム環境とファイル・タイプによって異なります。したがって、ランタイム環境とファイル・タイプの両方を変更するので、sysVar.resourceAssociation が使用されている箇所をすべて検討して、プログラムから提供される情報がご使用のネイティブ環境およびファイル・タイプに適合していることを確認してください。
  - EZEDESTP (EGL converseVar.printerAssociation)。VisualAge Generator の CICS 環境の場合、EZEDESTP の内容は印刷ファイルに関連付けられる宛先です。



EGL のネイティブ環境の場合、`converseVar.printerAssociation` の内容は印刷ファイルに関連付けられるファイル名です。ただし、情報のフォーマットはランタイム環境とファイル・タイプによって異なります。したがって、ランタイム環境とファイル・タイプの両方を変更するので、`converseVar.printerAssociation` が使用されている箇所をすべて検討して、プログラムから提供される情報がご使用のネイティブ環境およびファイル・タイプに適合していることを確認してください。

- EZELTERM (EGL `sysVar.terminalID`)。VisualAge Generator の CICS 環境の場合、EZELTERM の内容は CICS 端末 ID で、これは EZEUSR と同等です。EGL のネイティブ環境の場合、`sysVar.terminalID` は Java 仮想マシンのシステム・プロパティ `user.name` から初期設定されます。このプロパティが検索できない場合、`sysVar.terminalID` の内容はブランクになります。
- EZERCODE (EGL `sysVar.returnValue`)。VisualAge Generator の CICS 環境の場合、EZERCODE の値はシステムまたは呼び出し側プログラムに戻されません。EGL のネイティブ環境の場合、EZERCODE の値は無視されます。
- EZERT8 (EGL `sysVar.errorCode`)。VisualAge Generator の CICS 環境の場合、EZERT8 は次のどちらかの形式になります。
  - *RSnnnnnn*。ここで *nnnnnn* は、ファイル・アクセスと発生した問題に基づく VAGen 戻りコードです。
  - *nnnnnnnn*。ここで、最初の 2 文字は、CICS EXEC インターフェース・ブロックにある EIBFN の先頭バイトの 16 進表記です。残りの 6 文字は、CICS EXEC インターフェース・ブロックにある EIBRCODE のバイト 0 から 2 の 16 進表記です。

EGL のネイティブ環境の場合、戻りコード情報はファイル・タイプによって異なります。`sysVar.errorCode` を使用している箇所を検討して、検査する値がご使用の環境とファイル・タイプに適合していることを確認する必要があります。

- EZESEGT (EGL `sysVar.transactionID`)。VisualAge Generator の CICS 環境の場合、EZESEGT は現行トランザクション ID に初期設定され、また CONVERSE の後に有効になる新規トランザクション ID の設定にも使用されます。EZESEGT は、プログラム・ロジックの制御に使用できます。EGL のネイティブ環境の場合、EZESEGT は無視され、プログラム・ロジックの制御には使用できません。
- EZEUSR (EGL `sysVar.sessionID`)。VisualAge Generator の CICS 環境の場合、EZEUSR の内容は CICS 端末 ID で、これは EZELTERM と同等です。EGL のネイティブ環境の場合、`sysVar.sessionID` は Java 仮想マシンのシステム・プロパティ `user.name` から初期設定されます。このプロパティが検索できない場合、`sysVar.userID` の内容はブランクになります。
- EZEUSRID (EGL `sysVar.userID`)。VisualAge Generator の CICS 環境の場合、ユーザーがシステムにサインオンしていれば、EZEUSRID の内容は CICS ユーザー ID です。そうでなければ、内容はブランクです。EGL のネイティブ環境の場合、`sysVar.userID` は Java 仮想マシンのシステム・プロパティ `user.name` から初期設定されます。このプロパティが検索できない場合、`sysVar.userID` の内容はブランクになります。



## 生成される C++ と生成される Java の違い

C++ の生成から、Java の生成に変更する場合は、次の違いが生じます。

- 生成される Java は、VAGen によって生成される C++ プログラムと相互運用できません。Java 用に生成された EGL プログラムは、C++ 用に生成された VAGen プログラムとの間で相互に転送を行うことができません。Java 用に生成された EGL プログラムは、C++ 用に生成された VAGen の呼び出し先バッチ・プログラムを呼び出すことができます。
- 生成された Java プログラムからネイティブ C++ または VAGen 生成 C++ への呼び出しは、プログラムが同じワークステーションで実行されている場合であっても、常にリモート呼び出しになります。したがって、この状況に対するリンク・テーブル・エントリを追加する必要があります。
- VAGen によって生成される C++ プログラム内では、バイナリー・データは Intel フォーマット (逆方向のバイト・オーダー) で格納されます。EGL によって生成される Java プログラム内では、バイナリー・データは Intel フォーマットでは格納されません。生成される Java プログラムと、生成される C++ プログラムの間で呼び出し時に受け渡されるバイナリー・データは、EGL 変換テーブルによって変換されます。ただし、生成される Java プログラム内でファイルを使用する場合は、その前に、C++ プログラムによって書き込まれたファイル内のバイナリー・データを変換するプログラムを作成する必要があります。
- 生成される Java プログラムの `transfer` ステートメント、または `show` ステートメントのネーム・レゾリューション規則は次のとおりです。
  - `transfer` ステートメントまたは `show` ステートメントがパッケージ名とプログラム名を明示的に指定している場合は、このプログラムが使用されます (例: `transfer to program mypackage.program1`)。
  - `transfer` ステートメントまたは `show` ステートメントがプログラム名のみを明示的に指定している場合は、次のうち最初に該当する情報がプログラム名の解決に使用されます。
    - `transfer` ステートメントまたは `show` ステートメントを含むファイルが、パッケージとプログラムを明示的にインポートする (例: `import mypackage.program1` および `transfer to program program1`)。
    - 転送先プログラムが、`transfer` ステートメントまたは `show` ステートメントと同じパッケージ内にある。
    - `transfer` ステートメントまたは `show` ステートメントを含むファイルが、転送先プログラムを含むパッケージをインポートする (例: `import mypackage.*` および `transfer to program program1`)。
  - 生成時に使用されるリンク・オプション・パーツが、転送先プログラムとそのプログラムを含むパッケージを指定する、転送元プログラムの転送リンク・エントリを含んでいる。`show` ステートメントの場合、転送リンク・エントリは、転送先トランザクション・エントリの形式であることが必要です。
- `transfer` ステートメントまたは `show` ステートメントが `sysVar.transferName` を使用し、`transfer-to` プログラムが別のパッケージにある場合、リンク・オプション・パーツを使用して、`transfer-to` プログラムを含むパッケージを指定しなければなりません。
- 一般的な違いは、次のとおりです。

- VisualAge Generator によって生成された C++ コードを使用する場合、リソース関連は実行時に処理されます。EGL の場合は、生成時にリソース関連情報を指定して、この情報をプロパティ・ファイルに生成するオプションがあります。リソース関連ビルド記述子を設定して、リソース関連パーツを指示します。また、`genProperties` ビルド記述子を `GLOBAL` または `PROGRAM` に設定します。これらのビルド記述子オプションについて詳しくは、オンライン・ヘルプを参照してください。
- マップに関する違いは、次のとおりです。
  - テキスト書式では明滅はサポートされません。
  - `isDecimalDigit` プロパティは、文字フィールドのみに対してサポートされます。ハードウェア属性としてではなく、ソフトウェア編集としてインプリメントされます。数値フィールドもソフトウェア編集を使用します。詳しくは、81 ページの『マップ・フィールドと数値ハードウェア属性』を参照してください。
- 代替索引レコードが定義されている索引レコードの場合、`DUP` 入出力エラー値の設定は VisualAge Generator とは異なります。VisualAge Generator の場合、`SET record SCAN` に続いて `SCAN` または `SCANBACK` 入出力オプションを指定した場合、`SET record SCAN` ステートメントに対して `DUP` 入出力エラー値は設定されません。`DUP` 入出力エラー値は、最後に検索された重複キー・レコードを除く、それぞれの重複キー・レコードに対して設定されます。Java 生成の場合、`set record position` の後に `get next` ステートメントまたは `get previous` ステートメントが続いていると、最初に検索された重複キー・レコードに対してではなく、`set record position` に対して `duplicate` 入出力エラー状態が設定されます。その他の重複キー・レコードに対しては、VisualAge Generator の場合と同じように、`duplicate` 入出力エラー状態が設定されます。EGL の `duplicate` 状態は、最初と最後の重複キー・レコードを除くすべてのレコードに対して設定されます。索引レコードと代替索引レコードの詳細、および `set record position`、`get next`、`get previous` に対してこれらのレコードを使用する方法については、オンライン・ヘルプを参照してください。
- SQL に関する違いは、次のとおりです。
  - ODBC は EGL ではサポートされません。DB2 以外の SQL データベース・マネージャーを使用する場合は、ご使用のデータベース・マネージャー用の JDBC ドライバーを入手する必要があります。
  - JDBC は 2 フェーズ・コミットをサポートしません。このため、次の違いがあります。
    - コミットとロールバックに関して、SQL マネージャーと MQ series マネージャーの呼び出しは別個に行われます。このため、問題が発生した場合に、一方のリソースのコミットとロールバックが行われ、対応する他方のリソースのコミットとロールバックが行われない可能性があります。
    - `EZECONCT` (EGL `VGLib.connectionService`)。VisualAge Generator の場合は、`unit of work` 引数の `R` オプションにより、現行接続を終了せずに接続を別のデータベースに切り替えることができます。これにより、同じ作業単位内で複数のデータベースを更新できます。EGL では、この `R` オプションは、`unit of work` 引数の `D1C`、`D2A`、`D2C`、および `D2E` の各オプションと同様に、`D1E` を指定した場合と同じように扱われます。`D1E` は 1 フェーズ・コミットですが、データベース接続を自動的に解放しません。データバ

ースを切断するには、DISC、DCURRENT、または DALL オプションを明示的に要求する必要があります。詳しくは、VGLib.connectionService のオンライン・ヘルプを参照してください。

- EZESQISL (EGL VGVar.sqlIsolationLevel)。VisualAge Generator では、値 1 はカーソル固定の要求を表します。EGL では、値 1 は順序付け可能トランザクションの要求を表します。
- EZESQRRM (EGL VGVar.sqlerrmc) はサポートされません。
- EZESQWN6 (EGL VGVar.sqlwarn[7]) はサポートされません。
- EZESQLCA (EGL sysVar.sqlca) フィールドには制限があります。これらのフィールドには、EZESQRRM と EZESQWN6 の値を含めることはできません。
- EZE 特殊データ・ワードに関する違いは、次のとおりです。
  - EZECONVT (EGL sysVar.callConversionTable)。VisualAge Generator の C++ 生成の場合、変換テーブル名のフォーマットは ELAxxyyy です。ここで、xx はシステムを示し、yyy は言語を示します。EGL の Java 生成の場合、EGL が提供する変換テーブル名のフォーマットは CSOBxxxx です。ここで、CSO は固定の接頭部、B はターゲット・システムのバイト・オーダーを示し、xxxx はターゲット・システムのコード・ページを示します。B の有効な値は、Unix システムの場合は X、Intel システムの場合は I、EBCDIC システムの場合は E です。EGL が ELA テーブル名を CSO テーブル名に自動的に変換するため、コードの変更は必要ありません。ただし、ユーザー独自の CSO 変換テーブルを作成する機能は EGL にはありません。
  - EZERCODE (EGL sysVar.returnValue)。VisualAge Generator の C++ 生成の場合、EZERCODE はシステムまたは呼び出し側プログラムに戻されます。プログラムが異常終了した場合は、EZERCODE には値でなく VAGen 戻りコードが戻されます。EGL の Java 生成の場合、EZERCODE は無視されます。



---

## 第 7 部 付録





---

## 付録 A. 予約語

---

### EGL 予約語

EGL には大量の予約語があります。予約語をパーツ名として使用することはできません。パーツ名が EGL 予約語である場合、マイグレーション・ツールは関数、データ項目、レコード、およびマップの名前を変更します。マイグレーション・ツールは、テーブル、マップ・グループ、またはプログラムの名前を変更しません。EGL 予約語は、次のとおりです。

表 63. EGL 予約語

文字	予約語
A	absolute, add, all, and, any, as
B	bigInt, bin, bind, blob, boolean, by, byName, byPosition
C	call, case, char, clob, close, const, continue, converse, current
D	dataItem, dataTable, date, dbChar, decimal, decrement, delete, display, dli, dlicall
E	else, embed, end, escape, execute, exit, extends, externallyDefined
F	false, field, first, float, for, forEach, form, formGroup, forUpdate, forward, freeSql, from, function
G	get, goto, group
H	handler, hex, hold
I	if, implements, import, in, inOut, inParent, insert, int, interface, interval, into, is, isa
L	label, languageBundle, last, library, like
M	matches, mbChar, money, move
N	new, next, nil, no, noRefresh, not, nullable, num, number, numc
O	of, onEvent, onException, open, openUI, or, otherwise, out
P	pacf, package, PageHandler, passing, prepare, previous, print, private, program
R	record, ref, relative, replace, return, returning, returns
S	scroll, self, service, set, show, singleRow, smallFloat, smallInt, sql, sqlCondition, stack, static, string
T	this, time, timeStamp, to, transaction, transfer, true, try, type
U	unicode, update, url, use, using, usingKeys, usingPCB
W	when, where, while, with, wrap
Y	yes

注: EGL パーツ名は、EZE、# シンボル、または @ シンボルで始めることはできません。

---

### EGL 列挙型ワード

VAGen には、それぞれのプロパティごとに有効な値の固定リストがあります。EGL は、対応するプロパティの有効な値を指定する列挙型リストを提供しています。例えば、EGL は align プロパティの有効な値を AlignKind という名前の EGL 列挙型に格納します。また、EGL の場合はプロパティ値に変数と式を利用

できます。EGL がプロパティ値を解決する際に、EGL はまずその名前の変数を検索します。その名前の変数がない場合、EGL はプロパティ値を検証するために、そのプロパティに対応する列挙型を検索します。例えば、書式の中で変数フィールドのプロパティが `align=center` である場合、EGL はまず書式の中で `center` という名前の変数を検索します。書式に `center` という名前の変数がない場合、EGL は `AlignKind.center` を使用します。表 64 に、EGL 列挙型の名前、列挙型の有効な値のリスト、および変数名と列挙型の値の衝突を引き起こす可能性があるプロパティを使用するパーツを示します。

また、EGL ライブラリーの定数と変数を使用して設定できるプロパティがいくつかあります。表 65 に、EGL ライブラリーの定数と変数を使用する EGL プロパティを参照用にリストします。ただし、マイグレーション・ツールが使用する EGL ライブラリーの変数と定数は、表の中でアスタリスク (\*) によって示されているもののみです。

コードを新しく作成する際に競合が起こらないように、次のいずれかを行うことができます。

- 名前の衝突を起こす可能性がある変数を定義するときに、有効な値のリストのいずれかと一致する変数名を使用しない。
- プロパティの列挙値を常に修飾する (例えば、常に `align = AlignKind.center` を指定する)。
- 競合が生じたときに、プロパティの列挙値を修飾する (例えば、特定の書式に `center` という名前のフィールドが存在するときに、`align = AlignKind.center` を指定する)。

マイグレーション・ツールは、予想される使用頻度、マイグレーション・ツールが実際に使用するプロパティ、およびマイグレーションのパフォーマンスに関する考慮事項に基づいて、これら 3 つの技法を組み合わせ使用します。パーツと変数の名前を変更する必要性を最小限にするために、マイグレーション・ツールは次の処理を行います。

- YES と NO は EGL 予約語なので、YES または NO という名前のパーツまたは変数の名前変更を常に行います。
- PFn という名前のパーツまたは変数の名前変更を常に行います (ここで、*n* は 1 から 24 までです)。実質上、マイグレーション・ツールはこれらの値を予約語として扱います。これにより、マイグレーションのパフォーマンスに影響せずに、プログラムと書式の `helpKey` プロパティと `validationBypassKeys` プロパティの見やすさが向上します。
- FormGroup 内で、マイグレーション・ツールは次の処理を行います。
  - `deviceType` プロパティ値を常に完全に修飾します。これにより、FormGroup 内のすべての書式にあるすべてのフィールド名を検討する必要がなくなるので、パフォーマンスが向上します。
  - `helpKey` プロパティまたは `validationBypassKeys` プロパティは使用しません。これは、マップ・グループ用の同等な `VAGen` プロパティが存在しないからです。
- 書式内では、書式内の 1 つ以上のフィールド名が、書式に対して使用できる列挙値、EGL 定数、または EGL 変数のいずれかと同じ名前である場合に限り、マイグレーション・ツールは書式のプロパティを修飾します。フォームのいずれ

かのフィールドに関して競合が起こった場合、マイグレーション・ツールは、対応する列挙型またはライブラリー名のフォーム内にあるすべてのフィールドを対象に、すべてのプロパティ値を完全に修飾します。ただし、フィールドに競合があるかどうかを判別する際に、マイグレーション・ツールは `IndexOrientationKind` の値 (`across` または `down`) を無視します。これは、マイグレーション・ツールがこのプロパティを使用することはないからです。

- **VGUI** レコード内では、マイグレーション・ツールは、レコードの 1 つ以上のフィールド名が、**VGUI** レコードで使用できる列挙値または **EGL** 変数のいずれかと同じ名前である場合にのみ、プロパティを修飾します。レコードのいずれかのフィールドに関して競合が起こった場合、マイグレーション・ツールは、対応する列挙型またはライブラリー名を使用して、そのレコード内のすべてのフィールドについてすべてのプロパティ値を完全に修飾します。
- **PSB** レコード内で、マイグレーション・ツールは常に `pcbType` プロパティを完全修飾します。
- プログラム内で、マイグレーション・ツールは常に `callInterface` プロパティを完全修飾します。
- 他のパーツ型の中では、パーツ名と列挙値が競合する可能性はないので、プロパティ値の修飾を行いません。

注: 241 ページの表 64 と 242 ページの表 65 は参照用です。このため、マイグレーション・ツールによって作成されることのない `Library`、`ConsoleUI`、`PageHandler`、`Service`、および `Interface part` パーツがテーブルにインクルードされます。

表 64. EGL 列挙型

EGL 列挙型	変数名の制約となる有効な値のリスト	衝突を起こす可能性があるパーツ
<code>AlignKind</code>	<code>center</code> 、 <code>left</code> 、 <code>none</code> 、 <code>right</code>	書式
<code>CallingConventionKind</code>	<code>I4GL</code>	ライブラリー
<code>CaseFormatKind</code>	<code>defaultCase</code> 、 <code>lower</code> 、 <code>upper</code>	<code>ConsoleUI</code>
<code>ColorKind</code>	<code>black</code> 、 <code>blue</code> 、 <code>cyan</code> 、 <code>defaultColor</code> 、 <code>green</code> 、 <code>magenta</code> 、 <code>red</code> 、 <code>yellow</code> 、 <code>white</code>  注: <code>black</code> は <code>ConsoleUI</code> の場合のみ有効です。	書式 <code>ConsoleUI</code>
<code>CommTypeKind</code>	<code>LOCAL</code> 、 <code>TCPIP</code>	<code>Service Interface</code>
<code>DataSource</code>	<code>databaseConnection</code> 、 <code>reportData</code> 、 <code>sqlStatement</code>	<code>Report</code>
<code>DeviceTypeKind</code>	<code>doubleByte</code> 、 <code>singleByte</code>	<code>FormGroup</code>
<code>DisplayUseKind</code>	<code>button</code> 、 <code>hyperlink</code> 、 <code>input</code> 、 <code>output</code> 、 <code>secret</code> 、 <code>table</code>	<code>PageHandler</code> とともに使用されるレコード
<code>DLICallInterfaceKind</code>	<code>AIBTDLI</code> 、 <code>CBLTDLI</code>	プログラム
<code>EventKind</code>	<code>afterDelete</code> 、 <code>afterField</code> 、 <code>afterInsert</code> 、 <code>afterOpenUI</code> 、 <code>afterRow</code> 、 <code>beforeDelete</code> 、 <code>beforeField</code> 、 <code>beforeInsert</code> 、 <code>beforeOpenUI</code> 、 <code>beforeRow</code> 、 <code>menuAction</code> 、 <code>onKey</code>	<code>ConsoleUI</code>

表 64. EGL 列挙型 (続き)

EGL 列挙型	変数名の制約となる有効な値のリスト	衝突を起こす可能性があるパーツ
ExportFormat	html、pdf、text	ConsoleUI
HighlightKind	blink、defaultHighlight、noHighlight、reverse、underline	書式 ConsoleUI
IndexOrientationKind	across、down	書式
IntensityKind	bold、defaultHighlight、dim、invisible、normalIntensity	書式 ConsoleUI
LineWrapKind	character、compressed、word	ConsoleUI
OutlineKind	bottom、left、right、top 注: sysLib.box は、[left,right,top,bottom] と等価な定数です。sysLib.noOutline は、アウトラインがないことを示す定数です。	書式
PCBKind	TP、DB、GSAM	PSB レコード
PfKeyKind	pfn ここで (1 <= n <=24)	書式 FormGroup プログラム
ProtectKind	skip、no、yes	書式
SelectTypeKind	index、value	PageHandler とともに使用されるレコード
SignKind	leading、none、parens、trailing	書式 VGUI レコード PageHandler とともに使用されるレコード
UITypeKind	hidden、input、inputOutput、none、output、programLink、submit、submitBypass、uiForm	VGUI レコード

表 65. sysLib 定数および sysVar 変数を使用する EGL 列挙型

EGL プロパティ	EGL 定数および変数 (* = マイグレーション・ツールによって使用される値)	衝突を起こす可能性があるパーツ
dateFormat	strLib.defaultDateFormat* strLib.eurDateFormat strLib.isoDateFormat strLib.jisDateFormat strLib.usaDateFormat VGVar.systemGregorianDateFormat* VGVar.systemJulianDateFormat* 注: マイグレーション・ツールは、DataItem および VGUI レコード・パーツでは defaultDateFormat のみを使用します。	ConsoleUI フォーム、 VGUI レコード
fillCharacter	strLib.nullFill*	書式
outline	sysLib.box* sysLib.noOutline 注: outlineKind は、他のアウトライン値の列挙型です。	書式

表 65. sysLib 定数および sysVar 変数を使用する EGL 列挙型 (続き)

EGL プロパティ	EGL 定数および変数 (* = マイグレーション・ツールによって使用される値)	衝突を起こす可能性があるパーツ
timeFormat	strLib.defaultTimeFormat strLib.eurTimeFormat strLib.isoTimeFormat strLib.jisTimeFormat strLib.usaTimeFormat	PageHandler とともに使用されるレコード
timeStampFormat	strLib.db2TimeStampFormat strLib.odbcTimeStampFormat	ConsoleUI

## SQL 予約語

SQL 文節の中で使用できない SQL 予約語が多数存在します。パーツ名が SQL 予約語である場合、マイグレーション・ツールは関数、データ項目、レコード、およびマップの名前を変更します。マイグレーション・ツールは、テーブル、マップ・グループ、またはプログラムの名前を変更しません。SQL 予約語は、次のとおりです。

文字	予約語
A	absolute、action、add、alias、all、allocate、alter、and、any、are、as、asc、assertion、at、authorization、avg
B	begin、between、bigint、binaryLargeObject、bit、bit_length、blob、boolean、both、by
C	call、cascade、cascaded、case、cast、catalog、char、char_length、character、character_length、characterLargeObject、characterVarying、charLargeObject、charVarying、check、clob、close、coalesce、collate、collation、column、comment、commit、connect、connection、constraint、constraints、continue、convert、copy、corresponding、count、create、cross、current、current_date、current_time、current_timestamp、current_user、cursor
D	data、database、date、dateTime、day、deallocate、dec、decimal、declare、default、deferrable、deferred、delete、desc、describe、diagnostics、disconnect、distinct、domain、double、doublePrecision、drop
E	else、end、endExec、escape、except、exception、exec、execute、exists、explain、external、extract
F	false、fetch、first、float、for、foreign、found、from、full
G	get、getCurrentConnection、global、go、goto、grant、group
H	having、hour
I	identity、image、immediate、in、index、indicator、initially、inner、input、insensitive、insert、int、integer、intersect、into、is、isolation
J	join
K	key
L	language、last、leading、left、level、like、local、long、longint、lower、ltrim
M	match、max、min、minute、module、month
N	national、nationalCharacter、nationalCharacterLargeObject、nationalCharacterVarying、nationalCharLargeObject、nationalCharVarying、natural、nchar、ncharVarying、nclob、next、no、not、null、nullIf、number、numeric
O	octet_length、of、on、only、open、option、or、order、outer、output、overlaps
P	pad、partial、position、prepare、preserve、primary、prior、privileges、procedure、public
R	raw、read、real、references、relative、restrict、revoke、right、rollback、rows、rtrim、runtimeStatistics

文字	予約語
S	schema、scroll、second、section、select、session、session_user、set、signal、size、smallint、some、space、sql、sqlcode、sqlerror、sqlstate、substr、substring、sum、system_user
T	table、tablespace、temporary、terminate、then、time、timestamp、timezone_hour、timezone_minute、tinyint、to、trailing、transaction、translate、translation、trim、true
U	uncatalog、union、unique、unknown、update、upper、usage、user、using
V	values、varbinary、varchar、varchar2、varying、view
W	when、whenever、where、with、work、write
Y	year
Z	zone

## 特殊な処理を必要とする SQL 予約語

次の SQL 予約語を SQL 表名または列名として使用する場合、EGL では特殊な処理が必要です。

call、from、group、having、insert、order、select、set、union、update、values、where

これらの SQL 予約語を使用するには、次の手順で行います。

- sqlRecord 内の項目の column プロパティを指定するには、次のように指定します。

```
column = "¥"reservedWord¥"
```

例えば、次のようになります。

```
column = "¥"FROM¥"
```

- sqlRecord の tableNames プロパティを指定するには、次のように指定します。

```
tableNames = [{"¥"reservedWord2¥"}]
```

例えば、次のようになります。

```
tableNames = [{"¥"ORDER¥"}]
```

- レコードの defaultSelectCondition の中で SQL 列名として予約語のいずれかを使用するには、次のように指定します。

```
defaultSelectCondition = #sqlCondition{ "reservedWord" = ... }
```

例えば、次のようになります。

```
defaultSelectCondition = #sqlCondition{ "ORDER" = ... }
```

- SQL 入出力ステートメントの中で SQL 列名として予約語のいずれかを使用するには、次のように指定します。

```
... #sql{ select "reservedWord" from "reservedWord2" .... } ...
```

例えば、次のようになります。

```
... #sql{ select "FROM" from "ORDER" .... } ...
```



---

## Java 予約語

Java には、パッケージ名として使用できない予約語があります。Java の生成を行う場合は、次の名前の使用は避けてください。

文字	予約語
A	abstract
B	boolean、break、byte
C	case、catch、char、class、const、continue
D	default、do、double
E	else、extends
F	false、final、finally、float、for
G	goto
I	if、implements、import、instanceof、int、interface
L	long
N	native、new、null
P	package、private、protected、public
R	return
S	short、static、super、switch、synchronized
T	this、throw、throws、transient、true
V	void、volatile
W	while



---

## 付録 B. VisualAge Generator と EGL の言語エレメントの関係

この付録の表には、次の 3 つの欄があります。

- VisualAge Generator 4.5 -- この欄は、VAGen 言語エレメントを示しています。パーツ型に関連した部分では、表の編成と使用される用語は VAGen ユーザー・インターフェースに対応しています。ステートメント、EZE ワード、およびサービス・ルーチンの表は、ステートメント、EZE ワード、またはサービス・ルーチンのタイプ別に編成されています。
- マイグレーション・ツールによって作成される EGL -- この欄は、EGL 言語エレメントを示しています。この欄はマイグレーションに必要な情報のみを示しており、完全な EGL 構文を示すことは目的としていません。一部の EGL 言語エレメントに対しては、他のプロパティ、値、およびオプションを使用できる場合があります。例えば、EGL set ステートメントには、VisualAge Generator では使用できない追加のオプションがあります。この付録の表にリストされている set ステートメントのオプションは、VAGen 言語エレメントに対応するもののみです。この欄は、EGL 資料にある詳細情報を見付けるためのガイドとして使用してください。
- マイグレーション・ツールに関する考慮事項 -- この欄には、マイグレーション・ツールが VisualAge Generator から EGL への変換を処理する方法に関する追加情報があります。また、必要に応じて、関連パーツを使用したマイグレーション、関連パーツを使用しないマイグレーション、および VAGen 言語エレメントのマイグレーション時に起こりうる問題について詳しく説明する、未確定状態に関するセクションの参照先も示しています。

それぞれのパーツ型ごとに、セクションの最初にある表の 1 行目に次の情報があります。

- VisualAge Generator 4.5 - VisualAge Generator のパーツ型に対して、さまざまなウィンドウで指定できる情報の概要。
- EGL -- 対応する EGL パーツ型の EGL 構文の全体 (マイグレーション・ツールが使用する構文)。これとは異なる構文を使用できる場合があります。例えば、VAGen テーブルをマイグレーションする際に、マイグレーション・ツールは dataTable の内容を dataTable 構造の後に常に配置するので、『テーブル』のセクションにはこの構文が示されています。EGL 構文上は、dataTable の内容を dataTable 構造の前に配置することもできます。

表の中では、次の構文が使用されます。

- | - いくつかのオプションからの選択。選択の順序は、VisualAge Generator 4.5 と EGL の両方の欄で同じです。
- 黒丸付きリスト - オプションまたは値の長いリストからの選択。選択の順序は、VisualAge Generator 4.5 と EGL の両方の欄で同じです。
- 斜体 - VisualAge Generator からマイグレーションするときにマイグレーション・ツールが入力する値、または EGL ステートメントを新規に作成するときにユーザーが入力する値。
- 太字 - 示されたとおりに指定する必要がある EGL キーワードと記号。

- { } - 0 回から n 回繰り返すことができる情報を囲みます。
- { } - EGL プロパティー・リストを囲みます。プロパティーは常にコンマで区切られます。
- [ ] - オプションの情報を囲みます。
- [ ] - EGL の値のリストを囲みます。値は常にコンマで区切られます。

## 一般的な構文規則

VisualAge Generator と EGL の全体的な構文には、いくつかの違いがあります。

表 66. 一般的な構文規則

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>コメントは次のフォーマットで指定されます。</p> <ul style="list-style-type: none"> <li>• プログラム、テーブル、およびレコードの場合は、プロローグ。</li> <li>• 項目および関数の場合は、記述。</li> <li>• 関数内のコメントは、次の記号によって示されます。 <ul style="list-style-type: none"> <li>– セミコロン (;)。同じ行でセミコロンの後にあるものは、すべてコメントとして扱われます。</li> <li>– /*. 同じ行で /* の後にあるものは、すべてコメントとして扱われます。</li> </ul> </li> </ul>	<p>コメントは次のフォーマットで指定されます。</p> <ul style="list-style-type: none"> <li>• // は行コメントを示します。同じ行で // の後にあるものは、すべてコメントとして扱われます。コメントは 1 行のみです。</li> <li>• /* コメント */。/* の後にあるものは、次の /* まですべてコメントとして扱われます。複数行にわたるコメントを入力できます。</li> </ul>	<p>マイグレーション・ツールは、次のように変換を行います。</p> <ul style="list-style-type: none"> <li>• プロローグとパーツ記述は、EGL の // 行コメントに変換されます。</li> <li>• レコード、テーブル、マップ、関数ローカル・ストレージ、関数仮パラメーター、または関数からの戻り値などのフィールドとして使用される項目の記述は、EGL // 行コメントに変換されます。</li> <li>• 関数内のコメントは、/* コメント */に変換されます。</li> <li>• マイグレーション・ツールによって追加された通知コメントは EGL // 行コメントの形式になります。</li> </ul>
<p>小数点は、ロケールに応じてピリオドまたはコンマのどちらかを使用できます。</p>	<p>開発時の小数点は、常にピリオドです。生成時および実行時には、実行時のロケールと decimalSymbol ビルド記述子オプションに応じてピリオドまたはコンマのどちらかが使用されます。</p>	<p>デフォルトで小数点にコンマを使用するロケールの場合、またはマイグレーション構文の設定「小数点位のコンマを小数点に変換 (Convert decimal comma to decimal point)」を選択した場合は、マイグレーション時にマイグレーション・ツールがコンマをピリオドに変換します。</p>
<p>プロパティーは、専用のエディターで、チェック・ボックス、ドロップダウン・リストなどを使用して入力します。</p>	<p>プロパティーはテキスト・エディターで入力し、コンマで区切る必要があります。</p>	<p>特別な考慮事項なし。</p>
<p>値のリストは、専用のエディターで入力します。例えば、SQL レコードの表名は「SQL 行プロパティー」ウィンドウに入力します。</p>	<p>値のリストは大括弧 [ ] で囲む必要があります。</p>	<p>特別な考慮事項なし。</p>

表 66. 一般的な構文規則 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
他のパーツを参照するプロパティ値は、専用のエディターで入力します。例えば、マップ変数フィールドの編集ルーチンは、「変数フィールドのプロパティ (Variable Field Properties)」ウィンドウの「編集 (Edits)」ページに入力します。	他のパーツを参照するプロパティ値は、二重引用符で囲む必要があります。	特別な考慮事項なし。

## データ項目

データ項目に関するセクションは、次の表で構成されています。

- データ項目 - 一般構文、データ型、長さ、小数部、および説明 (249 ページの表 67)
- デフォルトのマップ・プロパティおよびユーザー・インターフェース・プロパティ - 一般情報 (252 ページの表 68)
- デフォルトのマップ・プロパティおよびユーザー・インターフェース・プロパティ - 一般編集 (252 ページの表 69)
- デフォルトのマップ・プロパティおよびユーザー・インターフェース・プロパティ - 数値の編集 (255 ページの表 70)
- デフォルトのマップ・プロパティおよびユーザー・インターフェース・プロパティ - エラー・メッセージ (255 ページの表 71)
- ユーザー・インターフェース・プロパティ - ラベルとヘルプ (256 ページの表 72)

注: EGL の場合、`dataItem` パーツの編集とメッセージのプロパティは、1 セットだけ存在します。マイグレーション・ツールは データ項目のマップ・プロパティと UI プロパティをマージします。

表 67. データ項目 — 一般構文、データ型、長さ、小数部、および説明

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>VAGen データ項目パーツ:</p> <ul style="list-style-type: none"> <li>itemName</li> <li>基本情報: <ul style="list-style-type: none"> <li>データ型</li> <li>長さ</li> <li>小数部</li> <li>説明</li> </ul> </li> <li>デフォルトのマップ・プロパティ</li> <li>ユーザー・インターフェース (UI) プロパティ</li> </ul>	<p>EGL 構文の例:</p> <pre>// Description DataItem itemName   dataType(lengthInformation)   { [{formattingProperties}]     [{validationProperties}]     [{pageHandlerFieldProperties}]   } end</pre>	<p>マイグレーション・ツールは、VAGen のデータ型、長さ、および小数部を使用して、EGL の <code>dataType</code> と <code>lengthInformation</code> を決定します。</p> <p>マイグレーション・ツールは、VAGen のデフォルトのマップ・プロパティと UI プロパティをマージして、単一の EGL フォーマット設定プロパティ、検証プロパティ、および <code>pageHandlerField</code> プロパティのセットにします。</p>

表 67. データ項目 — 一般構文、データ型、長さ、小数部、および説明 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>文字項目の型:</p> <ul style="list-style-type: none"> <li>• Char</li> <li>• Hex</li> <li>• DBCS</li> <li>• Mixed</li> <li>• Unicode (VisualAge for Java のみ)</li> </ul> <p>長さは文字数です。レコード・エディター内ではバイト数も表示できます。</p>	<p>対応する文字項目の型:</p> <ul style="list-style-type: none"> <li>• char</li> <li>• hex</li> <li>• dbchar</li> <li>• mbchar</li> <li>• unicode</li> </ul> <p>長さは文字数です。</p>	<p>マイグレーション・ツールは、文字データ項目を対応する型と長さに変換します。</p>
<p>数字 (ゾーン 10 進数) の型:</p> <ul style="list-style-type: none"> <li>• Num</li> <li>• Numc</li> </ul> <p>長さは総桁数で、最大は 18 です。小数部は、小数点の右側の桁数です。レコード・エディター内ではバイト数も表示できます。</p>	<p>対応する数値型:</p> <ul style="list-style-type: none"> <li>• num</li> <li>• numc</li> </ul> <p>精度は総桁数です。スケールは、小数点の右側の桁数です。</p> <p>num フィールドの最大精度は、デバッグおよび Java 生成の場合は 32、COBOL 生成の場合は 31 です。numc フィールドの最大精度は 18 です。</p>	<p>マイグレーション・ツールは、対応する型、精度、およびスケールへの変換を行います。小数部が 0 の場合、マイグレーション・ツールはスケールを省略します。</p>



表 67. データ項目 — 一般構文、データ型、長さ、小数部、および説明 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>パック 10 進数の型:</p> <ul style="list-style-type: none"> <li>• Pacf</li> <li>• Pack</li> </ul> <p>長さは総桁数で、最大は 18 です。小数部は、小数点の右側の桁数です。Pacf の長さは、奇数または 18 であることが必要です。Pack の長さは、偶数でも奇数でも構いません。長さ 18 の場合を除き、偶数の長さはデータ項目定義に記録されますが、テストおよび生成時、およびデータ項目エディターとレコード・エディター内では、次に大きな奇数の長さとして扱われます。SQL レコード・エディターのみが偶数の長さを表示し、SQL レコードのみがテストと生成の際に偶数の長さをサポートします。偶数の長さは、SQL <i>where</i> 文節と、Execution Time Statement Build を使用する SQL 関数内でのみ使用されます。レコード・エディター内ではバイト数も表示できます。</p>	<p>対応する数値型:</p> <ul style="list-style-type: none"> <li>• pacf</li> <li>• decimal</li> </ul> <p>精度は総桁数です。スケールは、小数点の右側の桁数です。</p> <p>pacf フィールドの最大精度は 18 です。decimal フィールドの最大精度は、デバッグおよび Java 生成の場合は 32、COBOL 生成の場合は 31 です。pacf の長さは、奇数または 18 であることが必要です。</p> <p>decimal の長さは、偶数でも奇数でも構いません。dataItem パーツ定義とすべてのレコード・タイプに対して、偶数の長さがサポートされます。</p> <p>テストおよび生成時に、VisualAge Generator 互換モードを使用している場合、EGL は偶数の精度をもつ 10 進項目に対して次の処理を行います。</p> <ul style="list-style-type: none"> <li>• すべてのレコードの精度を 1 だけ増やします。</li> <li>• EGL は、SQL <i>where</i> 文節または <i>prepare</i> ステートメント内で偶数の精度の一時変数を使用します。</li> </ul>	<p>マイグレーション・ツールは、対応する型、精度、およびスケールへの変換を行います。小数部が 0 の場合、マイグレーション・ツールはスケールを省略します。Pack 項目の場合、dataItem パーツ定義に偶数の長さが記録されていれば、マイグレーション・ツールは、デフォルトでその長さを偶数の長さとしてマイグレーションします。</p> <p>VAGen マイグレーション設定「項目または変数の <i>evensql=y</i> を受け入れない」を指定すると、マイグレーション・ツールは Pack 項目用の奇数精度 (項目が最大長である場合は 18) を自動的に使用し、影響のあるデータ項目パーツや非共用レコード項目に関する警告メッセージを出します。</p>
<p>バイナリー項目の型:</p> <ul style="list-style-type: none"> <li>• Bin、長さ 4、小数部なし</li> <li>• Bin、長さ 9、小数部なし</li> <li>• Bin、長さ 18、小数部なし</li> <li>• Bin、長さ 4、9、または 18、小数部あり</li> </ul>	<p>対応するバイナリー形式:</p> <ul style="list-style-type: none"> <li>• smallint (精度またはスケールなし)</li> <li>• int (精度またはスケールなし)</li> <li>• bigint (精度またはスケールなし)</li> <li>• bin (精度またはスケールあり)</li> </ul>	<p>マイグレーション・ツールは、小数部の長さと数に基づいて、バイナリー・データ項目を対応する型に変換します。bin 型は、小数部 (スケール) が指定されている場合のみ使用されます。</p>
説明	適用外。	マイグレーション・ツールは、項目の説明をコメントに変換し、dataItem 定義の前に配置します。

表 68. デフォルトのマップ・プロパティおよびユーザー・インターフェース・プロパティ - 一般情報

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>データ項目には、デフォルト・マップ・プロパティとユーザー・インターフェース (UI) プロパティの両方を指定できます。プロパティには次のものがあります。</p> <ul style="list-style-type: none"> <li>フォーマット設定編集</li> <li>検証編集</li> <li>エラー・メッセージ</li> </ul> <p>UI プロパティには、ラベルとヘルプ・テキストも含まれます。</p> <p>VisualAge Generator 内で一部のプロパティを明示的に設定すると、他のプロパティも設定されます。例えば、数字分離記号を設定すると、充てん文字、入力必須 (input required)、位置調整、通貨記号、および符号も明示的に設定されます。</p>	<p>dataItem パーツには次のプロパティを設定できます。</p> <ul style="list-style-type: none"> <li>フォーマット設定プロパティ</li> <li>検証プロパティ</li> <li>PageHandler フィールド・プロパティ</li> </ul> <p>一部のプロパティのカテゴリが、VisualAge Generator から変更されています。例えば、エラー・メッセージは検証プロパティと一緒にグループにあります。PageHandler フィールド・プロパティには、UI ラベルとヘルプ・テキストがインクルードされています。以下の表の EGL の列は、EGL プロパティのカテゴリを示します。</p>	<p>マイグレーション・ツールは、デフォルトのマップ・プロパティと UI プロパティをマージし、UI プロパティを優先します。検証編集とそれに関連したエラー・メッセージは、対してマイグレーションされます。マイグレーション・ツールは、VisualAge Generator 内で明示的に設定されたプロパティのみをマイグレーションします。EGL プロパティのデフォルト値は、ツールによって自動的に挿入されません。詳細と起こりうる問題については、61 ページの『共用編集とメッセージ』にある、マップと UI 編集のマージに関する情報を参照してください。</p> <p>また、詳細と起こりうる問題については、63 ページの『共用データ項目のマップ編集ルーチン』にある、共用データ項目のマップ項目編集に関する情報も参照してください。</p>

表 69. デフォルトのマップ・プロパティおよびユーザー・インターフェース・プロパティ - 一般編集

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>編集タイプ (UI のみ) - 値</p> <ul style="list-style-type: none"> <li>なし</li> <li>ブール</li> <li>日付</li> <li>時刻</li> </ul>	<p>EGL は、複数のプロパティをサポートします。</p> <ul style="list-style-type: none"> <li>適用外</li> <li>isBoolean = yes</li> <li>dateFormat = defaultDateFormat</li> <li>timeFormat = "HH:mm:ss"</li> </ul> <p>(フォーマット設定プロパティ)</p>	特別な考慮事項なし。
編集関数 (UI のみ)	validatorFunction (検証プロパティ)	特別な考慮事項なし。
編集テーブル (UI のみ)	validatorDataTable (検証プロパティ)	特別な考慮事項なし。
Web 上での編集関数の実行 (UI のみ)	runValidatorFromProgram	EGL プロパティは、VAGen プロパティの逆です。マイグレーション・ツールは、yes を no に、no を yes に変換します。

表 69. デフォルトのマッピング・プロパティおよびユーザー・インターフェース・プロパティ - 一般編集 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
編集ルーチン (マップのみ)	validatorFunction または validatorDataTable  (検証プロパティ)	<p>UI 編集関数と編集テーブルが指定されていなければ、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• マップ編集ルーチンが EZEC10 または EZEC11 ならば、validatorFunction プロパティを設定します。</li> <li>• 編集ルーチンが関数ならば、validatorFunction プロパティを設定します。</li> <li>• 編集ルーチンがテーブルならば、validatorDataTable プロパティを設定します。</li> </ul> <p>UI 編集関数または編集テーブルが指定されている場合、マイグレーション・ツールはマップ編集ルーチンをマイグレーションしません。</p> <p>マイグレーション時に編集ルーチンが使用できない場合は、特別な考慮事項が適用されます。詳細と起こりうる問題については、63 ページの『共用データ項目のマップ編集ルーチン』にある、マップ編集ルーチンに関する情報を参照してください。</p>
<p>位置調整 - 左そろえ   右そろえ   なし (マップのみ)</p> <p>注:</p> <ul style="list-style-type: none"> <li>• マップ項目のデフォルトは、数値フィールドの場合は右そろえ、その他のフィールドの場合は左そろえです。</li> <li>• UI 項目の場合、位置調整はサポートされません。</li> </ul>	<p>align = left   right   none</p> <p>(フォーマット設定プロパティ)</p> <p>注:</p> <ul style="list-style-type: none"> <li>• 書式フィールドのデフォルトは、数値フィールドの場合は右そろえ、その他のフィールドの場合は左そろえです。</li> <li>• PageHandler フィールドと VGUI レコード・フィールドについては、位置合わせはサポートされていません。</li> </ul>	特別な考慮事項なし。

表 69. デフォルトのマッピング・プロパティおよびユーザー・インターフェース・プロパティ - 一般編集 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>日付編集マスク (マップのみ)</p> <p>有効な値は以下のとおりです。</p> <ul style="list-style-type: none"> <li>• SYSGREGRN</li> <li>• SYSJULIAN</li> <li>• dateEditPattern</li> </ul>	<p>dateFormat = value</p> <p>有効な値は以下のとおりです。</p> <ul style="list-style-type: none"> <li>• systemGregorianCalendar</li> <li>• systemJulianCalendar</li> <li>• "dateEditPattern"</li> </ul> <p>(フォーマット設定プロパティ)</p> <p>注: dateEditPattern の中で、マイグレーション・ツールは次の EGL 表記への変換を行います。</p> <ul style="list-style-type: none"> <li>• 年を示す yy または yyyy。</li> <li>• 月を示す MM。</li> <li>• 日を示す dd。</li> <li>• 年の通算日を示す DDD。</li> </ul>	<p>UI 編集タイプが「日付」を指定していない場合、マイグレーション・ツールは VisualAge Generator 内で指定されている日付編集マスク (存在する場合) に基づいて dateFormat を設定します。UI 編集タイプが「日付」を指定している場合、マイグレーション・ツールはマップの日付編集マスクをマイグレーションしません。</p>
最小入力	minimumInput (検証プロパティ)	特別な考慮事項なし。
<p>充てん文字</p> <p>注:</p> <ul style="list-style-type: none"> <li>• UI レコード内で使用される項目のデフォルト充てん文字は、文字、混合、および数値の各フィールドの場合はブランクです。16 進フィールドの場合、デフォルト充てん文字はゼロです。DBCS および Unicode のフィールドの場合は、ブランクが必須の充てん文字です。NULL は無効な充てん文字です。</li> <li>• 文字、DBCS、または混合の各フィールドの場合、マップに使用される項目のデフォルト充てん文字は NULL です。デフォルト充てん文字は、数値フィールドの場合はブランク、16 進フィールドの場合はゼロです。</li> </ul>	<p>fillCharacter (フォーマット設定プロパティ)</p> <p>注:</p> <ul style="list-style-type: none"> <li>• 特定のページ、書式、または VGUI レコード内でオーバーライドされない限り、PageHandler フィールド、フォーム・フィールド、VGUI レコード・フィールドには同じデフォルト fillCharacter が使用されます。</li> <li>• strLib.nullFill は、NULL 充てん文字を表す EGL 定数です。あるいは、"" (2 つの連続する二重引用符) を使用します。</li> <li>• VGUI レコードでは、非ブランク文字が Unicode フィールド用に許可されています。</li> </ul>	<p>マイグレーション・ツールは、N を以下に変換します。</p> <ul style="list-style-type: none"> <li>• UI 充てん文字用の N</li> <li>• マップ・フィールド文字用の nullFill</li> </ul> <p>EGL にはデフォルトの充てん文字が 1 つしかないため、特殊な考慮事項が適用されます。詳細と起こりうる問題については、64 ページの『共用データ項目の充てん文字』の未確定データ項目と充てん文字に関する情報を参照してください。</p>
大文字変換	upperCase (フォーマット設定プロパティ)	特別な考慮事項なし。
16 進編集 (マップのみ)	isHexDigit (検証プロパティ)	特別な考慮事項なし。
入力必須 (Input required)	inputRequired (検証プロパティ)	特別な考慮事項なし。
SO/SI スペースの検査	needsSOSI (検証プロパティ)	特別な考慮事項なし。

表 70. デフォルトのマップ・プロパティおよびユーザー・インターフェース・プロパティ - 数値の編集

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
最小値と最大値 注: 最小値または最大値のどちらかを指定する場合は、両方を指定する必要があります。	<code>validValues = [[<i>minimumValue</i>, <i>maximumValue</i>]]</code> (検証プロパティ) 注: 複数の値の対、および単一値を <code>validValues</code> プロパティにリストできます。	マイグレーション・ツールは、最小値と最大値を EGL の <code>validValues</code> プロパティに結合します。
符号 - なし   先頭   末尾 注: デフォルト値は「なし」です。	<code>sign = none   leading   trailing</code> (フォーマット設定プロパティ) 注: デフォルト値は「なし」です。	特別な考慮事項なし。
通貨 (マップと UI 両方) 通貨記号 (UI のみ)	<code>currency = yes   no   currencySymbol = "symbol"</code> (フォーマット設定プロパティ) 注: <ul style="list-style-type: none"> <li><code>currencySymbol</code> は書式にも適用されます。</li> <li><code>currency = yes</code> で、ただし <code>currencySymbol</code> が指定されていない場合、実行時に実際に使用される通貨記号は、VisualAge Generator の場合と同じように設定されます。</li> </ul>	マイグレーション・ツールは、次のうち最初に該当するマイグレーションを行います。 <ul style="list-style-type: none"> <li>UI の通貨記号が指定されている場合、ツールは <code>currency = yes</code>, <code>currencySymbol = "symbol"</code> にマイグレーションします。</li> <li>UI の通貨編集が <code>yes</code> または <code>no</code> に設定されている場合、ツールは <code>currency</code> プロパティをそれぞれ <code>yes</code> または <code>no</code> に設定します。</li> <li>マップの通貨編集が <code>yes</code> または <code>no</code> に設定されている場合、ツールは <code>currency</code> プロパティをそれぞれ <code>yes</code> または <code>no</code> に設定します。</li> </ul>
セパレーター	<code>numericSeparator</code> (フォーマット設定プロパティ)	特別な考慮事項なし。
ゼロ編集	<code>zeroFormat</code> (フォーマット設定プロパティ)	特別な考慮事項なし。

表 71. デフォルトのマップ・プロパティおよびユーザー・インターフェース・プロパティ - エラー・メッセージ

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
編集テーブル (UI のみ)	<code>validatorDataTableMsgKey</code> (検証プロパティ)	特別な考慮事項なし。
EZE 関数 (UI のみ)	<code>validatorFunctionMsgKey</code> (検証プロパティ)	特別な考慮事項なし。

表 71. デフォルトのマップ・プロパティおよびユーザー・インターフェース・プロパティ - エラー・メッセージ (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
編集ルーチン (マップのみ)	validatorDataTableMsgKey または validatorFunctionMsgKey (検証プロパティ)	<p>特別な考慮事項が適用されます。マイグレーション・ツールがマップ編集ルーチン・メッセージをマイグレーションするかどうかを決定する方法については、61 ページの『共用編集とメッセージ』を参照してください。マイグレーション・ツールがマップ編集ルーチン・メッセージをマイグレーションする場合、ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>編集ルーチンが EZEC10 または EZEC11 ならば、 validatorFunctionMsgKey を設定します。</li> <li>編集ルーチンがテーブルならば、 validatorDataTableMsgKey を設定します。</li> <li>編集ルーチンが関数ならば、この状態では VisualAge Generator 内でメッセージは使用されないの、編集ルーチン・メッセージをマイグレーションしません。</li> </ul> <p>特別な考慮事項が適用されます。詳細と起こりうる問題については、63 ページの『共用データ項目のマップ編集ルーチン』にある、未確定データ項目とマップ編集ルーチンに関する情報を参照してください。</p>
最小入力	minimumInputMsgKey (検証プロパティ)	特別な考慮事項なし。
入力必須 (Input required)	inputRequiredMsgKey (検証プロパティ)	特別な考慮事項なし。
データ型	typeChkMsgKey (検証プロパティ)	特別な考慮事項なし。
数値範囲	validValuesMsgKey (検証プロパティ)	特別な考慮事項なし。

表 72. ユーザー・インターフェース・プロパティ - ラベルとヘルプ

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
UI ラベル	displayName (PageHandler フィールド・プロパティ)	特別な考慮事項なし。
ヘルプ・テキスト	ヘルプ (PageHandler フィールド・プロパティ)	特別な考慮事項なし。



## レコード

レコードに関するセクションは、次の表で構成されています。

- レコード - 一般構文、レコード・タイプ、プロパティ、およびプロローグ (257 ページの表 73)
- レコード - ほとんどのレコード・タイプのレコード構造 (259 ページの表 74)
- レコード - SQL プロパティと SQL レコード構造 (261 ページの表 75)
- レコード - DL/I プロパティと DL/I レコード構造 (265 ページの表 76)
- レコード - UI プロパティと UI レコード構造 (269 ページの表 77)
- レコード - UI 項目プロパティ - 一般 (270 ページの表 78)
- レコード - UI 項目プロパティ - 編集 (270 ページの表 79)
- レコード - UI 項目プロパティ - エラー・メッセージ (271 ページの表 80)
- レコード - UI 項目プロパティ - ヘルプ (272 ページの表 81)
- レコード - UI 項目プロパティ - 実行依頼 (272 ページの表 82)
- レコード - UI 項目プロパティ - プログラム・リンク (273 ページの表 83)

注: マイグレーション・ツールは、VAGen レコードを常に EGL 固定レコード・パーツに変換します。

表 73. レコード - 一般構文、レコード・タイプ、プロパティ、およびプロローグ

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>VAGen レコード・タイプ:</p> <ul style="list-style-type: none"> <li>• recordName</li> <li>• 基本情報 <ul style="list-style-type: none"> <li>– レコード・タイプ</li> <li>– レコード構造 (項目リスト)</li> </ul> </li> <li>• プロパティ (レコード・タイプによって異なる)</li> <li>• プロローグ</li> </ul> <p>注: 代替仕様レコードを指定するか、項目リストを組み込むことによって、レコード構造を指定できます。</p>	<p>EGL レコードの例:</p> <pre> **** Record=recordName**** // prolog //***** Record recordName type recordType { [ recordProperties ] } recordStructure end // end recordName </pre> <p>注: embed ステートメントを指定するか、項目リストを組み込むことによって、レコード構造を指定できます。</p>	<p>特別な考慮事項なし。</p>
<p>レコード・タイプ:</p> <ul style="list-style-type: none"> <li>• 作業用ストレージ</li> <li>• 再定義</li> <li>• シリアル</li> <li>• 索引付き</li> <li>• 相対</li> <li>• メッセージ・キュー</li> <li>• SQL 行</li> <li>• ユーザー・インターフェース</li> <li>• DL/I セグメント</li> </ul>	<p>EGL レコード・タイプ:</p> <ul style="list-style-type: none"> <li>• basicRecord</li> <li>• basicRecord</li> <li>• serialRecord</li> <li>• indexedRecord</li> <li>• relativeRecord</li> <li>• mqRecord</li> <li>• sqlRecord</li> <li>• VGUIRecord</li> <li>• DLISegment</li> </ul>	<p>マイグレーション・ツールは、再定義レコードを basicRecord にマイグレーションします。ツールは、レコード定義にコメントを組み込んで、再定義されたレコードの名前を示します。再定義レコードに関しては、特別な考慮事項が適用されます。詳細と起こりうる問題については、65 ページの『再定義レコード』を参照してください。</p>

表 73. レコード - 一般構文、レコード・タイプ、プロパティ、およびプロローグ (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>作業用ストレージ・レコードのプロパティ:</p> <ul style="list-style-type: none"> <li>代替仕様</li> </ul>	<p>basicRecord プロパティ:</p> <ul style="list-style-type: none"> <li>embed ステートメント</li> </ul>	<p>マイグレーション・ツールは、代替仕様を embed ステートメントにマイグレーションします。</p>
<p>再定義レコードのプロパティ:</p> <ul style="list-style-type: none"> <li>再定義</li> </ul> <p><b>注:</b> 再定義プロパティは、物理ストレージを提供する別のレコードの名前を指定します。現行レコードは、同じ物理ストレージに対して異なるデータ項目レイアウトを提供します。</p>	<p>basicRecord プロパティ:</p> <ul style="list-style-type: none"> <li>適用外。再定義情報は、レコードを使用するプログラム内でのみ指定されます。同じレコードを別のレコードの再定義として使用することも、通常のレコードとして使用することもできます。</li> </ul>	<p>マイグレーション・ツールは、レコード定義にコメントを組み込んで、再定義されたレコードの名前を示します。</p> <p>マイグレーション・ツールはまた、レコードを使用するプログラム内で、レコードに対する declaration ステートメントに redefines プロパティを組み込みます。</p> <p>レコードがプログラム内でどのように使用されるか、およびマイグレーション時にレコードが使用可能かどうかによって、特別な考慮事項が適用されます。詳細と起こりうる問題については、65 ページの『再定義レコード』を参照してください。</p>
<p>シリアル・レコードのプロパティ:</p> <ul style="list-style-type: none"> <li>ファイル名</li> <li>代替仕様</li> <li>可変長項目</li> <li>出現箇所項目</li> </ul>	<p>serialRecord プロパティ:</p> <ul style="list-style-type: none"> <li>fileName</li> <li>embed ステートメント</li> <li>lengthItem</li> <li>numElementsItem</li> </ul>	<p>マイグレーション・ツールは、代替仕様を embed ステートメントにマイグレーションします。</p>
<p>索引付きレコードのプロパティ:</p> <ul style="list-style-type: none"> <li>ファイル名</li> <li>レコード ID</li> <li>代替仕様</li> <li>可変長項目</li> <li>出現箇所項目</li> </ul>	<p>indexedRecord プロパティ:</p> <ul style="list-style-type: none"> <li>fileName</li> <li>keyItem</li> <li>embed ステートメント</li> <li>lengthItem</li> <li>numElementsItem</li> </ul>	<p>マイグレーション・ツールは、代替仕様を embed ステートメントにマイグレーションします。</p>
<p>相対レコードのプロパティ:</p> <ul style="list-style-type: none"> <li>ファイル名</li> <li>レコード ID</li> <li>代替仕様</li> </ul>	<p>relativeRecord プロパティ:</p> <ul style="list-style-type: none"> <li>fileName</li> <li>keyItem</li> <li>embed ステートメント</li> </ul>	<p>マイグレーション・ツールは、代替仕様を embed ステートメントにマイグレーションします。</p>

表 73. レコード - 一般構文、レコード・タイプ、プロパティ、およびプロログ (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
メッセージ・キュー・レコードのプロパティ: <ul style="list-style-type: none"> <li>ファイル名</li> <li>代替仕様</li> <li>トランザクションにメッセージを組み込む</li> <li>入力時にキューを排他使用で開く</li> <li>レコード長項目</li> <li>出現箇所項目</li> <li>キュー記述子レコード</li> <li>open オプション用レコード</li> <li>メッセージ記述子レコード</li> <li>get オプション用レコード</li> <li>put オプション用レコード</li> </ul>	mqRecord プロパティ: <ul style="list-style-type: none"> <li>queueName</li> <li>embed ステートメント</li> <li>includeMsgInTransaction</li> <li>openQueueExclusive</li> <li>lengthItem</li> <li>numElementsItem</li> <li>queueDescriptorRecord</li> <li>openOptionsRecord</li> <li>msgDescriptorRecord</li> <li>getOptionsRecord</li> <li>putOptionsRecord</li> </ul>	マイグレーション・ツールは、代替仕様を embed ステートメントにマイグレーションします。
SQL 行レコードのプロパティ: <ul style="list-style-type: none"> <li>261 ページの表 75 を参照。</li> </ul>	SQL 行レコードのプロパティ: <ul style="list-style-type: none"> <li>261 ページの表 75 を参照。</li> </ul>	特別な考慮事項なし。
DL/I セグメント・レコード・プロパティ: <ul style="list-style-type: none"> <li>265 ページの表 76 を参照してください。</li> </ul>	DL/I セグメント・レコード・プロパティ: <ul style="list-style-type: none"> <li>265 ページの表 76 を参照してください。</li> </ul>	特別な考慮事項なし。
UI レコード・プロパティ: <ul style="list-style-type: none"> <li>269 ページの表 77 を参照してください。</li> </ul>	UI レコード・プロパティ: <ul style="list-style-type: none"> <li>269 ページの表 77 を参照してください。</li> </ul>	特別な考慮事項なし。
プロログ	適用外。	マイグレーション・ツールは、プロログをコメントに変換し、レコード定義の前に配置します。

表 74. レコード - ほとんどのレコード・タイプのレコード構造

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
レコード構造 - バリエーション 1: 代替仕様。RecordA が代替仕様 RecordB を指定している場合、RecordB は RecordA の項目をすべて提供します。RecordA 内に項目構造はありません。  RecordB にレベル 77 項目がある場合、RecordA には RecordB のレベル 77 以外の項目のみがあります。	レコード構造 - バリエーション 1: EGL embed ステートメントは、現行レコードのフィールド構造を提供するレコードを指定します。RecordA は RecordB を組み込みます。例えば、次のようになります。 embed RecordB;	マイグレーション・ツールは、代替仕様を embed ステートメントにマイグレーションします。  作業用ストレージ・レコード内のレベル 77 項目には、特別な考慮事項が適用されます。詳細と起こりうる問題については、66 ページの『レコード内のレベル 77 項目』を参照してください。

表 74. レコード - ほとんどのレコード・タイプのレコード構造 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>レコード構造 - バリエーション 2 (共用項目を指定):</p> <ul style="list-style-type: none"> <li>• itemName</li> <li>• Occurs</li> <li>• 共用</li> <li>• levelNumber は非表示ですが、レコード内のデータ項目階層に基づいています。</li> </ul> <p>注: 型、長さ、小数部、および説明は、レコード・エディターに表示されますが、レコードに格納されません。</p>	<p>レコード構造 - EGL 型定義を使用したバリエーション 2 の例:</p> <pre>levelNumber itemName     itemName [occurs];</pre> <p>注: 型、長さ、小数部、および説明は、エディターには表示されません。</p>	<p>マイグレーション構文設定「共用データ項目をプリミティブ項目定義に変換」を選択した場合に、データ項目パーツが使用可能ならば、共用項目はマイグレーション・ツールによって EGL 変数に変換されます。この変数はデータ項目パーツに対して指定された型、長さ、および 10 進数に基づいたプリミティブ型定義を使用して定義されます。型、長さ、および 小数部の情報のマイグレーションは、249 ページの表 67 の説明と同じです。</p> <p>マイグレーション構文設定「選択済み共用データ項目をプリミティブ項目定義に変換」を選択しなかった場合、またはデータ項目パーツが使用不可の場合は、共用項目はマイグレーション・ツールによって EGL 変数に変換され、この変数は型定義を使用して定義されます。マイグレーション用の型定義は、常に項目名と同じです。</p> <p>occurs が 1 の場合、マイグレーション・ツールは occurs 情報を省略します。</p> <p>作業用ストレージ・レコード内のレベル 77 項目には、特別な考慮事項が適用されます。詳細と起こりうる問題については、66 ページの『レコード内のレベル 77 項目』を参照してください。</p>

表 74. レコード - ほとんどのレコード・タイプのレコード構造 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>レコード構造 - バリエーション 2 (非共用項目を指定):</p> <ul style="list-style-type: none"> <li>• itemName</li> <li>• Occurs</li> <li>• 型</li> <li>• 長さ</li> <li>• 小数部</li> <li>• 非共用</li> <li>• 説明</li> <li>• levelNumber は隠されていますが、レコード内のデータ項目階層を基にしています。</li> </ul> <p>注: 型、長さ、小数部、および説明は、項目とともにレコードに格納されます。</p>	<p>レコード構造 - EGL プリミティブ型を使用したバリエーション 2 の例:</p> <pre>levelNumber itemName   dataType(lengthInformation)   [occurs];   // Description</pre> <p>注: 型、長さ、小数部、および説明は、エディターに表示されます。</p>	<p>マイグレーション・ツールは非共用項目を、プリミティブ型を使用して定義される EGL 変数に変換します。型、長さ、および小数部の情報のマイグレーションは、249 ページの表 67 (データ項目 - 一般構文、データ型、長さ、小数部、および説明) の説明と同じです。</p> <p>occurs が 1 の場合、マイグレーション・ツールは occurs 情報を省略します。</p> <p>作業用ストレージ・レコード内のレベル 77 項目には、特別な考慮事項が適用されます。詳細と起こりうる問題については、66 ページの『レコード内のレベル 77 項目』を参照してください。</p>

表 75. レコード - SQL プロパティと SQL レコード構造

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>SQL レコード・プロパティ:</p> <ul style="list-style-type: none"> <li>• デフォルト・キー項目</li> <li>• 代替仕様</li> <li>• SQL 表: <ul style="list-style-type: none"> <li>– ラベル</li> <li>– 名前</li> </ul> </li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• レコードが代替仕様が指定しない場合、キー項目はレコード構造内で key=yes を指定している項目です。デフォルトのキー項目は無視されます。</li> <li>• レコードが代替仕様が指定している場合、キー項目は、現行レコード内のデフォルトのキー項目と、代替仕様レコード内で key=yes を指定している項目をマージしたものです。キーは、レコード構造内での項目の出現順にマージされます。現行レコード内のデフォルトのキー項目が、代替仕様レコード内でも key=yes として指定されている場合、その項目はマージされたキーのリストに 1 回だけ組み込まれます。</li> <li>• SQL 表名は、実際の表名 (通常の場合)、または実行時に置換される表名ホスト変数のどちらかです。表名ホスト変数はセミコロン (;) から始まります。</li> </ul>	<p>sqlRecord プロパティ:</p> <ul style="list-style-type: none"> <li>• keyItems</li> <li>• embed ステートメント</li> <li>• tableNames または tableNameVariables (あるいはその両方)</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• keyItems は、レコード内のキーすべてのリストです。レコード構造内の項目に対して、key=yes は指定されません。</li> <li>• 表名がホスト変数でない場合は、tableNames プロパティが表名と表ラベルのリストです。表名が実行時に置換されるホスト変数である場合は、tableNameVariables プロパティが表名と表ラベルのリストです。tableNameVariables プロパティ内の表名は、セミコロンから始まります。tableNames と tableNameVariables は、両方とも同じレコード定義の中で使用できます。</li> </ul>	<p>マイグレーション・ツールは、keyItems プロパティを次のように作成します。</p> <ul style="list-style-type: none"> <li>• VAGen 代替仕様が指定されていない場合、ツールはレコード構造にある key=yes を指定している項目すべてを使用しますが、VAGen のデフォルトのキー項目はインクルードしません。</li> <li>• VAGen 代替仕様が指定されている場合、ツールは代替仕様レコードにある key=yes を指定している項目と、現行レコードのデフォルトのキー項目をマージします。キーは、レコード構造内での項目の出現順と同じ順序でリストされます。現行レコードのデフォルトのキー項目が、代替仕様レコードにあるキー項目のいずれかと同じである場合、その項目は keyItems リストに 1 回だけ組み込まれます。</li> </ul> <p>マイグレーション・ツールは、tableNames と tableNameVariables のリストを次のように作成します。</p> <ul style="list-style-type: none"> <li>• 表名がホスト変数でない場合は、tableNames プロパティが表名と表ラベルから作成されます。</li> <li>• 表名がホスト変数である場合は、tableNameVariables プロパティが表名と表ラベルから作成されます。</li> </ul> <p>特別な考慮事項が適用されます。詳細と起こりうる問題については、68 ページの『代替仕様レコード』にある、SQL 代替仕様に関する情報を参照してください。</p>

表 75. レコード - SQL プロパティと SQL レコード構造 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>SQL デフォルト条件:</p> <ul style="list-style-type: none"> <li>• whereClauseText</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• SQL デフォルト条件には where 文節を指定できます (SQL 行レコード内で複数の表が使用されている場合の結合条件に対する指定が最も標準的)。構文は SQL 構文です。</li> <li>• !itemColumnName 変数が使用できます。この変数は、SQL 行レコード内の項目の名前を指定します。テストまたは生成時に、VisualAge Generator は対応する SQL 列名への置換を行います。</li> </ul>	<p>デフォルト選択条件の例</p> <pre>defaultSelectCondition =   #sqlCondition{     whereClauseText   }</pre> <p>注:</p> <ul style="list-style-type: none"> <li>• defaultSelectCondition は、VisualAge Generator と同じ目的で使用されます。</li> <li>• !itemColumnName 変数はサポートされません。実際の SQL 列名を使用する必要があります。</li> </ul>	<p>マイグレーション・ツールは、!itemColumnName 変数を対応する SQL 列名に変換します。</p> <p>特別な考慮事項が適用されます。詳細と起こりうる問題については、68 ページの『代替仕様レコード』にある、SQL 代替仕様に関する情報を参照してください。</p>
<p>レコード構造 - バリエーション 1: 代替仕様。RecordA が代替仕様 RecordB を指定している場合、RecordB は RecordA の項目をすべて提供します。RecordA 内に項目構造はありません。</p>	<p>レコード構造 - バリエーション 1: EGL embed ステートメントは、現行レコードのフィールド構造を提供するレコードを指定します。RecordA は RecordB を組み込みます。例えば、次のようになります。</p> <pre>embed RecordB;</pre>	<p>マイグレーション・ツールは、代替仕様を embed ステートメントにマイグレーションします。</p>



表 75. レコード - SQL プロパティと SQL レコード構造 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>レコード構造 - バリエーション 2 (共用項目を指定):</p> <ul style="list-style-type: none"> <li>• itemName</li> <li>• 読み取り専用</li> <li>• キー</li> <li>• SQL 列名</li> <li>• SQL コード</li> <li>• 共用</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• 型、長さ、小数部、および説明は、レコード・エディターに表示されますが、レコードに格納されません。</li> <li>• レベル番号は、SQL レコード内では使用されません。</li> <li>• バック・フィールドとバイナリー・フィールドの外部ソース形式には、SQL コードは組み込まれません。char、dbchar、または unicode のフィールドの外部ソース形式に SQL コードが組み込まれていない場合、フィールドは固定長フィールドとして処理されます。この状態は、VisualAge Generator の旧リリースからマイグレーションされたレコードが、VisualAge Generator 4.5 を使用して変更されなかった場合に起こります。</li> </ul>	<p>レコード構造 - EGL 型定義を使用したバリエーション 2 の例:</p> <pre>levelNumber itemName itemName { [sqlDataCode=sqlCodeNumber]   [column="SQLColumnName"]   [ isReadOnly=yes ]   [ isNullable = yes ]   [ sqlVariableLen = yes ] };</pre> <p>注:</p> <ul style="list-style-type: none"> <li>• 型、長さ、小数部、および説明は、エディターには表示されません。</li> <li>• レベル番号は、SQL レコード内ではオプションです。レベル番号が指定されている場合、SQL レコードは固定レコードです。レベル番号が指定されていない場合、SQL レコードは固定レコードではありません。非固定レコードでは、新規の EGL データ型 BLOB、CLOB、およびストリングを使用できます。ただし、非固定レコードのその他の振る舞いは、VAGen の振る舞いと互換性がありません。</li> </ul>	<p>マイグレーション構文設定「共用データ項目をプリミティブ項目定義に変換」を選択した場合、データ項目が使用可能であれば、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• 共用項目を、データ項目パーツに対して指定された型、長さ、および小数部に基いたプリミティブ定義を使用して定義された EGL 変数に変換します。</li> <li>• 16 進項目の sqlDataCode プロパティを組み合わせます。</li> <li>• 項目が可変長であることが VAGen SQL データ・コードによって示されている場合は、char、dbchar、または unicode のフィールドに対して sqlVariableLen=yes プロパティを設定します。項目が固定長であることが VAGen SQL データ・コードによって示されている場合、マイグレーション・ツールは sqlVariableLen プロパティを省略します。</li> </ul> <p>マイグレーション構文設定「共用データ項目をプリミティブ項目定義に変換」を選択しない場合、またはデータ項目パーツが使用不可の場合は、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• 型定義を使用して定義された EGL 変数に共用項目を変換します。マイグレーション用の型定義は、常に項目名と同じです。</li> <li>• 項目が外部ソース形式に含まれていて、VAGen バイナリー・フィールドまたはバック・フィールドのいずれかの値でなければ、sqlDataCode プロパティを組み合わせます。</li> <li>• 項目が可変長であることが VAGen SQL データ・コードによって示されている場合は、sqlVariableLen=yes プロパティを設定します。項目が固定長であることが VAGen SQL データ・コードによって示されている場合、マイグレーション・ツールは sqlVariableLen プロパティを省略します。</li> </ul> <p>マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• sqlRecord の EGL keyItems プロパティに、key=yes の項目をすべて組み合わせます。</li> <li>• レコードが固定レコードになるように、sqlRecord 内の項目にレベル番号を常に追加します。これにより、VAGen の振る舞いが保持されます。</li> </ul>

表 75. レコード - SQL プロパティと SQL レコード構造 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>レコード構造 - バリエーション 2 (非共用項目を指定):</p> <ul style="list-style-type: none"> <li>itemName</li> <li>型</li> <li>長さ</li> <li>小数部</li> <li>読み取り専用</li> <li>キー</li> <li>SQL 列名</li> <li>SQL コード</li> <li>非共用</li> <li>説明</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>型、長さ、小数部、および説明は、項目とともにレコードに格納されます。</li> <li>レベル番号は、SQL レコード内では使用されません。</li> <li>パック・フィールドとバイナリー・フィールドの外部ソース形式には、SQL コードは組み込まれません。char、dbchar、または unicode のフィールドの外部ソース形式に SQL コードが組み込まれていない場合、フィールドは固定長フィールドとして処理されます。この状態は、VisualAge Generator の旧リリースからマイグレーションされたレコードが、VisualAge Generator 4.5 を使用して変更されなかった場合に起こります。</li> </ul>	<p>レコード構造 - EGL プリミティブ型を使用したバリエーション 2 の例:</p> <pre>levelNumber itemName dataType(lengthInformation) // Description { [sqlDataCode=sqlCodeNumber]   [ column="SQLColumnName" ]   [ isReadOnly=yes ]   [ isNullable = yes ]   [ sqlVariableLen = yes ] };</pre> <p>注:</p> <ul style="list-style-type: none"> <li>型、長さ、小数部、および説明は、エディターに表示されます。</li> <li>レベル番号は、SQL レコード内ではオプションです。レベル番号が指定されている場合、SQL レコードは固定レコードです。レベル番号が指定されていない場合、SQL レコードは固定レコードではありません。非固定レコードでは、新規の EGL データ型 BLOB、CLOB、およびストリングを使用できます。ただし、非固定レコードのその他の振る舞いは、VAGen の振る舞いと互換性がありません。</li> </ul>	<p>マイグレーション・ツールは非共用項目を、プリミティブ型を使用して定義される EGL 変数に変換します。型、長さ、および小数部の情報のマイグレーションは、249 ページの表 67 の説明と同様です。</p> <p>マイグレーション・ツールは、sqlDataCode プロパティを 16 進項目のみに組み込みます。</p> <p>項目が可変長であることが VAGen SQL データ・コードによって示されている場合は、char、dbchar、または unicode のデータ項目に対してマイグレーション・ツールは sqlVariableLen=yes プロパティを設定します。項目が固定長であることが VAGen SQL データ・コードによって示されている場合、マイグレーション・ツールは sqlVariableLen プロパティを省略します。</p> <p>マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>sqlRecord の EGL keyItems プロパティに、key=yes の項目をすべて組み込みます。</li> <li>レコードが固定レコードになるように、sqlRecord 内の項目にレベル番号を常に追加します。これにより、VAGen の振る舞いが保持されます。</li> </ul>
<p>VAGen データ型 - Char</p> <ul style="list-style-type: none"> <li>データ・コード - 453</li> <li>データ・コード - 449 または 457</li> </ul>	<p>EGL データ型:</p> <ul style="list-style-type: none"> <li>char (sqlVariableLen は省略)</li> <li>varchar, sqlVariableLen = yes</li> </ul>	<p>特別な考慮事項なし。</p>
<p>VAGen データ型 - DBCS</p> <ul style="list-style-type: none"> <li>データ・コード - 469</li> <li>データ・コード - 465 または 473</li> </ul>	<p>EGL データ型:</p> <ul style="list-style-type: none"> <li>dbchar (sqlVariableLen は省略)</li> <li>vardbchar, sqlVariableLen = yes</li> </ul>	<p>特別な考慮事項なし。</p>
<p>VAGen データ型 - Unicode</p> <ul style="list-style-type: none"> <li>データ・コード - 469</li> <li>データ・コード - 465 または 473</li> </ul>	<p>EGL データ型:</p> <ul style="list-style-type: none"> <li>unicode (sqlVariableLen は省略)</li> <li>varunicode, sqlVariableLen = yes</li> </ul>	<p>特別な考慮事項なし。</p>
<p>SQL 列名</p> <p>注: SQL 列名は必須です。</p>	<p>column = "SQLColumnName"</p> <p>注: column プロパティはオプションです。column プロパティを省略した場合は、デフォルトでフィールド名に設定されます。</p>	<p>マイグレーション・ツールは、「列名を省略」の設定に基づいてマイグレーションを行います。</p> <ul style="list-style-type: none"> <li>この設定を選択した場合、ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>SQL 列名がフィールド名と同じならば、column プロパティを省略します。</li> <li>SQL 列名がフィールド名と異なっていれば、column プロパティを組み込みます。</li> </ul> </li> <li>この設定を選択しない場合、ツールは column プロパティを常に組み込みます。</li> </ul>

表 75. レコード - SQL プロパティと SQL レコード構造 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>対応するプロパティはありません。</p> <p>注: VisualAge Generator は、SQL 項目に常に NULL 標識変数を組み込みます。</p>	<p>isNullable = yes   no</p> <p>注: isNullable のデフォルトは no です。</p>	<p>マイグレーション・ツールは、「isNullable プロパティを省略」の設定に基づいてマイグレーションを行います。</p> <ul style="list-style-type: none"> <li>この設定を選択した場合、ツールは isNullable プロパティを組み込まず、このプロパティはデフォルトで no に設定されます。</li> <li>この設定を選択しない場合、ツールは isNullable=yes プロパティを常に組み込みます。これにより、VAGen の振る舞いが保持されます。</li> </ul>
<p>読み取り専用</p> <p>注: 「読み取り専用」は常に明示的に設定されます。SQL レコードに対して複数の表が指定されている場合、「読み取り専用」は常に yes にする必要があります。</p>	<p>isReadOnly = yes   no</p> <p>注: SQL レコードに対して複数の表が指定されている場合、ReadOnly はデフォルトで yes に設定されます。SQL レコードに対して表がただ 1 つ指定されている場合、ReadOnly はデフォルトで no に設定されます。</p>	<p>マイグレーション・ツールは、「isReadOnly プロパティを省略」の設定に基づいてマイグレーションを行います。</p> <ul style="list-style-type: none"> <li>この設定を選択した場合は、レコードに対して単一の表が指定されていて、VAGen の「読み取り専用」プロパティが yes に設定されている場合に限り、ツールは isReadOnly プロパティを組み込みます。</li> <li>この設定を選択しない場合は、VAGen の「読み取り専用」プロパティが yes に設定されていれば、ツールは isReadOnly プロパティを組み込みます。</li> </ul>

表 76. レコード - DLI プロパティと DLI レコード構造

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>DLI レコード・プロパティ</p> <ul style="list-style-type: none"> <li>キー項目</li> <li>代替仕様</li> <li>レコード長項目</li> </ul> <p>注: レコード名は DLI PSB 内のセグメント名と同じでなければなりません。</p>	<p>DLISegment プロパティ:</p> <ul style="list-style-type: none"> <li>keyItem</li> <li>embed ステートメント</li> <li>lengthItem</li> </ul> <p>注: EGL では DLI PSB のセグメント名と異なるレコード名が許されます。この状態で、EGL <i>segmentName</i> プロパティは DLI PSB で使用される名前を提供します。</p>	<p>マイグレーション・ツールは、EGL 予約語との競合、またはレコード名の先頭が # または @ シンボルであることが原因でレコード名を変更する場合、オリジナルのレコード名を提供するために、<i>segmentName</i> プロパティをインクルードします。</p>

表 76. レコード - *DL/I* プロパティと *DL/I* レコード構造 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>レコード構造 - バリエーション 1: 代替仕様。RecordA が代替仕様 RecordB を指定している場合、RecordB は RecordA の項目をすべて提供します。RecordA 内に項目構造はありません。</p> <p><b>注:</b> 組み込みレコードのフィールド名は、PSB の <i>DL/I</i> フィールド名と同じでなければなりません。</p>	<p>レコード構造 - バリエーション 1: EGL <code>embed</code> ステートメントは、現行レコードのフィールド構造を提供するレコードを指定します。RecordA は RecordB を組み込みます。例えば、次のようになります。</p> <p><code>embed RecordB;</code></p> <p>EGL ではレコードのフィールド名に、<i>DL/I</i> PSB のフィールド名と異なる名前を使用できます。組み込まれたレコードが <i>DLISegment</i> ではなく、フィールド名が異なる場合、<i>dliFieldName</i> プロパティを次のようにセットしてください。</p> <pre>embed RecordB { fieldInRecordB   {dliFieldName="nameInPSB"} } ;</pre>	<p>マイグレーション・ツールは、代替仕様を <code>embed</code> ステートメントにマイグレーションします。</p> <p>代替仕様レコードが <i>DL/I</i> セグメントでない場合、ツールは EGL 予約語との競合、またはフィールド名の先頭が # または @ シンボルであることが原因で名前変更された項目の <i>dliFieldName</i> プロパティを指定変更します。</p> <p>特別な考慮事項が適用されます。詳しくは、68 ページの『代替仕様レコード』を参照してください。</p>

表 76. レコード - *DL/I* プロパティと *DL/I* レコード構造 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>レコード構造 - バリエーション 2 (共用項目を指定):</p> <ul style="list-style-type: none"> <li>• itemName</li> <li>• Occurs</li> <li>• 共用</li> <li>• levelNumber は隠されていますが、レコード内のデータ項目階層を基にしています。</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• 型、長さ、小数部、および説明は、レコード・エディターに表示されますが、レコードに格納されません。</li> <li>• フィールド名は、PSB の <i>DL/I</i> フィールド名と同じでなければなりません。</li> </ul>	<p>レコード構造 - EGL 型定義を使用したバリエーション 2 の例:</p> <pre>levelNumber itemName   itemName [occurs]   {dliFieldName="nameInPSB"};</pre> <p>注:</p> <ul style="list-style-type: none"> <li>• 型、長さ、小数部、および説明は、エディターには表示されません。</li> <li>• EGL では、レコードのフィールド名に、<i>DL/I</i> PSB のフィールド名と異なる名前を使用できます。</li> </ul>	<p>マイグレーション構文設定「共用データ項目をプリミティブ項目定義に変換」を選択した場合に、データ項目パーツが使用可能ならば、共用項目はマイグレーション・ツールによって EGL 変数に変換されます。この変数はデータ項目パーツに対して指定された型、長さ、および 10 進数に基づいたプリミティブ型定義を使用して定義されます。型、長さ、および小数部の情報のマイグレーションは、249 ページの表 67 の説明と同じです。</p> <p>マイグレーション構文設定「選択済み共用データ項目をプリミティブ項目定義に変換」を選択しなかった場合、またはデータ項目パーツが使用不可の場合は、共用項目はマイグレーション・ツールによって EGL 変数に変換され、この変数は型定義を使用して定義されます。マイグレーション用の型定義は、常に項目名と同じです。</p> <p>occurs が 1 の場合、マイグレーション・ツールは occurs 情報を省略します。</p> <p>EGL 予約語との競合、またはレコード名の先頭が # または @ シンボルであることが原因で、項目が名前変更される場合、マイグレーション・ツールは dliFieldName プロパティをインクルードします。</p>

表 76. レコード - *DL/I* プロパティと *DL/I* レコード構造 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>レコード構造 - バリエーション 2 (非共用項目を指定):</p> <ul style="list-style-type: none"> <li>• itemName</li> <li>• Occurs</li> <li>• 型</li> <li>• 長さ</li> <li>• 小数部</li> <li>• 非共用</li> <li>• 説明</li> </ul> <p>levelNumber は隠されていますが、レコード内のデータ項目階層を基にしています。</p> <p>注:</p> <ul style="list-style-type: none"> <li>• 型、長さ、10 進数および記述は、レコードに保管されます。</li> <li>• フィールド名は、PSB の <i>DL/I</i> フィールド名と同じでなければなりません。</li> </ul>	<p>レコード構造 - EGL プリミティブ型を使用したバリエーション 2 の例:</p> <pre>levelNumber itemName dataType(lengthInformation) [occurs] {dliFieldName="nameInPSB"};</pre> <p>注:</p> <ul style="list-style-type: none"> <li>• 型、長さ、小数部、および説明は、エディターに表示されます。</li> <li>• EGL ではレコードのフィールド名に、<i>DL/I</i> PSB のフィールド名と異なる名前を使用できます。</li> </ul>	<p>マイグレーション・ツールは非共用項目を、プリミティブ型を使用して定義される EGL 変数に変換します。型、長さ、および 小数部の情報のマイグレーションは、249 ページの表 67 の説明と同じです。</p> <p>occurs が 1 の場合、マイグレーション・ツールは occurs 情報を省略します。</p> <p>EGL 予約語との競合、またはレコード名の先頭が # または @ シンボルであることが原因で、項目が名前変更される場合、マイグレーション・ツールは dliFieldName プロパティをインクルードします。</p>



表 77. レコード - UI レコード・プロパティとレコード構造

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
UI レコード・プロパティ: <ul style="list-style-type: none"> <li>一般               <ul style="list-style-type: none"> <li>UI タイトル</li> <li>値項目の実行依頼</li> <li>関数の編集</li> <li>ウェブでの関数編集の実行</li> </ul> </li> <li>編集順序の入力</li> <li>ヘルプ・テキスト</li> </ul>	VGUI レコード・プロパティ: <ul style="list-style-type: none"> <li>一般プロパティ               <ul style="list-style-type: none"> <li>タイトル</li> <li>commandValueItem</li> <li>validatorFunction</li> <li>runValidatorFromProgram</li> </ul> </li> <li>validationOrder</li> <li>ヘルプ</li> </ul> <p>注: validationOrder プロパティは、レコード内の各項目で指定されます。</p> <p>VGUI レコード定義の例は次のとおりです。</p> <pre>Record recordName type VGUIRecord {alias="originalVAGenName",  commandValueItem=itemX,  validatorFunction=functionY,  runValidatorFromProgram=yes,  title="Page Title",  help="help text line" } recordStructure end // end recordName</pre>	EGL runValidatorFromProgram プロパティは、VAGen プロパティの逆です。マイグレーション・ツールは、yes を no に、no を yes に変換します。
適用外。	別名	レコード名が EGL 予約語と競合するか、レコード名の先頭が # または @ シンボルである場合、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>UI レコードを名前変更する。</li> <li>alias プロパティをオリジナルの VAGen UI レコード名にセットする。</li> </ul> <p>特別な考慮事項が適用されます。詳しくは、71 ページの『予約語と UI レコード名』を参照してください。</p>
レコード構造は、それぞれのデータ項目ごとに追加情報がある点を除き、作業用ストレージ・レコードの構造に似ています。 <ul style="list-style-type: none"> <li>UI タイプ</li> <li>UI プロパティ</li> </ul>	レコード構造は、それぞれのフィールドごとに追加プロパティがある点を除き、基本レコードの構造に似ています。 <ul style="list-style-type: none"> <li>uiType</li> <li>検証プロパティ、フォーマット・プロパティ、および PageHandler フィールド・プロパティ</li> </ul>	特別な考慮事項なし。

表 78. レコード - UI 項目プロパティ - 一般

VisualAge Generator 4.5	EGL	マイグレーション・ツールに関する考慮事項
適用外	別名	<p>フィールド名が EGL 予約語と競合するか、フィールド名の先頭が # または @ シンボルである場合、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>フィールドを名前変更する。</li> <li>alias プロパティをオリジナルのフィールド名にセットする。</li> </ul> <p>特別な考慮事項が適用されます。詳しくは、71 ページの『予約語と UI レコード名』を参照してください。</p>
<p>UI タイプ - 値は次のとおり:</p> <ul style="list-style-type: none"> <li>書式</li> <li>隠しファイル</li> <li>入力</li> <li>Input/Output</li> <li>None</li> <li>Output</li> <li>プログラム・リンク</li> <li>実行依頼</li> <li>迂回の実行依頼</li> </ul>	<p>uiType - 値は次のとおり:</p> <ul style="list-style-type: none"> <li>uiForm</li> <li>隠しファイル</li> <li>input</li> <li>inputOutput</li> <li>none</li> <li>output</li> <li>programLink</li> <li>実行依頼</li> <li>submitBypass</li> </ul>	特別な考慮事項なし。
UI ラベル	displayName (PageHandler フィールド・プロパティ)	特別な考慮事項なし。
<p>配列項目:</p> <ul style="list-style-type: none"> <li>出現箇所項目</li> <li>選択済みインデックス項目</li> </ul>	<p>構造化フィールド配列:</p> <ul style="list-style-type: none"> <li>numElementsItem</li> <li>selectedIndexItem</li> </ul>	特別な考慮事項なし。

表 79. レコード - UI 項目プロパティ - 編集

VisualAge Generator 4.5	EGL	マイグレーション・ツールに関する考慮事項
<p>Edit タイプ - 値は次のとおり:</p> <ul style="list-style-type: none"> <li>None</li> <li>ブール</li> <li>日付</li> <li>時刻</li> </ul>	<p>EGL は、複数のプロパティをサポートします。</p> <ul style="list-style-type: none"> <li>適用外</li> <li>isBoolean = yes</li> <li>dateFormat = defaultDateFormat</li> <li>timeFormat = "HH:mm:ss"</li> </ul> <p>(フォーマット設定プロパティ)</p>	特別な考慮事項なし。
関数の編集	validatorFunction (検証プロパティ)	特別な考慮事項なし。

表 79. レコード - UI 項目プロパティ - 編集 (続き)

VisualAge Generator 4.5	EGL	マイグレーション・ツールに関する考慮事項
ウェブでの関数編集の実行	runValidatorFromProgram	EGL プロパティは、VAGen プロパティの逆です。マイグレーション・ツールは yes を no に、no を yes に変換します。
テーブルの編集	validatorDataTable (検証プロパティ)	特別な考慮事項なし。
最小入力	minimumInput (検証プロパティ)	特別な考慮事項なし。
充てん文字 <b>注:</b> <ul style="list-style-type: none"> <li>デフォルト充てん文字は、次のとおりです。               <ul style="list-style-type: none"> <li>文字、混合、および数値項目の場合のブランク。</li> <li>16 進項目の場合の 0。</li> </ul> </li> <li>DBCS およびユニコードの項目の充てん文字はブランクでなければなりません。</li> <li>NULL は無効な充てん文字です。</li> </ul>	fillCharacter (フォーマット設定プロパティ) <b>注:</b> <ul style="list-style-type: none"> <li>デフォルト充てん文字は、次のとおりです。               <ul style="list-style-type: none"> <li>文字、mbchar、および数値項目の場合のブランク。</li> <li>16 進項目の場合の 0。</li> </ul> </li> <li>dbchar の場合の充てん文字はブランクでなければなりません。ユニコード用充てん文字としては、どの文字も有効です。</li> <li>NULL は無効な充てん文字です。</li> </ul>	特別な考慮事項なし。
大文字変換	upperCase (フォーマット設定プロパティ)	特別な考慮事項なし。
入力必須 (Input required)	inputRequired (検証プロパティ)	特別な考慮事項なし。
SO/SI スペースの検査	needsSOSI (検証プロパティ)	特別な考慮事項なし。
通貨と通貨記号	currency = yes   no currencySymbol = "symbol"	特別な考慮事項なし。
最小値と最大値 <b>注:</b> 最小値または最大値のどちらかを指定する場合は、両方を指定する必要があります。	validValues = [[minimumValue, maximumValue]] (検証プロパティ)	マイグレーション・ツールは、最小値と最大値を EGL の validValues プロパティに結合します。
符号 - なし   先頭   末尾 <b>注:</b> デフォルトは「なし」です。	sign = none   leading   trailing (フォーマット設定プロパティ) <b>注:</b> デフォルトは「なし」です。	特別な考慮事項なし。
セパレーター	numericSeparator (フォーマット設定プロパティ)	特別な考慮事項なし。
ゼロ編集	zeroFormat (フォーマット設定プロパティ)	特別な考慮事項なし。

表 80. レコード - UI 項目プロパティ - エラー・メッセージ

VisualAge Generator 4.5	EGL	マイグレーション・ツールに関する考慮事項
テーブルの編集	validatorTableMsgKey (検証プロパティ)	特別な考慮事項なし。

表 80. レコード - UI 項目プロパティ - エラー・メッセージ (続き)

VisualAge Generator 4.5	EGL	マイグレーション・ツールに関する考慮事項
EZE 関数	validatorFunctionMsgKey (検証プロパティ)	特別な考慮事項なし。
最小入力	minimumInputMsgKey (検証プロパティ)	特別な考慮事項なし。
入力必須 (Input required)	inputRequiredMsgKey (検証プロパティ)	特別な考慮事項なし。
データ型	typeChkMsgKey (検証プロパティ)	特別な考慮事項なし。
数値範囲	validValuesMsgKey (検証プロパティ)	特別な考慮事項なし。

表 81. UI 項目プロパティ - ヘルプ

VisualAge Generator 4.5	EGL	マイグレーション・ツールに関する考慮事項
ヘルプ・テキスト	ヘルプ (PageHandler フィールド・プロパティ)	特別な考慮事項なし。

表 82. UI 項目プロパティ - 実行依頼

VisualAge Generator 4.5	EGL	マイグレーション・ツールに関する考慮事項
初期値	配列でないフィールドに関する次のフォーマットでのイニシャライザー。 <code>= "initialValue"</code>  構造化フィールド配列に関する次のフォーマットでのイニシャライザー。 <code>= ["initialValue1", "initialValue2"]</code>	特別な考慮事項なし。

表 83. レコード - UI 項目プロパティ - プログラム・リンク

VisualAge Generator 4.5	EGL	マイグレーション・ツールに関する考慮事項
<p>プログラム・リンク情報</p> <ul style="list-style-type: none"> <li>プログラム</li> <li>先頭 UI レコード</li> <li>新規ウィンドウとして開く</li> <li>パラメーターのリンク</li> </ul>	<p>プログラム・リンク情報</p> <ul style="list-style-type: none"> <li>programName</li> <li>uiRecordName</li> <li>newWindow</li> <li>linkParms</li> </ul> <p>EGL は link プロパティを結合して、次のような複素数プロパティにします。</p> <pre>@programLinkData {   programName = "PRGA",   uiRecordName = "MYUI",   newWindow = yes   [ , linkParmsInfo ] }</pre> <p>注: オプションの linkParmsInfo についての詳細は、以下を参照してください。</p>	<p>特別な考慮事項なし。</p>
<p>パラメーターのリンク:</p> <ul style="list-style-type: none"> <li>先頭 UI レコードの項目</li> <li>値           <ul style="list-style-type: none"> <li>リテラル</li> <li>現行レコードの項目</li> </ul> </li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>フォーム UI タイプ用のプログラム・リンク・カスタマイズを使用しているときには、現行フォーム内のデータは、次のプログラムの最初の UI レコードに名前ごとに自動的に移動させられます。先頭 UI レコード内の追加フィールドは、リンク・パラメーター内でリスト作成することで初期化することができます。</li> <li>プログラム・リンク UI タイプに関してプログラム・リンク・カスタマイズを使用しているときには、リンク・パラメーターに明示的にリストされた先頭 UI レコード内のフィールドのみが初期化されます。</li> </ul>	<p>EGL は VAGen リンク・パラメーターを結合して複素数プロパティにします。</p> <pre>linkParms = [   @LinkParameter     { name = "item1InFirstUI",       value = "literal" },   @LinkParameter     { name = "item2InFirstUI",       valueRef = "itemInCurrent"}]</pre> <p>注: uiForm と programLink UI の両方のタイプに関してプログラム・リンク・カスタマイズを行う場合、EGL は VisualAge Generator と同じ規則に従います。</p>	<p>特別な考慮事項なし。</p>

## テーブル

VAGen テーブルに関するセクションは、次の表で構成されています。

- テーブル - 一般構文、テーブル・タイプ、プロパティ、およびプロローグ (274 ページの表 84)
- VAGen テーブル - テーブル構造 (275 ページの表 85)
- VAGen テーブル - テーブルの内容 (276 ページの表 86)

表 84. テーブル - 一般構文、テーブル・タイプ、プロパティ、およびプロローグ

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
VAGen テーブル・パーツ: <ul style="list-style-type: none"><li>• tableName</li><li>• 基本情報<ul style="list-style-type: none"><li>- テーブル・タイプ</li><li>- テーブル構造 (項目リスト)</li></ul></li><li>• プロパティ</li><li>• プロローグ</li><li>• テーブルの内容</li></ul>	EGL 構文の例: <pre>/***/ DataTable=tableName***/ // prolog //***** DataTable tableName   type tableType   { [otherTableProperties]     [alias =       "originalTableName"] }   tableStructure   [{ contents =     [{rowContents}] }] end // end tableName</pre>	名前が EGL 予約語リストと競合していても、マイグレーション・ツールはテーブルの名前を変更しません。マイグレーション・ツールは alias プロパティを設定しません。テーブルの名前を変更する必要がある場合は、alias プロパティを使用して、VAGen テーブルのオリジナルの名前を指定してください。詳細については、72 ページの『予約語とテーブル名』のテーブル名に関する情報を参照してください。
テーブル・タイプ: <ul style="list-style-type: none"><li>• 指定なし</li><li>• 不適合</li><li>• 適合</li><li>• 範囲適合</li><li>• メッセージ</li></ul>	DataTable タイプ: <ul style="list-style-type: none"><li>• basicTable</li><li>• matchInvalidTable</li><li>• matchValidTable</li><li>• rangeChkTable</li><li>• msgTable</li></ul>	特別な考慮事項なし。
プロパティ - 実行時属性: <ul style="list-style-type: none"><li>• 常駐</li><li>• 共用</li></ul>	DataTable プロパティ: <ul style="list-style-type: none"><li>• resident</li><li>• shared</li></ul>	特別な考慮事項なし。
プロパティ - テーブルの内容の大文字変換	適用外。テーブルの内容を大文字にする必要がある場合は、内容を大文字で入力する必要があります。	VAGen テーブルがテーブルの内容を大文字に変換するように指定している場合、マイグレーション・ツールは、テーブルの内容にある char データ、16 進データ、および混合データが大文字に変換されるようにします。
プロローグ	適用外。	マイグレーション・ツールは、プロローグをコメントに変換し、DataTable 定義の前に配置します。



表 85. テーブル — テーブル構造

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>VAGen テーブル構造 - 共用項目を指定:</p> <ul style="list-style-type: none"> <li>• itemName</li> <li>• 共用</li> <li>• levelNumber は非表示ですが、レコード内のデータ項目階層に基づいています。</li> </ul> <p>注: 型、長さ、小数部、および説明は、テーブル・エディターに表示されますが、テーブルに格納されません。</p>	<p>DataTable 構造 - EGL 型定義を指定:</p> <pre>levelNumber itemName       itemName ;</pre> <p>注: 型、長さ、小数部、および説明は、エディターには表示されません。</p>	<p>マイグレーション構文設定「選択済み共用データ項目をプリミティブ項目定義に変換」を選択した場合に、データ項目パーツが使用可能ならば、共用項目はマイグレーション・ツールによって EGL 変数に変換され、この変数はデータ項目パーツに対して指定された型、長さ、および小数部に基づいたプリミティブ定義を使用して定義されます。型、長さ、および小数部の情報のマイグレーションは、249 ページの表 67 の説明と同じです。</p> <p>マイグレーション構文設定「選択済み共用データ項目をプリミティブ項目定義に変換」を選択しなかった場合、またはデータ項目パーツが使用不可の場合は、共用項目はマイグレーション・ツールによって EGL 変数に変換され、この変数は型定義を使用して定義されます。マイグレーション用の型定義は、常に項目名と同じです。</p>
<p>VAGen テーブル構造 — 非共用項目を指定:</p> <ul style="list-style-type: none"> <li>• itemName</li> <li>• 型</li> <li>• 長さ</li> <li>• 小数部</li> <li>• 非共用</li> <li>• 説明</li> <li>• levelNumber は非表示ですが、テーブル内のデータ項目階層に基づいています。</li> </ul> <p>注: 型、長さ、小数部、および説明は、項目とともにテーブルに格納されます。</p>	<p>DataTable 構造 — EGL プリミティブ型を指定:</p> <pre>levelNumber itemName       dataType(lengthInformation) ;       // Description</pre> <p>注: 型、長さ、小数部、および説明は、エディターに表示されます。</p>	<p>マイグレーション・ツールは非共用項目を、プリミティブ型を使用して定義される EGL 変数に変換します。型、長さ、および小数部の情報のマイグレーションは、249 ページの表 67 の説明と同じです。</p>

表 86. テーブル — テーブルの内容

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>テーブルの内容:</p> <ul style="list-style-type: none"> <li>• テーブルの内容は、フォーマット設定エディターに入力されます。テーブルの内容は、テーブル構造のトップレベル (親) 項目に対して入力されます。</li> <li>• 文字データと 16 進データは引用符で囲まれません。</li> </ul>	<p>DataTable の内容:</p> <ul style="list-style-type: none"> <li>• 各行の内容は大括弧で囲まれます。外側に、行のセット全体を囲む大括弧の組があります。</li> <li>• 行の内容に含まれる値は、コンマで区切る必要があります。</li> <li>• 文字データ (16 進データを含む) は、二重引用符で囲む必要があります。</li> </ul> <p>例:</p> <pre>contents = [ [ rowContents ]               { , [ rowContents ] }             ]</pre> <p>ここで、 rowContents = value { , value }</p>	<p>VAGen テーブルがテーブルの内容を大文字に変換するように指定している場合、マイグレーション・ツールは、テーブルの内容にある char データ、16 進データ、および混合データが大文字に変換されるようにします。</p> <p>マイグレーション・ツールはまた、文字データ (16 進データを含む) を二重引用符で囲みます。</p>

## マップ・グループ

マップ・グループに関するセクションは、次の表で構成されています。

- マップ・グループ — 一般情報 (276 ページの表 87)
- マップ・グループ — 一般構文と浮動域 (278 ページの表 88)
- マップ・グループ — 装置の名前、タイプ、およびサイズ (280 ページの表 89)

表 87. マップ・グループ — 一般情報

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>マップ・グループ・パーツは、浮動域が存在する場合のみ必要です。</p> <p>マップ・グループ・パーツがない場合は、マップ・グループ・パーツが存在する場合と同様に、VisualAge Generator によって同じマップ・グループ名のマップすべてが自動的に生成されます。</p>	<p>formGroup が必要です。</p>	<p>マイグレーション・ツールは、formGroup パーツがマイグレーション・セット内に存在しなければ、このパーツを作成します。</p>

表 87. マップ・グループ — 一般情報 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>マップ名は、マップ・グループ名とマップ名からなります。</p>	<p>書式名には、formGroup 名は含まれません。</p> <p>formGroup 内で書式を定義 (ネスト) できます。</p> <p>また、書式を formGroup の外部で定義することもできます。この場合、formGroup にはフォーム名を指定する use ステートメントと、フォームの位置指定をするパッケージをインポートする import ステートメントをインクルードする必要があります。この技法により、共通書式 (例えば、ポップアップ・リストの書式) の定義を 1 つ作成すれば、さまざまな formGroup 内でその書式を使用できます。</p>	<p>マイグレーション・ツールは、マップをすべて書式にマイグレーションします。マイグレーション・ツールは、複数のマップ・グループ間で共通する同一のマップ定義を識別しません。</p> <p>単一ファイル・モードでマイグレーションを行う場合、マイグレーション・ツールは formGroup 内のそれぞれの書式ごとに use ステートメントを組み込みます。該当する formGroup 内で書式がネストされるように、書式を移動する必要があります。</p> <p>ステージ 1 から 3 のマイグレーションを使用する場合、マイグレーション・ツールはすべての書式を formGroup 内で自動的にネストします。</p>
<p>プログラムがマップ・グループを指定していれば、プログラムはマップ名を参照するだけでマップ・グループ内の任意のマップを使用できます。</p>	<p>使用する formGroup を示す use ステートメントがプログラムに組み込まれていれば、プログラムは書式名を参照するだけで formGroup 内の任意のマップを参照できます。</p>	<p>特別な考慮事項なし。</p>

表 88. マップ・グループ — 一般構文と浮動域

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>マップ・グループ・パーツには、次の情報を含めることができます。</p> <ul style="list-style-type: none"> <li>マップ・グループ名</li> <li>浮動域情報 <ul style="list-style-type: none"> <li>装置名</li> <li>装置サイズ</li> <li>サイズ <ul style="list-style-type: none"> <li>行数</li> <li>列数</li> </ul> </li> <li>位置 <ul style="list-style-type: none"> <li>開始行</li> <li>開始列</li> </ul> </li> </ul> </li> </ul>	<p>formGroup には、次の情報を含めることができます。</p> <ul style="list-style-type: none"> <li>formGroup 名</li> <li>formGroup プロパティ</li> <li>画面浮動域の情報</li> <li>印刷浮動域の情報</li> <li>formGroup に組み込まれる書式の Use ステートメント</li> </ul> <p>formGroup のフォーマットは、例えば次のとおりです。</p> <pre> <b>FormGroup</b> groupName {   [ <b>alias</b>="generationName"]   [ <b>screenFloatingArea</b>     {screenFloatingAreaInformation}]   [<b>printFloatingArea</b>     {printFloatingAreaInformation}] } <b>Form</b> formName <b>type</b> textForm   {formProperties}   [variableFields]   [constantFields] <b>end</b> // end formName <b>use</b> formName2; <b>end</b> // end groupName </pre>	<p>マイグレーション・ツールは、VAGen 装置タイプを使用して、浮動域情報の対象が表示マップ (screenFloatingArea) またはプリンター・マップ (printFloatingArea) のどちらであるかを判別します。</p> <p>deviceType の設定については、280 ページの表 89 を参照してください。</p>
適用外。	別名	<p>マップ・グループ名が EGL 予約語と競合していても、マイグレーション・ツールはマップ・グループの名前を変更しません。特別な考慮事項が適用されます。詳細と起こりうる問題については、73 ページの『予約語と formGroup 名』を参照してください。</p>

表 88. マップ・グループ — 一般構文と浮動域 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>浮動域情報は、次のような情報です。</p> <ul style="list-style-type: none"> <li>• 装置名</li> <li>• 装置サイズ (行 x 列)</li> <li>• 浮動域仕様 <ul style="list-style-type: none"> <li>– サイズ <ul style="list-style-type: none"> <li>- 行数</li> <li>- 列数</li> </ul> </li> <li>– 位置 <ul style="list-style-type: none"> <li>- 開始行</li> <li>- 開始列</li> </ul> </li> </ul> </li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• VisualAge Generator では、浮動域のサイズと開始位置を定義します。</li> <li>• 推奨されませんが、同じサイズをもつ装置に対して異なる浮動域仕様を指定できます。</li> </ul>	<p>浮動域情報は、次のような情報です。</p> <ul style="list-style-type: none"> <li>• 装置サイズ</li> <li>• マージン情報</li> </ul> <p>印刷浮動域の情報にも装置タイプが含まれます。</p> <p>次に、テキスト書式に使用される画面浮動域の例を示します。</p> <pre>screenFloatingArea {   screenSize=[lines,columns],   topMargin=nn,   bottomMargin=nn,   leftMargin=nn,   rightMargin=nn }</pre> <p>次に、印刷書式に使用される印刷浮動域の例を示します。</p> <pre>printFloatingArea {   deviceType=singleByte,   pageSize=[lines,columns],   topMargin=nn,   bottomMargin=nn,   leftMargin=nn,   rightMargin=nn }</pre> <p>注: screenSize または pageSize に対して指定できる浮動域仕様はただ 1 つです。</p>	<p>マイグレーション・ツールは、VAGen 装置タイプを使用して、浮動域仕様の対象が表示マップ (screenFloatingArea) またはプリンター・マップ (printFloatingArea) のどちらであるかを判別します。</p> <p>マイグレーション・ツールは、マージン情報を次のように計算します。</p> <ul style="list-style-type: none"> <li>• topMargin は VAGen floatingAreaStartingLine - 1 に設定されます。</li> <li>• bottomMargin は VAGen deviceRows - (floatingAreaStartingLine + floatingAreaLines) + 1 に設定されます。</li> <li>• leftMargin は VAGen floatingAreaStartingColumn - 1 に設定されます。</li> <li>• rightMargin は VAGen deviceColumns - (floatingAreaStartingColumn + floatingAreaColumns) + 1 に設定されます。</li> </ul> <p>deviceType の設定については、280 ページの表 89 を参照してください。</p>
<p>プリンター・タイプは、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• プリンター</li> <li>• DBCS プリンター</li> </ul>	<p>deviceType=singleByte   doubleByte</p> <p>注: deviceType プロパティは、印刷書式に対してのみ指定できます。</p>	<p>マイグレーション・ツールは、VAGen プリンター・タイプに基づいて EGL deviceType プロパティを設定します。</p> <p>マイグレーション・ツールは、DeviceTypeKind を使用して deviceType 値を常に修飾します (例: deviceType = DeviceTypeKind.doubleByte)。これにより、formGroup 内の書式にある変数フィールドとの名前の競合が回避されます。</p>

表 89. マップ・グループ — 装置の名前、タイプ、およびサイズ

VisualAge Generator 装置名	装置サイズ (行 x 列)	装置タイプ	マイグレーション・ツールに関する考慮事項
3643-2	6 x 40	ディスプレイ	この装置サイズは、COBOL 生成の場合はサポートされません。
3277-1	12 x 40	ディスプレイ	この装置サイズは、COBOL 生成の場合はサポートされません。
3643-4	16 x 64	ディスプレイ	この装置サイズは、COBOL 生成の場合はサポートされません。
3278-1、3278-1B、 ANY-1D	12 x 80	ディスプレイ	特別な考慮事項なし。
3278-2、3278-2B、 ANY-2D	24 x 80	ディスプレイ	特別な考慮事項なし。
3278-3、3278-3B、 ANY-3D	32 x 80	ディスプレイ	特別な考慮事項なし。
3278-4、3278-4B、 ANY-4D	43 x 80	ディスプレイ	特別な考慮事項なし。
3278-5、3278-5B、 ANY-5D	27 x 132	ディスプレイ	特別な考慮事項なし。
ANY-D (62x160 として 構成された 3290)	255 x 160	ディスプレイ	この装置サイズは、COBOL 生成の場合はサポートされません。
5550D	24 x 80	DBCS ディスプレ イ	特別な考慮事項なし。
3767、PRINT-B、 PRINTER	255 x 132	プリンター	printFloatingArea に対して、EGL deviceType=singleByte
5550P	255 x 158	DBCS プリンター	printFloatingArea に対して、EGL deviceType=doubleByte

## マップ

マップに関するセクションは、次の表で構成されています。

- マップ — 一般情報 (281 ページの表 90)
- 表示マップ — 一般構文、マップ・タイプ、およびプロパティ (282 ページの表 91)
- プリンター・マップ — 一般構文、マップ・タイプ、およびプロパティ (285 ページの表 92)
- マップの定数フィールドと変数フィールド — 一般情報 (286 ページの表 93)
- マップの定数フィールドと変数フィールド — 一般構文、データ型、長さ、小数部、および説明 (289 ページの表 94)
- マップの定数フィールドと変数フィールド — 属性 (292 ページの表 95)
- マップ変数フィールド — 一般編集 (294 ページの表 96)
- マップ変数フィールド — 数値編集 (296 ページの表 97)
- マップ変数フィールド — エラー・メッセージ (297 ページの表 98)



表 90. マップ — 一般情報

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>マップには次の 2 種類があります。</p> <ul style="list-style-type: none"> <li>表示マップ</li> <li>プリンター・マップ</li> </ul>	<p>書式には次の 2 種類があります。</p> <ul style="list-style-type: none"> <li>テキスト書式</li> <li>印刷書式</li> </ul>	<p>特別な考慮事項なし。</p>
<p>マップ名は、マップ・グループ名とマップ名からなります。</p>	<p>書式名には、formGroup 名は含まれません。</p> <p>formGroup 内で書式を定義（ネスト）できます。</p> <p>また、書式を formGroup の外部で定義することもできます。この場合、formGroup にはフォーム名を指定する use ステートメントと、フォームの位置指定をするパッケージをインポートする import ステートメントをインクルードする必要があります。この技法により、共通書式（例えば、ポップアップ・リストの書式）の定義を 1 つ作成すれば、さまざまな formGroup 内でその書式を使用できます。</p>	<p>マイグレーション・ツールは、マップをすべて書式にマイグレーションします。マイグレーション・ツールは、複数のマップ・グループ間で共通する同一のマップ定義を識別しません。</p> <p>単一ファイル・モードでマイグレーションを行う場合、マイグレーション・ツールは formGroup 内のそれぞれの書式ごとに use ステートメントを組み込みます。該当する formGroup 内で書式がネストされるように、書式を移動する必要があります。</p> <p>ステージ 1 から 3 のマイグレーションを使用する場合、マイグレーション・ツールはすべての書式を formGroup 内で自動的にネストします。</p>
<p>プログラムがマップ・グループを指定していれば、プログラムはマップ名を参照するだけでマップ・グループ内の任意のマップを使用できます。</p>	<p>使用する formGroup を示す use ステートメントがプログラムに組み込まれていれば、プログラムは書式名を参照するだけで formGroup 内の任意のマップを参照できます。</p>	<p>特別な考慮事項なし。</p>

表 91. 表示マップ — 一般構文、マップ・タイプ、およびプロパティ

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>表示マップには、次の情報を含めることができます。</p> <ul style="list-style-type: none"> <li>マップ・グループ名とマップ名</li> <li>マップ・プロパティ <ul style="list-style-type: none"> <li>一般プロパティ <ul style="list-style-type: none"> <li>ヘルプ・マップ名</li> <li>ヘルプ・キー</li> <li>バイパス・キー</li> <li>変数フィールドの大文字への変換</li> </ul> </li> <li>レイアウト・プロパティ <ul style="list-style-type: none"> <li>マップ・サイズ</li> <li>開始位置</li> <li>浮動マップ</li> </ul> </li> <li>装置 <ul style="list-style-type: none"> <li>タイプ (ディスプレイまたは印刷)</li> <li>サポートされる装置</li> </ul> </li> </ul> </li> <li>定数フィールド</li> <li>変数フィールド</li> <li>変数フィールドのフィールド編集順序</li> </ul>	<p>テキスト書式パーツには、次の情報を含めることができます。</p> <ul style="list-style-type: none"> <li>書式名</li> <li>書式タイプ</li> <li>書式プロパティ</li> <li>定数フィールド</li> <li>変数フィールド</li> <li>変数フィールドの検証順序</li> </ul> <p>マイグレーション・ツールによって作成されるテキスト書式のフォーマットは、例えば次のとおりです。</p> <pre>Form mapName type textForm { screenSize=[sizeList],   formSize=[24,80], position=[1,1],   helpForm="helpFormName",   helpKey=pf1,   validationBypassKeys=[pf3],   msgField="VAGen_EZMSG"} [ variableFields ] [ constantFields ] end // end mapName</pre>	<p>マイグレーション・ツールは、VAGen 装置タイプを使用して、マップが表示マップ (テキスト書式) またはプリンター・マップ (印刷書式) のどちらであるかを判別します。</p> <p>装置がディスプレイまたはプリンターのどちらであるのかを判別するためには、280 ページの表 89 を参照してください。</p>
ヘルプ・マップ名	helpForm	特別な考慮事項なし。
ヘルプ・キー	helpKey	特別な考慮事項なし。
バイパス・キー	validationBypassKeys	特別な考慮事項なし。
マップに対して最大 5 つのバイパス・キーを指定できます。	書式に対して最大 5 つの validationBypassKeys を指定できます。	

表 91. 表示マップ — 一般構文、マップ・タイプ、およびプロパティ (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
変数フィールドの大文字への変換	書式に対してはサポートされません。書式にある char または mbchar の変数フィールドそれぞれに対して、ユーザーが入力するデータが自動的に大文字に変換されるかどうかを指定する必要があります。	マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>変数フィールドの大文字への変換がマップ全体に対して指定されている場合、マイグレーション・ツールはすべての文字フィールドと混合フィールドに <code>upperCase=yes</code> を組み込みます。</li> <li>変数フィールドの大文字への変換がマップ全体に対して指定されていない場合、マイグレーション・ツールは、それぞれの文字フィールドまたは混合フィールドごとに指定された大文字変換情報を使用して、そのフィールドに <code>upperCase</code> プロパティを設定するかどうかを決定します。</li> </ul>
マップ・サイズ — 行と列	<code>formSize = [Lines, Columns]</code>	特別な考慮事項なし。
開始位置 - 行と列  マップが浮動マップである場合は、NEXT,SAME が必要です。	<code>position = [Line, Column ]</code>  位置情報が省略された場合、書式は浮動書式になります。	浮動マップが選択された場合、マイグレーション・ツールは位置情報を省略します。
浮動マップ	適用外。位置情報が省略された場合、書式は浮動書式になります。	浮動マップが選択された場合、マイグレーション・ツールは位置情報を省略します。
装置タイプ - ディスプレイまたは DBCS ディスプレイ	タイプ <code>textForm</code>	マイグレーション・ツールは、装置タイプ情報を使用して、マップをテキスト書式または印刷書式のどちらにマイグレーションするかを決定します。
サポートされる装置  注: サポートされる装置は、装置タイプ、行数、および列数を指示します。	<code>screenSizes = [[Lines, Columns], [Lines, Columns]]</code>  注: 書式に対してサポートしたいそれぞれの画面サイズごとに、 <code>[Lines, Columns]</code> の対を組み込んでください。	マイグレーション・ツールは、装置タイプ情報を使用して、対応する <code>screenSizes</code> プロパティを決定します。複数の VAGen 装置に同じ画面サイズが指定されている場合、マイグレーション・ツールは画面サイズを 1 回だけ組み込みます。  VAGen によってサポートされる装置の一部は、EGL の COBOL 生成の場合にはサポートされないため、特別な考慮事項が適用されます。詳しくは、76 ページの『マップ・グループ、マップ、および装置サイズ』を参照してください。
適用外。 VisualAge Generator の場合、メッセージ・フィールドの名前は常に EZEMSG です。	<code>msgField</code>  これは、EGL エラー・メッセージを格納するフィールドの名前です。	EZEMSG がマップ上のどこかに存在する場合、マイグレーション・ツールは <code>msgField</code> プロパティを設定します。

表 91. 表示マップ — 一般構文、マップ・タイプ、およびプロパティ (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
適用外。	別名	<p>EGL 予約語との競合、またはマップ名の先頭が # または @ シンボルであることが原因で、マップ名を変更する必要がある場合、マイグレーション・ツールは alias プロパティをインクルードします。また、プログラムのメインのマップ・グループにあるマップ名との競合が原因でマップの名前を変更する必要がある場合、マイグレーション・ツールはプログラムのヘルプ・マップ・グループ内のマップに対して alias プロパティを組み込みます。</p> <p>特別な考慮事項が適用されます。詳しくは、77 ページの『マップ名とヘルプ・マップ名』を参照してください。</p>

表 92. プリンター・マップ — 一般構文、マップ・タイプ、およびプロパティ

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>プリンター・マップには、次の情報を含めることができます。</p> <ul style="list-style-type: none"> <li>マップ・グループ名とマップ名</li> <li>マップ・プロパティ <ul style="list-style-type: none"> <li>一般プロパティ <ul style="list-style-type: none"> <li>ヘルプ・マップ名</li> <li>ヘルプ・キー</li> <li>バイパス・キー</li> <li>変数フィールドの大文字への変換</li> <li>SO/SI が占める位置</li> </ul> </li> <li>レイアウト・プロパティ <ul style="list-style-type: none"> <li>マップ・サイズ</li> <li>開始位置</li> <li>浮動マップ</li> </ul> </li> <li>装置 <ul style="list-style-type: none"> <li>タイプ (ディスプレイまたは印刷)</li> <li>サポートされる装置</li> </ul> </li> </ul> </li> <li>定数フィールド</li> <li>変数フィールド</li> <li>変数フィールドのフィールド編集順序</li> </ul>	<p>印刷書式には、次の情報を含めることができます。</p> <ul style="list-style-type: none"> <li>書式名</li> <li>書式プロパティ</li> <li>定数フィールド</li> <li>変数フィールド</li> </ul> <p>マイグレーション・ツールによって作成されるテキスト書式のフォーマットは、例えば次のとおりです。</p> <pre>Form mapName type printForm {formSize=[255,158], position=[1,1],  addSpaceForSOSI=yes } [ variableFields ] [ constantFields ] end // end mapName</pre>	<p>マイグレーション・ツールは、VAGen 装置タイプを使用して、マップが表示マップ (テキスト書式) またはプリンター・マップ (印刷書式) のどちらであるかを判別します。</p> <p>マイグレーション・ツールは、印刷書式の次のプロパティを常に省略します。</p> <ul style="list-style-type: none"> <li>一般プロパティ <ul style="list-style-type: none"> <li>ヘルプ・マップ名</li> <li>ヘルプ・キー</li> <li>バイパス・キー</li> <li>変数フィールドの大文字への変換</li> </ul> </li> <li>装置 <ul style="list-style-type: none"> <li>サポートされる装置</li> </ul> </li> <li>変数フィールドのフィールド編集順序</li> </ul> <p>装置がディスプレイまたはプリンターのどちらであるのかを判別するためには、280 ページの表 89 を参照してください。</p>
ヘルプ・マップ名	印刷書式には適用されません。	マイグレーション・ツールは、印刷書式のこのプロパティを省略します。
ヘルプ・キー	印刷書式には適用されません。	マイグレーション・ツールは、印刷書式のこのプロパティを省略します。
バイパス・キー	印刷書式には適用されません。	マイグレーション・ツールは、印刷書式のこのプロパティを省略します。
変数フィールドの大文字への変換	印刷書式には適用されません。	マイグレーション・ツールは、印刷書式のこのプロパティを省略します。
SO/SI が占める位置	addSpaceForSOSI	特別な考慮事項なし。
マップ・サイズ — 行と列	formSize = [Lines, Columns]	特別な考慮事項なし。
開始位置 - 行と列	position = [Line, Column]	浮動マップが選択された場合、マイグレーション・ツールは位置情報を省略します。
マップが浮動マップである場合は、NEXT,SAME が必要です。	位置情報が省略された場合、書式は浮動書式になります。	
浮動マップ	適用外。位置情報が省略された場合、書式は浮動書式になります。	浮動マップが選択された場合、マイグレーション・ツールは位置情報を省略します。

表 92. プリンター・マップ — 一般構文、マップ・タイプ、およびプロパティ (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
装置タイプ - プリンターまたは DBCS プリンター	type printForm	マイグレーション・ツールは、装置タイプ情報を使用して、マップをテキスト書式または印刷書式のどちらにマイグレーションするかを決定します。
サポートされる装置	印刷書式には適用されません。	マイグレーション・ツールは、印刷書式のこのプロパティを省略します。
適用外。 VisualAge Generator の場合、メッセージ・フィールドの名前は常に EZEMSG です。	msgField これは、EGL エラー・メッセージを格納するフィールドの名前です。	EZEMSG がマップ上のどこかに存在する場合、マイグレーション・ツールは msgField プロパティを設定します。
適用外。	別名	EGL 予約語との競合、またはマップ名の先頭が # または @ シンボルであることが原因で、マップ名を変更する必要がある場合、マイグレーション・ツールは alias プロパティをインクルードします。また、プログラムのメインのマップ・グループにあるマップ名との競合が原因でマップの名前を変更する必要がある場合、マイグレーション・ツールはプログラムのヘルプ・マップ・グループ内のマップに対して alias プロパティを組み込みます。  特別な考慮事項が適用されます。詳しくは、77 ページの『マップ名とヘルプ・マップ名』を参照してください。

表 93. マップの定数フィールドと変数フィールド — 一般情報

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
マップ上の位置はすべて、次のいずれかとして記述する必要があります。 <ul style="list-style-type: none"> <li>変数フィールド</li> <li>定数フィールド</li> <li>定数フィールドまたは変数フィールドの先頭にある属性バイト</li> </ul>	書式上の位置は、すべて記述する必要はありません。デフォルト・プロパティ (noHighLight、normalIntensity、protect=skip、defaultColor、no outlining、および no cursor) をもつブランク定数は、指定する必要はありません。	マイグレーション・ツールは、デフォルト・プロパティをもつブランク定数を省略します。



表 93. マップの定数フィールドと変数フィールド — 一般情報 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>表示マップの定数フィールドには、実際には定数に適用しない属性を指定できます。例えば、次のようになります。</p> <ul style="list-style-type: none"> <li>• 無保護</li> <li>• 入力必須 (Input required)</li> <li>• 入力時充てん必須 (Require fill on input)</li> <li>• 数値属性</li> <li>• 変更データ・タグ</li> </ul>	<p>テキスト書式の定数フィールドには、定数に対して意味をなさないプロパティは指定できません。</p>	<p>サポートされないプロパティの場合、マイグレーション・ツールはテキスト書式の定数のプロパティを省略します。</p>
<p>プリンター・マップの定数フィールドには、実際にはプリンターに適用されない属性を指定できます。例えば、次のようになります。</p> <ul style="list-style-type: none"> <li>• 色</li> <li>• 輝度</li> <li>• 下線以外の強調表示</li> <li>• 保護</li> <li>• 初期カーソル・フィールド</li> <li>• ライト・ペン検出</li> </ul>	<p>印刷書式の定数フィールドには、定数に対して意味をなさないプロパティは指定できません。</p>	<p>サポートされないプロパティの場合、マイグレーション・ツールは印刷書式の定数のプロパティを省略します。</p>
<p>プリンター・マップの変数フィールドには、実際にはプリンターに適用されない属性を指定できます。例えば、次のようになります。</p> <ul style="list-style-type: none"> <li>• 色</li> <li>• 輝度</li> <li>• 下線以外の強調表示</li> <li>• 保護</li> <li>• 初期カーソル・フィールド</li> <li>• 入力必須 (Input required)</li> <li>• 入力時充てん必須 (Require fill on input)</li> <li>• 数値属性</li> <li>• 変更データ・タグ</li> <li>• ライト・ペン検出</li> </ul>	<p>印刷書式の変数フィールドには、印刷書式に対して意味をなさないプロパティは指定できません。</p>	<p>サポートされないプロパティの場合、マイグレーション・ツールは印刷書式の変数フィールドのプロパティを省略します。</p>

表 93. マップの定数フィールドと変数フィールド — 一般情報 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>プリンター・マップの変数フィールドには、実際にはプリンターに適用されない編集を指定できます。例えば、次のようになります。</p> <ul style="list-style-type: none"> <li>• 最小入力</li> <li>• 大文字変換</li> <li>• 16 進編集</li> <li>• 入力必須 (Input required)</li> <li>• 最小値</li> <li>• 最大値</li> <li>• 編集メッセージ</li> </ul>	<p>印刷書式の変数フィールドには、印刷書式に対して意味をなさないプロパティは指定できません。</p>	<p>サポートされないプロパティの場合、マイグレーション・ツールは印刷書式の変数フィールドのプロパティを省略します。</p>

表 94. マップの定数フィールドと変数フィールド — 一般構文、データ型、長さ、小数部、および説明

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>マップ上の変数には、次の情報があります。</p> <ul style="list-style-type: none"> <li>名前</li> <li>マップにドロップしたものに基づく次の情報: <ul style="list-style-type: none"> <li>データ型</li> <li>位置</li> </ul> </li> <li>基本情報: <ul style="list-style-type: none"> <li>説明</li> <li>初期値</li> <li>バイト単位の長さ</li> <li>配列指標</li> <li>数値編集</li> </ul> </li> <li>属性</li> <li>編集 (小数部の桁数など)</li> <li>エラー・メッセージ</li> </ul> <p><b>注:</b> 位置 は、属性バイトの位置です。バイト単位の長さは、フィールドの長さから属性バイトを除いたバイト数です。バイト単位の長さは、データ値の長さにも使用されます。</p>	<p>変数フィールドの情報には、次のものがあります。</p> <ul style="list-style-type: none"> <li>名前</li> <li>文字フィールドのタイプと文字数単位の長さ</li> <li>数値フィールドの型、精度、およびスケール</li> <li>位置</li> <li>バイト単位のフィールド長</li> <li>表示プロパティ</li> <li>フォーマット設定プロパティ</li> <li>検証プロパティ</li> <li>値</li> </ul> <p>一般に、次のことが当てはまります。</p> <ul style="list-style-type: none"> <li>VAGen の属性は、EGL の表示プロパティに対応します。</li> <li>VAGen の編集およびメッセージは、EGL のフォーマット設定プロパティ、または検証プロパティに対応します。</li> <li>ただし、VAGen の属性と編集の一部には、単一の EGL プロパティにマージされたものや、別のカテゴリーに移動したものがあります。</li> </ul> <p>EGL 変数フィールドの例を次に示します。</p> <pre>itemName dataType(lengthInformation) // description { position=[row,column],   fieldLen=length,   validationOrder=n,   [presentationProperties]   [formattingProperties]   [value="initialValue"]   [arrayInformation] }</pre> <p><b>注:</b> 位置 は、属性バイトの位置です。  <i>fieldLen</i> は、フィールドの長さから属性バイトを除いたバイト数です。  <i>dataType(lengthInformation)</i> に指定されるプリミティブ情報は、データ値の長さです。</p>	<p>マイグレーション・ツールは、EGL の <i>fieldLen</i> プロパティを VAGen のバイト単位の長さに設定します。ツールは、<i>dataType</i> の <i>lengthInformation</i> を次のように設定します。</p> <ul style="list-style-type: none"> <li><i>char</i>、<i>dbchar</i>、および <i>mbchar</i> のフィールドに対して、マイグレーション・ツールは <i>lengthInformation</i> をバイト数ではなく文字数に設定します。</li> <li>数値編集を指定している VAGen の <i>char</i> フィールドに対して、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>フィールドを EGL の <i>num</i> 型に変換します。</li> <li>精度を VAGen のバイト単位の長さに設定し、また VisualAge Generator のフィールドに対して小数部が指定されている場合は、精度を 1 だけ減らします。</li> <li>スケールを VisualAge Generator 内で指定されている小数部の桁数に設定します。</li> </ul> </li> </ul> <p>特別な考慮事項が適用されます。詳しくは、79 ページの『数値変数フィールド』を参照してください。</p>

表 94. マップの定数フィールドと変数フィールド — 一般構文、データ型、長さ、小数部、および説明 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>マップ上の定数には、次の情報があります。</p> <ul style="list-style-type: none"> <li>マップにドロップしたものに基づく次の情報: <ul style="list-style-type: none"> <li>データ型</li> <li>位置</li> </ul> </li> <li>基本情報: <ul style="list-style-type: none"> <li>初期値</li> <li>バイト単位の長さ</li> </ul> </li> <li>属性</li> </ul> <p>注: 位置 は、属性バイトの位置です。バイト単位の長さ は、フィールドの長さから属性バイトを除いたバイト数です。</p>	<p>定数フィールドの情報には、次のものがあります。</p> <ul style="list-style-type: none"> <li>位置</li> <li>フィールド長</li> <li>表示プロパティ</li> <li>値</li> </ul> <p>一般に、次のことが当てはまります。</p> <ul style="list-style-type: none"> <li>VAGen の属性は、EGL の表示プロパティに対応します。</li> <li>入力編集のみに適用される属性は、EGL 定数フィールドに対してはサポートされません。</li> </ul> <p>定数のデータ型は、<i>value</i> プロパティに基づいて決定されます。</p> <p>EGL 定数フィールドの例を次に示します。</p> <pre>{ position=[row,column],   fieldLen=length,   [presentationProperties]   [value="initialValue"] }</pre> <p>注: 位置 は、属性バイトの位置です。<i>fieldLen</i> は、フィールドの長さから属性バイトを除いたバイト数です。</p>	<p>マイグレーション・ツールは、EGL の <i>fieldLen</i> プロパティを VisualAge Generator の長さに設定します。</p>
<p>データ型:</p> <ul style="list-style-type: none"> <li>文字定数</li> <li>文字変数</li> <li>DBCS 定数</li> <li>DBCS 変数</li> <li>混合定数</li> <li>混合変数</li> <li>数値編集を選択した文字変数</li> </ul> <p>注: 型は、マップにドロップしたフィールドのタイプ、および「数値編集 (Numeric edit)」ボックスを選択したかどうかによって決まります。</p>	<p>EGL データ型:</p> <ul style="list-style-type: none"> <li>適用外</li> <li>char</li> <li>適用外</li> <li>dbchar</li> <li>適用外</li> <li>mbchar</li> <li>num</li> </ul>	<p>特別な考慮事項なし。</p>
<p>説明</p>	<p>適用外。</p>	<p>マイグレーション・ツールは、説明をコメントに変換し、データ型と長さの情報の後に配置します。</p>

表 94. マップの定数フィールドと変数フィールド — 一般構文、データ型、長さ、小数部、および説明 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
初期値	<p>value</p> <p>注:</p> <ul style="list-style-type: none"> <li>VisualAge Generator 互換モードでは、<i>value</i> プロパティは、値が割り当てられていないフィールドを画面に表示するときのみ使用されます。<i>value</i> プロパティを使用して、ストレージ内のフィールドの初期値が設定されることはありません。</li> <li>VisualAge Generator 互換モードが指定されていない場合、<i>value</i> プロパティはプログラムの開始時にプログラムのフィールドの初期値を指定します。</li> </ul>	特別な考慮事項なし。
長さ	<p>EGL 変数には次の情報が格納されます。</p> <ul style="list-style-type: none"> <li><i>length</i> (フィールド内の文字数または桁数)</li> <li><i>fieldLen</i> (マップ上でフィールドが占める、属性バイトを除いたスペース)</li> </ul>	マイグレーション・ツールは、VAGen の長さを使用して、EGL の <i>length</i> プロパティと EGL の <i>fieldLen</i> プロパティを両方とも設定します。数値フィールドに関しては、特別な考慮事項が適用されます。詳しくは、79 ページの『数値変数フィールド』を参照してください。
<p>配列指標</p> <p>注:</p> <ul style="list-style-type: none"> <li>配列サイズは、変数フィールドの最も大きい配列指標に基づいて決定されます。</li> <li>カーソル位置、色、強調表示、輝度、保護、カーソル位置など、配列要素の属性の一部はオーバーライドできます。</li> <li>配列要素の初期値をオーバーライドすることもできます。</li> </ul>	<pre>itemName datatype(lengthInfo) [ arraySize ] { properties for index 1,   this[n] { properties for index n } }</pre> <p>注:</p> <ul style="list-style-type: none"> <li>配列サイズは、データ型および長さの情報の直後に指定されます。</li> <li>カーソル位置、および色、強調表示、輝度、保護などの表示プロパティはオーバーライドできます。</li> <li><i>value</i> プロパティもオーバーライドできます。</li> <li>次のようにして、各要素の位置を指定できます。 <ul style="list-style-type: none"> <li><i>position=[row,column]</i> を使用して、各要素の明示的な位置を指定する。</li> <li>指標 1 に対して、追加プロパティ — <i>columns</i>、<i>linesBetweenRows</i>、<i>spacesBetweenColumns</i>、および <i>indexOrientation</i> を指定する。</li> </ul> </li> </ul>	マイグレーション・ツールは、配列の各要素の位置を常に明示的に設定します。

表 94. マップの定数フィールドと変数フィールド — 一般構文、データ型、長さ、小数部、および説明 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>フィールド編集順序</p> <p>注:</p> <ul style="list-style-type: none"> <li>フィールド編集順序は、「定義 (Define)」プルダウンから指定します。</li> <li>デフォルトのフィールド編集順序は、マップ上の変数フィールドの位置によって決まります (左から右、上から下の順)。</li> <li>システム共通プロダクトと VisualAge Generator の一部のバージョンは、フィールド編集順序を外部ソース形式で記録しません。</li> </ul>	<p>validationOrder</p> <p>注: デフォルトの validationOrder は、マップ上の変数フィールドの位置によって決まります (左から右、上から下の順)。</p>	<p>validationOrder がマップの外部ソース形式に含まれていない場合、マイグレーション・ツールは validationOrder を無視します。</p>

表 95. マップの定数フィールドと変数フィールド — 属性

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>輝度:</p> <ul style="list-style-type: none"> <li>通常 (Normal)</li> <li>低輝度 (Dark)</li> <li>高輝度 (Bright)</li> </ul>	<p>輝度:</p> <ul style="list-style-type: none"> <li>normalIntensity</li> <li>invisible</li> <li>bold</li> </ul> <p>(表示プロパティ)</p>	<p>特別な考慮事項なし。</p>
<p>強調表示:</p> <ul style="list-style-type: none"> <li>強調表示なし (No Highlight)</li> <li>明滅 (Blink)</li> <li>反転表示 (Reverse video)</li> <li>下線 (Underscore)</li> </ul>	<p>強調表示:</p> <ul style="list-style-type: none"> <li>noHighlight</li> <li>blink</li> <li>reverse</li> <li>underline</li> </ul> <p>(表示プロパティ)</p>	<p>特別な考慮事項なし。</p>
<p>保護:</p> <ul style="list-style-type: none"> <li>無保護</li> <li>保護</li> <li>自動スキップ</li> </ul>	<p>保護:</p> <ul style="list-style-type: none"> <li>no</li> <li>yes</li> <li>skip</li> </ul> <p>(表示プロパティ)</p>	<p>特別な考慮事項なし。</p>



表 95. マップの定数フィールドと変数フィールド — 属性 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
色: <ul style="list-style-type: none"> <li>モノクロ (Mono)</li> <li>青</li> <li>赤</li> <li>ピンク</li> <li>緑</li> <li>青緑色</li> <li>黄色</li> <li>白</li> </ul>	色: <ul style="list-style-type: none"> <li>defaultColor</li> <li>blue</li> <li>red</li> <li>magenta</li> <li>green</li> <li>cyan</li> <li>yellow</li> <li>white</li> </ul> (表示プロパティ)	特別な考慮事項なし。
初期カーソル・フィールド	cursor = yes   no  (書式フィールド・プロパティ)	特別な考慮事項なし。
入力必須 (Input required)	inputRequired (検証プロパティ)	<p>マイグレーション・ツールは、VAGen の「入力必須 (Input required)」属性と入力必須編集を次のようにマージします。</p> <ul style="list-style-type: none"> <li>「入力必須 (Input required)」属性または入力必須編集のどちらかが選択されている場合、マイグレーション・ツールは inputRequired を組み込みます。</li> <li>どちらも選択されていない場合、マイグレーション・ツールは inputRequired を省略します。</li> </ul>
入力時充てん必須 (Require fill on input)	fill (検証プロパティ)	特別な考慮事項なし。
数値属性  <b>注:</b> このプロパティは、CHA フィールド (「数値編集 (Numeric edit)」が選択されている CHA フィールドを含む) に対してサポートされています。	isDecimalDigit (検証プロパティ)  <b>注:</b> このプロパティは、char フィールドのみに対してサポートされます。	<p>数値属性が選択されている場合、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>char フィールドに isDecimalDigit を組み込みます。</li> <li>数値フィールドの isDecimalDigit を省略します。VAGen との互換性を保つために、EGL は数値フィールドのソフトウェア編集機能を備えています。</li> </ul> <p>詳しくは、81 ページの『マップ・フィールドと数値ハードウェア属性』を参照してください。</p>
変更データ・タグ	modified (表示プロパティ)	特別な考慮事項なし。
ライト・ペン検出	detectable (表示プロパティ)	特別な考慮事項なし。

表 95. マップの定数フィールドと変数フィールド — 属性 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
アウトライン: • 左 (left) • 右 (right) • 上 (over) • 下 (under) • ボックス (box)	outline: • 左 (left) • 右 (right) • top • bottom • ボックス (box) (表示プロパティ)	特別な考慮事項なし。

表 96. マップ変数フィールド — 一般編集

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
編集ルーチン	validatorFunction または validatorDataTable (検証プロパティ)	マイグレーション・ツールは次の処理を行います。 • マップ編集ルーチンが EZEC10 または EZEC11 ならば、validatorFunction プロパティを設定します。 • 編集ルーチンが関数ならば、validatorFunction プロパティを設定します。 • 編集ルーチンがテーブルならば、validatorDataTable プロパティを設定します。 注: マイグレーション時に編集ルーチンが使用できない場合は、特別な考慮事項が適用されます。詳細と起こりうる問題については、80 ページの『マップ変数フィールドと編集ルーチン』を参照してください。
位置調整 - 左そろえ   右そろえ   なし 注: マップ項目のデフォルトは、数値フィールドの場合は右そろえ、その他のフィールドの場合は左そろえです。	align = left   right   none (フォーマット設定プロパティ) 注: 書式フィールドのデフォルトは、数値フィールドの場合は右そろえ、その他のフィールドの場合は左そろえです。	特別な考慮事項なし。

表 96. マップ変数フィールド — 一般編集 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>日付編集マスク</p> <p>このオプションの値は、次のとおりです。</p> <ul style="list-style-type: none"> <li>• SYSGREGRN</li> <li>• SYSJULIAN</li> <li>• dateEditPattern</li> </ul>	<p>dateFormat = value</p> <p>このオプションの値は、次のとおりです。</p> <ul style="list-style-type: none"> <li>• systemGregorianCalendar</li> <li>• systemJulianDateFormat</li> <li>• "dateEditPattern"</li> </ul> <p>(フォーマット設定プロパティ)</p> <p>注: dateEditPattern の中で、マイグレーション・ツールは次の EGL 表記への変換を行います。</p> <ul style="list-style-type: none"> <li>• 年を示す yy または yyyy。</li> <li>• 月を示す MM。</li> <li>• 日を示す dd。</li> <li>• 年の通算日を示す DDD。</li> </ul>	<p>特別な考慮事項なし。</p>
<p>最小入力</p>	<p>minimumInput (検証プロパティ)</p>	<p>特別な考慮事項なし。</p>
<p>充てん文字</p> <p>注: 文字、DBCS、または混合の各フィールドの場合、マップに使用される項目のデフォルト充てん文字は NULL です。数値フィールドの場合はブランク、16 進フィールドの場合は 0 です。</p>	<p>fillCharacter (フォーマット設定プロパティ)</p> <p>注:</p> <ul style="list-style-type: none"> <li>• char, dbchar, または mbchar の各フィールドの場合、マップに使用される項目のデフォルト充てん文字は NULL です。数値フィールドの場合はブランク、16 進フィールドの場合は 0 です。</li> <li>• strLib.nullFill は、NULL 充てん文字を表す EGL 定数です。あるいは、"" (2 つの連続する引用符) を使用します。</li> </ul>	<p>特別な考慮事項なし。</p>
<p>大文字変換</p>	<p>upperCase (フォーマット設定プロパティ)</p>	<p>マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• 変数フィールドの大文字への変換がマップ全体に対して指定されている場合、マイグレーション・ツールはすべての文字フィールドと混合フィールドに upperCase=yes を組み込みます。</li> <li>• 変数フィールドの大文字への変換がマップ全体に対して指定されていない場合、マイグレーション・ツールは、それぞれの文字フィールドまたは混合フィールドごとに指定された大文字変換情報を使用して、そのフィールドに upperCase プロパティを設定するかどうかを決定します。</li> </ul>

表 96. マップ変数フィールド — 一般編集 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
16 進編集	isHexEdit (検証プロパティ)	特別な考慮事項なし。
入力必須 (Input required)	inputRequired (検証プロパティ)	<p>マイグレーション・ツールは、VAGen の「入力必須 (Input required)」属性と入力必須編集を次のようにマージします。</p> <ul style="list-style-type: none"> <li>「入力必須 (Input required)」属性または入力必須編集のどちらかが選択されている場合、マイグレーション・ツールは inputRequired を組み込みます。</li> <li>どちらも選択されていない場合、マイグレーション・ツールは inputRequired を省略します。</li> </ul>
SO/SI スペースの検査	needsSOSI (検証プロパティ)	特別な考慮事項なし。

表 97. マップ変数フィールド — 数値編集

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>最小値と最大値</p> <p>注: 最小値または最大値のどちらかを指定する場合は、両方を指定する必要があります。</p>	<p>validValues = [[minimumValue, maximumValue]]</p> <p>(検証プロパティ)</p> <p>注: 複数の値の対、および単一値を validValues プロパティにリストできます。</p>	<p>マイグレーション・ツールは、最小値と最大値を EGL の validValues プロパティに結合します。</p>
<p>符号:</p> <ul style="list-style-type: none"> <li>なし</li> <li>先頭</li> <li>末尾</li> </ul> <p>注: デフォルトは「なし」です。</p>	<p>sign:</p> <ul style="list-style-type: none"> <li>none</li> <li>leading</li> <li>trailing</li> </ul> <p>(検証プロパティ)</p> <p>注: デフォルトは「なし」です。</p>	特別な考慮事項なし。
通貨	<p>currency = yes   no</p> <p>currencySymbol = "symbol"</p> <p>(フォーマット設定プロパティ)</p> <p>注: currency = yes で、ただし currencySymbol が指定されていない場合、実行時に実際に使用される通貨記号は、VisualAge Generator の場合と同じように設定されます。</p>	<p>マイグレーション・ツールは、currency を yes または no のどちらかに必ず設定します。ツールが書式変数フィールドに対して currencySymbol="symbol" を設定することはありません。これは、VisualAge Generator に同等な情報が存在しないからです。</p>
セパレーター	numericSeparator (フォーマット設定プロパティ)	特別な考慮事項なし。
ゼロ編集	zeroFormat (フォーマット設定プロパティ)	特別な考慮事項なし。

表 98. マップ変数フィールド — エラー・メッセージ

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
編集ルーチン	validatorFunctionMsgKey OR validatorDataTableMsgKey (検証プロパティ)	<p>マイグレーション・ツールは、編集ルーチン・メッセージを次のようにマイグレーションします。</p> <ul style="list-style-type: none"> <li>編集ルーチンが EZEC10 または EZEC11 ならば、 validatorFunctionMsgKey を設定します。</li> <li>編集ルーチンがテーブルならば、 validatorDataTableMsgKey を設定します。</li> <li>編集ルーチンが関数ならば、この状態では VAGen 内でメッセージは使用されないため、編集ルーチン・メッセージをマイグレーションしません。</li> </ul> <p>詳細と起こりうる問題については、80 ページの『マップ変数フィールドと編集ルーチン』を参照してください。</p>
最小入力	minimumInputMsgKey (検証プロパティ)	特別な考慮事項なし。
入力必須 (Input required)	inputRequiredMsgKey (検証プロパティ)	特別な考慮事項なし。
データ型	typeChkMsgKey (検証プロパティ)	特別な考慮事項なし。
数値範囲	validValuesMsgKey (検証プロパティ)	特別な考慮事項なし。

## プログラム

プログラムに関するセクションは、次の表で構成されています。

- プログラム - 一般構文、プログラム・タイプ、呼び出し先パラメーター、およびプロローグ (298 ページの表 99)
- プログラム - プログラム仕様、プロパティ、テーブル、および追加レコード・リスト (300 ページの表 100)
- プログラム - main 関数とフロー・ステートメント (304 ページの表 101)

表 99. プログラム — 一般構文、プログラム・タイプ、呼び出し先パラメーター、およびプロローグ

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>プログラム・パーツ:</p> <ul style="list-style-type: none"> <li>• programName</li> <li>• プログラム・タイプ</li> <li>• 仕様 (プログラム・タイプによって異なる): <ul style="list-style-type: none"> <li>– 作業用ストレージ・レコード</li> <li>– PSB</li> <li>– 先頭マップ</li> <li>– 先頭 UI レコード</li> <li>– マップ・グループ</li> <li>– ヘルプ・マップ・グループ</li> </ul> </li> <li>• テーブルおよび追加レコード</li> <li>• 呼び出し先パラメーター</li> <li>• プロローグ</li> <li>• プロパティ (プログラム・タイプによって異なる)</li> <li>• 構造ダイアグラム <ul style="list-style-type: none"> <li>– main 関数</li> <li>– フロー・ステートメント (構造ダイアグラム内では非表示、ただし任意の main 関数に対して指定可能)</li> </ul> </li> </ul>	<p>EGL 構文の例:</p> <pre> /** *** Program=programName ** prolog ** ***** Program programName type eglProgramType //vagenProgramType [ ( calledParameters ) ] { [alias= "originalProgramName"] includeReferencedFunctions =yes, allowUnqualifiedItemReferences =yes, localSQLScope=yes, throwNrfEofExceptions=yes, handleHardIOErrors=no [ propertiesBasedOnType ] }  [ dataDeclarations ] [ useDeclarations ] function main ( ) { functionLabel: functionName( ) ; [{functionFlowStatements}]} end // end main end // end programName </pre>	<p>プログラム名が EGL 予約語リストと競合していても、マイグレーション・ツールはプログラムの名前を変更しません。マイグレーション・ツールは alias プロパティを設定しません。プログラムの名前を変更する必要がある場合は、alias プロパティを使用して、VAGen プログラムのオリジナルの名前を指定してください。86 ページの『プログラム名と予約語』を参照してください。</p> <p>マイグレーション・ツールは、プログラム定義にコメントとして VAGen プログラム・タイプを組み込みます。</p> <p>マイグレーション・ツールは、テーブルと追加レコードのリストを次のようにマイグレーションします。</p> <ul style="list-style-type: none"> <li>• レコードを dataDeclarations にマイグレーションします。</li> <li>• テーブルを useDeclarations にマイグレーションします。</li> </ul> <p>マイグレーション・ツールは、VAGen の振る舞いを保持するために常に次のプロパティを組み込みます。</p> <ul style="list-style-type: none"> <li>• includeReferencedFunctions</li> <li>• allowUnqualifiedItemReferences</li> <li>• localSQLScope</li> <li>• throwNrfEofExceptions</li> <li>• handleHardIOErrors</li> </ul>
<p>プログラム・タイプ:</p> <ul style="list-style-type: none"> <li>• メインのトランザクション</li> <li>• 呼び出し先トランザクション</li> <li>• メインのバッチ</li> <li>• 呼び出し先バッチ</li> <li>• Web トランザクション</li> </ul> <p>注: 詳しくは、この後の行にある『メインのトランザクションの実行モード値』を参照してください。</p>	<p>EGL プログラム・タイプ:</p> <ul style="list-style-type: none"> <li>• textUIProgram</li> <li>• textUIProgram</li> <li>• basicProgram</li> <li>• basicProgram</li> <li>• VGWebTransaction</li> </ul>	<p>マイグレーション・ツールは、プログラム定義にコメントとして VAGen プログラム・タイプを組み込みます。セグメンテーション値と EGL プロパティの対応については、300 ページの表 100 を参照してください。</p>



表 99. プログラム — 一般構文、プログラム・タイプ、呼び出し先パラメーター、およびプロローグ (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>呼び出し先パラメーター</p> <p>注:</p> <ul style="list-style-type: none"> <li>呼び出し先パラメーターは、専用のウィンドウに入力されます。</li> <li>パラメーター・タイプは、パラメーターが項目、レコード、またはマップのどれであるかを示します。</li> <li>パラメーター名は、常に別の VAGen パーツの名前です。EGL の型定義またはプリミティブ型に相当するものではありません。</li> <li>次の 2 つの特殊機能語がパラメーターとしてサポートされています。 <ul style="list-style-type: none"> <li>EZEDLPSB</li> <li>EZEDLPCB[n] (ここで <i>n</i> は数値リテラルです)</li> </ul> </li> </ul> <p>詳しくは、この表の次の行を参照してください。</p>	<p>EGL 呼び出し先パラメーターの例:</p> <pre>( parameterName typeInfo { , parameterName typeInfo } )</pre> <p>注:</p> <ul style="list-style-type: none"> <li>パラメーターはコンマで区切る必要があります。</li> <li>パラメーターは、dataItem、レコード、または書式のいずれかです。VAGen パラメーター型に直接対応するものではありません。</li> <li>EGL typeInfo は、次のいずれかです。 <ul style="list-style-type: none"> <li>dataItem のプリミティブ型</li> <li>dataItem、レコード、または書式の型定義</li> </ul> </li> </ul>	<p>マイグレーション・ツールは、オリジナルの VAGen パラメーターの型をコメントとして組み込みます。</p> <p>マイグレーション構文設定「選択済み共用データ項目をプリミティブ項目定義に変換」を選択した場合に、データ項目パーツが使用可能ならば、共用項目はマイグレーション・ツールによって EGL パラメーターに変換され、このパラメーターはデータ項目パーツに対して指定された型、長さ、および小数部に基づいたプリミティブ定義を使用して宣言されます。型、長さ、および小数部の情報のマイグレーションは、249 ページの表 67 の説明と同じです。</p> <p>マイグレーション構文設定「選択済み共用データ項目をプリミティブ項目定義に変換」を選択しなかった場合、またはデータ項目パーツが使用不可の場合は、共用項目はマイグレーション・ツールによって EGL パラメーターに変換され、この変数は型定義を使用して宣言されます。マイグレーション用の型定義は、常に項目名と同じです。</p> <p>特別な考慮事項が適用されます。詳細と起こりうる問題については、65 ページの『再定義レコード』を参照してください。</p>
<p>呼び出し先パラメーター: EZEDLPSB</p>	<p>PSB をパスするための EGL 呼び出し先パラメーターの例:</p> <pre>parameter リスト: ( psbData psbDataRecord ) program プロパティ: { @DLI { psbParm = "psbData" }}</pre>	<p>特別な考慮事項なし。</p>

表 99. プログラム — 一般構文、プログラム・タイプ、呼び出し先パラメーター、およびプロローグ (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>呼び出し先パラメーター: EZEDLPCB[n] (ここで <i>n</i> は数値リテラルです)</p>	<p>PCB をパスするための EGL 呼び出し先パラメーターの例:</p> <p>parameter リスト: ( <i>pcbName</i>     <i>pcbType_PCBRecord</i> )</p> <p>program プロパティ: { @DLI {     <i>pcbParms</i> = [ <i>pcbList</i> ] } }</p> <p>注:</p> <ul style="list-style-type: none"> <li>• <i>pcbList</i> は、PCB パラメーターの名前をプログラムの PSB 内の対応する位置に突き合わせるために使用されます。</li> <li>• <i>pcbType</i> の値は次のとおりです。 <ul style="list-style-type: none"> <li>– IO</li> <li>– ALT</li> <li>– DB</li> <li>– GSAM</li> </ul> </li> </ul>	<p>マイグレーション・ツールは、常に次のフォームで <i>pcbName</i> を使用します。 <i>pcbn</i> (ここで <i>n</i> は、VAGen 呼び出し先パラメーター・リストで 사용되는ものと同じ数値リテラルです。)</p> <p>プログラムの PSB パーツが使用可能な場合、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>• <i>pcbType</i> 情報をインクルードします。</li> <li>• それぞれの PCB パラメーターを、PSB パーツ内の対応する PCB に関連付けるための <i>pcbList</i> 情報をインクルードします。</li> </ul> <p>プログラムの PSB パーツが使用不可の場合、特別な考慮事項が適用されます。詳しくは、89 ページの『呼び出しパラメーター・リストに EZEDLPCB が含まれるプログラム』を参照してください。</p>
プロローグ	適用外。	マイグレーション・ツールは、プロローグをコメントに変換し、プログラム定義の前に配置します。

表 100. プログラム — プログラム仕様、プロパティ、テーブル、および追加レコード・リスト

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>一般情報:</p> <ul style="list-style-type: none"> <li>• VAGen プロパティと仕様の一部は、EGL プロパティ、データ宣言、または使用宣言にマイグレーションされます。</li> </ul>	<p>一般情報:</p> <ul style="list-style-type: none"> <li>• 後続の行によって、対応する EGL 言語エレメントがプログラム・プロパティ、データ宣言、または使用宣言のどれであるかが示されます。</li> </ul>	特別な考慮事項なし。
<p>メインのトランザクションの実行モード値:</p> <ul style="list-style-type: none"> <li>• 非セグメント化 (Nonsegmented)</li> <li>• セグメント化 (Segmented)</li> <li>• 単一セグメント (Single segment)</li> </ul> <p>注: 呼び出し先トランザクションは、常に非セグメント化モードで実行されます。</p>	<p>セグメント化 — 値:</p> <ul style="list-style-type: none"> <li>• segmented = no</li> <li>• segmented = yes</li> <li>• segmented = yes</li> </ul> <p>(プログラム・プロパティ)</p> <p>注: segmented プロパティは、呼び出し先プログラムに対しては指定されません。</p>	<p>セグメント化情報が外部ソース形式ファイルに含まれていない場合、マイグレーション・ツールはデフォルトの <i>segmented = no</i> を指定します。</p>

表 100. プログラム — プログラム仕様、プロパティ、テーブル、および追加レコード・リスト (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>作業用ストレージ・レコード (仕様)</p> <ul style="list-style-type: none"> <li>作業用ストレージ・レコードは、メインプログラムと呼び出し先プログラムの両方に対して指定できます。このレコードは、プログラムの 1 次作業用ストレージ・レコードとも呼ばれることもあります。</li> <li>1 次作業用ストレージ・レコードは、常に初期化されます。</li> </ul>	<p>inputRecord (プログラム・プロパティ)</p> <ul style="list-style-type: none"> <li>inputRecord プロパティは、メインプログラムに対してのみ指定できます。</li> <li>inputRecord は常に初期化されます。</li> <li>データ宣言も必要です。</li> </ul>	<p>マイグレーション・ツールは、1 次作業用ストレージ・レコードをメインプログラムの inputRecord プロパティに変換します。</p> <p>またマイグレーション・ツールは、メインプログラムと呼び出し先プログラムの両方に 1 次作業用ストレージ・レコードのデータ宣言を組み込みます。ツールは、呼び出し先プログラムのデータ宣言に <i>initialized = yes</i> プロパティを組み込みます。</p> <p>1 次作業用ストレージ・レコードにレベル 77 項目が含まれている場合、マイグレーション・ツールはレベル 77 レコードのデータ宣言ステートメントを組み込みます。</p> <p>詳細と起こりうる問題については、66 ページの『レコード内のレベル 77 項目』を参照してください。</p>
<p>PSB (仕様)</p> <p>注: VisualAge Generator では、PSB はパーツ・タイプです。</p>	<p>EGL は、PSB を指定するためにプログラム・プロパティとレコード宣言の両方を使用します。</p> <p>program プロパティ:</p> <pre>{ @DLI { psb = "psbName",   callInterface =     DLICallInterfaceKind.CBLTDLI,   handleHardDLIErrors = yes }}</pre> <p>program record declaration:</p> <pre>psbName psbPartName ;</pre> <p>注: EGL では、PSB はレコード・パーツのサブタイプです。</p>	<p>マイグレーション・ツールは、<i>psb</i> を <i>psb</i> プロパティの値として常時使用します。これにより、PSB で変数を参照する必要がある関数が、<i>psb</i> で変数を修飾できるようになります。</p> <p>マイグレーション・ツールは、IMS または DL/I プログラムが VAGen の振る舞いを保存できるように、常に次のプロパティをインクルードします。</p> <ul style="list-style-type: none"> <li>callInterface</li> <li>handleHardDLIErrors</li> </ul> <p>マイグレーション・ツールは、<i>psb</i> という変数の宣言をインクルードし、必要な名前変更を行った後で、<i>psbPartName</i> を VAGen PSB パーツの名前として指定します。</p>
先頭マップ (仕様)	inputForm (プログラム・プロパティ)	特別な考慮事項なし。
先頭 UI レコード (仕様)	inputUIRecord (プログラム・プロパティ)	特別な考慮事項なし。
マップ・グループ (仕様)	use formGroup (使用宣言)	特別な考慮事項なし。
ヘルプ・マップ・グループ (仕様)	use formGroup { helpGroup=yes } (使用宣言)	特別な考慮事項なし。

表 100. プログラム — プログラム仕様、プロパティ、テーブル、および追加レコード・リスト (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
メッセージ・テーブル接頭部 (プログラム・プロパティ)	msgTablePrefix (プログラム・プロパティ)	特別な考慮事項なし。
暗黙データ項目の許可 (プログラム・プロパティ)	サポートなし。	マイグレーション・ツールは、暗黙定義を自動的に作成しません。詳細と起こりうる問題については、86 ページの『プログラム内の暗黙データ項目』を参照してください。
<p>キー割り当て:</p> <ul style="list-style-type: none"> <li>ヘルプ・キー (1 つのキー)</li> <li>バイパス・キー (5 つまでのキー)</li> <li>F1-12 = F13-24</li> </ul> <p>(プログラム・プロパティ)</p> <p>注: キー割り当ては、プログラムに対して 1 回指定され、マップ・グループとヘルプ・マップ・グループの両方に適用されます。</p>	<p>EGL のキー割り当ての例:</p> <pre>use formGroup { [ helpGroup = yes, ]   helpKey = pfNumber,   validationBypassKeys =     [ pfNumberList ],   pfKeyEquate = yes   no } ;</pre> <p>(プログラムの formGroup とヘルプ formGroup の使用宣言プロパティ)</p> <p>注:</p> <ul style="list-style-type: none"> <li>validationBypassKeys リストの値は、コンマで区切る必要があります。</li> <li>validationBypassKeys プロパティは、プログラムのヘルプ formGroup に対しては指定されません。</li> </ul>	<p>マイグレーション・ツールは、formGroup とヘルプ formGroup の両方に対する使用宣言 ステートメントに、キー割り当て情報に相当する EGL ステートメントを組み込みます。マイグレーション・ツールは、ヘルプ formGroup の使用宣言から validationBypassKeys プロパティを省略します。</p>

表 100. プログラム — プログラム仕様、プロパティ、テーブル、および追加レコード・リスト (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>テーブルおよび追加レコード:</p> <ul style="list-style-type: none"> <li>レコード</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>再定義情報は、プログラムではなく VAGen 再定義レコードに格納されます。</li> <li>入出力オブジェクトとして使用されるレコードは、テーブルおよび追加レコードのリストには組み込まれません。</li> </ul>	<p>EGL 追加レコードの例:</p> <pre>recordName recordName   [{ redefines = "otherRecord"}];</pre> <p>(データ宣言)</p> <p>注:</p> <ul style="list-style-type: none"> <li>そのレコードが、別のレコードと同じ物理ストレージに対して異なるレコード・レイアウトを与えている場合は、プログラムのデータ宣言に再定義プロパティを指定する必要があります。</li> <li>データ宣言は、プログラム内で使用されるすべてのレコード (入出力レコードを含む) に必要です。</li> </ul>	<p>マイグレーション・ツールは、型定義と同じレコード名を常に使用します。</p> <p>VAGen レコードが再定義レコードとしてプログラム内で使用されている場合、マイグレーション・ツールは、データ宣言ステートメントに再定義プロパティを組み込みます。詳細と起こりうる問題については、65 ページの『再定義レコード』を参照してください。</p> <p>さらにマイグレーション・ツールは、プログラムによって入出力オブジェクトとして使用されているすべてのレコードにデータ宣言を組み込みます。</p> <p>マイグレーション・ツールは、プログラムによって入出力オブジェクトとして使用される MQ メッセージ・レコードの属性として指定されたレコードに、データ宣言を組み込みます。</p> <p>プログラムが PSB を指定した場合、マイグレーション・ツールは、PSB パーツで参照されたすべての DL/I セグメントに関して、データ宣言をインクルードします。</p> <p>Web トランザクション・プログラムの場合、マイグレーション・ツールは、CONVERSE I/O ステートメントまたは XFER ステートメントで参照された UI レコードに関して、データ宣言をインクルードします。</p>

表 100. プログラム — プログラム仕様、プロパティ、テーブル、および追加レコード・リスト (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>テーブルおよび追加レコード:</p> <ul style="list-style-type: none"> <li>• テーブル</li> <li>• それぞれのテーブルごとに、Keep After Use を指定できます。</li> </ul>	<p>EGL の 使用 宣言の例:</p> <pre>use tableName   [{deleteAfterUse = yes}]; (use declaration)</pre>	<p>マイグレーション・ツールは、テーブルおよび追加レコードのリストにあるテーブルを使用 宣言に変換します。</p> <p><i>DeleteAfterUse</i> は、VAGen の Keep After Use と反対の意味を持ちます。マイグレーション・ツールは、yes と no を反転します。</p> <p>VAGen マイグレーション設定「テーブルの <i>deleteAfterUse</i> を組み込まない」を選択すると、マイグレーション・ツールは自動的に <i>deleteAfterUse</i> プロパティを省略し、影響を受けるプログラムおよびテーブルに関して警告メッセージを出します。</p>

表 101. プログラム — main 関数とフロー・ステートメント

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>VAGen プログラムは、VAGen 構造ダイアグラムの最上位関数として main 関数を指定します。その他の関数はすべて、構造ダイアグラムを展開したときにのみ表示されます。</p> <p>それぞれの main 関数には、フロー・ステートメントが含まれていることがあります。これらのステートメントは構造ダイアグラムには表示されませんが、ダイアグラムからアクセス可能です。</p>	<p>EGL プログラムは、1 つの main 関数のみ指定します。この関数の名前は、常に <i>main</i> です。</p> <p>フロー・ステートメントは存在しません。</p> <p>プログラムの main 関数の構文は、例えば次のとおりです。</p> <pre>function main ( ) { functionLabel:   functionName ( ) ; [ { functionFlowStatements } ] } end // end main</pre>	<p>マイグレーション・ツールは、EGL の main 関数を作成します。ツールは、VAGen のそれぞれの main 関数ごとに、main 関数に次のものを組み込みます。</p> <ul style="list-style-type: none"> <li>• <i>functionLabel</i>。これにより、VAGen の main 関数を EGL の <i>exit stack functionLabel</i> ステートメント内で参照できます。ツールは、<i>functionLabel</i> を <i>functionName</i> に常に設定します。</li> <li>• VAGen の main <i>functionName</i> を呼び出すための関数呼び出しステートメント。</li> <li>• VAGen の main 関数のフロー・ステートメント (存在する場合)。</li> </ul> <p>フロー・ステートメントのマイグレーションについて詳しくは、以下を参照してください。</p> <ul style="list-style-type: none"> <li>• 326 ページの『ステートメント』</li> <li>• 342 ページの『EZE ワード』</li> <li>• 353 ページの『サービス・ルーチン』</li> </ul>



---

## 関数

次の表は、VAGen 演算部を EGL 演算部と比較したもので、マイグレーション・ツールによる変換の処理方法を示しています。

関数に関するセクションは、次の表で構成されています。

- 関数 - 一般構文、説明、パラメーター、戻り値、およびローカル・ストレージ (306 ページの表 102)
- 関数 - EXECUTE 入出力オプション (308 ページの表 103)
- 関数 - マップと UI レコードの入出力オプション (309 ページの表 104)
- 関数 - ファイルまたはデータベースの入出力 — 一般情報と入出力エラー・ルーチン (309 ページの表 105)
- 関数 - シリアル・レコード、索引付きレコード、相対レコード、およびメッセージ・キュー・レコードの入出力オプション (310 ページの表 106)
- 関数 - Execution Time Statement Build を使用しないデフォルト (未変更) の SQL ステートメントの入出力オプション (311 ページの表 107)
- 関数 - Execution Time Statement Build を使用しない変更済み SQL ステートメントの入出力オプション (314 ページの表 108)
- 関数 - Execution Time Statement Build を使用する SQL ステートメントの入出力オプション (318 ページの表 109)
- 関数 - デフォルト (未変更) の DL/I ステートメントの入出力オプション (320 ページの表 110)
- 関数 - 変更済み DL/I ステートメントのセグメント検索索引数 (322 ページの表 111)

表 102. 関数 — 一般構文、説明、パラメーター、戻り値、およびローカル・ストレージ

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>演算部には、次の情報を含めることができます。</p> <ul style="list-style-type: none"> <li>関数名</li> <li>入出力オプション</li> <li>入出力オブジェクト</li> <li>プロパティ: <ul style="list-style-type: none"> <li>エラー・ルーチン</li> <li>説明</li> </ul> </li> <li>関数からの戻り値</li> <li>関数仮パラメーター</li> <li>関数ローカル・ストレージ</li> <li>SQL ステートメント</li> <li>DL/I 呼び出し</li> <li>入出力オプションの前のステートメント</li> <li>入出力オプションの後のステートメント</li> </ul>	<p>演算部には、次の情報を含めることができます。</p> <ul style="list-style-type: none"> <li>functionName</li> <li>functionParameterList</li> <li>returnItemType</li> <li>dataDeclarations</li> <li>入出力ステートメントの前のステートメント</li> <li>入出力ステートメント</li> <li>入出力ステートメントの後のステートメント</li> </ul> <p>マイグレーション・ツールによって作成される関数の形式は、例えば次のとおりです。</p> <pre>// Description Function functionName (functionParamterList) [ returns( returnItemType ) ] [ dataDeclarations ] [ beforeStatements ] [ I/O Statement ] [ afterStatements ] end // end functionName</pre> <p>注:</p> <ul style="list-style-type: none"> <li>EGL の入出力ステートメントの作成には、VAGen の入出力オプション、入出力オブジェクト、エラー・ルーチン、SQL ステートメント、および DL/I 呼び出しが使用されます。</li> </ul>	<p>説明、関数からの戻り値、関数仮パラメーター、および関数ローカル・ストレージをマイグレーション・ツールが処理する方法については、この表の後続の行を参照してください。</p> <p>前後のステートメントのマイグレーションについて詳しくは、以下を参照してください。</p> <ul style="list-style-type: none"> <li>ステートメントについては、326 ページの『ステートメント』を参照。</li> <li>EZE ワードについては、342 ページの『EZE ワード』を参照。</li> <li>サービス・ルーチンについては、353 ページの『サービス・ルーチン』を参照。</li> </ul> <p>入出力オプションとエラー・ルーチンについて詳しくは、以下を参照してください。</p> <ul style="list-style-type: none"> <li>EXECUTE 入出力オプションについては、308 ページの表 103 を参照。</li> <li>マップと UI レコードの入出力オプションについては、309 ページの表 104 を参照。</li> <li>ファイルおよびデータベース入出力の一般情報については、309 ページの表 105 を参照。</li> <li>シリアル・レコード、索引付きレコード、相対レコード、およびメッセージ・キュー・レコードの入出力オプションについては、310 ページの表 106 を参照。</li> </ul> <p>SQL ステートメントについて詳しくは、このセクションにある次の表を参照してください。</p> <ul style="list-style-type: none"> <li>未変更の入出力ステートメントの入出力オプションについては、311 ページの表 107 を参照。</li> <li>Execution Time Statement Build を使用しない変更済みの SQL ステートメントの入出力オプションについては、314 ページの表 108 を参照。</li> <li>Execution Time Statement Build を使用する SQL ステートメントの入出力オプションについては、318 ページの表 109 を参照。</li> </ul> <p>DL/I 呼び出しについて詳しくは、このセクションにある次の表を参照してください。</p> <ul style="list-style-type: none"> <li>デフォルト (未変更) の DL/I ステートメントの入出力オプションについては、320 ページの表 110 を参照。</li> <li>変更済み DL/I ステートメントのセグメント検索索引数については、322 ページの表 111 を参照。</li> </ul>
説明	適用外	マイグレーション・ツールは、関数の説明をコメントに変換し、関数定義の前に配置します。

表 102. 関数 — 一般構文、説明、パラメーター、戻り値、およびローカル・ストレージ (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>関数仮パラメーター:</p> <ul style="list-style-type: none"> <li>関数仮パラメーターは、専用のウィンドウに入力されます。</li> <li>関数仮パラメーターとして使用される項目は、共用の場合も非共用の場合もあります。非共用項目の定義は、関数に格納されます。</li> </ul>	<p>関数仮パラメーター:</p> <ul style="list-style-type: none"> <li>パラメーターはコンマで区切る必要があります。</li> <li>それぞれのパラメーターに型情報が指定されます。</li> <li>オプションで、それぞれのパラメーターにパラメーター型情報が指定されます。</li> </ul> <p>関数仮パラメーターのフォーマットは、例えば次のとおりです。</p> <pre>( parameterName typeInfo   [ parameterType ]   { , parameterName typeInfo     [ parameterType ] } )</pre> <p>関数仮パラメーターの具体例を次に示します。</p> <pre>(parmSharedItem parmSharedItem   field,   parmNonSharedItem char(10)   nullable,   parmRecord parmRecord)</pre>	<p>関数仮パラメーター:</p> <p>マイグレーション・ツールは、型情報を次のように設定します。</p> <ul style="list-style-type: none"> <li>レコードの場合、typeInfo は同じレコード名を指定した型定義です。</li> <li>項目の型が VAGen Any* 型のいずれかである場合、typeInfo は対応する EGL 特殊項目型です。</li> <li>項目が共用データ項目の場合、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>マイグレーション構文設定「選択済み共用データ項目をプリミティブ項目定義に変換」を選択した場合に、データ項目パーツが使用可能ならば、共用項目はマイグレーション・ツールによって EGL 関数仮パラメーターに変換され、このパラメーターはデータ項目パーツに対して指定された型、長さ、および小数部に基づいたプリミティブ定義を使用して宣言されます。型、長さ、および小数部の情報のマイグレーションは、249 ページの表 67 の説明と同じです。</li> <li>マイグレーション構文設定「選択済み共用データ項目をプリミティブ項目定義に変換」を選択しなかった場合、またはデータ項目パーツが使用不可の場合は、共用項目はマイグレーション・ツールによって EGL 関数仮パラメーターに変換され、この変数は型定義を使用して宣言されます。マイグレーション用の型定義は、常に項目名と同じです。</li> </ul> </li> <li>項目が非共用データ項目の場合、typeInfo は項目の型、長さ、および小数部に基づいてマイグレーションされ、249 ページの表 67 で説明した規則に準拠します。</li> </ul>
<p>関数仮パラメーター:</p> <ul style="list-style-type: none"> <li>関数仮パラメーターの型: <ul style="list-style-type: none"> <li>レコード</li> <li>項目</li> <li>マップ項目</li> <li>SQL 項目</li> </ul> </li> </ul>	<p>関数仮パラメーター:</p> <ul style="list-style-type: none"> <li>関数仮パラメーターの型: <ul style="list-style-type: none"> <li>適用外</li> <li>適用外</li> <li>フィールド</li> <li>NULL 可能</li> </ul> </li> </ul>	
<p>関数仮パラメーター:</p> <ul style="list-style-type: none"> <li>特殊項目型、長さの指定なし: <ul style="list-style-type: none"> <li>AnyChar</li> <li>AnyDBCS</li> <li>AnyMix</li> <li>AnyHex</li> <li>AnyUnicode</li> <li>AnyNumeric</li> </ul> </li> </ul>	<p>関数仮パラメーター:</p> <ul style="list-style-type: none"> <li>特殊項目型、長さの指定なし: <ul style="list-style-type: none"> <li>char</li> <li>dbchar</li> <li>mbchar</li> <li>hex</li> <li>unicode</li> <li>number</li> </ul> </li> </ul>	
<p>関数からの戻り値:</p> <ul style="list-style-type: none"> <li>データ型</li> <li>長さ</li> <li>小数部</li> <li>説明</li> </ul>	<p>EGL の戻り値:</p> <ul style="list-style-type: none"> <li>次に、returns ステートメントのフォーマットの例を示します。</li> </ul> <pre>returns( returnType )   // Description</pre>	<p>関数に戻り値がある場合、マイグレーション・ツールは 249 ページの表 67 で説明した規則に従って、データ型、長さ、および小数部をマイグレーションします。</p>

表 102. 関数 — 一般構文、説明、パラメーター、戻り値、およびローカル・ストレージ (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>関数ローカル・ストレージ:</p> <ul style="list-style-type: none"> <li>関数ローカル・ストレージは、専用のウィンドウに入力されます。</li> <li>関数ローカル・ストレージとして使用される項目は、共用の場合も非共用の場合もあります。非共用項目の定義は、関数に格納されます。</li> </ul>	<p>関数の変数宣言:</p> <ul style="list-style-type: none"> <li>関数の変数宣言には、変数名とそれに関連した型情報が含まれている必要があります。</li> <li>関数の変数宣言のフォーマットは、例えば次のとおりです。</li> </ul> <pre>// Function Declarations variableName typeInfo ; { variableName typeInfo ; }</pre>	<p>マイグレーション・ツールは、typeInfo を次のように設定します。</p> <ul style="list-style-type: none"> <li>レコードの場合、typeInfo は同じレコード名を指定した型定義です。</li> <li>項目が共用データ項目の場合、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>マイグレーション構文設定「選択済み共用データ項目をプリミティブ項目定義に変換」を選択した場合に、データ項目パーツが使用可能ならば、共用項目はマイグレーション・ツールによって EGL 変数に変換され、この変数はデータ項目パーツに対して指定された型、長さ、および小数部に基づいたプリミティブ定義を使用して宣言されます。型、長さ、および小数部の情報のマイグレーションは、249 ページの表 67 の説明と同じです。</li> <li>マイグレーション構文設定「選択済み共用データ項目をプリミティブ項目定義に変換」を選択しなかった場合、またはデータ項目パーツが使用不可の場合は、共用項目はマイグレーション・ツールによって EGL 変数に変換され、この変数は型定義を使用して宣言されます。マイグレーション用の型定義は、常に項目名と同じです。</li> </ul> </li> <li>項目が非共用データ項目の場合、typeInfo は項目の型、長さ、および小数部に基づいてマイグレーションされ、249 ページの表 67 で説明した規則に準拠します。</li> </ul>
<p>関数ローカル・ストレージ:</p> <ul style="list-style-type: none"> <li>関数ローカル・ストレージのタイプ: <ul style="list-style-type: none"> <li>レコード</li> <li>項目</li> </ul> </li> </ul>	<p>関数ローカル・ストレージ:</p> <ul style="list-style-type: none"> <li>関数ローカル・ストレージのタイプ: <ul style="list-style-type: none"> <li>適用外</li> <li>適用外</li> </ul> </li> </ul>	

表 103. 関数 — EXECUTE 入出力オプション

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: なし</li> <li>入出力オプション: EXECUTE</li> </ul>	<p>同等なステートメントなし。</p>	<p>マイグレーション・ツールは、EXECUTE 入出力オプションを除去します。</p>

表 104. 関数 — マップと UI レコードの入出力オプション

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: mapName</li> <li>入出力オプション: DISPLAY</li> </ul> <p>注: DISPLAY は、表示マップとプリンター・マップの両方に使用されます。</p>	<p>テキスト書式を表示するには、<i>display</i> ステートメントを使用します。印刷書式を印刷するには、<i>print</i> ステートメントを使用します。</p> <p>次に、<i>display</i> ステートメントと <i>print</i> ステートメントの例を示します。</p> <pre>display mapName;  print mapName;</pre> <p>注: VisualAge Generator 互換モードでは、<i>display printForm</i> は <i>print printForm</i> と同様に扱われます。</p>	<p>マイグレーション・ツールは、マップのタイプに基づいて <i>display</i> ステートメントまたは <i>print</i> ステートメントへの変換を行います。詳細と起こりうる問題については、92 ページの『マップの DISPLAY ステートメント』を参照してください。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: mapName</li> <li>入出力オプション: CONVERSE</li> </ul>	<p><i>converse</i> ステートメントを使用します。</p> <p>次に、<i>converse</i> ステートメントの例を示します。</p> <pre>converse mapName;</pre>	<p>特別な考慮事項なし。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: UIRecordName</li> <li>入出力オプション: CONVERSE</li> </ul>	<p><i>converse</i> ステートメントを使用します。次に、<i>converse</i> ステートメントの例を示します。</p> <pre>converse UIRecordName;</pre>	<p>特別な考慮事項なし。</p>

表 105. 関数 — ファイルまたはデータベースの入出力 — 一般情報と入出力エラー・ルーチン

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>VAGen レコード入出力:</p> <ul style="list-style-type: none"> <li>入出力オプション</li> <li>入出力オブジェクト (常にレコード)</li> <li>入出力エラー・ルーチン (オプション)</li> </ul> <p>注: レコードは、シリアル・レコード、索引付きレコード、相対レコード、メッセージ・キュー・レコード、SQL 行レコード、DL/I セグメント・レコードのいずれかです。</p>	<p>EGL レコード入出力:</p> <ul style="list-style-type: none"> <li>入出力ステートメント</li> <li>レコード名</li> <li>エラー・ルーチン名を指定した <i>try onException end</i> ステートメント (オプション)</li> </ul> <p>入出力エラー・ルーチンを指定する場合は、ステートメントを <i>try...end</i> ブロックで囲みます。エラー・ルーチンを含むファイルまたはデータベースの入出力は、例えば次のようになります。</p> <pre>try   add recordName ; [onException error-routine ; ] end</pre>	<p>マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>VAGen の入出力オプションを、対応する EGL の入出力ステートメントに変更します。</li> </ul>

表 105. 関数 — ファイルまたはデータベースの入出力 — 一般情報と入出力エラー・ルーチン (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>ファイルまたはデータベースの入出力を行う関数の場合、エラー・ルーチンはオプションです。</p> <p><b>注:</b> ソフト・エラーが発生した場合、または EZEFEFEC = 1 の場合に、エラー・ルーチンが呼び出されます。</p>	<p>レコードの入出力を行う関数の場合、エラー・ルーチンはオプションです。エラー・ルーチンを使用しない入出力は、例えば次のとおりです。</p> <pre>add recordName;</pre> <p>エラー・ルーチンを使用する入出力は、例えば次のとおりです。</p> <pre>try   add recordName ; onException error-routine ; end</pre> <p><b>注:</b> ソフト・エラーが発生した場合、または handleHardIOErrors = 1 の場合には、onException ステートメントが呼び出されます。関数が DL/I I/O を行い、handleHardIOErrors = 1 の場合にも、onException ステートメントが呼び出されます。</p>	<p>マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>エラー・ルーチンが指定されていない場合、ツールは try、onException、または end の各ステートメントを組み込みません。</li> <li>エラー・ルーチンが指定されている場合、ツールは try ステートメントと end ステートメントを組み込みます。</li> <li>マイグレーション・ツールは、VAGen のエラー・ルーチン名に基づいて onException ステートメントへの変換を行います。マイグレーション・ツールがプログラムをマイグレーションする際に、VAGen の main 関数名は、main 関数ラベルと main 関数呼び出しステートメントの両方に常にマイグレーションされます。このため、関数の入出力エラー・ルーチンをマイグレーションするときに、mainFunctionLabel は常に mainFunctionName と一致します。</li> </ul> <p>関数名であるエラー・ルーチンのマイグレーションに関しては、特別な考慮事項が適用されます。詳細と起こりうる問題については、93 ページの『入出力エラー・ルーチン』を参照してください。</p>
<p>エラー・ルーチンの値:</p> <p>EZECLOS EZEFL0 EZERTN mainFunctionName  nonmainFunctionName</p>	<p>onException ブロック・ステートメント:</p> <pre>onException exit program; onException exit stack; Omit the onException statement. onException exit stack   mainFunctionLabel; onException   nonmainFunctionName();</pre>	

表 106. 関数 — シリアル・レコード、索引付きレコード、相対レコード、およびメッセージ・キュー・レコードの入出力オプション

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: ADD</li> </ul>	<p>add ステートメントを使用します。次に例を示します。</p> <pre>add recordName;</pre>	特別な考慮事項なし。
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: SCAN</li> </ul>	<p>get next ステートメントを使用します。次に例を示します。</p> <pre>get next recordName;</pre>	特別な考慮事項なし。
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: SCANBACK</li> </ul>	<p>get previous ステートメントを使用します。次に例を示します。</p> <pre>get previous recordName;</pre>	特別な考慮事項なし。



表 106. 関数 — シリアル・レコード、索引付きレコード、相対レコード、およびメッセージ・キュー・レコードの入出力オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: CLOSE</li> </ul>	<p><i>close</i> ステートメントを使用します。次に例を示します。</p> <pre>close recordName;</pre>	特別な考慮事項なし。
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: INQUIRY</li> </ul>	<p><i>get</i> ステートメントを使用します。次に例を示します。</p> <pre>get recordName;</pre>	特別な考慮事項なし。
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: UPDATE</li> </ul>	<p><i>get forUpdate</i> ステートメントを使用します。次に例を示します。</p> <pre>get recordName forUpdate;</pre>	特別な考慮事項なし。
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: DELETE</li> </ul>	<p><i>delete</i> ステートメントを使用します。次に例を示します。</p> <pre>delete recordName;</pre>	特別な考慮事項なし。
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: REPLACE</li> </ul>	<p><i>replace</i> ステートメントを使用します。次に例を示します。</p> <pre>replace recordName;</pre>	特別な考慮事項なし。

表 107. 関数 — *Execution Time Statement Build* を使用しないデフォルト (未変更) の SQL ステートメントの入出力オプション

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: ADD</li> </ul>	<p><i>add</i> ステートメントを使用します。次に例を示します。</p> <pre>add recordName;</pre>	特別な考慮事項なし。
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: SCAN</li> </ul>	<p><i>get next</i> ステートメントを使用します。次に例を示します。</p> <pre>get next recordName;</pre>	特別な考慮事項なし。
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: CLOSE</li> </ul>	<p><i>close</i> ステートメントを使用します。次に例を示します。</p> <pre>close recordName;</pre>	特別な考慮事項なし。
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: INQUIRY</li> </ul> <p>(Single row select を使用する場合、および使用しない場合)</p>	<p><i>get</i> ステートメントを使用します。single row select を実行する場合は、<i>singleRow</i> も使用します。single row select を使用しない例は、次のとおりです。</p> <pre>get recordName;</pre> <p>single row select を使用する例は、次のとおりです。</p> <pre>get recordName singleRow;</pre>	Single row select が VisualAge Generator 内で指定されている場合、マイグレーション・ツールは EGL の <i>singleRow</i> オプションを組み込みます。

表 107. 関数 — *Execution Time Statement Build* を使用しないデフォルト (未変更) の SQL ステートメントの入出力オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: UPDATE</li> </ul>	<p><i>get forUpdate</i> ステートメントを使用します。次に例を示します。</p> <pre>get recordName forUpdate     resultSetID;</pre>	<p>SQL UPDATE ステートメントをマイグレーションする際に、マイグレーション・ツールは <i>resultSetID</i> を常に組み込みます。ツールは、<i>resultSetID</i> を関数名に設定し、その後にユーザー指定の接尾部を付けます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</p> <p>レコードが SQL または非 SQL のどちらであるかマイグレーション・ツールが判別できない場合は、特別な考慮事項が適用されます。詳細と起こりうる問題については、100 ページの『複数の更新を行う SQL 入出力』を参照してください。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: DELETE</li> </ul>	<p><i>delete</i> ステートメントを使用します。次に例を示します。</p> <pre>delete recordName;</pre>	<p>特別な考慮事項なし。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: REPLACE</li> </ul> <p>(UPDATE/SETUPD functionName を使用する場合、または使用しない場合)</p>	<p><i>replace</i> ステートメントを使用します。次に、いくつかの例を示します。</p> <pre>replace recordName; replace recordName from     resultSetID;</pre>	<p>VisualAge Generator では UPDATE/SETUPD 関数名が組み込まれていた場合、マイグレーション・ツールは <i>resultSetID</i> を組み込み、<i>resultSetID</i> を UPDATE/SETUPD 関数名に設定して、その後にユーザー指定の接尾部を付けます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: SETINQ</li> </ul> <p>(Declare cursor with hold を使用する場合、および使用しない場合)</p>	<p><i>open</i> ステートメントを使用します。<i>Declare cursor with hold</i> を使用する場合は、<i>hold</i> オプションも使用します。次に、両タイプのステートメントの例を示します。</p> <pre>open resultSetID for recordName; open resultSetID hold     for recordName;</pre>	<p>マイグレーション・ツールは、<i>resultSetID</i> を関数名に設定し、その後にユーザー指定の接尾部を付けます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</p> <p><i>Declare cursor with hold</i> が VisualAge Generator で選択されていた場合、マイグレーション・ツールは EGL の <i>hold</i> オプションを組み込みます。</p>

表 107. 関数 — *Execution Time Statement Build* を使用しないデフォルト (未変更) の SQL ステートメントの入出力オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: SETUPD</li> </ul> <p>(Declare cursor with hold を使用する場合、および使用しない場合)</p>	<p><i>open forUpdate</i> ステートメントを使用します。 <i>Declare cursor with hold</i> を使用する場合は、<i>hold</i> オプションも使用します。次に、両タイプのステートメントの例を示します。</p> <pre>open resultSetID forUpdate for recordName; open resultSetID hold forUpdate for recordName;</pre>	<p>ツールは、resultSetID を関数名に設定し、その後にユーザー指定の接尾部を付けます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</p> <p><i>Declare cursor with hold</i> が VisualAge Generator で選択されていた場合、マイグレーション・ツールは EGL の <i>hold</i> オプションを組み込みます。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: SQLEXEC</li> </ul> <p>Model SQL Statement を使用する 場合</p> <p>注:</p> <ul style="list-style-type: none"> <li>この形式の SQLEXEC には、SQL レコード名が含まれています。</li> <li>モデル・タイプの値は、次のとおりです。 <ul style="list-style-type: none"> <li>None</li> <li>Update</li> <li>Delete</li> </ul> </li> <li>モデル・タイプが None の場合、VisualAge Generator は入出力を行いません。それでも入出力エラー・ルーチンは生成機能によって処理されますが、エラーは発生しません。</li> </ul>	<p><i>execute</i> ステートメントを使用します。次に例を示します。</p> <pre>execute modelType for recordName;</pre> <p>注: <i>modelType</i> は、update または delete のどちらかです。</p>	<p>マイグレーション・ツールは、VAGen の Model SQL Statement の値に基づいて EGL の <i>modelType</i> を設定します。</p> <p>VAGen の Model SQL Statement が None の場合、VAGen の入出力ステートメントは効力を持たないので、マイグレーション・ツールは入出力ステートメントを省略します。マイグレーション・ツールは、関数の入出力エラー・ルーチンに基づいて、<i>try</i>、<i>onException</i>、および <i>end</i> の各ステートメントを組み込みます。</p>

表 108. 関数 — *Execution Time Statement Build* を使用しない変更済み SQL ステートメントの入出力オプション

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>変更済み SQL ステートメントに関する一般情報:</p> <ul style="list-style-type: none"> <li>VisualAge Generator は、テストおよび生成時に SQL 行レコードから table 文節を作成します。table 文節は次のとおりです。 <ul style="list-style-type: none"> <li><b>insert into</b> <i>sqlTableName</i> (ADD 入出力オプションの場合)</li> <li><b>from</b> <i>sqlTableName</i> <i>sqlTableLabel</i> (INQUIRY、UPDATE、SETINQ、および SETUPD の各入出力オプションの場合)</li> <li><b>update</b> <i>sqlTableName</i> (REPLACE 入出力オプションの場合)</li> </ul> </li> <li>関数の最終変更時点によっては、他の SQL 文節が関数定義に保管されない場合があります。SQL 文節が保管されていない場合、VisualAge Generator は、入出力オブジェクトのレコード定義に基づいてデフォルト文節を作成します。</li> <li><i>!itemColumnName</i> 変数が使用できます。この変数は、SQL 行レコード内の項目の名前を指定します。テストまたは生成時に、VisualAge Generator は対応する SQL 列名への置換を行います。</li> <li><i>sqlClauses</i> は、SQL 構文で作成されます。</li> </ul>	<p>変更済み SQL ステートメントに関する一般情報:</p> <ul style="list-style-type: none"> <li>SQL 文節を変更する必要がある場合、EGL は文節すべての明示的な指定を必要とします。SQL ステートメントに table 文節を明示的に組み込む必要があります。table 文節は次のとおりです。 <ul style="list-style-type: none"> <li><b>insert into</b> <i>sqlTableName</i> (add ステートメントの場合)</li> <li><b>from</b> <i>sqlTableName</i> <i>sqlTableLabel</i> (get ステートメントと open ステートメントの場合)</li> <li><b>update</b> <i>sqlTableName</i> (replace ステートメントの場合)</li> </ul> </li> <li>SQL 文節が指定されている場合、EGL は文節すべての明示的な指定を必要とします。必要な SQL 文節は、入出力のタイプによって異なります。</li> <li>EGL は、SQL 列名が SQL ステートメントに明示的に含まれていることを必要とします。<i>!itemColumnName</i> 変数はサポートされません。</li> <li><i>sqlClauses</i> は、SQL 構文で作成されます。</li> </ul>	<p>マイグレーション・ツールは、SQL 行レコードからの表と表ラベルを使用して、EGL 入出力ステートメントの tables 文節を作成します。表名と表名ホスト変数の両方が、EGL 入出力ステートメントの table 文節に組み込まれます。</p> <p>必要な SQL 文節が関数定義に保管されていない場合、マイグレーション・ツールは VisualAge Generator の場合と同じように、レコード定義に基づいてデフォルト文節を作成します。</p> <p>マイグレーション・ツールは、<i>!itemColumnName</i> 変数を対応する SQL 列名に変換します。</p> <p>マイグレーション・ツールは、VAGen のコメント (/*) を SQL のコメント (--) に変換します。</p> <p>SQL レコードとその代替仕様レコード (存在する場合) がマイグレーション時に使用できない場合は、特別な考慮事項が適用されます。詳細と起こりうる問題については、95 ページの『SQL 入出力ステートメント』を参照してください。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: <i>recordName</i></li> <li>入出力オプション: ADD</li> </ul> <p>変更可能な文節:</p> <ul style="list-style-type: none"> <li>Columns</li> <li>VALUES</li> </ul>	<p><i>add</i> ステートメントを使用します。次に例を示します。</p> <pre>add recordName with #sql{   insert into   sqlTablename   (columnName1, columnName2)   values   (valueInfo1, valueInfo2) };</pre>	<p>マイグレーション・ツールは、レコード定義にある表名に基づいて <i>insert into</i> 文節を作成します。特別な考慮事項が適用されます。詳細と起こりうる問題については、95 ページの『SQL 入出力ステートメント』を参照してください。</p>

表 108. 関数 — *Execution Time Statement Build* を使用しない変更済み SQL ステートメントの入出力オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: INQUIRY</li> </ul> <p>(Single row select を使用する場合、および使用しない場合)</p> <p>変更可能な文節:</p> <ul style="list-style-type: none"> <li>SELECT</li> <li>INTO</li> <li>WHERE、GROUP BY、HAVING</li> <li>ORDER BY</li> </ul>	<p><i>get</i> ステートメントを使用します。</p> <p>次に、single row select を使用する場合の例を示します。</p> <pre> get recordName singleRow   with #sql{     select       Name1,       Name2,       Age     from       sqlTable1 sqlLabel1,       sqlTable2 sqlLabel2     where       Name1 = :NameX     order by       Age   } into   nameA, nameB, myage; </pre>	<p>Single row select が VisualAge Generator 内で指定されている場合、マイグレーション・ツールは EGL の singleRow オプションを組み込みます。</p> <p>マイグレーション・ツールは、レコード定義にある表名と表ラベルに基づいて <i>from</i> 文節を作成します。特別な考慮事項が適用されます。詳細と起こりうる問題については、95 ページの『SQL 入出力ステートメント』を参照してください。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: UPDATE</li> </ul> <p>変更可能な文節:</p> <ul style="list-style-type: none"> <li>SELECT</li> <li>INTO</li> <li>WHERE</li> <li>FOR UPDATE OF</li> </ul>	<p><i>get forUpdate</i> ステートメントを使用します。</p> <p>次に例を示します。</p> <pre> get recordName forUpdate   resultsetID   with #sql{     select       Name1, Name2, Age     from       sqlTable1 sqlLabel1     where       Name1 = :NameX     for update of       Name2, Age   } into   Name1, Name2, Age; </pre>	<p>マイグレーション・ツールは、SQL レコードの UPDATE ステートメントをマイグレーションする際に、resultSetID を常に組み込みます。ツールは、resultSetID を関数名に設定し、その後にユーザー指定の接尾部を付けます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</p> <p>マイグレーション・ツールは、レコード定義にある表名および表ラベルに基づいて、<i>from</i> 文節を作成します。特別な考慮事項が適用されます。詳細と起こりうる問題については、95 ページの『SQL 入出力ステートメント』を参照してください。</p>

表 108. 関数 — *Execution Time Statement Build* を使用しない変更済み SQL ステートメントの入出力オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: REPLACE</li> </ul> (オプションで UPDATE/SETUPD functionName)  指定できる文節: <ul style="list-style-type: none"> <li>SET</li> </ul>	<p><i>replace</i> ステートメントを使用します。</p> <p>次に、<i>replace</i> ステートメントの例を示します。</p> <pre> <b>replace</b> recordName <b>with #sql{</b>   <b>update</b>     sqlTableName   <b>set</b>     columnName1 = value1,     columnName2 = value2 <b>}</b> <b>from</b> resultSetID;           </pre>	<p>VisualAge Generator では UPDATE/SETUPD 関数名が組み込まれていた場合、マイグレーション・ツールは <i>from resultSetID</i> 文節を組み込みます。マイグレーション・ツールは、resultSetID を UPDATE/SETUPD 関数名に設定し、その後にユーザー指定の接尾部を付けます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</p> <p>マイグレーション・ツールは、レコード定義にある表名に基づいて <i>update</i> 文節を作成します。特別な考慮事項が適用されます。詳細と起こりうる問題については、95 ページの『SQL 入出力ステートメント』を参照してください。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オブジェクト: SETINQ</li> </ul> (Declare cursor with hold を使用する場合、または使用しない場合)  変更可能な文節: <ul style="list-style-type: none"> <li>SELECT</li> <li>INTO</li> <li>WHERE、GROUP BY、HAVING</li> <li>ORDER BY</li> </ul>	<p><i>open</i> ステートメントを使用します。 <i>Declare cursor with hold</i> を使用する場合は、<i>hold</i> オプションも使用します。</p> <p>次に、<i>hold</i> オプションを使用する <i>open</i> ステートメントの例を示します。</p> <pre> <b>open</b> resultSetID <b>hold</b> <b>with #sql{</b>   <b>select</b>     Name1, Name2   <b>from</b>     sqlTable1 sqlLabel1,     sqlTable2 sqlLabel2   <b>where</b>     Name1 &gt; :Name2   <b>order by</b>     Name1 <b>}</b> <b>into</b> Name1, Name2 <b>for</b> recordName;           </pre>	<p>ツールは、resultSetID を関数名に設定し、その後にユーザー指定の接尾部を付けます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</p> <p><i>Declare cursor with hold</i> が VisualAge Generator で選択されていた場合、マイグレーション・ツールは EGL の <i>hold</i> オプションを resultSetID の後に組み込みます。</p> <p>マイグレーション・ツールは、レコード定義にある表名と表ラベルに基づいて <i>from</i> 文節を作成します。特別な考慮事項が適用されます。詳細と起こりうる問題については、95 ページの『SQL 入出力ステートメント』を参照してください。</p>



表 108. 関数 — *Execution Time Statement Build* を使用しない変更済み SQL ステートメントの入出力オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: record</li> <li>入出力オプション: SETUPD</li> </ul> <p>(Declare cursor with hold を使用する場合、または使用しない場合)</p> <p>変更可能な文節:</p> <ul style="list-style-type: none"> <li>SELECT</li> <li>INTO</li> <li>WHERE</li> <li>FOR UPDATE OF</li> </ul>	<p><i>open forUpdate</i> ステートメントを使用します。次に、<i>hold</i> オプションの使用例を示します。</p> <pre> open resultSetID hold forUpdate with #sql{   select     Column1, Column2   from     sqlTable1 sqlLabel1   where     Column1 &gt; :Item1   for update of     Column2 } into Item1, Item2 for recordName; </pre>	<p>マイグレーション・ツールは、<i>resultSetID</i> を関数名に設定し、その後にユーザー指定の接尾部を付けます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</p> <p><i>Declare cursor with hold</i> が VisualAge Generator で選択されていた場合、マイグレーション・ツールは EGL の <i>hold</i> オプションを組み込みます。</p> <p>マイグレーション・ツールは、レコード定義にある表名および表ラベルに基づいて、<i>from</i> 文節を作成します。特別な考慮事項が適用されます。詳細と起こりうる問題については、95 ページの『SQL 入出力ステートメント』を参照してください。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: record</li> <li>入出力オプション: SQLEXEC</li> </ul> <p>Model SQL Statement を使用する場合</p> <p>注:</p> <ul style="list-style-type: none"> <li>この形式の SQLEXEC には、SQL レコード名がオプションに含まれています。</li> <li>モデル・タイプの値は、次のとおりです。 <ul style="list-style-type: none"> <li>None</li> <li>Update</li> <li>Delete</li> </ul> </li> </ul>	<p><i>execute</i> ステートメントを使用します。次に、このステートメントの例を示します。</p> <pre> execute modelType #sql{   UPDATE mysqltable   set Column1 = Column1 * 2   where Column2 = :Column2 } for recordName; </pre> <p>注: モデル・タイプの値には、Update と Delete があります。</p>	<p>マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>SQLEXEC を <i>execute</i> ステートメントに変換します。</li> <li>入出力オブジェクト (指定されている場合) を <i>for</i> 文節の <i>recordName</i> として使用します。</li> </ul> <p>マイグレーション・ツールは、VAGen の Model SQL Statement の値 (存在する場合) を、EGL の <i>execute</i> ステートメントのコメントとして組み込みます。</p> <p>マイグレーション・ツールは、VAGen の SQLEXEC 文節を EGL の SQL 文節にマイグレーションします。</p>

表 109. 関数 - Execution Time Statement Build を使用する変更済み SQL ステートメントの入出力オプション

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>Execution time statement build は、次の入出力オプションのみと組み合わせで使用できます。</p> <ul style="list-style-type: none"> <li>• INQUIRY</li> <li>• UPDATE</li> <li>• SETINQ</li> <li>• SETUPD</li> <li>• SQLEXEC</li> </ul> <p>Execution time statement build を指定すると、入出力ステートメントが実行されるたびに、SQL ステートメントが VisualAge Generator によって動的に準備されます。</p>	<p>EGL では、SQL ステートメントを動的に準備する必要が生じるごとに、SQL prepare ステートメントをコーディングします。さらに、prepare の後に open、execute、または get のいずれかのステートメントをコーディングする必要があります。例えば、Execution time statement build を使用する VAGen INQUIRY 入出力オプションに相当する EGL ステートメントは次のとおりです。</p> <pre>prepare prepID from   "sqlStatementString" for recordName; get recordName with prepID into itemList;</pre> <p>注:</p> <ul style="list-style-type: none"> <li>• prepare ステートメント内の sqlStatementString は、SQL 表記で書かれた定数と変数を連結したストリングです。列名と変数の両方を使用する where 文節の例は、次のとおりです。</li> </ul> <pre>[other clauses] + " where columnName = " + itemName + " AND columnName2 = " + itemName2 + .... [other clauses]</pre> <ul style="list-style-type: none"> <li>• この表で後に示す例には、二重引用符の外側に分割された変数は示されていません。</li> </ul>	<p>マイグレーション・ツールは、prepID を関数名に設定し、その後ろにユーザー指定の接尾部を付けます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</p> <p>マイグレーション・ツールは、関数内の SQL 文節と、レコード定義にある表名および表名変数を使用して、sqlStatementString を作成します。マイグレーション・ツールは、sqlStatementString を次のように作成します。</p> <ul style="list-style-type: none"> <li>• Execution time statement build が指定されなかった場合と同様に、次のような処理をすべて行います。 <ul style="list-style-type: none"> <li>– SQL 行レコードから得られる表名および表ラベルを使用して、EGL 入出力ステートメントの tables 文節を作成します。表名と表名ホスト変数の両方が、EGL 入出力ステートメントの tables 文節に組み込まれます。</li> <li>– レコード定義に基づいて、必要なデフォルト文節を作成します。</li> <li>– !itemName 変数を対応する SQL 列名に変換します。</li> <li>– VAGen コメント (/*) を、prepare ステートメント内の EGL コメント (//) に変換します。</li> </ul> </li> </ul> <p>その後、マイグレーション・ツールは次のような追加処理を行って、sqlStatementString を作成します。</p> <ul style="list-style-type: none"> <li>• 定数、列名、および SQL 演算子を二重引用符で囲みます。</li> <li>• 変数を二重引用符の外側に配置します。</li> <li>• + ストリング連結記号を使用して、ストリングと変数を連結します。</li> </ul> <p>SQL レコードとその代替仕様レコード (存在する場合) がマイグレーション時に使用できない場合は、特別な考慮事項が適用されます。詳細と起こりうる問題については、95 ページの『SQL 入出力ステートメント』を参照してください。</p>

表 109. 関数 - Execution Time Statement Build を使用する変更済み SQL ステートメントの入出力オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: record</li> <li>入出力オプション: INQUIRY</li> </ul> <p>(Single row select を使用する場合、および使用しない場合)</p> <p>変更可能な文節:</p> <ul style="list-style-type: none"> <li>SELECT</li> <li>INTO</li> <li>WHERE、GROUP BY、HAVING</li> <li>ORDER BY</li> </ul>	<p>prepare ステートメントを使用します。次に例を示します。</p> <pre>prepare prepID from   " select columnName "   + ", columnName2 "   + " from table1 t1 "   + "[ where whereClause ]"   + "[order by orderByClause ]" for recordName; get recordname with prepID into itemList;</pre>	<p>特別な考慮事項なし。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: record</li> <li>入出力オプション: UPDATE</li> </ul> <p>変更可能な文節:</p> <ul style="list-style-type: none"> <li>SELECT</li> <li>INTO</li> <li>WHERE</li> <li>FOR UPDATE OF</li> </ul>	<p>prepare ステートメントを使用します。次に例を示します。</p> <pre>prepare prepID from   " select columnName "   + ", columnName2 "   + " from table1 t1 "   + "[ where whereClause ]"   + " for Update of columnList " for recordName; get recordName forUpdate resultSetID with prepID into itemList;</pre>	<p>マイグレーション・ツールは、SQL レコードの UPDATE ステートメントをマイグレーションする際に、resultSetID を常に組み込みます。ツールは、resultSetID を関数名に設定し、その後にユーザー指定の接尾部を付けます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: record</li> <li>入出力オプション: SETINQ</li> </ul> <p>(Declare cursor with hold を使用する場合、または使用しない場合)</p> <p>変更可能な文節:</p> <ul style="list-style-type: none"> <li>SELECT</li> <li>INTO</li> <li>WHERE、GROUP BY、HAVING</li> <li>ORDER BY</li> </ul>	<p>prepare ステートメントを使用します。次に例を示します。</p> <pre>prepare prepID from   " select columnName "   + ", columnName2 "   + " from table1 t1 "   + "[ where whereClause ]"   + "[order by orderByClause ]" for recordName; open resultSetID [ hold ] with prepID into itemList for recordName;</pre>	<p>マイグレーション・ツールは、resultSetID を関数名に設定し、その後にユーザー指定の接尾部を付けます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</p> <p>Declare cursor with hold が VisualAge Generator で選択されていた場合、マイグレーション・ツールは EGL hold オプションを resultSetID の後に組み込みます。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: record</li> <li>入出力オプション: SETUPD</li> </ul> <p>(Declare cursor with hold を使用する場合、または使用しない場合)</p> <p>変更可能な文節:</p> <ul style="list-style-type: none"> <li>SELECT</li> <li>INTO</li> <li>WHERE</li> <li>FOR UPDATE OF</li> </ul>	<p>prepare ステートメントを使用します。次に例を示します。</p> <pre>prepare prepID from   " select columnName "   + ", columnName2 "   + " from table1 t1 "   + "[ where whereClause ] "   + " for update of columnList " for recordName; open resultSetID [ hold ] forUpdate with prepID into itemList for recordName;</pre>	<p>ツールは、resultSetID を関数名に設定し、その後にユーザー指定の接尾部を付けます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</p> <p>Declare cursor with hold が VisualAge Generator で選択されていた場合、マイグレーション・ツールは EGL hold オプションを resultSetID の後に組み込みます。</p>

表 109. 関数 - Execution Time Statement Build を使用する変更済み SQL ステートメントの入出力オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: record</li> <li>入出力オプション: SQLEXEC</li> </ul> <p>Model SQL Statement を使用する場合 注:</p> <ul style="list-style-type: none"> <li>この形式の SQLEXEC には、SQL レコード名がオプションで含まれています。</li> <li>モデル・タイプの値は、次のとおりです。 <ul style="list-style-type: none"> <li>None</li> <li>Update</li> <li>Delete</li> </ul> </li> </ul>	<p><i>prepare</i> ステートメントを使用します。次に例を示します。</p> <pre>prepare prepID from   " grant " + group_privileges   + " on " + table_name   + " to " + userid [ for recordName ]; execute prepID [ for recordName ]; // model = type</pre>	<p>マイグレーション・ツールは、VAGen の Model SQL Statement の値 (存在する場合) を、EGL の <i>execute</i> ステートメントのコメントとして組み込みます。</p> <p>マイグレーション・ツールは、VAGen の SQLEXEC 文節を EGL の SQL 文節に変換します。</p>

表 110. 関数 — デフォルト (未変更) の DLI ステートメントの入出力オプション

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: ADD</li> <li>PSB: psbName</li> <li>データベース ID: databaseName   pcbNumber</li> </ul>	<p><i>add</i> ステートメントを使用します。次に例を示します。</p> <pre>add recordName   [ usingPCB pcbInfo ];</pre>	<p>VAGen 関数でデータベース ID を指定すると、マイグレーション・ツールは <i>usingPCB</i> キーワードを組み込みます。マイグレーション・ツールは、VAGen 関数に保管されている情報に基づいて、<i>pcbInfo</i> の値を設定します。</p> <ul style="list-style-type: none"> <li><i>databaseName</i> には、カスタマー指定の接尾部が続きます。接尾部は、ステージ 2 の「VAGen マイグレーションの設定」によって制御できます。</li> <li><i>pcbn</i> (<i>n</i> は、VisualAge Generator で指定した <i>pcbNumber</i>)。</li> </ul>
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: SCAN</li> <li>PSB: psbName</li> <li>データベース ID: databaseName   pcbNumber</li> <li>SCAN オプション: <ul style="list-style-type: none"> <li>Scan update</li> <li>Scan parent</li> </ul> </li> </ul>	<p><i>get next</i> ステートメントを使用します。次に例を示します。</p> <pre>get next recordName   [ usingPCB pcbInfo ];</pre> <p>Scan update と Scan parent の両方を含む例:</p> <pre>get next inParent   recordName forUpdate   [ usingPCB pcbInfo ];</pre>	<p>VAGen 関数でデータベース ID を指定した場合は、ADD 入出力オプションについて説明されているように、マイグレーション・ツールによって <i>usingPCB</i> キーワードが組み込まれます。</p> <p>VAGen 関数で Scan parent オプションを指定した場合は、マイグレーション・ツールによって <i>inParent</i> キーワードが組み込まれます。</p> <p>VAGen 関数で Scan update オプションを指定した場合は、マイグレーション・ツールによって <i>forUpdate</i> キーワードが組み込まれます。</p>

表 110. 関数 — デフォルト (未変更) の DLI ステートメントの入出力オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: INQUIRY</li> <li>PSB: psbName</li> <li>データベース ID: databaseName   pcbNumber</li> </ul>	<p>get ステートメントを使用します。次に例を示します。</p> <pre>get recordName   [ usingPCB pcbInfo ] ;</pre>	<p>VAGen 関数でデータベース ID を指定した場合は、ADD 入出力オプションについて説明されているように、マイグレーション・ツールによって usingPCB キーワードが組み込まれます。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: UPDATE</li> <li>PSB: psbName</li> <li>データベース ID: databaseName   pcbNumber</li> </ul>	<p>get forUpdate ステートメントを使用します。次に例を示します。</p> <pre>get recordName forUpdate   [ usingPCB pcbInfo ] ;</pre>	<p>VAGen 関数でデータベース ID を指定した場合は、ADD 入出力オプションについて説明されているように、マイグレーション・ツールによって usingPCB キーワードが組み込まれます。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: DELETE</li> <li>PSB: psbName</li> <li>データベース ID: databaseName   pcbNumber</li> </ul>	<p>delete ステートメントを使用します。次に例を示します。</p> <pre>delete recordName   [ usingPCB pcbInfo ] ;</pre>	<p>VAGen 関数でデータベース ID を指定した場合は、ADD 入出力オプションについて説明されているように、マイグレーション・ツールによって usingPCB キーワードが組み込まれます。</p>
<ul style="list-style-type: none"> <li>入出力オブジェクト: recordName</li> <li>入出力オプション: REPLACE</li> <li>PSB: psbName</li> <li>データベース ID: databaseName   pcbNumber</li> </ul>	<p>replace ステートメントを使用します。次に例を示します。</p> <pre>replace recordName   [ usingPCB pcbInfo ] ;</pre>	<p>VAGen 関数でデータベース ID を指定した場合は、ADD 入出力オプションについて説明されているように、マイグレーション・ツールによって usingPCB キーワードが組み込まれます。</p>

表 111. 関数 - 変更済み DL/I ステートメントのセグメント検索指数

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>変更済み DL/I 呼び出し (SSA を使用) で指定される一般情報:</p> <ul style="list-style-type: none"> <li>入出力オプション</li> <li>入出力オブジェクト</li> <li>PSB</li> <li>データベース ID</li> <li>Scan update</li> <li>Scan parent</li> <li>次の SSA のうちゼロ、1 つ、またはそれ以上が組み込まれます。 <ul style="list-style-type: none"> <li>セグメント名</li> <li>コマンド・コード</li> <li>ブール演算子</li> <li>セグメント・フィールド</li> <li>比較演算子</li> <li>比較値項目</li> </ul> </li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>SSA は、専用の DL/I Call Editor に入ります。</li> <li>VisualAge Generator は、生成時に SSA のフォーマットを設定します。</li> </ul>	<p>変更済み DL/I 呼び出し (SSA を使用) の一般的なフォーマットでは、SSA は次のように中括弧の中に入ります。</p> <pre>with #dli { dliFunction   SSAList } ;</pre> <p>例えば、次のようになります。</p> <pre>ioOptionWithModifiers recordNameList [ usingPCB pcbInfo] with #dli { dliFunction   segmentName1*cmdCodes   (segmentFieldA   relationalOperatorA   :comparisonValueItemA   booleanOperator   segmentFieldB   relationalOperatorB   :comparisonValueItemB)   segmentName2*cmdCodes   (segmentFieldC   relationalOperatorC   :comparisonValueItemC) };</pre> <p>注:</p> <ul style="list-style-type: none"> <li>SSA はテキスト・エディターに入ります。</li> <li>EGL は、DL/I のフォーマット規則に従って、生成時に SSA のフォーマットを設定します。</li> </ul>	<p>マイグレーション・ツールは、入出力オブジェクトと SSA 用に指定されるコマンド・コードとの組み合わせに基づいて、recordNameList を作成します。</p>
<p>変更済み DL/I 呼び出し (SSA はすべて削除)。</p> <p>注: この手法は、データベース内のすべてのセグメントをスキャンする場合に使用されます。</p>	<p>変更済み DL/I 呼び出し (SSA はすべて削除) の一般的なフォーマット:</p> <pre>ioOptionWithModifiers recordNameList [ usingPCB pcbInfo] with #dli{ dliFunction };</pre>	<p>マイグレーション・ツールでは、recordNameList は入出力オブジェクトの名前に設定されます。</p>



表 111. 関数 - 変更済み DL/I ステートメントのセグメント検索指数 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>VisualAge Generator 入出力オプション:</p> <ul style="list-style-type: none"> <li>• ADD</li> <li>• SCAN</li> <li>• SCAN - Scan update</li> <li>• SCAN - Scan parent</li> <li>• SCAN - Scan update および Scan parent</li> <li>• INQUIRY</li> <li>• UPDATE</li> <li>• DELETE</li> <li>• REPLACE</li> </ul>	<p>対応する EGL の ioOptionWithModifiers:</p> <ul style="list-style-type: none"> <li>• add</li> <li>• get next</li> <li>• get next forUpdate</li> <li>• get next inParent</li> <li>• get next inParent forUpdate</li> <li>• get</li> <li>• get forUpdate</li> <li>• delete</li> <li>• replace</li> </ul>	<p>特別な考慮事項なし。</p> <p>ioOptionWithModifiers の値は、未変更の DL/I 入出力オプションの変換時に使用される値と同じです。</p>
<p>VisualAge Generator 入出力オプション:</p> <ul style="list-style-type: none"> <li>• ADD</li> <li>• SCAN</li> <li>• SCAN - Scan update</li> <li>• SCAN - Scan parent</li> <li>• SCAN - Scan update および Scan parent</li> <li>• INQUIRY</li> <li>• UPDATE</li> <li>• DELETE</li> <li>• REPLACE</li> </ul>	<p>対応する EGL の dliFunction:</p> <ul style="list-style-type: none"> <li>• ISRT</li> <li>• GN</li> <li>• GHN</li> <li>• GNP</li> <li>• GHNP</li> <li>• GU</li> <li>• GHU</li> <li>• DLET</li> <li>• REPL</li> </ul> <p>注: dliFunction は、ioOptionWithModifiers と整合性がなければなりません。</p>	<p>特別な考慮事項なし。</p>
<p>VisualAge Generator の入出力オプション、SSA、およびコマンド・コード</p> <p>注:</p> <ul style="list-style-type: none"> <li>• コマンド・コードはオプションです。SSA には、最大で 4 つのコマンド・コードがあります。</li> <li>• VisualAge Generator は、生成時およびランタイム時に特殊な処理を実行して、パス呼び出しに関連する D および N コマンド・コードをサポートします。</li> </ul>	<p>EGL の入出力オプション、recordNameList、SSA、およびコマンド・コード</p> <p>注:</p> <ul style="list-style-type: none"> <li>• * が指定されるのは、オプションのコマンド・コードが 1 つ以上指定されている場合に限られます。SSA には、最大で 4 つのコマンド・コードがあります。</li> <li>• EGL recordNameList は、SSA で指定される入出力オプションとコマンド・コードによって異なります。詳しくは、この表の次の行を参照してください。</li> </ul>	<p>マイグレーション・ツールは、入出力オブジェクトと SSA 用に指定されるコマンド・コードとの組み合わせに基づいて、recordNameList を作成します。</p>

表 111. 関数 - 変更済み DL/I ステートメントのセグメント検索指数 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>VisualAge Generator の入出力オブジェクトとコマンド・コード (D または N コマンド・コードは使用しない)</p> <p><b>注:</b> データベースで検索または更新されるのは、入出力オブジェクト (ターゲット DL/I セグメント) のみです。</p>	<p>EGL recordNameList は、ターゲット DL/I セグメントの名前です。</p>	<p>マイグレーション・ツールでは、recordNameList は入出力オブジェクトの名前に設定されます。</p>
<p>VisualAge Generator の ADD 入出力オプション (D コマンド・コードを使用)</p> <p><b>注:</b></p> <ul style="list-style-type: none"> <li>D コマンド・コードは、データベースに挿入される階層の先頭のセグメントを表します。すなわち、D コマンド・コードよりあとのセグメントは、すべて挿入されます。</li> <li>D コマンド・コードを指定できるセグメントは 1 つのみです。</li> <li>ターゲット・セグメントでは、D コマンド・コードが指定されることはありません。</li> </ul>	<p>add ステートメントを使用します。挿入されるすべてのセグメントを、入出力ステートメントのオブジェクトとしてリストする必要があります。以下は、DL/I Insert 呼び出しを指定する場合の例です。</p> <pre>add recordName1 ,     recordName2 ,     recordName3 [ usingPCB pcbInfo ] with #dli{ ISRT     recordName1*D     recordName2     recordName3 } ;</pre>	<p>マイグレーション・ツールでは、入出力ステートメントの recordNameList が作成され、そこには、D コマンド・コードを指定する DL/I セグメント・レコードと、階層内のそれ以降のすべての DL/I セグメント・レコードが含まれます。</p>
<p>VisualAge Generator の SCAN、INQUIRY、および UPDATE 入出力オプション (D コマンド・コードを使用)。</p> <p><b>注:</b></p> <ul style="list-style-type: none"> <li>D コマンド・コードは、階層内の、データベースから検索されるセグメントを表します。データベースから検索される、ターゲット・セグメント以外の各セグメントでは、D コマンド・コードを指定する必要があります。ターゲット・セグメントは、必ず検索されます。</li> <li>D コマンド・コードは、複数の SSA で指定できます。</li> <li>ターゲット・セグメントでは、D コマンド・コードが指定されることはありません。</li> </ul>	<p>入出力オプションに対応する get ステートメントのフォームを使用します。データベースから検索されるすべてのセグメントを、入出力ステートメントのオブジェクトとしてリストする必要があります。以下は、DL/I DL/I Get Hold Next in Parent 呼び出しを指定する場合の例です。</p> <pre>get next inParent     recordName1 ,     recordName3 forUpdate [ usingPCB pcbInfo ] with #dli{ GHNP     recordName1*D     recordName2     recordName3 } ;</pre>	<p>マイグレーション・ツールでは、入出力ステートメントの recordNameList が作成され、そこには、D コマンド・コードを指定する個々の DL/I セグメント・レコードと、ターゲット・セグメントが含まれます。</p>

表 111. 関数 - 変更済み DL/I ステートメントのセグメント検索指数 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>VisualAge Generator の REPLACE 入出力オプション (N コマンド・コードを使用)</p> <p>注:</p> <ul style="list-style-type: none"> <li>N コマンド・コードは、セグメントが、以前の SCAN または UPDATE で D コマンド・コードを使用して更新するために検索された場合でも、置換はされないことを意味します。</li> <li>N コマンド・コードは、複数のセグメントで指定できます。</li> <li>ターゲット・セグメントでは、N コマンド・コードが指定されることはありません。</li> </ul>	<p>replace ステートメントを使用します。ターゲット・セグメントのみは、recordNameList で指定されます。以下は、DL/I Replace 呼び出しを指定する場合の例です。</p> <pre>replace recordName3 [ usingPCB pcbInfo ] with #dli{ REPL recordName1*N recordName3 } ;</pre>	<p>マイグレーション・ツールでは、recordNameList は入出力オブジェクトの名前に設定されます。</p>
<p>VisualAge Generator 比較演算子 (SSA 用):</p> <ul style="list-style-type: none"> <li>EQ および =</li> <li>GT および &gt;</li> <li>LT および &lt;</li> <li>GE および &gt;= および =&gt;</li> <li>LE および &lt;= および =&lt;</li> <li>NE および ^= および ^=^</li> </ul>	<p>対応する EGL 比較演算子 (SSA 用):</p> <ul style="list-style-type: none"> <li>=</li> <li>&gt;</li> <li>&lt;</li> <li>&gt;=</li> <li>&lt;=</li> <li>!=</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>EGL がサポートするのは、SSA 用比較演算子の 1 つのバリエーションのみです。</li> <li>EGL は、!= を、DL/I で許容される不等演算子に変換します。</li> </ul>	<p>特別な考慮事項なし。</p>
<p>VisualAge Generator ブール演算子 (SSA 用):</p> <ul style="list-style-type: none"> <li>AND および &amp;</li> <li>OR および  </li> </ul> <p>注: VisualAge Generator は、従属の OR はサポートしていません。</p>	<p>対応する EGL ブール演算子 (SSA 用):</p> <ul style="list-style-type: none"> <li>&amp;</li> <li>  (垂直バー)</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>EGL がサポートするのは、SSA 用ブール演算子の 1 つのバリエーションのみです。</li> <li>EGL は、従属の OR (# 記号) もサポートしています。</li> </ul>	<p>特別な考慮事項なし。</p>

表 111. 関数 - 変更済み DL/I ステートメントのセグメント検索指数 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>比較値項目</p> <p>注:</p> <ul style="list-style-type: none"> <li>比較値項目は、プログラム内の任意の項目にすることができます。</li> <li>その項目が修飾されていない場合、VisualAge Generator はまず、セグメント・レコード内で、現在の SSA に関連付けられている項目を探します。</li> </ul>	<p>比較値項目</p> <p>注:</p> <ul style="list-style-type: none"> <li>比較値項目は、リテラルまたはプログラム内の任意の項目にすることができます。</li> <li>その項目が修飾されていない場合、EGL はまず、ターゲット・セグメントを調べます。これは、階層内で最下位にあるセグメントです。</li> <li>比較値項目の前には、ホスト変数を表すセミコロンを付ける必要があります。例えば、次のようになります。</li> </ul> <p style="text-align: center;">:qualifier.itemName</p> <p>EGL は、生成時にセミコロンを除去します。</p>	<p>比較値項目が修飾されていない場合、マイグレーション・ツールは、現在の SSA に関連付けられている DL/I セグメント・レコードを検査して、項目がそのレコードに含まれているかどうかを判断します。含まれていれば、マイグレーション・ツールは、その DL/I セグメント・レコードを比較値項目の修飾子として組み込みます。</p> <p>DL/I セグメント・レコードが使用できない場合、あるいは比較値項目がそのレコード内に存在しない場合は、特別な考慮事項が適用されます。詳細および起こりうる問題については、101 ページの『DL/I I/O および比較値項目』を参照してください。</p>

## ステートメント

ステートメントに関するセクションは、次の表で構成されています。

- 一般規則 - データ項目修飾と数値リテラル (327 ページの表 112)
- 関数呼び出し (328 ページの表 113)
- 割り当て、MOVE、および MOVEA (328 ページの表 114)
- SET, 330 ページの表 115
- RETRIEVE と FIND (333 ページの表 116)
- EZE Aid、EZESYS、および入出力エラー値を含む IF、WHILE および TEST (335 ページの表 117)
- CALL (340 ページの表 118)
- DXFR (340 ページの表 119)
- XFER (341 ページの表 120)

表 112. ステートメント - 一般規則 - データ項目修飾と数値リテラル

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>項目修飾規則: 項目が修飾されていない場合、VisualAge Generator は次の順序で項目を検索します。</p> <ul style="list-style-type: none"> <li>関数のローカル・ストレージまたはパラメーター・リストにある項目。</li> <li>関数のローカル・ストレージまたはパラメーター・リストにある、関数の入出力オブジェクトとレコード。項目名がこのカテゴリ内で固有でない場合は、項目名を修飾する必要があります。</li> <li>プログラムの 1 次作業用ストレージ・レコード、呼び出し先パラメーター・リスト、テーブルおよび追加レコードのリスト、およびすべての入出力オブジェクト内にある、レコード、マップ、およびテーブル。項目名がこのカテゴリ内で固有でない場合は、項目名を修飾する必要があります。</li> <li>項目名がプログラム内で見付からず、プログラムが暗黙項目を許容している場合は、VisualAge Generator が項目の使用法に基づいてデータ項目定義を作成します。</li> </ul>	<p>項目修飾規則: 項目が修飾されていない場合、EGL は次の順序で項目を検索します。</p> <ul style="list-style-type: none"> <li>関数のローカル・ストレージまたはパラメーター・リストにある項目。</li> <li>関数の入出力ステートメント内で使用されているレコードと書式、および関数のローカル・ストレージまたはパラメーター・リストにあるレコード。項目名がこのカテゴリ内で固有でない場合は、項目名を修飾する必要があります。</li> <li>プログラムのデータ宣言、使用宣言、およびパラメーター・リストにあるレコード、書式、および dataTable。項目名がこのカテゴリ内で固有でない場合は、項目名を修飾する必要があります。</li> <li>EGL は、暗黙項目を許容しません。すべての項目を明示的に定義する必要があります。</li> </ul>	<p>詳細と起こりうる問題については、66 ページの『レコード内のレベル 77 項目』、および 86 ページの『プログラム内の暗黙データ項目』を参照してください。</p>
<p>数値リテラル:</p> <ul style="list-style-type: none"> <li>引用符で囲まれません。</li> <li>各国語に応じて、ピリオド (.) またはコンマ (,) のどちらかを小数点として使用します。</li> </ul>	<p>数値リテラル:</p> <ul style="list-style-type: none"> <li>引用符で囲まれません。</li> <li>ピリオドを小数点として使用する必要があります。生成時に、decimalSymbol ビルド記述子オプションによって、生成される Java コードまたは COBOL コード内でピリオドまたはコンマのどちらを小数点として使用するかが決まります。</li> </ul>	<p>マイグレーション・ツールは、書式変数フィールドの初期値を除いて、小数点として使用されるコンマをピリオドに変換します。</p>

表 113. ステートメント — 関数呼び出し

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
VAGen 構文の例: functionName( [argumentList] );	EGL 構文の例: functionName( [argumentList] );	同等な EGL のシステム・ライブラリー関数については、342 ページの『EZE ワード』を参照してください。  入出力エラー・ルーチンからの関数呼び出しについては、309 ページの表 105 を参照してください。
フロー・ステートメント内: functionName();	フロー・ステートメントはサポートされません。 <b>goto</b> functionName;	特別な考慮事項なし。

表 114. ステートメント — 割り当て、MOVE、および MOVEA

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
VAGen 構文の例: target = functionName ( [argumentList] );	EGL 構文の例: target = functionName ( [argumentList] );	同等な EGL のシステム・ライブラリー関数については、342 ページの『EZE ワード』を参照してください。
target = numericExpression; または target = numericExpression (R;	target = numericExpression ; または target = <b>mathLib.round</b> ( numericExpression ) ;	(R オプションが指定されている場合、マイグレーション・ツールはこのオプションを EGL の round システム関数に変換します。



表 114. ステートメント — 割り当て、MOVE、および MOVEA (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>target = source; または MOVE source [ TO ] target;</p> <p>注:</p> <ul style="list-style-type: none"> <li>ターゲットは、レコード、マップ、データ項目、または一部の EZE データ・ワードのいずれかです。</li> <li>ソースは、レコード、マップ、リテラル、データ項目、または一部の EZE データ・ワードのいずれかです。</li> <li>ターゲットがレコードまたはマップである場合は、ソースもレコードまたはマップであることが必要です。move corresponding が実行されます。</li> </ul>	<p>target = source ; または move source to target byName ; または move source to target ;</p> <p>注:</p> <ul style="list-style-type: none"> <li>代入ステートメントの場合: <ul style="list-style-type: none"> <li>ターゲットは、レコード、項目、または一部のシステム変数のいずれかです。ターゲットがレコードである場合は、ソースもレコードであることが必要です。ソースは、バイトごとにターゲットに移動されます。</li> <li>ソースは、レコード、リテラル、項目、または一部のシステム変数のいずれかです。</li> <li>書式は、代入ステートメント内では使用できません。</li> <li>代入ステートメントに対しては、move corresponding は実行されません。</li> </ul> </li> <li>move ステートメントの場合: <ul style="list-style-type: none"> <li>ターゲットとソースは、VisualAge Generator の代入ステートメントまたは MOVE ステートメントの場合と同じです。</li> <li>byName を指定した場合、EGL は move corresponding を実行します。</li> <li>修飾子を指定しない場合は、ソースのパーツ型に応じて、項目間移動または move corresponding のどちらかの移動が行われます。</li> </ul> </li> </ul> <p>データ変換と切り捨ての規則は、VisualAge Generator の場合と同じです。</p>	<p>マイグレーション・ツールは、代入ステートメントと move ステートメントをマイグレーションする際に、次の EGL 規則を考慮します。</p> <ul style="list-style-type: none"> <li>EGL は、項目間移動の場合は代入ステートメントの使用を選択します。</li> <li>VAGen の move corresponding の振る舞いを保持するために、レコードまたは書式が関係する移動には move byName ステートメントが必要です。</li> <li>修飾子を指定しない移動が許容され、ソースのパーツ型に応じて、項目間移動または move corresponding として処理されます。</li> </ul> <p>このため、マイグレーション・ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>次のいずれかの場合は、代入ステートメントに変換します。 <ul style="list-style-type: none"> <li>ソースまたはターゲットが EZE データ・ワードである (例: EZEAPP)。</li> <li>ソースがリテラルである。</li> <li>ソースまたはターゲットが修飾項目または添え字付き項目である。</li> <li>ソースまたはターゲットが、関数のパラメーター・リストまたはローカル・ストレージ内の項目である。</li> </ul> </li> <li>ソースまたはターゲットが関数の入出力オブジェクトであるか、関数のパラメーター・リストまたはローカル・ストレージ内のレコードである場合は、move byName に変換します。</li> <li>その他のすべての状況では、修飾子を指定しない move に変換します。</li> </ul> <p>詳細と起こりうる問題については、103 ページの『代入ステートメント』を参照してください。</p>

表 114. ステートメント — 割り当て、MOVE、および MOVEA (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>MOVEA source [T0] target; または MOVEA source [T0] target FOR occurrence;</p> <p>注: source は、配列またはスカラーのどちらかです。</p>	<p><b>move source to target for all ;</b> または <b>move source to target for occurrence ;</b></p>	<p>マイグレーション・ツールは、MOVEA ステートメントを変換して、for 修飾子を指定した move ステートメントにします。さらに、ツールは次の処理を行います。</p> <ul style="list-style-type: none"> <li>FOR occurrence オプションが VisualAge Generator 内で指定されていない場合は、for all オプションを組み込みます。</li> <li>FOR occurrence オプションが VisualAge Generator 内で指定されている場合は、for occurrence オプションを組み込みます。</li> <li>添え字が前に指定されていなかった場合は、ターゲットの添え字を 1 に設定します。</li> <li>ソースは配列またはスカラー項目のどちらかなので、ソースに対しては添え字を 1 に設定しません。</li> </ul>

表 115. ステートメント — SET

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>一般情報:</p> <ul style="list-style-type: none"> <li>単一の SET ステートメントに複数のオプションを指定するには、コンマまたはブランクを区切りに使用できます。</li> </ul>	<p>一般情報:</p> <ul style="list-style-type: none"> <li>単一の set ステートメントに複数のオプションを指定するには、コンマで区切る必要があります。</li> </ul>	<p>特別な考慮事項なし。</p>
<p>SET record SCAN;</p> <p>または</p> <p>SET record EMPTY;</p> <p>注: SET record EMPTY は、レベル 77 項目に影響しません。</p>	<p><b>set record position;</b></p> <p>または</p> <p><b>set record empty;</b></p>	<p>マイグレーション・ツールは、レベル 77 レコードに対するステートメントを追加しません。</p>
<p>SET sqlItem NULL;</p> <p>注: sqlItem は、SQL 行レコード内の項目、または関数の SQLITEM パラメーターです。</p>	<p><b>set sqlItem null;</b></p> <p>注: sqlItem は、SQL 行レコード内の nullable=yes 項目、または関数の NULL 可能パラメーターのどちらかです。</p>	<p>特別な考慮事項なし。</p>

表 115. ステートメント — SET (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<pre>SET map [ ALARM             [ CLEAR   EMPTY ] ] ;</pre> <p>注:</p> <ul style="list-style-type: none"> <li>• CLEAR は EMPTY は相互に排他的です。</li> <li>• ALARM と、CLEAR または EMPTY は、PAGE オプションと組み合わせることができます。</li> </ul>	<pre>set form [alarm             [ initial   empty ] ] ;</pre> <p>注:</p> <ul style="list-style-type: none"> <li>• <i>Initial</i> と <i>empty</i> は相互に排他的です。</li> <li>• PAGE オプションの置換表現は、他のオプションと組み合わせることはできません。</li> </ul>	<p>ALARM、CLEAR、または EMPTY が PAGE オプションと組み合わせて使用されている場合、マイグレーション・ツールは VAGen ステートメントを 2 つの EGL ステートメントに分割します。</p>
<pre>SET map PAGE ;</pre> <p>注: PAGE は、ALARM、および CLEAR または EMPTY のどちらかと組み合わせることができます。</p>	<pre>converseLib.clearScreen();            // display form または converseLib.pageEject();            // printer form</pre> <p>注: PAGE オプションの置換表現は、他のオプションと組み合わせることはできません。</p>	<p>マイグレーション・ツールは、SET map PAGE を次のようにマイグレーションします。</p> <ul style="list-style-type: none"> <li>• SET map PAGE を他のオプションと組み合わせて使用する場合、マイグレーション・ツールは VAGen ステートメントを 2 つの EGL ステートメントに分割します。</li> <li>• マップが表示マップである場合、マイグレーション・ツールはステートメントを <code>converseLib.clearScreen();</code> に変換します。</li> <li>• マップがプリンター・マップである場合、ツールはステートメントを <code>converseLib.pageEject();</code> に変換します。</li> <li>• マップを使用してマップ・タイプを判別できない場合、マイグレーション・ツールはステートメントを <code>converseLib.EZE_SETPAGE();</code> に変換します。</li> </ul> <p>詳細と起こりうる問題については、106 ページの『SET map PAGE ステートメント』を参照してください。</p>

表 115. ステートメント — SET (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<pre>SET mapItem   [ CURSOR   FULL       [ NORMAL   DEFINED ] ] ;</pre> <p>注:</p> <ul style="list-style-type: none"> <li>• <i>mapItem</i> は、マップのフィールド、または関数の MAPITEM パラメーターのどちらかです。</li> <li>• NORMAL と DEFINED は相互に排他的です。</li> <li>• CURSOR と FULL は、NORMAL または DEFINED のどちらかと組み合わせることができます。</li> <li>• VisualAge Generator は、印刷マップに対して CURSOR、FULL、NORMAL、および DEFINED の設定を許容しますが、これらの設定は印刷出力に影響しません。</li> </ul>	<pre>set formField   [ cursor   full       [ normal   initialAttributes ] ] ;</pre> <p>注:</p> <ul style="list-style-type: none"> <li>• <i>formField</i> は、書式の変数フィールド、または関数のフィールド・パラメーターのどちらかです。</li> <li>• <i>normal</i> と <i>initialAttributes</i> は相互に排他的です。</li> <li>• <i>cursor</i> と <i>full</i> は、<i>normal</i> または <i>initialAttributes</i> のどちらかと組み合わせることができます。</li> <li>• EGL は、印刷書式に対する <i>cursor</i>、<i>full</i>、<i>normal</i>、または <i>initialAttributes</i> の設定をサポートしません。</li> </ul>	<p>マイグレーション・ツールは、<i>formField</i> がテキスト書式と印刷書式のどちらにあるかに関係なく、それぞれのオプションと同等な EGL オプションへのマイグレーションを行います。詳細と起こりうる問題については、106 ページの『SET mapItem 属性』を参照してください。</p>
<pre>SET mapItem   [ CURSOR   FULL       color   extendedHighlight       MODIFIED       [ BRIGHT   DARK ]       [ PROTECT   AUTOSKIP ] ] ;</pre> <p>注:</p> <ul style="list-style-type: none"> <li>• <i>mapItem</i> は、マップのフィールド、または関数の MAPITEM パラメーターのどちらかです。</li> <li>• BRIGHT は DARK は相互に排他的です。</li> <li>• PROTECT と AUTOSKIP は相互に排他的です。</li> <li>• その他のオプションは、すべて組み合わせ可能です。</li> <li>• VisualAge Generator は、印刷マップに対してこれらの属性の設定を許容します。ただし、印刷出力に影響を及ぼすものは USCORE の拡張強調表示オプションのみです。</li> </ul>	<pre>set formField   [ cursor   full       color   extendedHighlight       modified       [ bold   invisible ]       [ protect   skip ] ] ;</pre> <p>注:</p> <ul style="list-style-type: none"> <li>• <i>formField</i> は、書式の変数フィールド、または関数のフィールド・パラメーターのどちらかです。</li> <li>• <i>Bold</i> と <i>invisible</i> は相互に排他的です。</li> <li>• <i>Protect</i> と <i>skip</i> は相互に排他的です。</li> <li>• その他のオプションは、すべて組み合わせ可能です。</li> <li>• 下線の拡張強調表示オプションを除いて、EGL は印刷書式に対してこれらの属性の設定をサポートしません。</li> </ul>	<p>マイグレーション・ツールは、<i>formField</i> がテキスト書式と印刷書式のどちらにあるかに関係なく、それぞれのオプションと同等な EGL オプションへのマイグレーションを行います。詳細と起こりうる問題については、106 ページの『SET mapItem 属性』を参照してください。色と <i>extendedHighlight</i> の情報については、この表の後続行を参照してください。</p>

表 115. ステートメント — SET (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
color: MONO   BLUE   GREEN   PINK   RED   TURQ   YELLOW   WHITE	color: defaultColor   blue   green   magenta   red   cyan   yellow   white	特別な考慮事項なし。
extendedHighlight: NOHILITE   BLINK   RVIDEO   USCORE	extendedHighlight: noHighLight   blink   reverse   underline	特別な考慮事項なし。

表 116. ステートメント — RETRIEVE と FIND

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
RETR dataItem1 table[.searchColumn] dataItem2 [ returnColumn ] ;  <b>注:</b> <ul style="list-style-type: none"> <li>• <i>searchColumn</i> が指定されていない場合、デフォルトはテーブルの先頭列です。</li> <li>• <i>returnColumn</i> が指定されていない場合、デフォルトはテーブルの 2 列目です。</li> </ul>	<b>if</b> ( <i>dataItem1</i> in <i>dataTable.searchColumn</i> ) <i>dataItem2</i> = <i>dataTable.returnColumn</i> [ <b>sysVar.arrayIndex</b> ]; <b>end</b>  <b>注:</b> <i>searchColumn</i> と <i>returnColumn</i> が必要です。	マイグレーション・ツールは、RETR ステートメントを <i>if</i> ステートメントと代入ステートメントに変換します。  マイグレーション時にテーブルが使用できない場合は、特別な考慮事項が適用されます。詳細と起こりうる問題については、105 ページの『RETR ステートメント』を参照してください。

表 116. ステートメント — RETRIEVE と FIND (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>FIND dataItem table[.searchColumn] trueStatement;</p> <p>または</p> <p>FIND dataItem table[.searchColumn] , falseStatement ;</p> <p>OR</p> <p>FIND dataItem table[.searchColumn] trueStatement [,] falseStatement ;</p> <p>注:</p> <ul style="list-style-type: none"> <li>• <i>searchColumn</i> が指定されていない場合、デフォルトはテーブルの先頭列です。</li> <li>• FIND がプログラム・フローの中で使用されている場合、<i>trueStatement</i> と <i>falseStatement</i> は、main 関数の名前、または EZECLOS のどちらかです。</li> <li>• FIND が関数の中で使用されている場合、<i>trueStatement</i> と <i>falseStatement</i> は、任意の関数の名前、EZECLOS、EZEFL0、または EZERTN のいずれかです。</li> </ul>	<p>if (dataItem in dataTable.searchColumn) EGLtrueStatement ; end</p> <p>または</p> <p>if (dataItem in dataTable.searchColumn) else EGLfalseStatement ; end</p> <p>または</p> <p>if (dataItem in dataTable.searchColumn) EGLtrueStatement ; else EGLfalseStatement ; end</p> <p>注: <i>searchColumn</i> が必要です。</p>	<p>マイグレーション・ツールは、FIND ステートメントを変換して、if ステートメント、および true と false の両ステートメントに相当する EGL ステートメントにします。trueStatement および falseStatement から対応する EGL ステートメントへの変換については、この表の後続行を参照してください。</p>
<p>フロー内の true/falseStatement:</p> <ul style="list-style-type: none"> <li>• functionName() (main のみ)</li> <li>• EZECLOS</li> </ul>	<p>対応する同等な EGL ステートメント:</p> <ul style="list-style-type: none"> <li>• goto functionName;</li> <li>• exit program;</li> </ul>	<p>特別な考慮事項なし。</p>
<p>関数内の true/falseStatement:</p> <ul style="list-style-type: none"> <li>• functionName (任意の関数)</li> <li>• EZECLOS</li> <li>• EZEFL0</li> <li>• EZERTN</li> </ul>	<p>対応する同等な EGL ステートメント:</p> <ul style="list-style-type: none"> <li>• functionName();</li> <li>• exit program;</li> <li>• exit stack;</li> <li>• return;</li> </ul>	<p>特別な考慮事項なし。</p>



表 117. ステートメント — EZEAD、EZESYS、および入出力エラー値を含む IF、WHILE および TEST

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<pre>IF logicalExpression ;   { statement ; } [ ELSE;   { statement ; } ] END;</pre>	<pre>if ( EGLLogicalExpression )   { EGLStatement ; } [ else   { EGLStatement ; } ] end</pre>	VAGen 論理式と EGL 論理式の関係については、この表の後続行を参照してください。
<pre>WHILE logicalExpression ;   { statement ; } END;</pre>	<pre>while ( EGLLogicalExpression )   { EGLStatement ; } end</pre>	VAGen 論理式と EGL 論理式の関係については、この表の後続行を参照してください。
<pre>TEST testCondition   trueStatement ;  TEST testCondition   , falseStatement ;  TEST testCondition   trueStatement   [, ] falseStatement ;</pre> <p>注:</p> <ul style="list-style-type: none"> <li>• TEST ステートメントは、IF ... IS ステートメントと同様です。この例外は TEST mapItem nnn、+nnn、および -nnn で、これと同等な IF ステートメントはありません。</li> <li>• TEST がプログラム・フローの中で使用されている場合、<i>trueStatement</i> と <i>falseStatement</i> は、main 関数の名前、または EZECLAS のどちらかです。</li> <li>• TEST が関数の中で使用されている場合、<i>trueStatement</i> と <i>falseStatement</i> は、任意の関数の名前、EZECLAS、EZEFLO、または EZERTN のいずれかです。</li> </ul>	<pre>if ( EGLLogicalExpression )   EGLtrueStatement ; end  if ( EGLLogicalExpression ) else   EGLfalseStatement ; end  if ( EGLLogicalExpression )   EGLtrueStatement ; else   EGLfalseStatement ; end</pre>	<p>TEST mapItem nnn、+nnn、および -nnn を除き、マイグレーション・ツールは TEST ステートメントを変換して、同等な if ... is ステートメントと、EGL の同等な true ステートメントおよび false ステートメントにします。</p> <p>VAGen 論理式と EGL 論理式の関係については、この表の後続行を参照してください。</p> <p><i>trueStatement</i> および <i>falseStatement</i> から対応する EGL ステートメントへの変換については、この表の後続行を参照してください。</p>
<p>IF と WHILE 用の VisualAge Generator プール演算子:</p> <ul style="list-style-type: none"> <li>• AND</li> <li>• OR</li> </ul>	<p>対応する if と while 用の EGL プール演算子:</p> <ul style="list-style-type: none"> <li>• &amp;&amp;</li> <li>•   </li> </ul>	特別な考慮事項なし。

表 117. ステートメント — *EZE Aid*、*EZE Sys*、および入出力エラー値を含む *IF*、*WHILE* および *TEST* (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p><i>IF</i> と <i>WHILE</i> 用の VisualAge Generator 比較演算子:</p> <ul style="list-style-type: none"> <li>• EQ および =</li> <li>• NE および ^=</li> <li>• LE および &lt;= および =&lt;</li> <li>• LT および &lt;</li> <li>• GE および &gt;= および =&gt;</li> <li>• GT および &gt;</li> </ul>	<p>対応する <i>if</i> と <i>while</i> 用の EGL 比較演算子:</p> <ul style="list-style-type: none"> <li>• ==</li> <li>• !=</li> <li>• &lt;=</li> <li>• &lt;</li> <li>• &gt;=</li> <li>• &gt;</li> </ul>	<p>特別な考慮事項なし。</p>
<p><i>IF</i> と <i>WHILE</i> 用の VisualAge Generator 状態演算子:</p> <ul style="list-style-type: none"> <li>• IS</li> <li>• NOT</li> </ul>	<p>対応する <i>if</i> と <i>while</i> 用の EGL 状態演算子:</p> <ul style="list-style-type: none"> <li>• is</li> <li>• not</li> </ul>	<p>マイグレーション・ツールは、VAGen の <i>TEST</i> ステートメントを EGL の <i>if ... is</i> ステートメントに常にマイグレーションします。</p>
<p><i>IF</i> と <i>WHILE</i> 用の VisualAge Generator 配列演算子:</p> <ul style="list-style-type: none"> <li>• IN</li> </ul>	<p>対応する <i>if</i> と <i>while</i> 用の EGL 状態演算子:</p> <ul style="list-style-type: none"> <li>• in</li> </ul>	<p>特別な考慮事項が適用されます。 107 ページの『IN がリテラルかスカラーかの検査』を参照してください。</p>
<p>VisualAge Generator <i>mapItem</i> の状態条件:</p> <ul style="list-style-type: none"> <li>• BLANK または BLANKS</li> <li>• CURSOR</li> <li>• DATA</li> <li>• MODIFIED</li> <li>• NULL または NULLS</li> <li>• NUMERIC</li> </ul>	<p>対応する EGL <i>formField</i> の状態条件:</p> <ul style="list-style-type: none"> <li>• blanks</li> <li>• cursor</li> <li>• data</li> <li>• modified</li> <li>• blanks</li> <li>• numeric</li> </ul>	<p>マイグレーション・ツールは、同等な EGL 状態条件への変換を行います。</p> <p><i>mapItem</i> NULL には特別な考慮事項が適用されます。詳細と起こりうる問題については、108 ページの『SQL 項目とマップ項目が NULL かどうかの検査』を参照してください。</p>

表 117. ステートメント — EZE Aid、EZESYS、および入出力エラー値を含む IF、WHILE および TEST (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>TEST ステートメントの特殊な mapItem の状態条件: nnn   +nnn   -nnn</p> <p>注: この条件は、ユーザーが入力したデータの長さを nnn と比較します。nnn、+nnn、または -nnn に対応して、=、&gt;、または &lt; のテストが行われます。</p>	<p>EGL は、この状態条件のサポートを直接は提供していません。ただし、次のようにすれば同等な機能を実現できます。</p> <ul style="list-style-type: none"> <li>システム・ライブラリー関数 <i>converseLib.fieldInputLength</i> を使用します。この関数は、ユーザーが入力したデータの長さを戻します。</li> <li>if ステートメントを使用して、得られた長さを nnn、+nnn、または -nnn に対応する ==、&gt;、または &lt; によって比較します。</li> </ul>	<p>プログラムをマイグレーションする際に、マイグレーション・ツールは常に次の宣言を組み込みます。</p> <pre>&lt;custPrefix&gt;EZE_ITEMLEN</pre> <p>マイグレーション・ツールは、TEST nnn、+nnn、または -nnn に対して次の処理を行います。</p> <ul style="list-style-type: none"> <li>TEST ステートメントの直前にステートメントを追加して、次の設定を行います。</li> </ul> <pre>&lt;custPrefix&gt;EZE_ITEMLEN</pre> <p>この設定には、システム・ライブラリー関数 <i>converseLib.fieldInputLength(item)</i> を使用します。</p> <ul style="list-style-type: none"> <li>TEST ステートメントを if ステートメントに変換し、</li> </ul> <pre>&lt;custPrefix&gt;EZE_ITEMLEN</pre> <p>を == nnn、&gt; nnn、または &lt; nnn と比較します。</p>
<p>VisualAge Generator マップの状態条件:</p> <ul style="list-style-type: none"> <li>MODIFIED</li> </ul>	<p>対応する EGL 書式の状態条件:</p> <ul style="list-style-type: none"> <li>modified</li> </ul>	<p>特別な考慮事項なし。</p>
<p>VisualAge Generator EZE Aid の状態条件:</p> <ul style="list-style-type: none"> <li>ENTER</li> <li>BYPASS</li> <li>PAn (ここで、n = 1, 2, 3)</li> <li>PFn (ここで n は 1 から 24)</li> <li>PA</li> <li>PF</li> </ul>	<p>対応する EGL <i>converseVar.eventKey</i> の状態条件:</p> <ul style="list-style-type: none"> <li>enter</li> <li>bypass</li> <li>pan (ここで、n = 1, 2, 3)</li> <li>pfm (ここで n は 1 から 24)</li> <li>pakey</li> <li>pfkey</li> </ul>	<p>特別な考慮事項なし。</p>
<p>VisualAge Generator <i>sqlItem</i> の状態条件:</p> <ul style="list-style-type: none"> <li>BLANK または BLANKS</li> <li>NULL</li> <li>NUMERIC</li> <li>TRUNC</li> </ul>	<p>対応する EGL <i>sqlItem</i> の状態条件:</p> <ul style="list-style-type: none"> <li>blanks</li> <li>null</li> <li>numeric</li> <li>trunc</li> </ul>	<p>マイグレーション・ツールは、同等な EGL 状態条件への変換を行います。</p> <p><i>sqlItem</i> NULL には特別な考慮事項が適用されます。詳細と起こりうる問題については、108 ページの『SQL 項目とマップ項目が NULL かどうかの検査』を参照してください。</p>

表 117. ステートメント — *EZE Aid*、*EZESYS*、および入出力エラー値を含む *IF*、*WHILE* および *TEST* (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>VisualAge Generator レコードの状態条件:</p> <ul style="list-style-type: none"> <li>• DED</li> <li>• DUP</li> <li>• EOF</li> <li>• ERR</li> <li>• FMT</li> <li>• FNA</li> <li>• FNF</li> <li>• FUL</li> <li>• HRD</li> <li>• LOK</li> <li>• NRF</li> <li>• UNQ</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• DUP は、SQL レコードと非 SQL レコードの両方に対してサポートされます。</li> <li>• SQL レコードの場合、DUP と UNQ は同等で、常にハード・エラーです。</li> <li>• 非 SQL レコードの場合、DUP と UNQ は同等ではなく、両方ともソフト・エラーです。</li> <li>• LOK は、OS/400 上でのみサポートされるソフト・エラーです。</li> </ul>	<p>対応する EGL レコードの状態条件:</p> <ul style="list-style-type: none"> <li>• deadLock</li> <li>• duplicate または unique</li> <li>• endOfFile</li> <li>• ioError</li> <li>• invalidFormat</li> <li>• fileNotAvailable</li> <li>• fileNotFound</li> <li>• full</li> <li>• hardIOError</li> <li>• deadLock</li> <li>• noRecordFound</li> <li>• unique</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• <i>duplicate</i> は、非 SQL レコードのみに対してサポートされるソフト・エラーです。</li> <li>• <i>unique</i> は、SQL レコードと非 SQL レコードの両方に対してサポートされるハード・エラーです。</li> <li>• LOK は <i>deadLock</i> に変換され、常にハード・エラーです。</li> </ul>	<p>マイグレーション・ツールは、同等な EGL 状態条件への変換を行います。</p> <p>レコード・タイプに応じて、DUP のマイグレーションには特別な考慮事項が適用されます。詳細と起こりうる問題については、110 ページの『入出力エラー値 UNQ および DUP』を参照してください。</p> <p>LOK のマイグレーションにも、特別な考慮事項が適用されます。詳細と起こりうる問題については、112 ページの『入出力エラー値 LOK』を参照してください。</p>
<p>UI レコード状態条件:</p> <ul style="list-style-type: none"> <li>• MODIFIED</li> </ul>	<p>対応する EGL VGUI レコードの条件:</p> <ul style="list-style-type: none"> <li>• modified</li> </ul>	<p>特別な考慮事項なし。</p>
<p>VisualAge Generator <i>dataItem</i> の状態条件:</p> <ul style="list-style-type: none"> <li>• BLANK または BLANKS</li> <li>• NUMERIC</li> </ul>	<p>対応する EGL <i>dataItem</i> の状態条件:</p> <ul style="list-style-type: none"> <li>• blanks</li> <li>• numeric</li> </ul>	<p>特別な考慮事項なし。</p>

表 117. ステートメント — *EZEAD*、*EZESYS*、および入出力エラー値を含む *IF*、*WHILE* および *TEST* (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>VisualAge Generator EZESYS の状態条件:</p> <ul style="list-style-type: none"> <li>• AIX</li> <li>• AIXCICS</li> <li>• HP</li> <li>• IMSBMP</li> <li>• IMSVS</li> <li>• MVS BATCH</li> <li>• MVSCICS</li> <li>• NTCICS</li> <li>• OS2</li> <li>• OS2CICS</li> <li>• OS2GUI</li> <li>• OS400</li> <li>• SCO</li> <li>• SOLACICS</li> <li>• SOLARIS</li> <li>• TSO</li> <li>• VMCMS</li> <li>• VMBATCH</li> <li>• VSEBATCH</li> <li>• VSECICS</li> <li>• WINGUI</li> <li>• WINNT</li> <li>• ITF</li> </ul>	<p>対応する <i>sysVar.systemType</i> の状態条件:</p> <ul style="list-style-type: none"> <li>• aix</li> <li>• AIXCICS</li> <li>• hpux</li> <li>• imsbmp</li> <li>• imsvs</li> <li>• zosbatch</li> <li>• zoscics</li> <li>• NTCICS</li> <li>• OS2</li> <li>• OS2CICS</li> <li>• OS2GUI</li> <li>• iseriesc</li> <li>• SCO</li> <li>• SOLACICS</li> <li>• solaris</li> <li>• TSO</li> <li>• VMCMS</li> <li>• VMBATCH</li> <li>• vsebatch</li> <li>• vsecics</li> <li>• WINGUI</li> <li>• win</li> <li>• debug</li> </ul>	<p>マイグレーション・ツールは、同等な EGL 状態条件への変換を行います。</p> <p>EZESYS の状態の検査には、特別な考慮事項が適用されます。詳細と起こりうる問題については、114 ページの『EZESYS』を参照してください。</p> <p>注: 一部の VAGen ランタイム環境はサポートされません。ただし、マイグレーション・ツールは常に同等な値への変換を行います (その値が EGL では有効でなくても)。前の VAGen 値がサポートされていない場合は、「問題」ビューにエラーが表示されます。</p>
<p>フロー内の <b>true/falseStatement:</b></p> <ul style="list-style-type: none"> <li>• functionName() (main のみ)</li> <li>• EZECLAS</li> </ul>	<p>対応する同等な EGL ステートメント:</p> <ul style="list-style-type: none"> <li>• <b>goto</b> functionName ;</li> <li>• <b>exit program;</b></li> </ul>	<p>特別な考慮事項なし。</p>
<p>関数内の <b>true/falseStatement:</b></p> <ul style="list-style-type: none"> <li>• functionName (任意の関数)</li> <li>• EZECLAS</li> <li>• EZEFLD</li> <li>• EZERTN</li> </ul>	<p>対応する同等な EGL ステートメント:</p> <ul style="list-style-type: none"> <li>• functionName();</li> <li>• <b>exit program;</b></li> <li>• <b>exit stack;</b></li> <li>• <b>return;</b></li> </ul>	<p>特別な考慮事項なし。</p>

表 118. ステートメント — CALL

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
CALL <i>programName</i> <i>argument</i> [ { [,] <i>argument</i> } ] [ ( <i>options</i> ) ] ;  または  CALL <i>serviceRoutine</i> <i>argument</i> [ { [,] <i>argument</i> } ] [ ( <i>options</i> ) ] ;  <b>注:</b> 引数を区切るコンマはオプションです。	<b>call</b> <i>programName</i> <i>argument</i> [ { , <i>argument</i> } ] [ <i>options</i> ] ;  <b>注:</b> <ul style="list-style-type: none"> <li>引数を区切るコンマは必要です。</li> <li><i>programName</i> が予約語であってはなりません。プログラムが非 EGL プログラムならば、リンケージ・テーブル・エントリーを使用して実名を指定します。</li> </ul>	オプションから、対応する EGL のステートメントまたはオプションへの変換については、この表の後続行を参照してください。  サービス・ルーチン用の CALL ステートメントのマイグレーションについては、353 ページの『サービス・ルーチン』を参照してください。
REPLY オプション	REPLY オプションが VisualAge Generator で指定されている場合、対応する EGL のステートメントは次のとおりです。  <b>try</b> <b>call</b> <i>programName</i> <i>argument</i> [ { , <i>argument</i> } ] [ <i>otherOptions</i> ] ; <b>end</b>	REPLY オプションが指定されている場合、マイグレーション・ツールは <i>try...end</i> ブロックを組み込みます。
otherOptions: <ul style="list-style-type: none"> <li>NOMAPS</li> <li>NONCSP</li> </ul>	対応する EGL <i>otherOptions</i> : <ul style="list-style-type: none"> <li>noRefresh</li> <li>externallyDefined</li> </ul>	特別な考慮事項なし。

表 119. ステートメント — DXFR

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
DXFR <i>target</i> [ <i>recordName</i> ] [ (NONCSP) ] ;  ここで、 <i>target</i> は次のとおりです。  <i>programName</i> または  EZEAPP  <b>注:</b> <ul style="list-style-type: none"> <li>任意のレコードを渡すことができます。</li> <li>作業用ストレージ・レコードが渡される場合に、レベル 77 項目は含まれません。</li> </ul>	<b>transfer to program</b> <i>target</i> [ <b>passing</b> <i>recordName</i> ] [ <b>externallyDefined</b> ] ;  ここで、 <i>target</i> は次のとおりです。  <i>programName</i> または  <i>sysVar.transferName</i>  <b>注:</b> 任意のレコードを渡すことができます。	特別な考慮事項なし。



表 120. ステートメント — XFER

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>バリエーション 1 - Transfer へのマイグレーション (マップまたは UI レコードなし)</p> <p>XFER target [ recordName ] [ (NONCSP) ;</p> <p>ここで、target は次のとおりです。</p> <p>transactionName</p> <p>または</p> <p>EZEAPP</p> <p>注:</p> <ul style="list-style-type: none"> <li>このフォーマットの XFER には、マップまたは UI レコードは指定されません。</li> <li>任意のレコードを渡すことができます。作業用ストレージ・レコードが渡される場合に、レベル 77 項目は含まれません。</li> <li>transactionName は、非トランザクションのランタイム環境ではプログラム名です。</li> </ul>	<p>transfer ステートメントの EGL 構文は、次のとおりです。</p> <p><b>transfer to transaction target</b> [ passing recordName ] [ externallyDefined ] ;</p> <p>ここで、target は次のとおりです。</p> <p>transactionName</p> <p>または</p> <p>sysVar.transferName</p> <p>注:</p> <ul style="list-style-type: none"> <li>任意のレコードを渡すことができます。</li> <li>transactionName は、非トランザクションのランタイム環境ではプログラム名です。</li> </ul>	<p>ステートメントにコンマが入っていない場合、マイグレーション・ツールは XFER を EGL の transfer to transaction ステートメントに変換します。</p>

表 120. ステートメント — XFER (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>バリエーション 2 - Show へのマイグレーション (マップを指定した XFER または UI レコードを指定した XFER)</p> <pre>XFER target   [ recordName ]   , mapName   [ (NONCSP ) ; OR XFER target   [ recordName ]   , UIRecordName ;</pre> <p>ここで、<i>target</i> は次のとおりです。</p> <ul style="list-style-type: none"> <li>• transactionName</li> <li>• EZEAPP</li> <li>• UI レコードを指定した XFER の ‘ ’</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• 任意のレコードを渡すことができます。作業用ストレージ・レコードが渡される場合に、レベル 77 項目は含まれません。</li> <li>• transactionName は、非トランザクション・ターゲット環境ではプログラム名です。</li> <li>• (NONCSP は、マップを指定した XFER でのみサポートされます。)</li> </ul>	<p><i>show</i> ステートメントの場合、EGL 構文はフォームまたは VGUI レコードが使用されているかどうかによって異なります。</p> <pre>show formName   returning to target   [ passing recordName ]   [ externallyDefined ] ; OR show UIRecordName   [ returning to target ]   [ passing recordName ] ;</pre> <p>ここで、<i>target</i> は次のとおりです。</p> <ul style="list-style-type: none"> <li>• transactionName</li> <li>• sysVar.transferName</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• 任意のレコードを渡すことができます。</li> <li>• transactionName は、非トランザクション・ターゲット環境ではプログラム名です。</li> <li>• VGUI レコードを指定した <i>show</i> の場合、ターゲットが ‘ ’ であれば、ターゲット文節への戻りは省略されます。</li> </ul>	<p>ステートメントにコンマが入っている場合、マイグレーション・ツールは XFER ステートメントを EGL <i>show</i> ステートメントに変換します。</p>

## EZE ワード

### プログラム・フローの EZE ワード

表の左側の欄には、VisualAge Generator 4.5 の EZE ワードを示します。右側の欄には、マイグレーション・ツールが EZE ワードを EGL に変換した結果を示します。

表 121. プログラム・フローの EZE ワード

VisualAge Generator 4.5 の EZE ワード	EGL
EZECLOS	<p>これは、ロケーションによって次のように異なります。</p> <ul style="list-style-type: none"> <li>入出力エラー・ルーチンとして使用されている場合、マイグレーション・ツールは EZECLOS を try...end ブロック内で次のように変換します。</li> </ul> <pre>onException exit program;</pre> <ul style="list-style-type: none"> <li>他の場所で使用されている場合 (TEST や FIND の true または false のオペランドとして使用されている場合など)、マイグレーション・ツールは EZECLOS を次のように変換します。</li> </ul> <pre>exit program;</pre> <p>注: 出口プログラムの戻りコードは sysVar.returnValue です。これは、EZERCODE と同等です。このデフォルトは、VisualAge Generator と同じ機能を実現します。</p>
EZEFLO 注: EZEFLO は、フロー・ステートメント内では使用できません。	<p>これは、ロケーションによって次のように異なります。</p> <ul style="list-style-type: none"> <li>入出力エラー・ルーチンとして使用されている場合、マイグレーション・ツールは EZEFLO を try...end ブロック内で次のように変換します。</li> </ul> <pre>onException exit stack;</pre> <ul style="list-style-type: none"> <li>他の場所で使用されている場合 (TEST や FIND の true または false のオペランドとして使用されている場合など)、マイグレーション・ツールは EZEFLO を次のように変換します。</li> </ul> <pre>exit stack;</pre>

表 121. プログラム・フローの EZE ワード (続き)

VisualAge Generator 4.5 の EZE ワード	EGL
<p>EZERTN または EZERTN(return value) 注:</p> <ul style="list-style-type: none"> <li>• EZERTN は、フロー・ステートメント内では使用できません。</li> <li>• EZERTN(return value) は、入出力エラー・ルーチンとしては使用できません。</li> </ul>	<p><b>EZERTN</b> — これは、ロケーションに応じて次のようになります。</p> <ul style="list-style-type: none"> <li>• 入出力エラー・ルーチンとして使用されている場合、マイグレーション・ツールは try...end ブロックを組み込みますが、onException ステートメントを省略します。</li> <li>• 他の場所で使用されている場合、マイグレーション・ツールは EZERTN を次のように変換します。</li> </ul> <pre>return;</pre> <p>または</p> <pre>return(returnValue);</pre> <p>注: returnValue が EZESYS である場合は、他の考慮事項について EZESYS を参照してください。</p>

## SQL EZE ワード

表の左側の欄には、VisualAge Generator 4.5 の EZE ワードを示します。右側の欄には、マイグレーション・ツールが EZE ワードを EGL に変換した結果を示します。

表 122. SQL EZE ワード

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZECONCT	<p>VGLib.connectionService</p> <p>引数は VAGen の場合と同じです。ただし、デバッグと Java 生成の場合、作業単位引数の値の一部はサポートされません。JDBC は、単一フェーズ・コミットのみをサポートします。</p>
EZESQCOD	<p>sysVar.sqlcode</p>
<p>EZESQISL</p> <p>注:</p> <ul style="list-style-type: none"> <li>• VisualAge Generator 4.5 の場合、EZESQISL は ODBC 環境での使用がサポートされています。</li> <li>• それ以外のすべての環境では、EZESQISL はサポートされないか VisualAge Generator 内で無視されますが、互換性のために保持されます。</li> </ul>	<p>VGVar.sqlIsolationLevel</p> <p>注: EGL は次の場合に、VGVar.sqlIsolationLevel をサポートします。</p> <ul style="list-style-type: none"> <li>• VAGen 互換モードが選択済みかどうかに関わらず connectionService の場合</li> <li>• VAGen 互換モードが選択済みのときにのみ connect の場合</li> </ul>

表 122. SQL EZE ワード (続き)

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZESQLCA	sysVar.sqlca 注: sysVar.sqlca は、EGL では部分的にのみサポートされます。デバッグおよび Java 生成の場合、EGL は VGVar.sqlerrmc と VGVar.sqlwarn[7] の値を格納するフィールドを sysVar.sqlca 内で設定しません。
EZESQRD3	VGVar.sqlerrd[3] 注: マイグレーション・ツールは、これを配列参照に変更します。
EZESQRRM	VGVar.sqlerrmc 注: sqlerrmc は、デバッグまたは Java 生成に対してはサポートされません。
EZESQWN1	VGVar.sqlwarn[2] 注: マイグレーション・ツールは、これを配列参照に変更します。
EZESQWN6	VGVar.sqlwarn[7] 注: マイグレーション・ツールは、これを配列参照に変更します。sqlwarn[7] は、デバッグまたは Java 生成に対してはサポートされません。
N/A	VGVar.sqlState 注: これは EGL 用の新規のもので、VisualAge Generator 4.5 に同等なワードはありません。マイグレーション・ツールによってこれに変換されるワードはありません。

## DL/I EZE ワード

表の左側の欄には、VisualAge Generator 4.5 EZE ワードを示します。右側の欄には、マイグレーション・ツールが EZE ワードを EGL に変換した結果を示します。

表 123. DL/I EZE ワード

VisualAge Generator 4.5 の EZE ワード	EGL
EZEDLCER	dliVar.cicsError
EZEDLCON	dliVar.cicsCondition
EZEDLDBD	dliVar.dbName
EZEDLERR	dliVar.handleHardDLIErrors
EZEDLKEY	dliVar.keyArea[1:dliVar.keyAreaLen]
EZEDLKYL	dliVar.keyAreaLen
EZEDLLEV	dliVar.segmentLevel

表 123. DLI EZE ワード (続き)

VisualAge Generator 4.5 の EZE ワード	EGL
<b>EZEDLPCB</b> <b>注:</b> これは配列です。デフォルトの添え字は 1 です。	<p>マイグレーション・ツールは、プログラムの PSB を <i>psb</i> に宣言する変数を常にセットします。このため、ステートメントでは、EZEDLPCB[<i>n</i>] は以下のように変換されます。</p> <ul style="list-style-type: none"> <li>• EZEDLPCB[0] は <i>psb.iopcb</i> に変換されます。</li> <li>• 1 はデフォルトの添え字であるため、EZEDLPCB は <i>psb.pcb1</i> に変換されます。</li> <li>• EZEDLPCB[<i>n</i>] (ここで <i>n</i> は数値リテラル) は <i>psb.pcbn</i> に変換されます。</li> </ul> <p>プログラムの呼び出し先パラメーター・リストでは、特別な考慮事項が適用されます。詳しくは、298 ページの表 99 を参照してください。</p>
<b>EZEDLPRO</b>	<i>dliVar.procOptions</i>
<b>EZEDLPSB</b>	<p><i>call</i> ステートメント以外のステートメントでは、EZEDLPSB は次のように変換されます。</p> <p><i>dliLib.psbData.psbName</i></p> <p><i>call</i> ステートメントでは、EZEDLPSB は次のように変換されます。</p> <p><i>dliLib.psbData</i></p> <p>プログラムの呼び出し先パラメーター・リストでは、特別な考慮事項が適用されます。詳しくは、298 ページの表 99 を参照してください。</p>
<b>EZEDLRST</b>	<i>dliVar.cicsRestart</i>
<b>EZEDLSEG</b>	<i>dliVar.segmentName</i>
<b>EZEDLSSG</b>	<i>dliVar.numSensitiveSegs</i>
<b>EZEDLSTC</b>	<i>dliVar.statusCode</i>
<b>EZEDLTRM</b> <b>注:</b> EZEDLTRM は EZECNVCM と同等です。両方とも、 <i>converseVar.commitOnConverse</i> に変換されます。	<i>converseVar.commitOnConverse</i>

## 日付と時刻の EZE ワード

表の左側の欄には、VisualAge Generator 4.5 の EZE ワードを示します。右側の欄には、マイグレーション・ツールが EZE ワードを EGL に変換した結果を示します。



表 124. 日付と時刻の EZE ワード

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZEDAY	VGVar.currentShortJulianDate
EZEDAYL	VGVar.currentJulianDate
EZEDAYLC	VGVar.currentFormattedJulianDate
EZEDTE	VGVar.currentShortGregorianDate
EZEDTEL	VGVar.currentGregorianDate
EZEDTELC	VGVar.currentFormattedGregorianDate
EZETIM	VGVar.currentFormattedTime

## その他のデータの EZE ワード

表の左側の欄には、VisualAge Generator 4.5 の EZE ワードを示します。右側の欄には、マイグレーション・ツールが EZE ワードを EGL に変換した結果を示します。

表 125. その他のデータの EZE ワード

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZE Aid	converseVar.eventKey
EZE APP	sysVar.transferName
EZE CNVCM	converseVar.commitOnConverse
EZE CONVT	sysVar.callConversionTable
record.EZEDEST	record.resourceAssociation 注: 修飾は引き続きレコード名です。
EZEDESTP	converseVar.printerAssociation
EZE FEC	VGVar.handleHardIOErrors
EZE LOC	sysVar.remoteSystemID
EZE LTERM	テキスト・プログラム用 sysVar.terminalID。  VGWebTransaction 用 sysVar.conversationID。  注: <ul style="list-style-type: none"> <li>テキスト・プログラムでは、sysVar.terminalID と sysVar.conversationID の両方が terminalID 情報を提供します。</li> <li>VGWebTransaction プログラムでは、sysVar.terminalID と sysVar.conversationID の両方が conversationID 情報を提供します。</li> </ul>

表 125. その他のデータの EZE ワード (続き)

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZEMNO	<ul style="list-style-type: none"> <li>• EZEMNO が MOVE または割り当てのターゲット として使用されている場合は、次のようになります。 <ul style="list-style-type: none"> <li>– EZEMNO が 9999 以外の数値リテラルから設定されている場合、EZEMNO は次のようになります。  <pre>converseLib.validationFailed(numericLiteral);</pre> </li> <li>– EZEMNO が数値リテラル 9999 から設定されている場合、EZEMNO は次のようになります。  <pre>converseLib.validationFailed();</pre> </li> <li>– EZEMNO が項目から設定されている場合、EZEMNO は次のようになります。  <pre>if (itemName == 9999)     converseLib.validationFailed(); else     converseLib.validationFailed(itemName); end</pre> </li> </ul> </li> <li>• EZEMNO がその他の場所で使用されている場合は、次のように置き換えられます。  <pre>converseVar.validationMsgNum</pre> </li> </ul>
EZEMSG <b>注:</b> データ項目としての EZEMSG は、マップに配置されている場合のみ存在します。複数のマップに配置されている場合は、EZEMSG を修飾する必要があります。	<custPrefix>EZEMSG <b>注:</b> <ul style="list-style-type: none"> <li>• &lt;custPrefix&gt; と EZEMSG の間にドットは入りません。</li> <li>• EZEMSG が関数の中で使用されている場合、マイグレーション・ツールは、これらの関数の中で EZEMSG に使用されていたものと同じ修飾を &lt;custPrefix&gt;EZEMSG に対して維持します。例えば、xxxx.EZEMSG は xxxx.&lt;custPrefix&gt;EZEMSG になります。</li> <li>• EZEMSG がマップ内で使用されている場合、マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>– フィールド名を &lt;custPrefix&gt;EZEMSG に変更します。</li> <li>– マップの msgField プロパティを &lt;custPrefix&gt;EZEMSG に設定します。</li> </ul> </li> </ul>
EZEOVER	VGVar.handleOverflow
EZEOVERS	sysVar.overflowIndicator
EZERCODE <b>注:</b> 推奨されませんが、VisualAge Generator では EZERCODE に対して負の値と 512 を超える値を指定できます。	sysVar.returnValue <b>注:</b> EGL では、sysVar.returnValue に対して負の値または 512 を超える値を指定できません。
EZEREPLY	VGVar.handleSysLibraryErrors

表 125. その他のデータの EZE ワード (続き)

VisualAge Generator 4.5 の EZE ワード	EGL の定義
<b>EZERT2</b> <b>注:</b> VisualAge Generator 4.5 では、EZERT2 は MQ Series アクセス用の条件コード としてのみ使用されます。	VGVar.mqConditionCode
<b>EZERT8</b> <b>注:</b> EZERT8 が設定される対象は、次のとおりです。 <ul style="list-style-type: none"> <li>• REPLY オプションが指定されている場合は、CALL ステートメント。</li> <li>• EZEREPLY が 1 に設定されている場合は、EZE システム関数呼び出し。</li> <li>• シリアル・レコード、索引付きレコード、相対レコード、およびメッセージ・キュー・レコードの入出力オプション。</li> </ul>	sysVar.errorCode <b>注:</b> sysVar.errorCode が設定される対象は、次のとおりです。 <ul style="list-style-type: none"> <li>• すべての CALL ステートメント。</li> <li>• すべての sysLib システム関数呼び出し。</li> <li>• 一部の strLib、mathLib、および VGLib システム機能呼び出し</li> <li>• シリアル・レコード、索引付きレコード、相対レコード、およびメッセージ・キュー・レコードの入出力オプション。</li> </ul> EGL の sysVar.errorCode の値が変更される頻度は、VisualAge Generator の場合より高くなります。
<b>EZESEGM</b>	converseVar.segmentedMode
<b>EZESEGTR</b>	sysVar.transactionID
<b>EZESYS</b>	if ステートメントまたは while ステートメントの中で EGL の値を使用するには、以下を使用します。 sysVar.systemType 他のステートメント内で使用するために以前の VAGen の値を取得するには、以下を使用します。 myItem = VGLib.getVAGSysType(); その後、ステートメント内で myItem を使用します。 マイグレーション済みの VAGen プログラム内で以前の VAGen の値を使用する必要がある場合は、以下を使用します。 <custPrefix>EZESYS ここで、<custPrefix> はマイグレーションのステージ 2 で指定した名前変更接頭部です。VAGen マイグレーション設定「以前の EZESYS 値を初期化しない (Do not initialize old EZESYS values)」に基づいて、マイグレーション・ツールは、<custPrefix>EZESYS のデータ宣言および以前の VAGen 値への初期化ステートメントを組み込んだり、省略したりします。 詳細と起こりうる問題については、114 ページの『EZESYS』を参照してください。

表 125. その他のデータの EZE ワード (続き)

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZETST 注: IF...IN、および MOVEA に対して設定されます。 EZETST は 2 バイトのバイナリーです。	sysVar.arrayIndex 注: arrayIndex は int (4 バイトのバイナリー) です。
EZEUSR	sysVar.sessionID
EZEUSRID	sysVar.userID

## 一般的な関数の EZE ワード

表の左側の欄には、VisualAge Generator 4.5 の EZE ワードを示します。右側の欄には、マイグレーション・ツールが EZE ワードを EGL に変換した結果を示します。注記のある場合を除き、引数リストは VisualAge Generator と EGL のどちらでも同じなので、表からは省略しています。

表 126. 一般的な関数の EZE ワード

VisualAge Generator 4.5 の EZE ワード	EGL の定義
result = EZEBYTES(itemOrRecord) 注: VisualAge Generator の資料には、項目とレコードのみが EZEBYTES の引数として使用できると記述されています。ただし、VisualAge Generator は EZEBYTES の引数としてマップを許容します。	result = sysLib.bytes(itemOrRecord) 注: EGL は、sysLib.bytes の引数としてのフォームをサポートしません。マイグレーション・ツールは、引数が項目、レコード、またはマップのどれであるかに関係なく、引数を変換します。
EZECOMIT()	sysLib.commit()
EZECONV(target,direction,conversionTable)	sysLib.convert
EZEC10(xxx, yyy, zzz)	sysLib.verifyChkDigitMod10
EZEC11(xxx, yyy, zzz)	sysLib.verifyChkDigitMod11
EZEG10(xxx, yyy, zzz)	sysLib.calculateChkDigitMod10
EZEG11(xxx, yyy, zzz)	sysLib.calculateChkDigitMod11
EZEPURGE(queueName)	sysLib.purge
EZEROLLB()	sysLib.rollback()
EZEWAIT(variableName) 注: variableName は、100 分の 1 秒単位で時間を指定します。	sysLib.wait(variableName); 注: <ul style="list-style-type: none"> <li>variableName は、秒単位で時間を指定します。</li> <li>マイグレーション・ツールは、時間を秒数に変換します。詳細と起こりうる問題については、116 ページの『EZEWAIT』を参照してください。</li> </ul>

## ストリング EZE ワード

表の左側の欄には、VisualAge Generator 4.5 の EZE ワードを示します。右側の欄には、マイグレーション・ツールが EZE ワードを EGL に変換した結果を示します。引数リストは VisualAge Generator と EGL のどちらでも同じなので、表からは省略しています。

表 127. ストリング EZE ワード

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZESBLKT	strLib.setBlankTerminator
EZESCCWS	strLib.concatenateWithSeparator
EZESCMPR	VGLib.compareBytes
EZESCNCCT	VGLib.concatenateBytes
EZESCOPY	VGLib.copyBytes
EZESFIND	strLib.findStr
EZESNULLT	strLib.setNullTerminator
EZESSET	strLib.setSubStr
EZESTLEN	strLib.strLen
EZESTOKN	strLib.getNextToken

## 数学 EZE ワード

表の左側の欄には、VisualAge Generator 4.5 の EZE ワードを示します。右側の欄には、マイグレーション・ツールが EZE ワードを EGL に変換した結果を示します。注記のある場合を除き、引数リストは VisualAge Generator と EGL のどちらでも同じなので、表からは省略しています。

表 128. 数学 EZE ワード — 一般的な数学関数

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZEABS	mathLib.abs
EZECEIL	mathLib.ceiling
EZEEXP	mathLib.exp
EZEFLOOR	mathLib.floor
EZEFREXP	mathLib.frexp
EZELDEXP	mathLib.ldexp
EZELOG	mathLib.log
EZELOG10	mathLib.log10
EZEMAX	mathLib.maximum
EZEMIN	mathLib.minimum
EZEMODF	mathLib.modf
EZENCMPR	mathLib.compareNum
EZEPOW	mathLib. pow
EZEPRSCN	mathLib.precision

表 128. 数学 EZE ワード — 一般的な数学関数 (続き)

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZEROUND	<code>mathLib.round</code> <b>注:</b> <code>mathLib.round</code> は、 <i>(R</i> オプションを指定した VAGen ステートメントを置き換えるためにも使用されます。これらのステートメントの代入ステートメントへのマイグレーションは、次の構文を使用して行われます。 <code>result = mathLib.round(numericExpression);</code>
EZESQRT	<code>mathLib.sqrt</code>

表 129. 数学 EZE ワード — 三角関数

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZEACOS	<code>mathLib.acos</code>
EZEASIN	<code>mathLib.asin</code>
EZEATAN	<code>mathLib.atan</code>
EZEATAN2	<code>mathLib.atan2</code>
EZECOS	<code>mathLib.cos</code>
EZECOSH	<code>mathLib.cosh</code>
EZESIN	<code>mathLib.sin</code>
EZESINH	<code>mathLib.sinh</code>
EZETAN	<code>mathLib.tan</code>
EZETANH	<code>mathLib.tanh</code>

表 130. 数学 EZE ワード — 浮動小数点数学関数

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZEFLADD	<code>mathLib.floatingSum</code>
EZEFLDIV	<code>mathLib.floatingQuotient</code>
EZEFLMOD	<code>mathLib.floatingMod</code>
EZEFLMUL	<code>mathLib.floatingProduct</code>
EZEFLSET	<code>mathLib.floatingAssign</code>
EZEFLSUB	<code>mathLib.floatingDifference</code>

## ユーザー・インターフェースの EZE ワード

表の左側の欄には、VisualAge Generator 4.5 EZE ワードを示します。右側の欄には、マイグレーション・ツールが EZE ワードを EGL に変換した結果を示します。引数リストは VisualAge Generator でも EGL でも同じであるため、表からは省略しています。

表 131. ユーザー・インターフェースの EZE ワード

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZEUIERR	<code>sysLib.setError</code>
EZEUILOC	<code>sysLib.setLocale</code>



## オブジェクト・スクリプトの EZE ワード

表 132. オブジェクト・スクリプトの EZE ワード

VisualAge Generator 4.5 の EZE ワード	EGL の定義
EZESCRPT(targetScriptName)	マイグレーション・ツールはエラー・メッセージを出し、演算部は正しくマイグレーションできません。オリジナルの EZESCRPT ステートメントは、EGL ソース・コードにコメントとして組み込まれます。

## サービス・ルーチン

サービス・ルーチンに関するセクションは、次の表で構成されています。

- サービス・ルーチン - 一般構文 (353 ページの表 133)
- サービス・ルーチン - VisualAge Generator ルーチンおよび同等な EGL ルーチン (353 ページの表 134)

表 133. サービス・ルーチン - 一般構文

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
CALL serviceRoutine [ argumentList ] ;	<pre>sysLib.EGLSystemFunction ( [ argumentList ] );</pre> <p>注: EGL システム関数は、VisualAge Generator の場合と同じ引数リストを使用します。</p>	特別な考慮事項なし。
CALL serviceRoutine [ argumentList ] (REPLY ;	<pre>try   sysLib.EGLSystemFunction   ( [ argumentList ] ); end;</pre> <p>注: EGL システム関数は、VisualAge Generator の場合と同じ引数リストを使用します。</p>	VisualAge Generator では (REPLY オプションが組み込まれていた場合、マイグレーション・ツールは try... end ブロックを組み込みます。

表 134. サービス・ルーチン - VisualAge Generator ルーチンおよび同等な EGL ルーチン

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
CALL AUDIT	sysLib.audit	特別な考慮事項なし。
CALL COMMIT	sysLib.commit	特別な考慮事項なし。
CALL CREATX	sysLib.startTransaction	特別な考慮事項なし。
CALL CSPTDLI	VGLib.VGTDLI  注: EGL は EGLTDLI および AIBTDLI もサポートしています。	特別な考慮事項なし。

表 134. サービス・ルーチン - VisualAge Generator ルーチンおよび同等な EGL ルーチン (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
CALL EZCHART	<b>call EZCHART [ arguments ] externallyDefined ;</b>  注: EGL に EZCHART の置換表現はありません。	VAGen マイグレーション・ツールは、EZCHART を外部定義プログラムへの呼び出しに変換します。  VisualAge Generator では REPLY オプションが指定されていた場合、マイグレーション・ツールは <i>try... end</i> ブロック内に call ステートメントをネストします。
CALL RESET	sysLib.rollback	特別な考慮事項なし。

## PSB

VisualAge Generator では、PSB はパーツ・タイプです。PSB は、IMS または DL/I PSB の情報のサブセットを含みます。TP PCB に関連付けされた名前はありません。DB と GSAM PCB に関連付けするデータベース名は、固有である必要はありません。DL/I I/O 関数は、データベース名か、PCB 番号のいずれかによって、PSB 内の特定の PCB を参照することができます。ステートメントおよびプログラムの呼び出し先パラメーター・リストで、EZEDLPCB 特殊機能語により番号で PCB を参照することができます。I/O PCB は VAGen PSB に明示的にインクルードされていませんが、IMSVS、IMSBMP、および MVS Batch ターゲット環境では常に表示されています。I/O PCB は PCB 番号 0 と見なされます。

EGL では、PSB はレコード・パーツ・タイプのサブタイプです。PSBRecord は、*non-fixed* レコードです。PSBRecord 内のそれぞれの PCB 変数は固有でなければなりません。DL/I I/O 関数は、PSBRecord 内で PCB 変数に与えられた名前を使用して、特定の PCB を参照することができます。同様に、ステートメントおよびプログラムのパラメーター・リストでは、PSB 内で PCB 変数に与えられた名前を使用します。

マイグレーション・ツールは、VAGen PSB 内の数値位置を基に、各 TP PCB の変数名を作成します。このツールは、VAGen PSB、PCB のタイプを指示するカスタマー指定サフィックス、および、必要な場合、固有変数名を作成する番号にあるデータベース名の組み合わせを使用して、それぞれの DB または GSAM PCB ごとに変数名を作成します。このツールはまた、DB および GSAM PCB を再定義する変数を作成します。再定義変数は、VAGen PSB 内の PCB の数値位置を基にしています。これによりマイグレーション・ツールは、マイグレーション実行中にどちらの変数 (データベース名または PCB 番号) も使用可能になります。

表 135. PSB

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>PSB 情報:</p> <ul style="list-style-type: none"> <li>名前</li> <li>PCB 情報は PCB タイプにより異なります <ul style="list-style-type: none"> <li>番号</li> <li>型 <ul style="list-style-type: none"> <li>TP</li> <li>DB</li> <li>GSAM</li> </ul> </li> <li>データベース</li> <li>セグメント</li> <li>親</li> <li>索引キー</li> </ul> </li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>PSB は分離パーツ・タイプです。</li> <li>I/O PCB は明示的に指定されませんが、IMSVS、IMS BMP、および MVS Batch 環境では DL/I PSB 内になければなりません。</li> </ul>	<p>EGL 構文の例:</p> <pre>Record psbName type PSBRecord {defaultPSBName =   "originalPSBName"} iopcb IO_PCBRecord; { @PCB { pcbType=PCBKind.TP }}; pcb0 IO_PCBRecord { redefines="iopcb" }; [ otherPCBInformation ] end // end psbName</pre> <p>注:</p> <ul style="list-style-type: none"> <li>PSB はレコードのタイプです。</li> <li>I/O PCB は明示的に指定しなければなりません。</li> <li>EGL は @PCB 複素数プロパティを使用して、PCB タイプを指定します。</li> <li>EGL は次のレコードに関するレコード定義を提供します。 <ul style="list-style-type: none"> <li>IO_PCBRecord</li> <li>ALT_PCBRecord</li> <li>DB_PCBRecord</li> <li>GSAM_PCBRecord</li> </ul> </li> <li>EGL は、CICS 環境用の生成中には、I/O PCB 変数を無視 (除去) します。</li> </ul>	<p>予約語のため、または名前の先頭が # または @ シンボルであることが原因で、PSB の名前変更が必要な場合、マイグレーション・ツールは defaultPSBName プロパティのみをインクルードします。</p> <p>マイグレーション・ツールは常にすべての PSB に変数 iopcb と pcb0 再定義を追加します。</p>
<p>PCB タイプ - TP</p> <ul style="list-style-type: none"> <li>番号</li> </ul>	<pre>ELAALT ALT_PCBRecord {@PCB { pcbType = PCBKind.TP }}; pcb1 ALT_PCBRecord { redefines = "ELAALT" }; ELAEXP ALT_PCBRecord {@PCB { pcbType = PCBKind.TP }}; pcb2 ALT_PCBRecord { redefines = "ELAEXP" }; pcb<i>n</i> ALT_PCBRecord { @PCB { pcbType = PCBKind.TP }};</pre> <p>注:</p> <ul style="list-style-type: none"> <li>EGL は CICS 環境用の生成中には、代替 PCB 変数を無視 (除去) します。</li> </ul>	<p>マイグレーション・ツールは、VAGen PSB の最初の 2 つの TP PCB の名前として、ELAALT と ELAEXP を使用します。マイグレーション・ツールはまた、番号で参照ができるようにこれらの 2 つの TP PCB に関する再定義を作成します。</p> <p>追加の TP PCB がある場合、マイグレーション・ツールは、PSB 内の PCB 番号を基に TP PCB の変数名を作成します。これによりマイグレーション・ツールは、EZEDLPCB[<i>n</i>] の置き換えとして pcb<i>n</i> を使用できるようになります。</p>

表 135. PSB (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>PCB タイプ - DB</p> <ul style="list-style-type: none"> <li>番号</li> <li>データベース</li> <li>セグメント</li> <li>親</li> <li>索引キー</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>PSB の複数の PCB に同じデータベース名を使用できます。</li> </ul>	<pre>DBName_dbSuffix DB_PCBRecord { @PCB { pcbType = PCBKind.DB,   secondaryIndex = "indexKeyName",   secondaryIndexItem =     "renamedIndexKey",   hierarchy =     [ @Relationship       { segmentRecord =         "segmentName",         parentRecord =         "parentSegmentName" } ] } };</pre> <pre>pcbn DB_PCBRecord { redefines = "DBName_dbSuffix" };</pre> <p>注:</p> <ul style="list-style-type: none"> <li>PSB はレコードであり、それぞれのデータベース名がレコード内でフィールドになるため、それぞれのデータベース名は固有でなければなりません。</li> </ul>	<p>マイグレーション・ツールは、サフィックスのついた VAGen データベース名に基づき、DB PCB の変数名を作成します。サフィックスは、ステージ 2 の「VAGen マイグレーション・データベースの I/O 設定」によって指定できます。</p> <p>必要な場合、マイグレーション・ツールは、固有の DBName を作成するために番号をインクルードします。例: DBName_n_dbSuffix。ここでは <i>n</i> は <i>pcbn</i> 再定義の番号と同じです。</p> <p>EGL 予約語との競合、または名前の先頭が # または @ シンボルであることが原因で、VAGen 索引キーの名前変更が必要な場合、マイグレーション・ツールは secondaryIndexItem プロパティのみをインクルードします。</p> <p>マイグレーション・ツールはまた、VAGen PSB 内の PCB 番号に基づいて DB PCB の再定義を作成します。</p>
<p>PCB タイプ - GSAM</p> <ul style="list-style-type: none"> <li>番号</li> <li>データベース</li> </ul>	<pre>DBName_gsamSuffix GSAM_PCBRecord { @PCB   { pcbType = PCBKind.GSAM } }; pcbn GSAM_PCBRecord { redefines =   "DBName_gsamSuffix" };</pre> <p>注:</p> <ul style="list-style-type: none"> <li>PSB はレコードであり、それぞれのデータベース名がレコード内でフィールドになるため、それぞれのデータベース名は固有でなければなりません。</li> <li>EGL は CICS 環境用の生成中に、GSAM PCB 変数を見捨て (除去) します。</li> </ul>	<p>マイグレーション・ツールは、サフィックスのついた VAGen データベース名に基づき、GSAM PCB の変数名を作成します。サフィックスは、ステージ 2 の「VAGen マイグレーション・データベースの I/O 設定」によって指定できます。</p> <p>必要な場合、マイグレーション・ツールは、固有の DBName を作成するために番号をインクルードします。例: DBName_n_gsamSuffix。ここでは <i>n</i> は <i>pcbn</i> 再定義の番号と同じです。</p> <p>マイグレーション・ツールはまた、VAGen PSB 内の PCB 番号に基づいて GSAM PCB の再定義を作成します。</p>

## 制御パーツ

VisualAge Generator では、制御パーツはフリー・フォームのテキスト・エディターを使用して入力されます。制御パーツは、生成時に実際に使用されるまでは検証されません。大文字または小文字は区別されません。EGL では、制御パーツは .eglbl ファイルに XML 表記で格納され、制御パーツのタイプごとに専用のエディターを使用して作成されます。

EGL の場合、大文字と小文字は区別されます。このセクションの表では、VisualAge Generator のフリー・フォームのテキスト・エディターに入力する情報と、EGL で使用される XML タグまたは属性値を比較しています。表にはタグまたは属性値のみが示されており、実際の XML 構文は示されていません。

### 注:

- マイグレーション・ツールは、対応する EGL 置換表現は存在しないが EGL に必要な関連情報の判別に役立つ可能性がある生成オプション、リンケージ・テーブル・オプション、およびリソース関連オプションを、コメントとして組み込みます。例えば、マイグレーション・ツールは、生成オプション /jspreldir をコメントとして組み込みます。これらのコメントは、通常の EGL ビルド・パーツ・エディターを使用する際は表示されません。テキスト・エディターを使用してファイルを開くと、これらのコメントを表示できます。
- マイグレーション・ツールは、情報が現在または将来の EGL オプションの判別に使用できない場合は、対応する EGL の置換表現がない生成オプションを除去します。例えば、*/lineinfo* の置換表現は存在しません。このオプションは、IBM サポートによる VAGen 生成プログラムのデバッグを支援するためのものです。このオプションは、EGL 生成プログラムには使用できないので、マイグレーション・ツールはこのオプションをコメントとして組み込みません。
- マイグレーション・ツールは、次の場合を除いて制御パーツの名前を変更しません。
  - マイグレーション・ツールは、バインド制御パーツ名の末尾から .BND 接尾部を除去します。
  - マイグレーション・ツールは、リンク・エディット・パーツ名の末尾から .LKG 接尾部を除去します。
  - マイグレーション・ツールは、制御パーツ名に含まれるその他のドットを下線に変更します。
  - さらにこのツールは、/OPTIONS、/RESOURCE、および /LINKEDIT 生成オプション内で参照されている制御パーツ名のドットを下線に変更します。
  - マイグレーション・ツールは、パーツ名が EGL 予約語と競合している場合は、エラー・メッセージを出します。

制御パーツに関するセクションは、次の表で構成されています。

- 制御パーツの一般情報 (358 ページの表 136)
- 生成オプション (359 ページの表 137)
- 生成オプション - 変換テーブルの値 (378 ページの表 138)
- :calllink のリンケージ・テーブル・オプション (379 ページの表 139)

- :filelink のリンケージ・テーブル・オプション (384 ページの表 140)
- :crtxlink のリンケージ・テーブル・オプション (385 ページの表 141)
- :dxfrlink のリンケージ・テーブル・オプション (386 ページの表 142)
- リソース関連 (388 ページの表 143)
- リンク・エディット・オプション (392 ページの表 144)
- バインド制御 (393 ページの表 145)
- パーツ関連のシンボリック・パラメーター (393 ページの表 146)
- ファイル関連のシンボリック・パラメーター (394 ページの表 147)
- ユーザー定義のシンボリック・パラメーター (395 ページの表 148)

表 136. 制御パーツの一般情報

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<b>VAGen 制御パーツ名:</b> <ul style="list-style-type: none"> <li>• 名前にピリオド (.) を含むことができます。</li> <li>• バインド・パーツとリンク・エディット・パーツの場合、名前のうち最初のピリオドの後にある部分はすべて接尾部として扱われます。接尾部は、/bind と /linkedit の生成オプションの後に指定できます。</li> </ul>	<b>EGL ビルド・パーツ:</b> <ul style="list-style-type: none"> <li>• ピリオド (.) は、ビルド・パーツ名の中では無効です。</li> </ul>	マイグレーション・ツールは、ピリオド (.) を下線 (_) に変更します。
<b>VAGen 制御パーツのタグと値の中では、大文字と小文字は区別されません。</b>	<b>EGL 制御パーツのタグと値の中では、大文字と小文字が区別されます。</b>	マイグレーション・ツールは、制御パーツのタグと値を EGL に必要な正しい大/小文字に変換します。



## 生成オプション・パーツ

表 137. 生成オプション

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>VAGen 生成オプション・パーツ:</p> <ul style="list-style-type: none"> <li>1 つ以上の生成オプションを指定します。</li> <li>/options 生成オプションを使用してチェーニングできます。</li> <li>生成時に、ワークスペースに含まれる他の制御パーツを参照できます。参照される制御パーツは、生成オプション・パーツの関連パーツとは見なされません。</li> </ul>	<p>EGL ビルド記述子パーツ:</p> <ul style="list-style-type: none"> <li>1 つ以上のビルド記述子オプションを指定します。</li> <li>nextBuildDescriptor ビルド記述子オプションを使用してチェーニングできます。</li> <li>次のいずれかが当てはまる場合に限り、他のビルド・パーツを参照できます。 <ul style="list-style-type: none"> <li>ビルド・パーツが、同じ .eglblid ファイルに含まれている。</li> <li>ビルド・パーツが、.eglblid ファイルによってインポートされるファイル内にある。</li> </ul> </li> </ul>	<p>VAGen 制御パーツがすべて同じ VisualAge Java パッケージ、または VisualAge Smalltalk アプリケーション内にある場合、制御パーツは同じ .eglblid ファイルに配置されます。この場合、import ステートメントは必要ありません。</p> <p>VAGen 制御パーツが異なる VisualAge Java パッケージまたは VisualAge Smalltalk アプリケーション内にある場合、マイグレーション・ツールは import ステートメントを作成しません。ユーザーが import ステートメントを追加する必要があります。他の制御パーツへの参照を EGL が解決できない場合は、「問題」ビューにエラーが表示されます。</p>
VAGen 生成オプションの値は、ディレクトリ名またはファイル名に特殊文字が含まれている場合に限り、二重引用符で囲まれます。	EGL ビルド記述子オプションの値は、二重引用符で囲む必要があります。ただし、EGL ビルド・パーツ・エディターを使用する場合、エディターは引用符を XML ソースに自動的に挿入します。エディターに引用符は表示されません。	マイグレーション・ツールは、.eglblid ファイルの XML ソースを作成する際に、引用符を自動的に組み込みます。
<p>多くの VAGen 生成オプションは、生成オプションの肯定または否定を反映して /xxxx または /noxxxx のように指定できます。次に例を示します。</p> <ul style="list-style-type: none"> <li>/prep は、準備ステップを生成直後に自動的に開始するように指示します。</li> <li>/noprep は、準備ステップを後で実行するために、自動的に開始しないように指示します。</li> </ul>	<p>多くの EGL ビルド記述子オプションは、ビルド記述子オプションの肯定または否定を反映して xxxx="YES" または xxxx="NO" のように指定できます。次に例を示します。</p> <ul style="list-style-type: none"> <li>prep="YES" は、準備ステップを生成直後に自動的に開始するように指示します。</li> <li>prep="NO" は、準備ステップを後で実行するために、自動的に開始しないように指示します。</li> </ul>	<p>マイグレーション・ツールは、オプションを次のように処理します。</p> <ul style="list-style-type: none"> <li>他に指示のない限り、マイグレーション・ツールは /xxxx を対応する xxxx="YES" オプションに変換します。</li> <li>他に指示のない限り、マイグレーション・ツールは /noxxxx を対応する xxxx="NO" オプションに変換します。</li> </ul>
/ansisql	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/bidicontable=xxxx	bidiConversionTable="xxxx"	特別な考慮事項なし。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>/bind=xxxx</p> <p>VisualAge Generator の場合、xxxx はバインド・パーツの接尾部です。プログラムのバインド・パーツの名前は、pgmname.xxxx です。ここで、xxxx は /bind オプションによって指定される接尾部です。/bind=suffix を指定する理由としては、次のことが考えられます</p> <ol style="list-style-type: none"> <li>1. プログラムを複数の DB2 プランにバインドするので、プログラムに特殊なバインドが必要。</li> <li>2. VisualAge Generator では、プログラムとまったく同じ名前のバインド・パーツを容易に作成できない。</li> </ol>	<p>bind="xxxx"</p> <p>bind オプションの意味は、VisualAge Generator とは異なります。EGL の場合、xxxx はバインド・パーツのフルネームです。bind オプションは、バインド・パーツ名がプログラムと異なる場合にのみ指定する必要があります。ほとんどの場合、プログラムとバインド・パーツの名前は同じなので、bind オプションを組み込む必要はありません。</p> <p>複数のランタイム環境用に同じプログラムを生成し、それぞれの環境ごとに特殊なバインド・コマンドが必要な場合のみ、bind オプションが必要です。</p> <p>bind オプションのもう 1 つの用途は、バインド・コマンドのテンプレートを含むパーツ名の指定です。プロジェクト管理担当者または DBA は、メンバー固有のパラメーター用に置換可能な SYMPARMS を組み込んで、バインド・パーツを定義できます。EGL の bind オプションを使用して、このテンプレート・パーツを指定できます。それぞれのプログラムごとにパッケージをバインドする場合は、この技法が役に立ちます。</p>	<p>VisualAge Generator と EGL ではバインドの値の意味が異なるので、マイグレーション・ツールはこのオプションをマイグレーションできません。マイグレーション・ツールは、/bind をコメントとして組み込みます。</p>
<p>/checktype=xxxx</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• none</li> <li>• low</li> <li>• all</li> </ul>	<p>checkType="xxxx"</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• NONE</li> <li>• LOW</li> <li>• ALL</li> </ul>	<p>特別な考慮事項なし。</p>
<p>/cicsdbcs</p>	<p>サポートなし。</p>	<p>現在、サポートされるすべての CICS 変換プログラムに DBCS のサポートが組み込まれているので、マイグレーション・ツールはこのオプションをコメントとして組み込みません。</p>
<p>/cicsentries=xxxx</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• none</li> <li>• rdo</li> <li>• macro</li> </ul>	<p>cicsEntries="xxxx"</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• NONE</li> <li>• RDO</li> <li>• MACRO</li> </ul>	<p>特別な考慮事項なし。</p>

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
/cobollevel=le   vs	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
commentlevel= <i>n</i> または /commentlevel= <i>commentText</i>  <i>n</i> または <i>commentText</i> は、次のいずれかです。 <ul style="list-style-type: none"> <li>• 0 または minimum</li> <li>• 1 または info</li> <li>• 2 または logic</li> <li>• 3 または data</li> <li>• 4 または statements</li> </ul> <b>注:</b> <ul style="list-style-type: none"> <li>• 数値または同等な <i>commentText</i> のどちらかを指定できます。</li> <li>• 0 = 生成オプションのコメントのみ</li> <li>• 1 = 別名、標準的な生成情報</li> <li>• 2 = プログラムとテーブルのプロローグ、および関数の説明</li> <li>• 3 = レコードのプロローグとデータ項目の説明</li> <li>• 4 = ソース・ステートメントとコメント</li> <li>• C++ の場合、有効な値は 0 = none と 1 = comments のみです。</li> </ul>	commentLevel=" <i>n</i> "  <i>n</i> は次のいずれかです。 <ul style="list-style-type: none"> <li>• 0</li> <li>• 1</li> <li>• 1</li> <li>• 1</li> <li>• 1</li> </ul> <b>注:</b> <ul style="list-style-type: none"> <li>• 0 = コメントなし</li> <li>• 1 = コメントが含まれる</li> </ul>	マイグレーション・ツールは、/commentlevel=0 または minimum を 0 にマイグレーションし、他のすべての値を 1 にマイグレーションします。
/configmapname="xxxx"  xxxx は、VisualAge Smalltalk 構成マップの名前です。	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。これは、このオプションが、ソース・コード・リポジトリに 1 単位としてチェックインする必要がある、関連した EGL プロジェクトのグループの判別に役立つ可能性があるからです。
/configmapversion="xxxx"  xxxx は、/configmapname によって指定される、VisualAge Smalltalk 構成マップのバージョン名です。	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。これは、このオプションが、ソース・コード・リポジトリに 1 単位としてチェックインする必要がある、関連した EGL プロジェクトのグループの判別に役立つ可能性があるからです。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
/contable=xxxx xxxx は、変換テーブルの名前です。	clientCodeSet="yyyy" serverCodeSet="zzzz" yyyy と zzzz は、それぞれクライアントとサーバーの変換テーブルの名前です。	マイグレーション・ツールは、 <i>clientCodeSet</i> と <i>serverCodeSet</i> の両オプションを、VAGen の /contable 生成オプションから設定します。VAGen と EGL の値の対応については、378 ページの表 138 を参照してください。/contable=xxxx の値が 378 ページの表 138 に示されていない場合、マイグレーション・ツールは <i>clientCodeSet</i> と <i>serverCode</i> の両方を xxxx に設定します。
/createdds	genDDSFile="YES"   "NO" 注: これは ISERIESC 用です。	特別な考慮事項なし。
/currency=xxx (1 文字から 3 文字)	currencySymbol="xxx"	特別な考慮事項なし。
/data = 24   31	data="24"   "31"	特別な考慮事項なし。
/dbms=xxxx xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• db2</li> <li>• oracle</li> <li>• odbc</li> </ul>	dbms="xxxx" xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• DB2</li> <li>• ORACLE</li> <li>• DB2</li> </ul> 注: EGL の Java 生成機能は、ODBC ではなく JDBC をサポートします。	マイグレーション・ツールは、 <i>odbc</i> を <i>DB2</i> に変更し、警告メッセージを出します。
/dbpassword=xxxx	sqlPassword="xxxx"	マイグレーション・ツールは、VAGen の /dbpassword オプションと /sqlpassword オプションを、EGL の <i>sqlPassword</i> オプションにマージします。生成オプション・パーツに /dbpassword と /sqlpassword の両方が含まれている場合、マイグレーション・ツールは <i>sqlPassword</i> を 2 回組み込みます。「問題」ビューにエラーが示されます。
/dbuser=xxxx	sqlID="xxxx"	マイグレーション・ツールは、VAGen の /dbuser オプションと /sqlID オプションを EGL の <i>sqlID</i> オプションにマージします。生成オプション・パーツに /dbuser と /sqlID の両方が含まれている場合、マイグレーション・ツールは <i>sqlID</i> を 2 回組み込みます。「問題」ビューにエラーが示されます。
/debugtrace	debugTrace="YES"   "NO"	特別な考慮事項なし。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
/destaccount=xxxx	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/destdir=xxxx	destDirectory="xxxx"	特別な考慮事項なし。
/desthost=xxxx	destHost="xxxx"	特別な考慮事項なし。
/destlib=xxxx	destLibrary="xxxx" 注: これは ISERIESC 用です。	特別な考慮事項なし。
/destpassword=xxxx	destPassword="xxxx"	特別な考慮事項なし。
/destuid=xxxx	destUserID="xxxx"	特別な考慮事項なし。
/dxfrcancel	cancelAfterTransfer="YES"   "NO"	特別な考慮事項なし。
/dxfrxctl	useXctlForTransfer="YES"   "NO"	特別な考慮事項なし。
/ejbgroup=xxxx	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/endcommarea	endCommarea="YES"   "NO"	特別な考慮事項なし。
/errdest=xxxx	errorDestination="xxxx" 注: これは IMS 用です。	特別な考慮事項なし。
/fastpath	imsFastPath="YES"   "NO" 注: これは IMS 用です。	特別な考慮事項なし。
/fold	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/ftptranslationcmddbs=xxxx	サポートなし。 EGL は、ホストへのファイル転送用に TCP/IP のみをサポートしています。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/ftptranslationcmdsbs=xxxx	サポートなし。 EGL は、ホストへのファイル転送用に TCP/IP のみをサポートしています。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/genauthortimevalues /nogenauthortimevalues	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/genhelpmaps	genHelpFormGroup="YES"   "NO"	特別な考慮事項なし。
/genmaps	genFormGroup="YES"   "NO"	特別な考慮事項なし。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
/genout=xxxx	genDirectory="xxxx"	<p>マイグレーション・ツールは、/genout を変換してその結果をオリジナルのビルド記述子と、<i>secondaryTargetBuildDescriptor</i> オプションによって参照される新規のビルド記述子パーツの両方に置きます。</p> <p>Java を生成する場合は、genProject ビルド記述子オプションを genDirectory オプションに、追加としてあるいは代わりとして指定することが必要になる場合があります。genProject は、次の場合は必須です。</p> <ul style="list-style-type: none"> <li>• HP-UX または SOLARIS 向けに生成する場合</li> <li>• VGWebTransactions または VGUI レコードを生成する場合</li> </ul>
/genproperties /nogenproperties	<p>genProperties="GLOBAL" genProperties="NO"</p> <p>EGL は、genProperties="PROGRAM" も提供しています。</p>	<p>マイグレーション・ツールは、/genproperties を EGL の <i>genProperties="GLOBAL"</i> に変換します。これは、生成結果に関してこれが最も近い値だからです。</p>
/genresourcebundle /nogenresourcebundle	genResourceBundle="YES"   "NO"	<p>マイグレーション・ツールは、/genresourcebundle を変換してその結果をオリジナルのビルド記述子と、<i>secondaryTargetBuildDescriptor</i> オプションによって参照される新規のビルド記述子パーツの両方に置きます。</p>
/genret	genReturnImmediate="YES"   "NO"	特別な考慮事項なし。
/gentables	genDataTables="YES"   "NO"	特別な考慮事項なし。
/genuirecords	genVGUIRecords="YES"   "NO"	特別な考慮事項なし。
/groupname=xxxx	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/inedit=all /inedit=inonly	<p>validateOnlyIfModified="NO" validateOnlyIfModified="YES"</p>	特別な考慮事項なし。



表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>/initaddws</p> <p>VisualAge Generator の場合、1 次作業用ストレージ・レコードは常に初期化されます。/initaddws 生成オプションは、テーブルおよび追加レコードのリストに指定されている、その他の作業用ストレージ・レコードの初期化を行います。</p>	<p>initNonIOData="YES"   "NO"</p> <p>EGL の場合、メインプログラムに対して inputRecord として指定されたレコードは常に初期化されます。initNonIOData ビルド記述子オプションは、プログラムのデータ宣言に指定されている、その他の基本レコードの初期化を行います。また EGL では、プログラム内の任意のレコード宣言に対して initialized プロパティを指定できます。initialized プロパティを使用すると、initNonIOData ビルド記述子オプションよりも精密な制御が可能です。</p>	<p>マイグレーション・ツールは、メインプログラムの 1 次作業用ストレージ・レコードを EGL の inputRecord プロパティにマイグレーションし、さらに initialized プロパティを指定しないレコード宣言を組み込みます。マイグレーション・ツールは、呼び出し先プログラムの 1 次作業用ストレージ・レコードを、initialized プロパティを指定したレコード宣言にマイグレーションします。レベル 77 項目用の追加レコードをマイグレーション・ツールが作成した場合、マイグレーション・ツールはそのレコードのデータ宣言を組み込み、さらに initialized プロパティも組み込みます。これにより、VisualAge Generator の場合と同じ 1 次作業用ストレージ・レコードの初期化が行われます。その他の基本レコードはすべて、initNonIOData ビルド記述子オプションに基づいて初期化されます。</p>
/initrecd	initIORecords="YES"   "NO"	特別な考慮事項なし。
/javadestdir=xxxx	destDirectory="xxxx"	マイグレーション・ツールは、/javadestdir を変換してその結果を secondaryTargetBuildDescriptor オプションによって参照される新規のビルド記述子パーツに置きます。
/javadesthost=xxxx	destHost="xxxx"	マイグレーション・ツールは、/javadesthost を変換してその結果を secondaryTargetBuildDescriptor オプションによって参照される新規のビルド記述子パーツに置きます。
/javadestpassword=xxxx	destPassword="xxxx"	マイグレーション・ツールは、/javadestpassword を変換してその結果を secondaryTargetBuildDescriptor オプションによって参照される新規のビルド記述子パーツに置きます。
/javadestuid=xxxx	destUserID="xxxx"	マイグレーション・ツールは、/javadestuid を変換してその結果を secondaryTargetBuildDescriptor オプションによって参照される新規のビルド記述子パーツに置きます。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
/jvasystem=xxxx  xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• AIX</li> <li>• LINUX</li> <li>• OS2</li> <li>• OS390</li> <li>• OS400</li> <li>• SOLARIS</li> <li>• WINNT</li> </ul>	system="xxxx"  対応する EGL の値は、次のとおりです。 <ul style="list-style-type: none"> <li>• AIX</li> <li>• LINUX</li> <li>• サポートなし</li> <li>• USS</li> <li>• ISERIESJ</li> <li>• SOLARIS</li> <li>• WIN</li> </ul>	マイグレーション・ツールは、サポートされている /jvasystem の値を変換してその結果を <i>secondaryTargetBuildDescriptor</i> オプションによって参照される新規のビルド記述子パーツに置きます。  このツールは、非サポートの値をコメントとしてオリジナルのビルド記述子パーツに組み込みます。
/jobcard=xxxx	サポートなし。以下のような、同等の機能が提供されています。 <ul style="list-style-type: none"> <li>• z/OS および iSeries ビルド・サーバーが jobcard を処理します。これらの環境では、JOB CARD シンボリック・パラメーターは無視されます。</li> <li>• VSE は JOB CARD シンボリック・パラメーターをサポートします。</li> </ul>	マイグレーション・ツールは、このオプションを JOB CARD シンボリック・パラメーターに変換します。
/jobname=xxxx	サポートなし。以下のような、同等の機能が提供されています。 <ul style="list-style-type: none"> <li>• z/OS および iSeries では、\$USERID をビルド・スクリプトのジョブ名として使用できます。EGL 生成機能は、destUserID ビルド記述子オプションの値を番号と連結して得られる固有のジョブ名を使用して、\$USERID を置き換えます。これらの環境では、JOB NAME シンボリック・パラメーターは無視されます。</li> <li>• VSE は JOB NAME シンボリック・パラメーターをサポートします。</li> </ul>	マイグレーション・ツールは、このオプションを JOB NAME シンボリック・パラメーターに変換します。
/jspreldir="xxxx"	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/leftjust	leftAlign="YES"   "NO"	特別な考慮事項なし。
/lineinfo	サポートなし。	このオプションは IBM サポートが VAGen 生成プログラムをデバッグする場合にのみ意味があるので、マイグレーション・ツールはこのオプションをコメントとして組み込みません。生成される COBOL に影響はありません。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
/lines=nn	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/linkage=xxxx xxxx は、VAGen リンケージ・テーブル・パーツの名前です。	linkage="xxxx" xxxx は、EGL リンケージ・オプション・パーツの名前です。	特別な考慮事項なし。
/linkedit=xxxx  VisualAge Generator の場合、xxxx はリンク・エディット・パーツの接尾部です。プログラムのリンク・エディット・パーツの名前は、pgmname.xxxx です。ここで、xxxx は /linkedit オプションによって指定される接尾部です。/linkedit=suffix を指定する理由としては、次のことが考えられます。  1. PL/I への静的リンク・エディットを行う場合など、特殊なリンク・エディットがプログラムに必要。  2. VisualAge Generator では、プログラムとまったく同じ名前のリンク・エディット・パーツを容易に作成できない。	linkEdit="xxxx"  linkEdit の意味は、VisualAge Generator とは異なります。EGL の場合、xxxx はリンク・エディット・パーツのフルネームです。linkEdit オプションは、バインド・パーツ名がプログラムと異なる場合にのみ指定する必要があります。ほとんどの場合、プログラムとリンク・エディット・パーツの名前は同じなので、linkedit オプションを組み込む必要はありません。  複数のランタイム環境用に同じプログラムを生成し、それぞれの環境ごとに特殊な linkedit コマンドが必要な場合のみ、linkEdit オプションが必要です。	VisualAge Generator と EGL ではリンク・エディットの値の意味が異なるので、マイグレーション・ツールはこのオプションをマイグレーションできません。マイグレーション・ツールは、/linkedit をコメントとして組み込みます。
/listing /listingonerror /nolisting  注: これは 3 者択一のスイッチです。	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/locvalid	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/log=xx または /nolog	imsLogID="xx" または include /nolog as a comment  注: これは IMS 用です。	マイグレーション・ツールは、このオプションを次のように処理します。  • /log=xx は imsLogID="xx" に変換されます。  • /nolog はコメントに変換されます。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<code>/math=xxxxx</code> <code>xxxxx</code> は、次のいずれかです。 <ul style="list-style-type: none"> <li>• <code>cobol</code></li> <li>• <code>cspae</code></li> </ul>	<code>math="xxxxx"</code> <code>xxxxx</code> は、次のいずれかです。 <ul style="list-style-type: none"> <li>• <code>COBOL</code></li> <li>• <code>CSPA</code></li> </ul>	特別な考慮事項なし。
<code>/mfsdev =</code> <code>    ('deviceName',</code> <code>      'MFSInfo',</code> <code>      'eAI'</code> <code>    )</code> <b>注:</b> <ul style="list-style-type: none"> <li>• VAGen <code>deviceName</code> が使用されます。 <code>MFSInfo</code> は、 VAGen 装置で使用するために対応する MFS 情報を提供します。 <code>eAI</code> は、拡張属性情報を提供します。</li> <li>• <code>eAI</code> の値は、 <code>EATTR</code>、 <code>NOEATTR</code>、 および <code>NCD</code> です。</li> <li>• 複数の <code>MFSInfo</code> および <code>eAI</code> の値を、単一の <code>deviceName</code> に提供することができます。</li> <li>• この生成オプションの詳細については、「<i>VisualAge Generator Server Guide for MVS, VSE, and VM</i>」を参照してください。</li> </ul>	<code>&lt;mfsDevice</code> <code>    width="nn", height="nn",</code> <code>    devStmtParms="MFSInfo",</code> <code>    extendedAttributes="eAI"</code> <code>&gt;/&gt;</code> <b>注:</b> <ul style="list-style-type: none"> <li>• EGL 装置サイズ (幅および高さ) が使用されます。 <code>MFSInfo</code> および <code>eAI</code> は、 VisualAge Generator と同じ情報を提供します。</li> <li>• <code>eAI</code> の値は、 <code>YES</code>、 <code>NO</code>、 および <code>NCD</code> です。</li> <li>• 複数の <code>MFSInfo</code> および <code>eAI</code> の値を、単一の装置サイズに提供することができます。</li> <li>• このビルド記述子オプションの詳細については、「<i>EGL 解説書</i>」を参照してください。</li> </ul>	<p>マイグレーション・ツールは、 VAGen <code>deviceName</code> を対応する幅および高さに変換します。装置名とサイズの関係については、 280 ページの表 89 を参照してください。 2 つの <code>deviceName</code> が同一の幅および高さに変換され、 <code>MFSInfo</code> および <code>eAI</code> に同一の値が指定されている場合、マイグレーション・ツールは 1 つのエントリーのみを組み込みます。</p> <p>マイグレーション・ツールは、 <code>MFSInfo</code> の値を変更しません。</p> <p>マイグレーション・ツールは、 <code>eAI</code> の値に対応する EGL 値に変換します。</p>
<code>/mfseattr</code> <code>/nomfseattr</code> <code>/mfseattrnncd</code>  VisualAge Generator の場合、これら 3 つのオプションは、MFS フォーマットのマップの拡張属性サポートを生成するために必要な情報を提供する、3 者択一のスイッチです。	<code>mfsExtendedAttr="YES"</code> <code>mfsExtendedAttr="NO"</code> <code>mfsExtendedAttr="NCD"</code>  <b>注:</b> これは IMS 用です。	特別な考慮事項なし。
<code>/mfsignore</code>	<code>mfsIgnore="YES"   "NO"</code> <b>注:</b> これは IMS 用です。	特別な考慮事項なし。
<code>/mfstest</code>	<code>mfsUseTestLibrary="YES"   "NO"</code> <b>注:</b> これは IMS 用です。	特別な考慮事項なし。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>/msgtableprefix=xxxx</p> <p>VisualAge Generator の場合、メッセージ・テーブル接頭部はプログラム上で指定されます。UI レコードを単独で生成する場合は、生成時にメッセージ・テーブル接頭部を指定する必要があります。</p>	<p>msgTablePrefix = "xxxx"</p> <p>EGL では、msgTablePrefix に関して VisualAge Generator と同じ考慮事項が適用されます。</p>	<p>マイグレーション・ツールは、/msgtableprefix を変換して、その結果をオリジナルのビルド記述子と、 <i>secondaryTargetBuildDescriptor</i> オプションによって参照される新規のビルド記述子パーツの両方に置きます。</p> <p>VGUI レコードを使用するプログラムを生成しないで、VGUI レコードのみを生成する場合は、メッセージ・テーブルの接頭部が付いたパッケージ名を組み込まなければなりません (例えば、msgTablePrefix = "packageName.prefixID")。</p>
<p>/msp=xxxx</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• all</li> <li>• gsam</li> <li>• mfs</li> <li>• seq</li> </ul>	<p>formServicePgmType="xxxx"</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• ALL</li> <li>• GSAM</li> <li>• MFS</li> <li>• SEQ</li> </ul> <p>注: これは IMS 用です。</p>	<p>特別な考慮事項なし。</p>
/nullfill	fillWithNulls="YES"   "NO"	特別な考慮事項なし。
/numovfl	checkNumericOverflow="YES"   "NO"	特別な考慮事項なし。
<p>/options=xxxx</p> <p>xxxx は、別の VAGen 生成オプション・パーツの名前です。</p>	<p>nextBuildDescriptor="xxxx"</p> <p>xxxx は、別の EGL ビルド記述子パーツの名前です。</p>	<p>特別な考慮事項なし。</p>
/packagename=xxxx	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
<p>/possign=x</p> <p>x は次のいずれかです。</p> <ul style="list-style-type: none"> <li>• f</li> <li>• c</li> </ul>	<p>positiveSignIndicator="x"</p> <p>x は次のいずれかです。</p> <ul style="list-style-type: none"> <li>• F</li> <li>• C</li> </ul> <p>注: これは ISERIESC 用です。</p>	<p>特別な考慮事項なし。</p>
/prep	prep ="YES"   "NO"	特別な考慮事項なし。
/prepfiler	buildPlan="YES"   "NO"	特別な考慮事項なし。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
/printdest=xxxx  xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• ezeprint</li> <li>• termid</li> </ul>	printDestination="xxxx"  xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• PROGRAMCONTROLLED</li> <li>• TERMINALID</li> </ul>	特別な考慮事項なし。
/project="xxxx"[,"version"]  xxxx は VisualAge for Java プロジェクトの名前、version は指定したプロジェクトのバージョン名です。	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。これは、このオプションが、ソース・コード・リポジトリに 1 単位としてチェックインする必要がある、関連した EGL プロジェクトのグループの判別に役立つ可能性があるからです。
/projectid=xxxx	projectID="xxxx"	特別な考慮事項なし。
/recovery	restoreCurrentMsgOnError="YES"   "NO" <b>注:</b> これは IMS 用です。	特別な考慮事項なし。
/resource=xxxx  xxxx は、VAGen リソース関連パーツの名前です。	resourceAssociations="xxxx"  xxxx は、EGL リソース関連パーツの名前です。	特別な考慮事項なし。
/resourcebundlelocale=xxxx	resourceBundleLocale = "xxxx"	マイグレーション・ツールは、/resourcebundlelocale を変換して、その結果をオリジナルのビルド記述子と、 <i>secondaryTargetBuildDescriptor</i> オプションによって参照される新規のビルド記述子パーツの両方に置きます。
/resvword=xxxx	reservedWord="xxxx"	特別な考慮事項なし。
/rt=xxxx	returnTransaction="xxxx"	特別な考慮事項なし。
/runfile	genRunFile="YES"   "NO"	特別な考慮事項なし。
/sendtranslationcmddbcs=xxxx	サポートなし。  <b>注:</b> EGL は、ホストへのファイル転送用に TCP/IP のみをサポートしています。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/session=xxxx	サポートなし。  <b>注:</b> EGL は、ホストへのファイル転送用に TCP/IP のみをサポートしています。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/setfull	setFormItemFull="YES"   "NO"	特別な考慮事項なし。
/sp	checkToTransaction="YES"   "NO"	特別な考慮事項なし。



表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
/spa=xxxx,ADF,yyyy  <b>注:</b> ADF はオプションです。 yyyy はオプションであるため、VisualAge Generator では以下のすべてが有効です。  /spa=xxxx /spa=xxxx,ADF,yyyy /spa=xxxx,,yyyy	EGL では、次の 3 つの別個のオプションがあります。  spaSize="xxxx" spaADF= "YES"   "NO" spaStatusBytePosition="yyyy"	マイグレーション・ツールは、/spa オプションを 3 つの EGL オプションに分割します。  マイグレーション・ツールは、値が YES の場合にのみ spaADF を組み込みます。
/spzero	spacesZero="YES"   "NO"	特別な考慮事項なし。
/sqldb=xxxx	sqlDB="xxxx"	特別な考慮事項なし。
/sqlid=xxxx	sqlID="xxxx"	マイグレーション・ツールは、VAGen の /dbuser オプションと /sqlID オプションを EGL の <i>sqlID</i> オプションにマージします。生成オプション・パーツに /dbuser と /sqlID の両方が含まれている場合、マイグレーション・ツールは sqlID を 2 回組み込みます。「問題」ビューにエラーが示されます。
/sqlpassword=xxxx	sqlPassword="xxxx"	マイグレーション・ツールは、VAGen の /dbpassword オプションと /sqlpassword オプションを、EGL の <i>sqlPassword</i> オプションにマージします。生成オプション・パーツに /dbpassword と /sqlpassword の両方が含まれている場合、マイグレーション・ツールは sqlPassword を 2 回組み込みます。「問題」ビューにエラーが示されます。
/sqlvalid	validateSQLStatements="YES"   "NO"	特別な考慮事項なし。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
/symparm=pppppppp,'vvvv' <ul style="list-style-type: none"> <li>pppppppp は、シンボリック・パラメーターの名前です。 pppppppp は 1 文字から 8 文字までです。</li> <li>vvvv は値です。値の中で、2 つの連続した単一引用符は、1 つの単一引用符を表します。</li> </ul>	EGL は、VisualAge Generator と同じ事前定義シンボリック・パラメーターを多数サポートします。新しい EGL のシンボリック・パラメーターと競合しない限り、任意のユーザー定義シンボリック・パラメーターを使用することもできます。	マイグレーション・ツールは、シンボリック・パラメーターを次のように処理します。 <ul style="list-style-type: none"> <li>マイグレーション・ツールは、VAGen 定義のシンボリック・パラメーターを、対応する EGL のシンボリック・パラメーターに変換します。</li> <li>対応する EGL のシンボリック・パラメーターが存在しない場合、マイグレーション・ツールは VAGen 定義のシンボリック・パラメーターを EGL のシンボリック・パラメーターの構文に変換し、パラメーターの名前や値は変更しません。また、マイグレーション・ツールはエラー・メッセージを出します。</li> <li>マイグレーション・ツールは、ユーザー定義のシンボリック・パラメーターを EGL のシンボリック・パラメーターの構文に変換し、パラメーターの名前や値は変更しません。</li> </ul>
/SYMPARM=EZALTXTR,'xxx'	transferErrorTransaction="xxx"	特別な考慮事項なし。
/SYMPARM=EZONEAS2,'xxx'	oneFormItemCopybook="YES"	特別な考慮事項なし。
/syncdxfr	synchOnPgmTransfer="YES"   "NO" 注: これは CICS 環境の DLI 用です。	特別な考慮事項なし。
/syncxfer	synchOnTrxTransfer="YES"   "NO"	特別な考慮事項なし。
/syscodes	sysCodes="YES"   "NO"	特別な考慮事項なし。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>/system=xxxx</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• MVSBATCH</li> <li>• MVSCICS</li> <li>• IMSBMP</li> <li>• IMSVS</li> <li>• AIX</li> <li>• JAVALINUX</li> <li>• JAVAOS390</li> <li>• JAVAOS400</li> <li>• JAVAWINNT</li> <li>• JAVAWRAPPER</li> <li>• WINNT</li> <li>• LINUX</li> <li>• OS400</li> <li>• VSEBATCH</li> <li>• VSECICS</li> <li>• HP</li> <li>• SOLARIS</li> </ul> <p>VAGen では次に示す環境も指定できますが、これらの環境はマイグレーション・ツールによって変換されません。</p> <p>JAVA、JAVAGUI、WINGUI、OS2GUI、OS2、OS2CICS、AIXCICS、NTCICS、SOLACICS、TSO、VMCMS、VMBATCH</p>	<p>system="xxxx"</p> <p>対応する EGL の値は、次のとおりです。</p> <ul style="list-style-type: none"> <li>• ZOSBATCH</li> <li>• CICS for z/OS</li> <li>• IMSBMP</li> <li>• IMSVS</li> <li>• AIX</li> <li>• LINUX</li> <li>• USS</li> <li>• ISERIESJ</li> <li>• WIN</li> <li>• WIN</li> <li>• WIN</li> <li>• LINUX</li> <li>• ISERIESC</li> <li>• VSEBATCH</li> <li>• VSECICS</li> <li>• HPUX</li> <li>• SOLARIS</li> </ul>	<p>マイグレーション・ツールは、このオプションを次のように処理します。</p> <ul style="list-style-type: none"> <li>• /system=xxxx に対応する値が EGL に存在すれば、マイグレーション・ツールは対応する EGL の値にマイグレーションします。</li> <li>• /system=xxxx に対応する値が EGL に存在しなければ、マイグレーション・ツールは /system=xxxx をコメントとして組み込みます。</li> <li>• /system=JAVAWRAPPER の場合、マイグレーション・ツールは EGL のビルド記述子オプション <i>enableJavaWrapperGen="ONLY"</i> も組み込みます。これは、プログラムのために Java ラッパーのみを生成するように指定します。</li> <li>• COBOL 環境の場合、マイグレーション・ツールは、<i>destPort</i> ビルド記述子オプションの指定が必要であることを示す警告メッセージを出します。</li> </ul>
<p>/targnls=xxx</p> <p>xxx は、3 文字の各国語コードです。</p>	<p>targetNLS="xxx"</p> <p>xxx は、3 文字の各国語コードです。ENP (大文字英語) を除くすべての値は、VisualAge Generator と EGL で同一です。ENP に相当するものは EGL には存在しません。</p>	<p>マイグレーション・ツールは、/targnls を変換して、その結果をオリジナルのビルド記述子と、<i>secondaryTargetBuildDescriptor</i> オプションによって参照される新規のビルド記述子パーツの両方に置きます。</p> <p>マイグレーション・ツールは、VAGen 値を targetNLS 値として使用します。値が ENP である場合は、「問題」ビューにエラーが示されます。<i>.eglbld</i> ファイルを編集して、値を変更できます。ENP の置換値として、ENU (大/小文字混合英語) を使用できます。</p>

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>/templates=xxxx</p> <p>VisualAge Generator の場合は、テンプレートを使用して準備 JCL とランタイム JCL が生成され、また CICS トランザクションとプログラムのエントリーが生成されます。</p>	<p>templateDir="xxxx"</p> <p>EGL の場合、準備テンプレートの代わりにビルド・スクリプトが使用されます。ZOSBATCH、VSEBATCH、IMSBMP、および ISERIESC のターゲット環境用のランタイム JCL を生成する場合にのみ、テンプレートが使用されます。</p>	<p>特別な考慮事項なし。</p>
<p>/trace=xxxx,yyyy</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• none</li> <li>• sqlerr</li> <li>• sqlio</li> </ul> <p>yyyy はオプションです。yyyy が存在する場合は、<i>stmt</i> に設定されます。</p> <p><i>none</i>、<i>sqlerr</i>、または <i>sqlio</i> (<i>stmt</i> の指定あり、または指定なし) の任意の組み合わせが有効です。</p>	<p>/trace は、次のように複数のビルド記述子オプションに分割されます。</p> <ul style="list-style-type: none"> <li>• <i>sqlerr</i> が指定されている場合 -- <i>sqlErrorTrace="YES"</i></li> <li>• <i>sqlio</i> が指定されている場合 --- <i>sqlIOTrace="YES"</i></li> <li>• <i>stmt</i> が指定されている場合 --- <i>statementTrace="YES"</i></li> </ul>	<p>特別な考慮事項なし。</p>
<p>/transfertype=xxxx</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• tcpip</li> <li>• sna</li> </ul>	<p>サポートなし。</p> <p>注: EGL は、ホストへのファイル転送用に TCP/IP のみをサポートしています。</p>	<p>マイグレーション・ツールは、このオプションをコメントとして組み込みます。</p>
<p>/transid=primaryID,restartID</p> <p>VisualAge Generator の場合は、<i>/transid=,restart</i> が有効で、1 次トランザクションのデフォルトはプログラム名の先頭 4 文字です。</p>	<p>/transid は、次のように複数のビルド記述子オプションに分割されます。</p> <ul style="list-style-type: none"> <li>• <i>primary</i> が指定されている場合 -- <i>startTransactionID="primaryID"</i></li> <li>• <i>,restart</i> が指定されている場合 --- <i>restartTransactionID="restartID"</i></li> </ul>	<p>特別な考慮事項なし。</p>
<p>/twaoff=nnnn</p>	<p>twaOffset="nnnn"</p>	<p>特別な考慮事項なし。</p>
<p>/unload</p> <p>VisualAge Generator の場合、<i>/unload</i> を指定すると、現行生成プロセスのために要求されているプロジェクトまたは生成マップのロード前に、VAGen パーツを含むすべての VisualAge Java プロジェクト、または VisualAge Smalltalk 構成マップが、バッチ生成機能によってアンロードされていました。</p>	<p>サポートなし。</p>	<p>マイグレーション・ツールは、このオプションに関するコメントを組み込みません。</p>

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
/validmix	validateMixedItems="YES"   "NO"	特別な考慮事項なし。
/vmloadlib=xxxx	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
/vselib=xxxx	vseLibrary="xxxx"	特別な考慮事項なし。
/workdb=xxxx  xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• aux</li> <li>• main</li> <li>• dli</li> <li>• sql</li> </ul>	workDBType="xxxx"  xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• AUX</li> <li>• MAIN</li> <li>• DLI</li> <li>• SQL</li> </ul>	特別な考慮事項なし。
使用されません。	vagCompatibility="YES"	VAGen マイグレーション設定「互換モードを設定しない ( <i>Do not set compatibility mode</i> )」に基づいて、マイグレーション・ツールはそれぞれのビルド記述子パーツについて、このオプションを追加あるいは省略を行いません。
使用されません。	itemsNullable="NO"	マイグレーション・ツールは、このオプションを常にすべてのビルド記述子パーツに追加します。この手法は、VAGen の振る舞いを保持するために、デフォルトの NO を使用する必要があることを明示的に示すために使用されます。
VisualAge Generator の場合、 <i>decimalSymbol</i> は、Java で CHA と NUM の値を割り当てまたは比較するときに使用されるランタイム・プロパティです。	<i>decimalSymbol</i> ="x"  x は次のいずれかです。 <ul style="list-style-type: none"> <li>• ピリオド ( . )</li> <li>• コンマ ( , )</li> </ul> EGL の場合、この情報は生成時または実行時に指定できます。	マイグレーション・ツールは、 <i>decimalSymbol</i> を設定しません。Java ソースの生成を予定している場合は、このプロパティをビルド記述子パーツに追加できます。このようにすると、すべての EGL プロパティ・ファイル内で <i>decimalSymbol</i> が生成されます。代わりに、生成された EGL プロパティ・ファイルにプロパティを直接追加することもできます。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
使用されません。	<p>destPort="xxxx"</p> <p>EGL の場合、<i>destPort</i> は、生成出力を実行準備のためにホスト・システムに転送するときに使用するポートを指定します。<i>destPort</i> ビルド記述子オプションは、COBOL 生成のターゲット環境に必要です。</p>	<p>マイグレーション・ツールは、<i>destPort</i> を設定しません。デフォルト値は、以下に示すようにターゲット環境によって異なります。</p> <ul style="list-style-type: none"> <li>• z/OS 環境の場合、<i>destPort</i> のデフォルト値はありません。<i>destPort</i> ビルド記述子オプションを追加し、その値が z/OS ビルド・サーバーを開始する JCL (ジョブ制御言語) で使用する値と一致しなければなりません。z/OS ビルド・サーバーを開始するサンプル JCL (ジョブ制御言語) はポート 5555 を使用します。</li> <li>• iSeries 環境の場合、<i>destPort</i> のデフォルト値はありません。<i>destPort</i> ビルド記述子オプションを追加する必要があります。値は、iSeries ビルド・サーバーによって使用される値と一致しなければなりません。</li> <li>• VSE 環境の場合、<i>destPort</i> のデフォルト値は 21 です。<i>destPort</i> ビルド記述子オプションを指定する必要があるのは、値が 21 とは異なる場合のみです。</li> </ul>
使用されません。	genProject="xxxx"	<p>Java を生成する場合は、<i>genProject</i> ビルド記述子オプションを <i>genDirectory</i> オプションに、追加としてあるいは代わりとして指定することが必要になる場合があります。<i>genProject</i> は、次の場合は必須です。</p> <ul style="list-style-type: none"> <li>• HP-UX または SOLARIS 向けに生成する場合、あるいは</li> <li>• VGWebTransactions または VGUI レコードを生成する場合</li> </ul>



表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
サポートなし。	<code>tempDirectory="xxxx"</code>	VGUI レコードを生成する場合、 <code>genProject</code> ディレクトリーに同名の JSP が既に存在するときには、 <code>tempDirectory</code> オプションを使用することにより、生成された JSP を置くディレクトリーを指定することができます。 <code>tempDirectory</code> を指定しない場合は、JSP は <code>genProject</code> に生成されますが、その名前は <code>newxxxx.JSP</code> になります。ここで、 <code>xxxx</code> は VGUI レコードの名前です。
サポートなし。  VisualAge Generator では、プログラムが <code>EZESYS</code> の値を検査する場合であっても、プログラムを生成する対象であるすべてのターゲット環境に対して、すべての VAGen ソース・コードが有効でなければなりません。	<code>eliminateSystemDependentCode="YES"   "NO"</code>  EGL では、プログラムが <code>systemType</code> の値を検査する場合は、現行ターゲット生成環境では実行されることがありえないソース・コードを省略できます。これにより、得られる COBOL または Java のソース・コードを小さくすることができます。	マイグレーション・ツールは、 <code>eliminateSystemDependentCode</code> を設定しません。デフォルト値は "YES" です。
使用されません。	<code>sessionBeanID="xxxx"</code>	マイグレーション・ツールは、 <code>sessionBeanID</code> を設定しません。Java または Java ラッパーを生成する場合、 <code>sessionBeanID</code> ビルド記述子オプションを設定する必要があるかどうか判別するには、EGL のオンライン・ヘルプを参照してください。

表 137. 生成オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
VisualAge Generator の場合は、SQL JDBC ドライバー・クラス、JNDI 名、および接続 URL 情報を、実行時に使用されるプロパティー・ファイルに組み込みます。	<pre>sqlJDBCDriverClass="xxxx" sqlValidationConnectionURL="xx"</pre> <p>EGL の場合、この情報は生成時または実行時に指定できます。</p>	<p>マイグレーション・ツールは、<i>sqlJDBCDriverClass</i> および <i>sqlValidationConnectionURL</i> を設定しません。これらの値を生成時に指定する必要がある場合は、次のようにして指定できます。</p> <ul style="list-style-type: none"> <li>ワークスペース設定を指定する。 この技法は、Eclipse 環境で生成を行う場合のみ使用できます。</li> <li>ビルド記述子パーツにビルド記述子オプションを指定する。この技法は、Eclipse 環境で生成を行う場合にも、バッチ生成を行う場合にも使用できます。</li> </ul> <p>どちらの場合も、プロパティー・ファイルが生成されるように、<i>genProperties</i>="GLOBAL" または "PROGRAM" ビルド記述子オプションをさらに組み込む必要があります。</p> <p>値を実行時に指定する必要がある場合は、プロパティー・ファイルのランタイム・プロパティーを変更できます。</p>

表 138. 生成オプション - 変換テーブルの値

言語	変換テーブルの VAGen <i>/contable</i> 値	EBCDIC 文字セットの EGL <i>serverCodeSet</i>	ASCII 文字セットの EGL <i>clientCodeSet</i>
アラビア語	ELACNARA	IBM-420	IBM-1256
中国語 (簡体字)	ELACNCHS	IBM-935	IBM-1381
中国語 (簡体字)	ELACNGBK	IBM-1388	IBM-1386
中国語 (繁体字)	ELACNCHT	IBM-937	IBM-950
デンマーク語	ELACNDKN	IBM-277	IBM-1252
東ヨーロッパ言語	ELACN870	IBM-870	IBM-1250
英語 (英国)	ELACN285	IBM-285	IBM-1252
英語 (US)	ELACNENU	IBM-037	IBM-1252
フィンランド語	ELACNFIN	IBM-298	IBM-1252
フランス語	ELACNFRA	IBM-297	IBM-1252
ドイツ語	ELACNDEU	IBM-273	IBM-1252
ギリシャ語	ELACNGRE	IBM-875	IBM-1253
ヘブライ語	ELACNHEB	IBM-424	IBM-1255
イタリア語	ELACNITA	IBM-280	IBM-1252

表 138. 生成オプション - 変換テーブルの値 (続き)

言語	変換テーブルの VAGen /contable 値	EBCDIC 文字セットの EGL serverCodeSet	ASCII 文字セットの EGL clientCodeSet
日本語、カタカナ	ELACNJPN	IBM-930	IBM-943
日本語、ローマ字	ELACNJPL	IBM-939	IBM-943
韓国語	ELACNKOR	IBM-933	IBM-949
ノルウェー語	ELACNDKN	IBM-277	IBM-1252
ポルトガル語	ELACNPTB	IBM-037	IBM-1252
ロシア語	ELACNCYR	IBM-1025	IBM-1251
スペイン語	ELACNESP	IBM-284	IBM-1252
スウェーデン語	ELACNSWE	IBM-278	IBM-1252
スイス・ドイツ語	ELACNDES	IBM-500	IBM-1252
トルコ語	ELACNTUR	IBM-1026	IBM-1254
ユーザー定義 (上記 リスト以外)	XXXXXXXX	XXXXXXXX	XXXXXXXX

## リンケージ・テーブル・パーツ

リンケージ・テーブル・パーツは、Calllink、Filelink、Crtxlink、および Dxfrlink です。

### callLink

表 139. :calllink のリンケージ・テーブル・オプション

VisualAge Generator 4.5	マイグレーション・ツールによって生成 される EGL	マイグレーション・ツールに関する考 慮事項
:calllink	callLink	特別な考慮事項なし。
linktype=xxxx  xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>dynamic</li> <li>static</li> <li>cicslink</li> <li>remote</li> <li>csocall</li> <li>sessionejb</li> </ul>	呼び出しのタイプ。ここで、同等な EGL オプションは次のとおりです。 <ul style="list-style-type: none"> <li>localCall</li> <li>localCall</li> <li>localCall</li> <li>remoteCall</li> <li>remoteCall</li> <li>ejbCall</li> </ul>	VAGen の linktype が省略されている 場合、マイグレーション・ツールは localCall を使用します。マイグレーシ ョン・ツールは、EGL の CallLink 情 報に関するその他のプロパティを設 定するために、他の場所でも linktype を使用します。
appName=programName  programName は、呼び出される プログラムの名前です。ワイルド カードを使用できます。	pgmName="programName"	特別な考慮事項なし。

表 139. :calllink のリンケージ・テーブル・オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
externalname=applname	alias="applname"	名前が EGL の予約語であるために VAGen プログラムの名前を変更する必要がある場合は、プログラム定義またはリンケージ・テーブルのどちらかに <i>alias</i> プロパティを使用して、生成されるプログラムの名前としてプログラムの元の VAGen 名を指定できます。どちらかの技法を使用すれば、VAGen プログラムを呼び出す非 VAGen プログラムを変更する必要がなくなります。
package=packageName	package="packageName"	Java の生成を行う場合に、呼び出し側と呼び出し先のプログラムが別々のパッケージにあるときは、パッケージ名を呼び出し先プログラムのリンケージ・エントリーに組み込むことができます。あるいは、CALL ステートメントを変更して、プログラムをパッケージ名によって明示的に修飾するか、CALL ステートメントを含むファイルにパッケージの import 文を組み込みます。
library=libraryName  または  dllname=libraryName  VisualAge Generator の場合、library と dllname は同義語として扱われます。	library="libraryName"	マイグレーション・ツールは、VAGen の library または dllname を EGL の library プロパティにマージします。
linktype=xxxx  xxxx は、次のいずれかです。 • dynamic • static • cicslink	linkType="xxxx"  xxxx は、次のいずれかです。 • DYNAMIC • STATIC • CICSLINK	特別な考慮事項なし。
parmform=xxxx  xxxx は、次のいずれかです。 • oslink • commptr • commdata • cicsoslink	parmForm="xxxx"  xxxx は、次のいずれかです。 • OSLINK • COMMPTR • COMMDATA • CICSOSLINK	特別な考慮事項なし。

表 139. :calllink のリンケージ・テーブル・オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>contable=xxxx</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• conversionTableName</li> <li>• *</li> <li>• EZECONVT</li> <li>• BINARY</li> <li>• NONE</li> </ul>	<p>conversionTable="xxxx"</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• conversionTableName</li> <li>• *</li> <li>• PROGRAMCONTROLLED</li> <li>• サポートなし</li> <li>• サポートなし</li> </ul>	<p>マイグレーション・ツールは、EGL の CallLink 情報を作成するときに、同じ conversionTableName を使用します。</p> <p>マイグレーション・ツールは、VAGen の contable=BINARY を BINARY にマイグレーションしますが、この値は EGL ではサポートされません。また、マイグレーション・ツールはエラー・メッセージを出します。「問題」ビューにエラーが示されます。.eglblb ファイルを編集し、サポートされる値を選択して使用することによって、エラーを訂正する必要があります。</p> <p>VAGen contable=NONE の場合、マイグレーション・ツールは conversionTable プロパティを省略します。</p>
<p>location=xxxx</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• systemName</li> <li>• EZELOC</li> </ul>	<p>location="xxxx"</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• systemName</li> <li>• PROGRAMCONTROLLED</li> </ul>	<p>特別な考慮事項なし。</p>

表 139. :calllink のリンケージ・テーブル・オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>remotecomtype=xxxx</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• appcims</li> <li>• ca400</li> <li>• cicsclient</li> <li>• dce</li> <li>• dcesecure</li> <li>• direct</li> <li>• exci</li> <li>• ipc</li> <li>• java400</li> <li>• lu2</li> <li>• tcpip</li> </ul>	<p>remoteComType="xxxx"</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• サポートなし</li> <li>• サポートなし</li> <li>• CICSECI</li> <li>• サポートなし</li> <li>• サポートなし</li> <li>• DIRECT</li> <li>• サポートなし</li> <li>• DISTINCT</li> <li>• JAVA400</li> <li>• サポートなし</li> <li>• TCPIP</li> </ul>	<p>マイグレーション・ツールは、cicsclient を対応する最も近い EGL 値である CICSECI に変換します。</p> <p>VAGen の :calllink エントリーが ctgport と ctglocation をまだ指定していない場合、マイグレーション・ツールはエラー・メッセージを出してこれらの値の指定を促します。</p> <p>マイグレーション・ツールは、リストに「サポートなし」と示されている値を「現状のまま」マイグレーションし、メッセージを出します。この時点で、使用する通信プロトコルを決定し、正しい情報を使用して EGL の CallLink エントリーを更新する必要があります。CallLink 情報を訂正するまで、「問題」ビューにエラーが示されます。</p> <p>CICSSSL を使用する場合は、ctgPort、ctgLocation、ctgKeyStore、および ctgKeyStorePassword の各プロパティを EGL の CallLink 情報に追加する必要があります。</p> <p>CICSJ2C を使用する場合は、pgmName、conversionTable、remotePgmType、luwControl、remoteBind、location、および parmForm の各プロパティを EGL の CallLink 情報に追加する必要があります。</p> <p>マイグレーション・ツールは、APPCIMS を「現状のまま」マイグレーションします。その理由は、APPCIMS がサポートされていないため、およびその他のプロパティの値がまったく異なるためです。APPCIMS に最適な置換値は IMSTCP です。</p>



表 139. :calllink のリンケージ・テーブル・オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<b>remoteapptype=xxxx</b> xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• vg</li> <li>• nonvg</li> <li>• vgjava</li> <li>• itf</li> </ul>	<b>remotePgmType="xxxx"</b> xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• EGL</li> <li>• EXTERNALLYDEFINED</li> <li>• 適用外</li> <li>• サポートなし</li> </ul>	VisualAge Generator <b>remoteapptype=vgjava</b> の場合、マイグレーション・ツールは :calllink エントリーをマイグレーションしますが、 <i>remotePgmType</i> プロパティを省略します。  <b>remoteapptype=itf</b> の場合、マイグレーション・ツールは :calllink エントリー全体をコメント化します。
<b>serverid=serverName</b>	<b>serverID="serverName"</b>	特別な考慮事項なし。
<b>luwcontrol=xxxx</b> xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• client</li> <li>• server</li> </ul>	<b>luwControl="xxxx"</b> xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• CLIENT</li> <li>• SERVER</li> </ul>	特別な考慮事項なし。
<b>remotebind=xxxx</b> xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• generation</li> <li>• runtime</li> </ul>	<b>remoteBind="xxxx"</b> xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• GENERATION</li> <li>• RUNTIME</li> </ul>	特別な考慮事項なし。
<b>providerURL=URLName</b>	<b>providerURL="URLName"</b>	特別な考慮事項なし。
<b>ctglocation='tcpipInfo'</b>	<b>ctgLocation="tcpipInfo"</b>	特別な考慮事項なし。
<b>ctgport=portID</b>	<b>ctgPort="portID"</b>	特別な考慮事項なし。
<b>bitmode=nn</b> nn は、次のいずれかです。 <ul style="list-style-type: none"> <li>• 16</li> <li>• 32</li> </ul>	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。
<b>binform=xxxx</b> xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• intel</li> <li>• host</li> </ul>	サポートなし。	マイグレーション・ツールは、このオプションをコメントとして組み込みます。

表 139. :calllink のリンケージ・テーブル・オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
サポートなし。  VisualAge Generator では、呼び出し先プログラムが画面にマップを送らない場合に、CALL ステートメントに NOMAPS オプションを指定するとパフォーマンスが向上します。	refreshScreen="YES"   "NO"	マイグレーション・ツールは、このプロパティを設定しません。以前、VAGen の call ステートメントに NOMAPS を指定していた場合は、 <i>vagCompatibility="YES"</i> ビルド記述子オプションを使用すれば、EGL の CALL ステートメントに <i>noRefresh</i> オプションを引き続き使用できます。代わりに、呼び出し先プログラムに対する CallLink エントリーに <i>refreshScreen="NO"</i> を指定しても、同じサポートを利用できます。
使用されません。  VisualAge Generator によってサポートされる通信プロトコルは、この情報を必要としません。	ctgKeyStore ctgKeyStorePassword	マイグレーション・ツールは、このプロパティを設定しません。 <i>remoteComType="CICSSSL"</i> を使用する場合は、 <i>ctgKeyStore</i> と <i>ctgKeyStorePassword</i> が必要です。
使用されません。  VisualAge Generator では、呼び出し先バッチ・プログラム用の Java ラッパーを生成する場合には、/system=JAVAWRAPPER 生成オプションを使用します。	javaWrapper="YES"   "NO"	マイグレーション・ツールは、このプロパティを設定しません。呼び出し先プログラムを生成するたびに Java ラッパーが生成されるようにする場合は、 <i>javaWrapper="YES"</i> を指定する必要があります。

## fileLink

表 140. :filelink のリンケージ・テーブル・オプション

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
:filelink	fileLink	特別な考慮事項なし。
linktype=xxxx  xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• local</li> <li>• remote</li> </ul> VisualAge Generator のデフォルトは local です。	ファイルのタイプ。ここで、同等な EGL オプションは次のとおりです。 <ul style="list-style-type: none"> <li>• localFile</li> <li>• remoteFile</li> </ul>	VAGen の linktype が指定されていない場合、マイグレーション・ツールは <i>localFile</i> に変換します。
filename=fileName  fileName は、VAGen レコード定義にあるファイルの名前です。ワイルドカードを使用できます。	fileName="fileName"	特別な考慮事項なし。

表 140. :filelink のリンケージ・テーブル・オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
contable=xxxx xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• conversionTableName</li> <li>• *</li> <li>• EZECONVT</li> <li>• BINARY</li> </ul>	conversionTable="xxxx" xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• conversionTableName</li> <li>• *</li> <li>• PROGRAMCONTROLLED</li> <li>• サポートなし</li> </ul>	マイグレーション・ツールは、EGL の FileLink 情報を作成するときに、同じ conversionTableName を使用します。 マイグレーション・ツールは、VAGen の contable=BINARY を BINARY にマイグレーションしますが、この値は EGL ではサポートされません。また、マイグレーション・ツールはエラー・メッセージを出します。「問題」ビューにエラーが示されます。.eglbl ファイルを編集し、サポートされる値を選択して使用することによって、エラーを訂正する必要があります。
location=xxxx xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• CICS</li> <li>• EZELOC</li> </ul>	locationSpec="xxxx" xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• CICS</li> <li>• PROGRAMCONTROLLED</li> </ul>	特別な考慮事項なし。

## Crtxlink

表 141. :crtxlink のリンケージ・テーブル・オプション

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
:crtxlink	asynchLink	特別な考慮事項なし。
linktype=xxxx xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>• local</li> <li>• remote</li> </ul> 注: VisualAge Generator のデフォルトは local です。	ファイルのタイプ。ここで、同等な EGL オプションは次のとおりです。 <ul style="list-style-type: none"> <li>• localAsynch</li> <li>• remoteAsynch</li> </ul>	VAGen の linktype が指定されていない場合、マイグレーション・ツールは localAsynch に変換します。
recdname=recordName recordName は、VAGen レコード定義の名前です。ワイルドカードを使用できます。	recordName="recordName"	特別な考慮事項なし。

表 141. :crtxlink のリンケージ・テーブル・オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
contable=xxxx xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>conversionTableName</li> <li>*</li> <li>EZECONVT</li> <li>BINARY</li> </ul>	conversionTable="xxxx" xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>conversionTableName</li> <li>*</li> <li>PROGRAMCONTROLLED</li> <li>サポートなし</li> </ul>	マイグレーション・ツールは、EGL の AsynchLink 情報を作成するときに、同じ conversionTableName を使用します。  マイグレーション・ツールは、VAGen の contable=BINARY を BINARY に変換しますが、この値は EGL ではサポートされません。また、マイグレーション・ツールはエラー・メッセージを出します。「問題」ビューにエラーが示されます。.eglblt ファイルを編集し、サポートされる値を選択して使用することによって、エラーを訂正する必要があります。
location=xxxx xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>CICS</li> <li>EZELOC</li> </ul>	locationSpec="xxxx" xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>CICS</li> <li>PROGRAMCONTROLLED</li> </ul>	特別な考慮事項なし。
package=packageName	package="packageName"	特別な考慮事項なし。

## Dxfalink

表 142. :dxfalink のリンケージ・テーブル・オプション

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
:dxfalink	transferToProgram	特別な考慮事項なし。
fromappl=programName programName は、DXFR を使用して別のプログラムへの転送を行うプログラムの名前です。ワイルドカードは使用できません。	fromPgm="programName"	特別な考慮事項なし。
toappl=programName2 programName2 は、転送先のプログラムの名前です。	toPgm="programName2"	特別な考慮事項なし。

表 142. :dxfrlink のリンケージ・テーブル・オプション (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>linktype=xxxx</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• dynamic</li> <li>• static</li> <li>• noncsp</li> </ul>	<p>linkType="xxxx"</p> <p>xxxx は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• DYNAMIC</li> <li>• STATIC</li> <li>• EXTERNALLYDEFINED</li> </ul>	<p>以前、VAGen の DXFR ステートメントに NONCSP を指定していた場合は、ビルド記述子オプションに <i>vagCompatibility="YES"</i> を組み込めば、EGL の transfer to program ステートメントに <i>externallyDefined</i> オプションを引き続き使用できます。代わりに、転送先のプログラムに対する <i>transferToProgram</i> エントリーに <i>linkType= "EXTERNALLYDEFINED"</i> を指定しても、同じサポートを利用できます。</p>
<p>サポートなし。</p>	<p>alias="applname"</p>	<p>名前が EGL の予約語であるために VAGen プログラムの名前を変更する必要がある場合は、プログラム定義またはリンケージ・テーブルのどちらかに <i>alias</i> プロパティーを使用して、生成されるプログラムの名前としてプログラムの元の VAGen 名を指定できます。どちらの技法を使用しても、VAGen プログラムに転送する非 VAGen プログラムを変更する必要がなくなります。</p>

## リソース関連パーツ

表 143. リソース関連

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>VisualAge Generator では、リソース関連パーツは、特定のターゲット環境に対してファイルをインプリメントする方法を指定します。このファイルは、VAGen レコード定義に指定されているファイル名です。</p> <p>リソース関連パーツは、特定のターゲット環境に対して印刷出力をインプリメントする方法も指定できます。</p> <p>プログラムの生成時には、それぞれの索引付き出力、シリアル出力、相対出力、または印刷出力ごとに、fileName がリソース関連パーツと突き合わせられます。fileName と生成ターゲット環境を基準として最初にマッチングしたエントリーが、そのファイルに対して使用されるエントリーになります。</p>	<p>EGL のリソース関連パーツは、特定のターゲット環境に対してファイルをインプリメントする方法を指定します。このファイルは、EGL レコード定義に指定されている <i>fileName</i> プロパティです。</p> <p>リソース関連パーツは、特定のターゲット環境に対して印刷出力をインプリメントする方法も指定できます。</p> <p>プログラムの生成時には、それぞれの索引付き出力、シリアル出力、相対出力、または印刷出力ごとに、fileName がリソース関連パーツと突き合わせられます。fileName と生成ターゲット環境を基準として最初にマッチングしたエントリーが、そのファイルに対して使用されるエントリーになります。</p>	<p>特別な考慮事項なし。</p>
<p>VisualAge Generator の場合、C++ を生成すると、リソース関連ファイルは実行時にも使用されます。</p>	<p>EGL の場合、リソース関連情報は EGL パーツに格納されます。</p>	<p>マイグレーション・ツールは、VAGen リソース関連ファイル内でのみ有効だった、その他のオプションの変換をサポートしています。</p>
<pre>file = fileName         EZEPRINT</pre>	<pre>fileName="fileName"         "printer"</pre>	<p>特別な考慮事項なし。</p>



表 143. リソース関連 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p><code>/system=targetSystem</code></p> <p><code>targetSystem</code> は次のいずれかです。</p> <ul style="list-style-type: none"> <li>• AIX *</li> <li>• AIXCICS *</li> <li>• HP-UX *</li> <li>• IMSBMP</li> <li>• IMSVS</li> <li>• LINUX **</li> <li>• MVS BATCH</li> <li>• MVSCICS</li> <li>• NTCICS *</li> <li>• OS2 *</li> <li>• OS2CICS</li> <li>• OS400</li> <li>• SCO *</li> <li>• SOLACICS *</li> <li>• SOLARIS *</li> <li>• TSO</li> <li>• VMCMS</li> <li>• VMBATCH</li> <li>• VSEBATCH</li> <li>• VSECICS</li> <li>• WINNT **</li> </ul> <p>注:</p> <ul style="list-style-type: none"> <li>• * — C++ 生成の場合に使用される環境を示します。</li> <li>• ** — Java 生成の場合に使用される環境を示します。</li> <li>• <code>/system</code> はオプションです。</li> <li>• VisualAge Generator は、ターゲット・システムのワイルドカードとして * をサポートします。(例: MVS* または *CICS)</li> </ul>	<p>これは EGL ターゲット環境です。</p> <p>対応する環境の値は、次のとおりです。</p> <ul style="list-style-type: none"> <li>• aix</li> <li>• サポートなし</li> <li>• hpux</li> <li>• imsbmp</li> <li>• imsvs</li> <li>• linux</li> <li>• zosbatch</li> <li>• zoscics</li> <li>• サポートなし</li> <li>• サポートなし</li> <li>• サポートなし</li> <li>• iseriesc</li> <li>• サポートなし</li> <li>• サポートなし</li> <li>• solaris</li> <li>• サポートなし</li> <li>• サポートなし</li> <li>• vsebatch</li> <li>• vsecics</li> <li>• win</li> </ul> <p>注: ワイルドカードはサポートされません。</p>	<p>マイグレーション・ツールは、<code>/system</code> オプションを次のように処理します。</p> <ul style="list-style-type: none"> <li>• リストに「サポートなし」と示されているターゲット・システムの場合、マイグレーション・ツールは、VAGen リソース関連エントリーの情報をコメントとして EGL リソース関連パーツに組み込みます。これにより、情報が可能なかぎり保持されます。</li> <li>• <code>/system</code> オプションが VAGen リソース関連エントリーから省略された場合、マイグレーション・ツールは <i>any</i> を EGL リソース関連ターゲット環境として使用します。</li> <li>• <code>/system</code> オプションにワイルドカードが使用されている場合、マイグレーション・ツールはワイルドカードを含めてオプションをそのままマイグレーションします (例: mvs* または *cics)。また、マイグレーション・ツールはエラー・メッセージを出します。</li> </ul>

表 143. リソース関連 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
<p>/filetype=<i>fileType</i></p> <p><i>fileType</i> は、次のいずれかです。</p> <ul style="list-style-type: none"> <li>• BTRIEVE</li> <li>• GSAM</li> <li>• IBMCOBOL</li> <li>• MFCOBOL</li> <li>• MMSGQ</li> <li>• MQ</li> <li>• OS2COBOL</li> <li>• SEQ</li> <li>• SEQRS</li> <li>• SMSGQ</li> <li>• SPOOL</li> <li>• TEMPAUX</li> <li>• TEMPMAIN</li> <li>• TRANSIENT</li> <li>• VSAM</li> <li>• VSAMRS</li> </ul>	<p>EGL ファイル・タイプの対応する値を、次のリストに示します。</p> <ul style="list-style-type: none"> <li>• サポートなし</li> <li>• gsam</li> <li>• ibmcobol</li> <li>• サポートなし</li> <li>• mmsgq</li> <li>• mq</li> <li>• サポートなし</li> <li>• seq または seqws</li> <li>• seqrs</li> <li>• smsgq</li> <li>• spool</li> <li>• tempaux</li> <li>• tempmain</li> <li>• transient</li> <li>• vsam</li> <li>• vsamrs</li> </ul>	<p>マイグレーション・ツールは、/filetype オプションを次のように処理します。</p> <ul style="list-style-type: none"> <li>• /filetype オプションが VAGen リソース関連エントリーから省略された場合、マイグレーション・ツールは <i>default</i> を EGL ファイル・タイプとして使用します。</li> <li>• /system オプションがホスト・ターゲット環境を指定している場合、マイグレーション・ツールは VAGen の SEQ ファイル・タイプを EGL の seq ファイル・タイプに変換します。</li> <li>• /system オプションがワークステーション環境である場合、マイグレーション・ツールは VAGen の SEQ ファイル・タイプを EGL の seqws ファイル・タイプに変換します。</li> <li>• サポートされないファイル・タイプ値の場合、リソース関連の対象の /system がサポートされていると、マイグレーション・ツールは VAGen ファイル・タイプを使用して EGL リソース関連エントリーを作成し、エラー・メッセージを出します。「問題」ビューにもエラーが示されます。EGL リソース関連パーツを使用するには、その前にこのエラーを修正する必要があります。</li> </ul>
/sysname=systemName	systemName="systemName"	マイグレーション・ツールは、/sysname オプションの中で使用されているシンボリック・パラメーターを、対応する EGL の置換シンボリック・パラメーターに変換します。
/replace /noreplace	replace="YES" replace="NO"	特別な考慮事項なし。
/dup /nodup	duplicates="YES" duplicates="NO"  注: これは ISERIESC 用です。	特別な考慮事項なし。
/commit /nocommit  これらのオプションは、OS/400 ターゲット環境の場合のみ使用されます。	commit="YES" commit="NO"  注: これは ISERIESC 用です。	特別な考慮事項なし。

表 143. リソース関連 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
/blksize=xxxx,yyyy,zzzz  VisualAge Generator では、このオプションは VSE ターゲット環境の場合のみ使用されます。	blockSize="xxxx,yyyy,zzzz"	特別な考慮事項なし。
/sysnum=xxxx  VisualAge Generator では、このオプションは VSE ターゲット環境の場合のみ使用されます。	systemNumber="xxxx"	特別な考慮事項なし。
/label /nolabel  VisualAge Generator では、このオプションは VSE ターゲット環境の場合のみ使用されます。	standardLabel="YES" standardLabel="NO"	特別な考慮事項なし。
/pcbno=n  これは、IMSVS または IMSBMP のターゲット環境、またはファイル・タイプが GSAM ならば MVS バッチの場合のみ有効です。	pcbName="pcbn"	マイグレーション・ツールは、リテラルの pcb と PCB 番号を連結することで、PCB 番号を名前に変換します。
/noff  VisualAge Generator に /FF オプションはありません。このオプションは、VAGen リソース関連ファイル内でのみサポートされます。	FormFeedOnClose="NO" FormFeedOnClose="YES"	マイグレーション・ツールは、/noff を <i>FormFeedOnClose="NO"</i> に変換します。
/text  VisualAge Generator に /NOTEXT オプションはありません。このオプションは、VAGen リソース関連ファイル内でのみサポートされます。	text="YES" text="NO"	マイグレーション・ツールは、/text を <i>text="YES"</i> に変換します。
/contable=xxxx  xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>conversionTableName</li> <li>EZECONVT</li> </ul> このオプションは、VAGen リソース関連ファイル内でのみサポートされます。	conversionTable="xxxx"  xxxx は、次のいずれかです。 <ul style="list-style-type: none"> <li>conversionTableName</li> <li>PROGRAMCONTROLLED</li> </ul>	マイグレーション・ツールは、EGL のリソース関連エントリを作成するときに、同じ <i>conversionTableName</i> を使用します。

表 143. リソース関連 (続き)

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
/keys=xxxx  VisualAge Generator では、このオプションは /filetype=BTRIEVE を指定した場合のみ使用できます。このオプションは、VAGen リソース関連ファイル内でのみサポートされます。	KEYS="xxxx"	BTRIEVE はサポートされるターゲット環境で使用されるので、マイグレーション・ツールは /keys オプションを EGL の keys オプションにマイグレーションします。
/basename=xxxx  VisualAge Generator では、このオプションは OS/2 <sup>®</sup> ターゲット環境の場合のみ使用されます。このオプションは、VAGen リソース関連ファイル内でのみサポートされます。	サポートなし。	マイグレーション・ツールは、OS/2 ターゲット環境に関するエントリをすべてコメント化します。

## リンク・エディット・パーツ

表 144. リンク・エディット・パーツ

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
VisualAge Generator では、リンク・エディット・パーツの標準の名前は <i>programName.suffix</i> です。ここで、名前の先頭部分は VAGen プログラム名と同じで、接尾部は LKG です。VAGen の /linkedit 生成オプションが、接尾部の値を指定します。	デフォルトでは、EGL のリンク・エディット・パーツ名はプログラム名と同じである必要があります。これが当てはまる場合、リンク・エディット・ビルド記述子オプションを指定する必要はありません。  プログラムに関して複数のリンク・エディット・パーツが存在する場合は、プログラムのリンク・エディット・ビルド記述子オプション・パーツに異なるパーツ名を使用する必要があります。この場合は、完全なリンク・エディット・パーツ名をリンク・エディット・ビルド記述子オプションに指定する必要があります。	接尾部が .LKG ならば、マイグレーション・ツールは新規の EGL リンク・エディット・パーツを作成する際に接尾部を除去します。接尾部が .LKG 以外のものならば、ピリオド (.) は EGL パーツ名の中では無効な文字なので、マイグレーション・ツールは <i>.suffix</i> を <i>_suffix</i> に変更します。
VAGen のリンク・エディット・パーツは、ホスト環境での準備プロセス中にプログラムのリンク・エディットに必要なリンク・エディット・ステートメントを格納しています。	EGL のリンク・エディット・パーツは、ホスト環境のビルド・プロセス中にプログラムをリンク・エディットするために必要なリンク・エディット・ステートメントを含んでいます。	マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>リンク・エディット・パーツの中で使用されているシンボリック・パラメーターを、対応する EGL の置換シンボリック・パラメーターに変換します。</li> <li>VAGen パーツ内と同じ字下げを使用します。</li> </ul>

## バインド制御パーツ

表 145. バインド制御パーツ

VisualAge Generator 4.5	マイグレーション・ツールによって生成される EGL	マイグレーション・ツールに関する考慮事項
VisualAge Generator では、バインド制御パーツの標準の名前は <i>programName.suffix</i> です。ここで、名前の先頭部分は VAGen プログラム名と同じで、接尾部は BND です。VAGen の /bind 生成オプションが、接尾部の値を指定します。	デフォルトでは、EGL のバインド制御パーツ名はプログラム名と同じであることが必要です。これが当てはまる場合、バインド制御ビルド記述子オプションを指定する必要はありません。  プログラムに対して複数のバインド制御パーツが存在する場合は、プログラムのバインド制御パーツに異なるパーツ名を使用する必要があります。この場合は、完全なバインド制御パーツ名をバインド・ビルド記述子オプションに指定する必要があります。	接尾部が .BND ならば、マイグレーション・ツールは新規の EGL バインド制御パーツを作成する際に接尾部を除去します。接尾部が .BND 以外のものならば、ピリオド (.) は EGL パーツ名の中では無効な文字なので、マイグレーション・ツールは <i>.suffix</i> を <i>_suffix</i> に変更します。
VAGen のバインド制御パーツは、MVS ホスト環境での準備プロセス中に行われる、プログラムへの DB2 データベース・リソース・モジュール (DBRM) のバインドに必要な DB2 バインド・コマンドを格納しています。	EGL のバインド制御パーツは、z/OS ホスト環境でのビルド・プロセス中に行われる、プログラムへの DBRM のバインドに必要なバインド・コマンドを格納しています。	マイグレーション・ツールは次の処理を行います。 <ul style="list-style-type: none"> <li>バインド制御パーツの先頭にコマンドを追加します。これらのコマンドは、ビルド・サーバーが必要とするものです。</li> <li>バインド制御パーツの中で使用されているシンボリック・パラメーターを、対応する EGL の置換シンボリック・パラメーターに変換します。</li> <li>VAGen パーツ内と同じ字下げを使用します。</li> </ul>

## シンボリック・パラメーター

次の表に、VAGen のシンボリック・パラメーターと EGL のシンボリック・パラメーターの関係を示します。

表 146. パーツ関連のシンボリック・パラメーター

パーツ関連のシンボリック・パラメーター	対応する EGL のシンボリック・パラメーター
EZECOBOLTYPE	サポートなし
EZEDATA	DATA
EZEDBCS	サポートなし
EZEDESTLIB	サポートなし
EZEDESTNAME	サポートなし
EZEDLI	EZEDLI — DL/I のみ
EZEENTRY	サポートなし
EZEENV	SYSTEM

表 146. パーツ関連のシンボリック・パラメーター (続き)

パーツ関連のシンボリック・パラメーター	対応する EGL のシンボリック・パラメーター
EZEGDATE	EZEGDATE
EZEGENOUT	サポートなし
EZEGMBR	EZEGMBR
EZEGTIME	EZEGTIME
EZEMBR	リンク・エディット・パーツまたはバインド・パーツの場合は、EZEALIAS。それ以外の場合は、EZEMBR。
EZEMBRPATH	サポートなし
EZEMSG	サポートなし
EZEPID	EZEPID
EZEPREPDESTACCOUNT	サポートなし
EZEPREPDESTHOST	サポートなし
EZEPREPDESTDIR	サポートなし
EZEPREPDESTPASSWORD	サポートなし
EZEPREPDESTUID	サポートなし
EZEPREPFTPCMDSBCS	サポートなし
EZEPREPFTPCMDBCS	サポートなし
EZEPREPSEND CMDDBCS	サポートなし
EZEPREPSESSION	サポートなし
EZEPREPSP	サポートなし
EZEPREPSQLDB	サポートなし
EZEPREPWORKDB	サポートなし
EZEPSB	サポートなし — DL/I および IMS のみ
EZEPTYPE	サポートなし
EZESQL	EZESQL
EZETBLNAME	サポートなし
EZETPROC	サポートなし
EZETRAN	サポートなし
EZETRANSFERTYPE	サポートなし
EZETWASIZE	サポートなし
EZEUSERID	サポートなし
EZEVMLoadLIB	適用外 — VM のみ
EZEVSELIB	EZEVSELIB — VSE のみ
EZEXAPP	サポートなし。

表 147. ファイル関連のシンボリック・パラメーター

ファイル関連のシンボリック・パラメーター	対応する EGL のシンボリック・パラメーター
EZEBLK	EZEBLK
EZEDBD	サポートなし



表 147. ファイル関連のシンボリック・パラメーター (続き)

ファイル関連のシンボリック・パラメーター	対応する EGL のシンボリック・パラメーター
EZEDD	EZEDD
EZEDLBL	EZEDLBL — VSE のみ
EZEDSN	EZEDSN
EZELRECL	EZELRECL
EZERECFM	EZERECFM

表 148. ユーザー定義のシンボリック・パラメーター

ユーザー定義のシンボリック・パラメーター	対応する EGL のシンボリック・パラメーター
COB2LIB	COBCICS
COBLIST	サポートなし
DBDLIB	DBDLIB — DL/I のみ
DSNLOAD	DSNLOAD
DSYS	DSYS
ELA	ELA
EZALTXTR	標準ビルド・オプションへの特殊なマイグレーション — 『生成オプション』セクションの transferErrorTransaction="xxx" を参照
EZONEAS2	標準ビルド・オプションへの特殊なマイグレーション - 『生成オプション』のセクションの oneFormItemCopybook="YES" を参照
EZUAUTH	EZUAUTH
EZUINST	EZUINST
PSBLIB	PSBLIB — DL/I のみ
PROCLIB	PROCLIB -- VSE のみ
PWRCLASS	PWRCLASS -- VSE のみ
RESLIB	RESLIB — DL/I および IMS のみ
SQLDBNAM	SQLDBNAM — VSE のみ
SQLPKGNM	SQLPKGNM — VSE のみ
SQLPROPT	SQLPROPT — VSE のみ
SQLSTMDE	SQLSTMDE — VSE のみ
SQLSTOPT	SQLSTOPT — VSE のみ
SQLUSRPW	SQLUSRPW — VSE のみ
VMFMODE	適用外 — VM のみ
VMDISKADDR	適用外 — VM のみ
VUSERLIB	VUSERLIB — VSE のみ



---

## 付録 C. マイグレーション・ツールからのメッセージ

ここでは、マイグレーション・ツールから出されるメッセージについて説明します。メッセージは、接頭部別に次に示すセクションに記載されています。

- HPT.EGL.00xxx - ステージ 1 共通メッセージ
- HPT.EGL.01xxx - VisualAge for Java のステージ 1
- HPT.EGL.02xxx - VisualAge Smalltalk のステージ 1
- IWN.MIG - EGL のステージ 2 および 3

それぞれのメッセージ番号の末尾にある文字は、メッセージの重大度を示す接尾部です。

- *i* - 情報メッセージ。状況を示したり、VisualAge Generator と EGL の言語が異なるために、マイグレーション・ツールがマイグレーション中に情報を除去したことを通知したりします。ユーザー・アクションは必要ありません。
- *w* - 警告メッセージ。起こりうる問題を示します。例えば、マイグレーション・ツールが EGL 構文について最適な推測を行った場合などです。検証または生成時にエラーが検出された場合のみ、ユーザー・アクションが必要です。
- *e* - エラー・メッセージ。マイグレーション・ツールは、EGL 構文について適切な推測を行うことができませんでした。ユーザー・アクションを行って、欠落している情報、または不完全な情報を提供する必要があります。
- *t* - トレース・メッセージ。情報メッセージよりも詳細な状況を示します。トレース・メッセージには、コミット点が記録された時点についての詳しい情報があります。トレース・メッセージは説明を必要としないので、本マイグレーション・ガイドには記載されていません。

---

### VisualAge Generator から EGL マイグレーション・ツールへのメッセージ — ステージ 1

ステージ 1 マイグレーション・ツールは、サンプルとして出荷されます。メッセージは、サンプル・ツール自体の中では翻訳されていません。ただし、本マイグレーション・ガイドの中では、**サンプルに付属するメッセージ**が翻訳されています。

#### ステージ 1 共通メッセージ

次のメッセージは、VisualAge for Java と VisualAge Smalltalk の両方の VAGen マイグレーション・ツールに共通です。

---

**HPT.CM.215.e** ファイル *filename* を開くことができません。戻りコードは *returnCode* (*returnCodeText*) です。

**説明:** 指定されたファイルを開くことができません。*returnCode* と *returnCodeText* が、理由を示しています。*returnCode* 2 は、見付からないファイルを示します。

**ユーザーの処置:** マイグレーション・ツールに有効なファイルを指定します。

---

**HPT.EGL.0001.w** テーブル名 *tableName* は予約語です。名前を変更する必要があります。

**説明:** マイグレーション・ツールは、テーブルの名前を自動的に変更しません。

**ユーザーの処置:** テーブルの名前と、それに対するすべての参照 (プログラムのテーブルおよび追加レコード・リスト、ロジック・ステートメント、データ項目編集ルーチン、マップ編集ルーチン、および UI 編集テーブルの中での参照など) を変更する必要があります。

VAGen 以外による *dataTable* 名の参照 (CICS プログラム定義など) をすべて変更したことを確認してください。この代わりに、マイグレーションが完了するまで待ち、EGL で *dataTable* を名前変更し、EGL *alias* プロパティを使用して元のテーブル名を指定することもできます。

---

**HPT.EGL.0002.w マップ・グループ名 *mapGroupName*** は予約語です。名前を変更する必要があります。

**説明:** マイグレーション・ツールは、マップ・グループの名前を自動的に変更しません。

**ユーザーの処置:** マップ・グループの名前、マップ・グループ内のマップすべての名前、およびマップ・グループの参照すべて (プログラムのマップ・グループ、またはヘルプ・マップ・グループとしての参照を含む) を変更する必要があります。VAGen 以外によるマップ・グループ名の参照 (CICS プログラム定義など) をすべて変更したことを確認してください。この代わりに、マイグレーションが完了するまで待ち、EGL で *formGroup* を名前変更し、EGL *alias* プロパティを使用して元のマップ・グループ名を指定することもできます。

---

**HPT.EGL.0003.w プログラム名 *programName*** は予約語です。名前を変更する必要があります。

**説明:** マイグレーション・ツールは、プログラムの名前を自動的に変更しません。

**ユーザーの処置:** プログラムの名前と、それに対するすべての参照 (CALL、DXFR、および XFER の各ステートメントでの参照、およびリンケージ・テーブル・パーツの中での参照など) を変更する必要があります。また、このプログラムに対応するバインド制御パーツ、またはリンク・エディット・パーツの名前も変更します。VAGen 以外によるプログラム名の参照 (CICS プログラム定義など) をすべて変更したことを確認してください。この代わりに、マイグレーションが完了するまで待ち、EGL でプログラムを名前変更し、EGL *alias* プロパティを使用して元のプログラム名を指定することもできます。

---

**HPT.EGL.0004.w 制御パーツ名 *partName*** は予約語です。名前を変更する必要があります。

**説明:** 指定された制御パーツ名はドット表記を使用していますが、ドットの前にある名前が予約語です。マイグ

レーション・ツールは、ドットの前にある名前がプログラム名であり、この制御パーツがプログラムに密接に関連していることを想定します。マイグレーション・ツールはプログラムの名前を変更しないので、ドット表記の制御パーツの名前も変更しません。

**ユーザーの処置:** プログラムの名前と、それに対するすべての参照 (CALL、DXFR、および XFER の各ステートメントでの参照、およびリンケージ・テーブル・パーツの中での参照など) を変更する必要があります。また、このプログラムに対応するバインド制御パーツ、またはリンク・エディット・パーツの名前も変更します。VAGen 以外によるプログラム名の参照 (CICS プログラム定義など) をすべて変更したことを確認してください。この代わりに、マイグレーションが完了するまで待ち、EGL でプログラムを名前変更し、EGL *alias* プロパティを使用して元のプログラム名を指定することもできます。EGL 予約語のリストについては、239 ページの『付録 A. 予約語』を参照してください。

---

**HPT.EGL.0005.w UI レコード *recordName*** は、予約語であるか、# または @ シンボルで始まっています。名前を変更する必要があります。

**説明:** ステージ 1 マイグレーション・ツールは、UI レコードの名前を変更しません。しかし、ステージ 2 ツールは、ステージ 2 の「接頭部の名前変更」を使用してレコードの名前を変更します。ステージ 2 ツールには、VGUI レコード用の *alias* プロパティも含まれているため、EGL 生成の出力内の名前は、VisualAge Generator の出力内の名前と同一になります。ステージ 3 ツールも、VGUI レコードが含まれているファイルの名前を変更します。単一ファイル・モードでマイグレーションを行う場合、マイグレーション・ツールは同じ変更を行います。

**ユーザーの処置:** なし。推奨される方法は、ステージ 2 マイグレーション・ツールでユーザーのレコードの名前を変更できるようにすることです。これにより、レコードとファイル名に対するすべての参照も変更されます。

---

**HPT.EGL.0006.i *preferenceFile* のマイグレーションにより、*outputList* が作成されます。**

**説明:** *preferenceFile* のマイグレーションにより、*outputList* が作成されます。可能な出力は、マイグレーション・プラン、レポート、およびデータベースの更新です。

**ユーザーの処置:** なし。

---

**HPT.EGL.0007.w** 現行フィルターを基にしたマイグレーション・ファイルは作成されませんでした。

**説明:** 現行フィルターを基にしたマイグレーション・ファイルは作成されませんでした。

**ユーザーの処置:** フィルター設定を変更します。

---

**HPT.EGL.0008.e** 設定オプション *preferenceOption* に対して *PreferenceValue* は無効な値です。

**説明:** この値は、この設定オプションに対して無効です。

**ユーザーの処置:** 設定ファイル内の設定オプション値を変更します。

---

**HPT.EGL.0009.e** マイグレーション・セット *migrationSetName* に対しては、スパン・マップ接尾部の設定値を指定する必要があります。

**説明:** 指定されたマイグレーション・セットには、複数のプロジェクトまたは複数のパッケージにわたるマップ・グループが 1 つ以上含まれています。マップ・グループに必要なプロジェクトまたはパッケージをマイグレーション・ツールが作成できるように、スパン・マップ接尾部の設定値を指定する必要があります。

**ユーザーの処置:** ステージ 1 マイグレーション設定ファイルを編集します。「マッピング (Mapping)」ページの「スパン・マップ (Spanning Maps)」セクションで、「プロジェクト接尾部 (Project suffix)」フィールドと「パッケージ接尾部 (Package suffix fields)」フィールドの値を指定します。詳しくは、Java の場合は 127 ページの『「マッピング (Mapping)」ページ』、Smalltalk の場合は 150 ページの『「マッピング (Mapping)」ページ』を参照してください。

---

**HPT.EGL.0010.w** マイグレーション・アクションが要求されませんでした。

**説明:** ステージ 1 マイグレーション・ツールの出力オプションを選択していません。

**ユーザーの処置:** 1 つ以上のオプションを選択します。これらのオプションにより、マイグレーション・プラン・ファイルの作成、レポートの作成、またはデータベースの更新が可能です。

---

---

**HPT.EGL.0011.i** マイグレーション・セット *migrationSetName* のデータベース・クリーンアップを開始します。

**説明:** マイグレーション・データベースは、指定されたマイグレーション・セットに関する情報をすでに含んでいます。マイグレーション・ツールは、一連の新しい設定を使用してステージ 1 を実行する準備のために、マイグレーション・セット情報を削除しました。

**ユーザーの処置:** なし。

---

**HPT.EGL.0012.i** マイグレーション・セット *migrationSetName* のデータベース・クリーンアップが完了しました。

**説明:** マイグレーション・データベースは、指定されたマイグレーション・セットに関する情報をすでに含んでいます。マイグレーション・ツールは、一連の新しい設定を使用してステージ 1 を実行する準備のために、マイグレーション・セット情報を削除しました。

**ユーザーの処置:** なし。

---

**HPT.EGL.0013.e** それぞれの名前変更規則に、固有の順序値が必要です。

**説明:** 複数の名前変更規則の順序番号が同じです。

**ユーザーの処置:** ステージ 1 マイグレーション設定ファイルを編集して、それぞれの規則のオーダー番号が固有になるように名前変更規則を変更します。

---

**HPT.EGL.0014.i** マイグレーション・セット *migrationSetName-migrationSetVersion* は、*n* 個のエラー・メッセージ、*n* 個の警告メッセージ、および *n* 個の情報メッセージを生成しました。

**説明:** *n* は、指定されたマイグレーション・セットに対してステージ 1 マイグレーション・ツールが発行したメッセージの数です。情報メッセージの数は、メッセージ HPT.EGL.0014.i を含みます。

**ユーザーの処置:** なし。

---

**HPT.EGL.0015.e** 得られた EGL プロジェクト名 *eglProjectName* は、無効文字 *characterList* を含んでいます。名前変更規則を変更してください。

**説明:** ユーザーが指定した名前変更規則を使用して、ステージ 1 マイグレーション・ツールは提示された EGL プロジェクト名を作成しましたが、この名前は EGL プロジェクトの命名規則に適合しません。無効な文字は、*characterList* に示されています。

---



**ユーザーの処置:** ステージ 1 マイグレーション設定ファイルを編集して、有効な EGL プロジェクト名が得られるようにプロジェクトの名前変更規則を変更します。名前変更規則を変更する際には、マッピング・コンテキストとして *both* を指定する名前変更規則の影響を必ず検討してください。

---

#### HPT.EGL.0016.e 得られた EGL パッケージ名

*eglPackageName* は、無効文字 *characterList* を含んでいます。名前変更規則を変更してください。

**説明:** ユーザーが指定した名前変更規則を使用して、マイグレーション・ツールは提示された EGL パッケージ名を作成しましたが、この名前は EGL パッケージの命名規則に適合しません。無効な文字は、*characterList* に示されています。

**ユーザーの処置:** ステージ 1 マイグレーション設定ファイルを編集して、有効な EGL パッケージ名が得られるようにパッケージの名前変更規則を変更します。名前変更規則を変更する際には、マッピング・コンテキストとして *both* を指定する名前変更規則の影響を必ず検討してください。

---

#### HPT.EGL.0017.e 得られた EGL プロジェクト名

*eglProjectName* は、末尾がピリオド (.) であってはなりません。名前変更規則を変更してください。

**説明:** ユーザーが指定した名前変更規則を使用して、マイグレーション・ツールは提示された EGL プロジェクト名を作成しましたが、この名前の末尾がピリオドです。この名前は、EGL プロジェクトの命名規則に適合しません。

**ユーザーの処置:** ステージ 1 マイグレーション設定ファイルを編集して、有効な EGL プロジェクト名が得られるようにプロジェクトの名前変更規則を変更します。名前変更規則を変更する際には、マッピング・コンテキストとして *both* を指定する名前変更規則の影響を必ず検討してください。また、ストリング・コンテキストとして *any*、*back*、または *token* を指定する名前変更規則の影響も考慮してください。

---

#### HPT.EGL.0018.e 得られた EGL パッケージ名

*eglPackageName* は、先頭が数字、または末尾がピリオド (.) であってはなりません。名前変更規則を変更してください。

**説明:** ユーザーが指定した名前変更規則を使用して、マイグレーション・ツールは提示された EGL パッケージ名を作成しましたが、この名前の末尾がピリオドであるか、先頭が数字です。この名前は、EGL パッケージの

命名規則を満たしていません。

**ユーザーの処置:** ステージ 1 マイグレーション設定ファイルを編集して、有効な EGL パッケージ名が得られるようにパッケージの名前変更規則を変更します。名前変更規則を変更する際には、マッピング・コンテキストとして *both* を指定する名前変更規則の影響を必ず検討してください。

---

#### HPT.EGL.0019.i マイグレーション・フィーチャー

*featureName* *versionName* がロードされました。

**説明:** この情報メッセージは、現在 Java ワークスペースまたは Smalltalk イメージにロードされている、マイグレーション・フィーチャーの名前とバージョンを示します。

**ユーザーの処置:** なし。

---

#### HPT.EGL.0020.i マイグレーション・フィーチャー

*featureName* *versionName* はロードされませんでした。*listOfNames* がロードされていません。

**説明:** この情報メッセージは、Java ワークスペースに追加する必要がある、または Smalltalk イメージにロードする必要があるマイグレーション・フィーチャーの名前とバージョンを示します。ただし、1 つ以上の Java パッケージまたは Smalltalk アプリケーションが、マイグレーション・フィーチャーに対して予期されるバージョンではありません。*listOfNames* は、現在ロードされている、予期されるバージョンではない Java パッケージまたは Smalltalk アプリケーションのリストです。

**ユーザーの処置:** ステージ 1 マイグレーション・ツールを変更していなければ、Java の場合はマイグレーション・フィーチャーの再追加、Smalltalk の場合はマイグレーション・フィーチャーの再ロードを試みてください。ステージ 1 マイグレーション・ツールを変更した場合、このメッセージは、変更した Java パッケージまたは Smalltalk アプリケーションを知らせるためのものです。

---

#### HPT.EGL.0021.e 外部化された EGL 予約語リストをロードできません。 *fileName* を検査してください。

**説明:** 指定された *fileName* を見付けることができません。*fileName* はファイルの絶対パス名です。121 ページの『第 4 章 ステージ 1 — Java からの抽出』、または 145 ページの『第 5 章 ステージ 1 — Smalltalk からの抽出』の説明のとおりステージ 1 マイグレーション・ツールをインストールした場合、ファイルの名前とロケーションは正しく設定されます。



**ユーザーの処置:** EGL 予約語のファイルが正しいパスにあり、正しいファイル名が指定されていることを確認します。正しくない場合は、ステージ 1 マイグレーション・ツールのインストール手順を確認します。

---

**HPT.EGL.0022.e** パーツ *partName* の外部ソース形式は無効です。

**説明:** ステージ 1 マイグレーション・ツールは、リポジトリから外部ソース形式の抽出を 3 回試行しましたが、パーツの開始タグと終了タグが有効な対 (例えば、:record と :erecord) ではありません。ステージ 1 マイグレーション・ツールは処理を継続します。ただし、指定された *partName* に対する正しい外部ソース形式がマイグレーション・データベースに格納されません。

**ユーザーの処置:** リポジトリ内の外部ソース形式を調べて、パーツ内に明らかなエラーがないかどうか確認し

ます。リモート・リポジトリを使用している場合は、リポジトリをローカル・ドライブにコピーして、ステージ 1 マイグレーション・ツールの再実行を試してください。問題を解決できない場合には、IBM サポートに連絡してください。

---

**HPT.EGL.0023.e** 重複するパーツがあるために、パーツ *partName* の外部ソース形式は無効です。

**説明:** 同じパーツ型のパーツに、同じパーツ名のものが 1 つ以上あります。マイグレーション・ツールは、マイグレーション・データベースに格納する外部ソース形式を判別できません。

**ユーザーの処置:** マイグレーション・セット内に重複するパーツ名がないように、マイグレーション・セットを変更します。

## VisualAge for Java のステージ 1

次のメッセージは、VisualAge Generator to EGL マイグレーション・ツールの VisualAge for Java バージョンでのみ出されます。

---

**HPT.EGL.0101.e** 現行パッケージ名 *vagenPackageName* から、EGL パッケージ名 *eglPackageName* が作成されましたが、名前の先頭が # または @ であるか、予約語 *reservedWordList* を使用しています。名前を変更する必要があります。

**説明:** EGL パッケージ名のドット表記で、いずれかのワードとして EGL 予約語を使用することはできません。

**ユーザーの処置:** ステージ 1 名前変更規則を使用して、EGL の命名上の制約に違反しない EGL パッケージ名を作成します。EGL 予約語のリストについては、239 ページの『付録 A. 予約語』を参照してください。作成される EGL パッケージ名の先頭が # または @ 記号にならないようにします。

---

**HPT.EGL.0102.e** マイグレーション・セット *migrationSetName* - *migrationSetVersion* は、プロジェクト *projectName* のバージョン *projectVersion1* と *projectVersion2* を参照しています。マイグレーション・セットは作成できませんでした。

**説明:** マイグレーション・ツールは、指定されたマイグレーション・セット・バージョンの上位 PLP プロジェクトを展開しました。展開された上位 PLP プロジェクトは、同じプロジェクト名の複数バージョンを含んでいます。マイグレーションは継続できません。

**ユーザーの処置:** PLP プロジェクトを使用している場合は、PLP チェーンに各プロジェクトのバージョンがただ 1 つ含まれるように、上位 PLP プロジェクトと、そのプロジェクトが参照する下位 PLP プロジェクトを変更します。マイグレーション・プラン・ファイルを手作業で作成した場合は、マイグレーション・セットに対して各プロジェクトのバージョンがただ 1 つ指定されるように、マイグレーション・プラン・ファイルを変更します。

---

**HPT.EGL.0103.e** データベース・ドライバーのロード中にエラーが発生しました。ドライバー: *driverName*。db2java.zip ファイルがクラスパス内にあることを確認してください。

**説明:** ステージ 1 設定ファイルに指定されたデータベース・ドライバーを見付けることができませんでした。

**ユーザーの処置:** 正しいドライバー名とロケーションを指定するように、ステージ 1 設定ファイルを変更します。

---

**HPT.EGL.0104.e** データベースへの接続中にエラーが発生しました。データベース: *databaseName*。

**説明:** ステージ 1 マイグレーション・ツールは、マイグレーション・データベースに接続できませんでした。

**ユーザーの処置:** 指定されたデータベースが作成済みであることを確認してください。また、ステージ 1 設定

ファイル内のユーザー ID とパスワードの設定値を検出して、これらが正しいことを確認してください。

---

**HPT.EGL.0105.e データベース接続のクローズ中にエラーが発生しました。**

**説明:** ステージ 1 マイグレーション・ツールは、マイグレーション・データベースへの接続をクローズできませんでした。

**ユーザーの処置:** ステージ 1 マイグレーション・ツールは、データベース接続のクローズを試行する前にすべてのコミットを行います。VisualAge Generator をシャットダウンすれば、接続のクローズを強制できます。

---

**HPT.EGL.0106.e メソッド *methodName* 内でリポジトリへのアクセス中にエラーが発生しました。**

**説明:** ステージ 1 マイグレーション・ツール用の指定されたメソッドが、リポジトリにアクセスできませんでした。

**ユーザーの処置:** リモート・リポジトリを使用している場合は、リポジトリがアクセス可能であり、ネットワークに問題がないことを確認してください。その後、マイグレーションを再試行します。

---

**HPT.EGL.0107.e XML ファイル *fileName* の書き込み中にエラーが発生しました。**

**説明:** ステージ 1 マイグレーション・ツールは、指定されたファイル名を書き込むことができませんでした。

**ユーザーの処置:** そのファイルのために十分なスペースが使用可能であることを確認してください。その後、マイグレーションを再試行します。

---

**HPT.EGL.0108.w 名前 *partName* に無効な空白があるため、*partType* パーツはマイグレーションから除外されました。このパーツは、パッケージ *packageName*, *versionName* にあります。**

**説明:** VisualAge Generator では、パーツ名の末尾にブランクを含むパーツを作成できます。この場合、一方のパーツ名の末尾にブランクがあることを除いて、同じ名前をもつパーツが 2 つ作成されることがよくあります。名前が少し異なっているので、これらは重複パーツではありません。ただし、外部ソース形式ファイル内では、パーツ名以外のソース・コードが異なっても、これらのパーツ名は同一です。ステージ 1 マイグレーション・ツールは、ブランクを含むパーツ名をマイグレーション・セットから省略します。これは、他の VAGen パーツがブランクで終わるパーツ名を参照でき

ないからです。類似した名前のパーツが存在しない場合は、この手法によって、他のパーツが参照できないパーツがマイグレーションされないようになります。類似した名前のパーツが存在する場合は、この手法によって、ステージ 2 マイグレーション・ツールが正しいパーツ定義を EGL に変換します。

**ユーザーの処置:** なし。ただし、VisualAge Generator 内でパーツを検討して、類似した名前のパーツが存在するかどうか判別することをお勧めします。VAGen パーツ・ブラウザーを使用し、*partName\** を指定してマイグレーション・セット内のすべてのパーツを検索します。

---

**HPT.EGL.0109.e 予期しない例外が発生しました:**  
*javaExceptionStackTrace*

**説明:** ステージ 2 またはステージ 3 のマイグレーション・ツールの実行中に、予期しないエラーが発生しました。

**ユーザーの処置:** *javaExceptionStackTrace* を検討してください。エラーに応じて、無視できる場合も訂正できる場合もあります。例えば、次のようになります。

- 変換できない文字が置換文字によって置き換えられたことを示すメッセージは、無視できます。このメッセージは、外部ソース形式の中で無効文字が検出された場合に発生します。マイグレーション・データベース内では、文字はブランクに置き換えられます。ステージ 1 マイグレーション・ツールは処理を継続します。このマイグレーション・データベースは、ステージ 2 と 3 に使用できます。
- EGL ファイル名を格納するには SQL 列が短すぎることを示すメッセージの場合は、問題を訂正できます。この場合、マイグレーション・データベースの情報は無効なので、ステージ 1 マイグレーション・ツールは処理を停止します。SQL 表定義を変更して列の長さを増やし、ステージ 1 を再度実行することによって、問題を訂正できます。ただし、SQL 列の長さを増やす前に、マイグレーション後にこれらの長い名前をスクロールすることになって構わないかどうか、また長い名前が EGL の制限を超えないかどうかを検討してください。名前変更規則または SQL 列を変更した後、ステージ 1 マイグレーションを再度実行します。

問題を解決できない場合は、IBM サポートに連絡して支援を受けてください。

---

**HPT.EGL.0110.e プロジェクト *projectName* パージョン *versionName* は、リポジトリでは定義されません。**

**説明:** ステージ 1 マイグレーション・ツールは、PLP プロジェクトのチェーンを含むマイグレーション・セッ

トの上位 PLP を展開しました。指定済みプロジェクトおよびバージョンは PLP チェーンで参照されますが、リポジトリでは使用不可です。

**ユーザーの処置:** プロジェクトおよびバージョンをマイグレーション・セットに組み込む必要があるかどうか判別してください。組み込む必要があれば、プロジェクトの要求されたバージョンがリポジトリからパージされていないかどうか確認します。パージされている場合、プロジェクトを復元してから、もう一度マイグレーションします。問題を解決できない場合は、IBM サポートに連絡して支援を受けてください。

---

#### HPT.EGL.0111.e 元の VAGen プロジェクト名

*projectName* から、空の派生 EGL プロジェクト名が作成されました。名前変更規則を変更してください。

**説明:** マイグレーション・ツールが、ユーザーによって指定された名前変更規則を使用して、提示された EGL プロジェクト名を作成しましたが、この名前には文字が含まれていません。

**ユーザーの処置:** ステージ 1 マイグレーション設定フ

## VisualAge Smalltalk のステージ 1

次のメッセージは、VisualAge Generator to EGL マイグレーション・ツールの VisualAge Smalltalk バージョンでのみ出されます。

---

#### HPT.EGL.0201.e 現行アプリケーション名

*vagenApplicationName* から、EGL パッケージ名 *eglPackageName* が作成されましたが、先頭が # または @ であるか、予約語 *reservedWordList* を使用しています。名前を変更する必要があります。

**説明:** EGL パッケージ名のドット表記で、いずれかのワードとして EGL 予約語を使用することはできません。

**ユーザーの処置:** ステージ 1 名前変更規則を使用して、EGL の命名上の制約に違反しない EGL パッケージ名を作成します。EGL 予約語のリストについては、239 ページの『付録 A. 予約語』を参照してください。作成される EGL パッケージ名の先頭が # または @ 記号にならないようにします。

---

#### HPT.EGL.0202.e マイグレーション・セット

*migrationSetName* が構成マップ *configurationMapName* を参照していますが、このマップはリポジトリ内で定義されていません。

**説明:** ステージ 1 マイグレーション・ツールは、指定されたマイグレーション・セットの上位構成マップを展開

ファイルを編集して、有効な EGL プロジェクト名が得られるようにプロジェクトの名前変更規則を変更します。名前変更規則を変更する際には、マッピング・コンテキストとして *both* を指定する名前変更規則の影響を必ず検討してください。

---

#### HPT.EGL.0112.e 元の VAGen パッケージ名

*packageName* から、空の派生 EGL パッケージ名が作成されました。名前変更規則を変更してください。

**説明:** マイグレーション・ツールが、ユーザーによって指定された名前変更規則を使用して、提示された EGL パッケージ名を作成しましたが、この名前には文字が含まれていません。

**ユーザーの処置:** ステージ 1 マイグレーション設定ファイルを編集して、有効な EGL パッケージ名が得られるようにパッケージの名前変更規則を変更します。名前変更規則を変更する際には、マッピング・コンテキストとして *both* を指定する名前変更規則の影響を必ず検討してください。

開しました。しかし、ツールが上位構成マップを展開し、必須のマップとアプリケーションのチェーンを展開したときに、1 つ以上の必須マップがライブラリー内で使用できませんでした。

**ユーザーの処置:** 必要なマップをマイグレーション・セットに組み込む必要があるかどうか判別してください。組み込む必要があれば、構成マップの要求されたバージョンがライブラリーからパージされていないかどうか確認します。パージされている場合は、要求されている構成マップを修復し、マイグレーションを再度実行します。

---

#### HPT.EGL.0203.e ProgramContext が、データベース・エラー errorMessage を検出しました。

**説明:** データベース・エラーが発生しました。考えられる問題は、無効なスキーマ名、ユーザー権限の制限、または SQL 表の欠落です。

**ユーザーの処置:** マイグレーション設定ファイルを訂正してください。問題が解消しない場合は、IBM サポートに連絡して支援を受けてください。

---

#### HPT.EGL.0204.e データベースへの接続中にエラーが発生しました。ErrorMessage。



**説明:** 接続時にデータベース・エラーが発生しました。考えられる問題は、無効なユーザー ID やパスワード名、または無効なデータベース名です。

**ユーザーの処置:** マイグレーション設定ファイルを訂正してください。問題が解消しない場合は、IBM サポートに連絡して支援を受けてください。

---

**HPT.EGL.0205.i** マイグレーションにより、構成マップ *configMapName* の *v* 個のバージョンから、*n* 個のマイグレーション・セットが作成されました。設定ファイルは、バージョン深さを *d* と指定しています。

**説明:** ステージ 1 マイグレーション設定ファイルは、マイグレーションするバージョンの数を *d* と指定しています。ステージ 1 マイグレーション・ツールは、*d* 個のマイグレーション・セット (指定した構成マップのバージョンごとに 1 つずつ) を作成する必要があります。しかし、マイグレーション・ツールは *n* に示されている数のマイグレーション・セットしか作成していません。*v* に示されている数は、マイグレーション・ツールがライブラリー内で検出した、指定された構成マップのバージョン数です。*v* が *d* より小さい場合は、構成マップのバージョン数が予想した数より少なかったことを示しています。この場合、*n* と *v* は等しくなり、これは構成マップのすべてのバージョンからマイグレーション・セットが作成されたことを示しています。*v* が *d* より大きい場合は、ライブラリー内に構成マップのバージョンがさらに多く存在することを示します。この場合、*n* と *d* は等しくなり、これはバージョン深さの設定が満たされたことを示しています。

**ユーザーの処置:** なし。

---

**HPT.EGL.0206.e** マイグレーション・セット *migrationSetName* にロード・エラーが発生しました。

**説明:** マイグレーション・セットをイメージにロードできませんでした。この原因としては、マイグレーション・セットに重複するパーツ名が存在することが考えられます。

**ユーザーの処置:** システム・トランスクリプトを調べて、エラーの原因を判別してください。問題を訂正して、ステージ 1 マイグレーションを再度実行してください。

---

**HPT.EGL.0207.w** 名前 *partName* に無効な空白があるため、*partType* パーツはマイグレーションから除外されました。このパーツは、アプリケーション *applicationName*、*versionName* にあります。

**説明:** VisualAge Generator では、パーツ名の末尾にブランクを含むパーツを作成できます。この場合、一方のパーツ名の末尾にブランクがあることを除いて、同じ名前をもつパーツが 2 つ作成されることがよくあります。名前が少し異なっているので、これらは重複パーツではありません。ただし、外部ソース形式ファイル内では、パーツ名以外のソース・コードが異なっても、これらのパーツ名は同一です。ステージ 1 マイグレーション・ツールは、ブランクを含むパーツ名をマイグレーション・セットから省略します。これは、他の VAGen パーツがブランクで終わるパーツ名を参照できないからです。類似した名前のパーツが存在しない場合は、この手法によって、他のパーツが参照できないパーツがマイグレーションされないようになります。類似した名前のパーツが存在する場合は、この手法によって、ステージ 2 マイグレーション・ツールが正しいパーツ定義を EGL に変換します。

**ユーザーの処置:** なし。ただし、VisualAge Generator 内でパーツを検討して、類似した名前のパーツが存在するかどうか判別することをお勧めします。VAGen パーツ・ブラウザーを使用し、*partName\** を指定してマイグレーション・セット内のすべてのパーツを検索します。

---

**HPT.EGL.0208.e** データベース列 *schemaName.tableName.columnName* のデータは切り捨てられました。

**説明:** 1 つ以上の名前変更規則により、対応する SQL 列に収まらない長さの EGL プロジェクト名、パッケージ名、またはバージョン名が作成されました。

**ユーザーの処置:** より短い EGL プロジェクト名、パッケージ名、またはバージョン名が作成されるように、名前変更規則を変更します。また、DB2 表を変更して、SQL 列の長さを増やすこともできます。ただし、SQL 列の長さを増やす前に、マイグレーション後にこれらの長い名前をスクロールすることになっても構わないかどうか、また長い名前が EGL の制限を超えないかどうかを検討してください。名前変更規則または SQL 列を変更した後、ステージ 1 マイグレーションを再度実行します。

---

**HPT.EGL.0211.e** 元の VAGen 構成マップ名 *configurationMapName* から、空の派生 EGL プロジェクト名が作成されました。名前変更規則を変更してください。

**説明:** マイグレーション・ツールが、ユーザーによって指定された名前変更規則を使用して、提示された EGL プロジェクト名を作成しましたが、この名前には文字が含まれていません。

**ユーザーの処置:** ステージ 1 マイグレーション設定フ

ファイルを編集して、有効な EGL プロジェクト名が得られるように構成マップの名前変更規則を変更します。名前変更規則を変更する際には、マッピング・コンテキストとして *both* を指定する名前変更規則の影響を必ず検討してください。

---

**HPT.EGL.0212.e** 元の VAGen アプリケーション名 *applicationName* から、空の EGL パッケージ名が作成されました。(Original VAGen application name *applicationName* - results in a derived empty EGL package name.) 名前変更規則を変更してください。

.....

---

## VisualAge Generator から EGL マイグレーション・ツールへのメッセージ ステージ 2

EGL 予約語に関して必要な名前変更を行うまで、メッセージには常に VAGen パーツ名が挿入されます。

---

**IWN.MIG.0001.e** 外部ソース形式ファイル *fileName* の構文解析中に例外が発生しました - 無効な外部ソース形式ヘッダー

**説明:** マイグレーション・ツールは、VisualAge Generator 4.5 からエクスポートされた外部ソース形式のみを処理します。指定された外部ソース形式ファイルの先頭行に、VisualAge Generator 4.5 外部ソース形式ファイルの正しいヘッダーがありません。

**ユーザーの処置:** 外部ソース形式ファイルを VisualAge Generator 4.5 にインポートします。これにより、現行パーツが VisualAge Generator 4.5 形式に変換されます。その後、VisualAge Generator 4.5 を使用してパーツをエクスポートし、マイグレーションを再実行します。

---

**IWN.MIG.0002.e** 外部ソース形式ファイル *fileName*, *partType*, *partName* の構文解析中の例外 - *exceptionText*

**説明:** VisualAge Generator からの外部ソース形式構文の解析中に問題が発生しました。この問題の考えられる原因は次のとおりです。

- 引用符のミスマッチ (例: データ項目の通貨フィールド)。
- 制御パーツ内のコメント区切りのミスマッチ。
- 使用しているロケールに対して無効な各国語文字。例えば、2 バイト・ロケール用に設定されていないワー

**説明:** マイグレーション・ツールが、ユーザーによって指定された名前変更規則を使用して、提示された EGL パッケージ名を作成しましたが、この名前には文字が含まれていません。

**ユーザーの処置:** ステージ 1 マイグレーション設定ファイルを編集して、有効な EGL プロジェクト名が得られるようにアプリケーションの名前変更規則を変更します。名前変更規則を変更する際には、マッピング・コンテキストとして *both* を指定する名前変更規則の影響を必ず検討してください。

クステーション上で、中国語などの 2 バイト文字を使用する VAGen ソース・コードをマイグレーションしようとした場合など。

**ユーザーの処置:** VisualAge Generator 内でパーツを訂正し、外部ソース形式を再度エクスポートします。その後、ステージ 2 マイグレーション・ツールを実行して、ファイルを処理します。VisualAge Generator 内でパーツを訂正できない場合は、IBM サポートに連絡して援助を受けてください。このファイルの外部ソース形式のソースをあらかじめ用意しておいてください。

---

**IWN.MIG.0003.e** ファイル *fileName*, *partType*, *partName* の EGL への変換中に例外が発生しました - *exceptionText*

**説明:** EGL ソースの作成中に問題が発生しました。*exceptionText* が、発生した問題を具体的に示しています。

**ユーザーの処置:** IBM サポートに連絡して支援を受けてください。このファイルの外部ソース形式のソースをあらかじめ用意しておいてください。

---

**IWN.MIG.0004.e** 出力ファイル *fileName1* と UI レコードは、名前が同じです - UI レコードは *fileName2* に入ります。

**説明:** 単一ファイルのマイグレーション中は、*fileName1* は指定済みの出力ファイルです。外部ソース・フォーマットのファイルは、指定済み出力ファイルと同じ名前を持つ UI レコードを含んでいます。このフ

ファイルは他のいくつかのパーツも含んでいます。マイグレーション・ツールは、UI レコードを処理する前に指定済み出力ファイルに他のパーツを置きました。このツールは、続いて、*fileName2* という名前の EGL ファイルに UI レコードを置きました。「問題」ビューにエラーが表示されます。

**ユーザーの処置:** *fileName1* に異なる名前を付けます。VGUI レコード名と同じ名前を *fileName2* に付けます。

---

**IWN.MIG.0047.i** マイグレーション・セット *Name* —  
マイグレーションが開始しました。

**説明:** これは、マイグレーション・ツールから出される、状況を示す情報メッセージです。

**ユーザーの処置:** なし。

---

**IWN.MIG.0048.i** マイグレーション・セット *Name* -  
マイグレーションが完了しました。

**説明:** これは、マイグレーション・ツールから出される、状況を示す情報メッセージです。

**ユーザーの処置:** なし。

---

**IWN.MIG.0049.i** EGL *projectName*, *packageName*,  
*fileName* の *partType* *partName* - マイグ  
レーションが開始されました

**説明:** これは、マイグレーション・ツールから出される、状況を示す情報メッセージです。 *partType* は、プログラム、マップ・グループ、またはテーブルのいずれかです。指定された *partName* の関連パーツが、同時にマイグレーションされます。関連パーツは、マイグレーション・データベース内の情報に基づいて、*partName* と同じファイル、あるいは異なるプロジェクト、パッケージ、またはファイルに入っています。プログラムのマイグレーションが開始されると、それぞれの関連したマップ・グループがマイグレーションされ、次にそれぞれの関連したテーブルがマイグレーションされます。最後に、残りの関連パーツ (レコード、共用項目、および関数) がマイグレーションされます。

**ユーザーの処置:** なし。

---

**IWN.MIG.0050.i** プログラム *programName* - その他の  
関連パーツのマイグレーションが開始され  
ました

**説明:** これは、マイグレーション・ツールから出される、状況を示す情報メッセージです。プログラムのマイグレーションが開始されると、それぞれの関連したマップ・グループがマイグレーションされ、次にそれぞれの関連したテーブルがマイグレーションされます。最後

に、残りの関連パーツ (レコード、共用項目、および関数) がマイグレーションされます。メッセージ IWN.MIG.0050.i は、プログラムの残りの関連パーツのマイグレーションが開始されたときに出力されます。

**ユーザーの処置:** なし。

---

**IWN.MIG.0051.e** マイグレーション・セット *planName*,  
*partType*, *partName* の構文解析中に例外が  
発生しました - 無効な外部ソース形式ヘ  
ッダー。

**説明:** マイグレーション・ツールは、VisualAge Generator 4.5 からエクスポートされた外部ソース形式のみを処理します。指定されたパーツの外部ソース形式の先頭行に、VisualAge Generator 4.5 外部ソース形式ファイルの正しいヘッダーがありません。サンプルのステージ 1 マイグレーション・ツールを変更した場合、またはマイグレーション・データベースをロードするためのステージ 1 マイグレーション・ツールをユーザーが独自に作成した場合に、この問題が起こる可能性があります。

**ユーザーの処置:** 外部ソース形式ファイルを VisualAge Generator 4.5 にインポートします。これにより、現行パーツが VisualAge Generator 4.5 形式に変換されます。その後、ステージ 1 マイグレーション・ツールを使用してマイグレーション・セットをエクスポートします。

---

**IWN.MIG.0052.e** マイグレーション・セット *planName*,  
*partType*, *partName* の構文解析中に例外が  
発生しました - *exceptionText*。

**説明:** VisualAge Generator からの外部ソース形式構文の解析中に問題が発生しました。この問題の考えられる原因は次のとおりです。

- 引用符のミスマッチ。例:
  - データ項目の通貨フィールド
- 制御パーツ内のコメント区切りのミスマッチ。
- 使用しているロケールに対して無効な各国語文字。例えば、2 バイト・ロケール用に設定されていないワークステーション上で、中国語などの 2 バイト文字を使用する VAGen ソース・コードをマイグレーションしようとした場合など。

**ユーザーの処置:** VisualAge Generator 内でパーツを訂正し、ステージ 1 マイグレーション・ツールを再実行してデータベースを訂正します。その後、ステージ 2 マイグレーション・ツールを再実行して、更新したパーツを処理します。VisualAge Generator 内でパーツを訂正できない場合は、IBM サポートに連絡して援助を受けてください。問題のあるパーツを含む小さなリポジトリ



ー (.dat ファイル) をあらかじめ用意しておいてください。

---

**IWN.MIG.0053.e** マイグレーション・セット *planName*, *partType*, *partName* の EGL への変換中に例外が発生しました - *exceptionText*。

**説明:** EGL ソースの作成中に問題が発生しました。*exceptionText* が、発生した問題を具体的に示しています。

**ユーザーの処置:** IBM サポートに連絡して支援を受けてください。このパーツの外部ソース形式のソースをあらかじめ用意しておいてください。

---

**IWN.MIG.0054.e** マイグレーション・セット *migrationSetName*, *partType*, *partName* の外部ソース形式が無効です。

**説明:** 指定されたパーツ用に保管されている外部ソース形式が無効です。マイグレーション・ツールは、指定されたマイグレーション・セット内でその他のパーツの処理を継続します。関連パーツを使用したマイグレーションのために、マイグレーション・ツールは指定されたパーツを使用不可と見なします。マイグレーション・ツールは、指定されたパーツに対して、意図的に無効な EGL をマイグレーション・データベースに格納します。格納される EGL は、EZE\_UNKNOWN\_PARTTYPE *partName* です。これにより、「問題」ビューにエラーが示されるようになります。

**ユーザーの処置:** 指定されたパーツを VisualAge Generator 内で検討してください。パーツの外部ソース形式をエクスポートし、単一ファイル・モードでパーツのマイグレーションを試行します。問題を解決できない場合は、IBM サポートに連絡して支援を受けてください。問題のあるパーツを含む小さなリポジトリ (.dat ファイル) または 外部ソース・ファイルをあらかじめ用意しておいてください。

---

**IWN.MIG.0055.e** マイグレーション停止 - エラー限度を超えました。

**説明:** 無効な外部ソース形式のパーツに関するエラーしきい値を超えました。マイグレーション・ツールは処理を停止します。

**ユーザーの処置:** メッセージ IWN.MIG.0054.e のすべての出現箇所を検討してください。ユーザー独自のツールを作成してマイグレーション・データベースをロードしている場合は、ツールが外部ソース形式のコードをマイグレーション・データベースにロードする方法に問題があることが考えられます。問題の原因の判別に役立つ照会については、441 ページの『付録 G. マイグレーション・データベース』を参照してください。

---

**IWN.MIG.0060.e** データベース・ドライバーのロード中にエラーが発生しました。ドライバー: *driverName*

**説明:** 指定されたデータベース・ドライバーを見付けることができません。

**ユーザーの処置:** データベース・ドライバー名を訂正します。また、データベース・ドライバーのロケーションが正しいことを確認してください。

---

**IWN.MIG.0061.e** データベースへの接続中にエラーが発生しました。データベース: *databaseName.errorText*

**説明:** マイグレーション・ツールは、指定されたスキーマ名を使用して、指定されたデータベースに接続できません。 *errorText* フィールドに、接続が失敗した理由についてさらに詳しい情報があります。

**ユーザーの処置:** データベース名を訂正します。リモート・データベースに接続している場合は、データベースがローカル側でカタログ済みであることを確認してください。

---

**IWN.MIG.0063.e** データベース接続のクローズ中にエラーが発生しました。

**説明:** マイグレーションは正常に完了しましたが、マイグレーション・ツールはデータベース接続をクローズできませんでした。

**ユーザーの処置:** EGL 開発環境をシャットダウンしてから、データベースをバックアップしてください。

---

**IWN.MIG.0070.e** ユーザー出口メソッド *renameUserExitName* [*partName*] が存在しません。

**説明:** 名前変更ユーザー出口に対して指定した JAR ファイル、または JAR ファイル内のパッケージとクラスが見付かりませんでした。

**ユーザーの処置:** VAGen マイグレーション設定の中で名前変更ユーザー出口に対して指定した、JAR ファイルのロケーション、パッケージ名、およびクラス名を確認します。

---

**IWN.MIG.0071.e** ユーザー出口メソッド *renameUserExitName* [*partName*] には、必要なメソッド・シグニチャーがありません。

**説明:** 名前変更ユーザー出口を使用するには、シグニチャー *renameUserExit(String s, Connection c)* をもつメソッドを組み込む必要があります。VAGen マイグレーション

ョン設定の中で名前変更ユーザー出口に対して指定した JAR ファイル、パッケージ、およびクラスに、このメソッドが存在しませんでした。

**ユーザーの処置:** クラス定義を検討し、必要なメソッド・シグニチャーを組み込んだことを確認します。また、VAGen マイグレーション設定の中で、名前変更ユーザー出口に対して正しい JAR ファイルのロケーション、パッケージ名、およびクラス名を指定したことを確認します。

---

#### IWN.MIG.0072.e ユーザー出口メソッド

`renameUserExitName [partName]` は、Java 言語アクセス制御を実行しますが、基本メソッドにアクセスできません。

**説明:** マイグレーション・アプリケーションは、指定されたユーザー出口クラスの定義にアクセスできません。

**ユーザーの処置:** 名前変更ユーザー出口クラスが `public` として定義されていて、指定されたパッケージ内にあることを確認します。

---

#### IWN.MIG.0073.e ユーザー出口メソッド

`renameUserExitName [partName]` は、例外をスローして予期せず終了しました。

**説明:** メソッド `renameUserExit(String s, Connection c)` は `NULL` 値を返します。マイグレーションは継続します。マイグレーション・ツールは、オリジナルの VAGen パーツ名を使用します。

**ユーザーの処置:** `try ... catch` ブロックを使用して、名前変更ユーザー出口のコードを囲みます。このメッセージを回避するには、例外が発生した場合にオリジナルの VAGen パーツ名を返します。

---

#### IWN.MIG.0080.i VAGen マイグレーション設定ファイル `pref_store.ini` が見付かりません。デフォルトが想定されます。

**説明:** VAGen マイグレーション設定ファイルがありません。マイグレーション・ツールは、デフォルト値を設定に使用します (例えば、名前変更接尾部とヘルプ・マップ接尾部)。この原因としては、マイグレーション中に新しいワークスペースを指定したために、設定が存在しないことが考えられます。設定ファイルは、次の場所にあります。

```
workspace-directory¥.metadata¥.plugins
¥org.eclipse.core.runtime¥.settings
¥com.ibm.etools.egl.vagenmigration.prefs
```

**ユーザーの処置:** マイグレーション設定のデフォルト値については、171 ページの『VAGen マイグレーションの設定』を参照してください。

---

#### IWN.MIG.0081.i ファイル `fileName` - マイグレーションが完了しました。

**説明:** マイグレーション・ツールは、指定されたファイルの処理を完了しました。

**ユーザーの処置:** ログ・メッセージを検討して、マイグレーションの結果を確認してください。

---

#### IWN.MIG.0082.e ファイル `fileName` - 必要パラメーターが指定されていません。

**説明:** 1 つ以上の必要パラメーターが指定されていません。 `-importFile` パラメーターは常に必要です。

`-importFile` パラメーターが外部ソース形式ファイルを指定している場合は、`-eglFile` パラメーターと `-package` パラメーターも必要です。

**ユーザーの処置:** バッチ・コマンド・ファイルを検討して、指定されていないパラメーターを判別してください。パラメーターを追加し、バッチ・コマンド・ファイルを再実行します。

---

#### IWN.MIG.0083.e ファイル `fileName` - パラメーター `parmName` に値が割り当てられていません。

**説明:** `parmName` は、`-importFile`、`-eglFile`、`-package` のいずれかです。

**ユーザーの処置:** バッチ・コマンド・ファイルを訂正し、再実行します。

---

#### IWN.MIG.0084.e ファイル `fileName` - パラメーター `parmName`、値 `value` が無効です。

**説明:** `parmName` は、`-importFile`、`-eglFile`、`-package` のいずれかです。パラメーター名には大/小文字の区別があります。

**ユーザーの処置:** バッチ・コマンド・ファイルを訂正し、再実行します。

---

#### IWN.MIG.0085.e ファイル `fileName` - パラメーター・リストの中で無効なパラメーターが渡されました。

**説明:** バッチ・コマンド・ファイルに問題があります。1 つ以上のパラメーターが誤って入力されました。有効なパラメーターは、`-importFile`、`-eglFile`、`-package`、および `-overwrite` のみです。

**ユーザーの処置:** バッチ・コマンド・ファイルを訂正し、再実行します。

---

**IWN.MIG.0095.e 関数 *functionName* - EZESCRPT**  
は、マイグレーション対象としてサポート  
されていません。

**説明:** 指定された関数は、EZESCRPT 特殊機能語を使用するステートメントを含んでいます。EZESCRPT は、VAGen マイグレーション・ツールによって現在サポートされていません。マイグレーション・ツールはこの関数をマイグレーションしますが、EZESCRPT を使用するステートメントをコメント化します。

**ユーザーの処置:** EGL 関数を検討してください。このリリースでは、この関数を使用するプログラムは生成または実行できません。

---

**IWN.MIG.0101.e データ項目 *dataItemName* -  
*editRoutineName* の編集ルーチン・タイプ**  
を判別できません。関数が想定されます。

**説明:** VisualAge Generator は、データ項目のマッピング編集ルーチンとして、EZEC10、EZEC11、関数、またはテーブルをサポートします。EGL は、*dataItem* に対して *validatorFunction* と *validatorDataTable* の両プロパティをサポートします。マイグレーション・ツールは、マッピング編集ルーチンを次のように変換します。

- EZEC10 と EZEC11 は、*validatorFunction* プロパティにマイグレーションされます。
- *editRoutineName* に指定されたパーツがマイグレーション時に使用可能で、このパーツが関数ならば、*editRoutineName* は *validatorFunction* プロパティにマイグレーションされます。また、*editRoutineName* が 7 文字より長い場合も、マイグレーション・ツールは編集ルーチンを *validatorFunction* プロパティにマイグレーションします。これは、テーブル名が VisualAge Generator では 7 文字に制限されているからです。
- *editRoutineName* に指定されたパーツがマイグレーション時に使用可能で、このパーツがテーブルならば、*editRoutineName* は *validatorDataTable* プロパティにマイグレーションされます。また、編集メッセージが項目に対して指定されている場合も、マイグレーション・ツールは編集ルーチンを *validatorDataTable* プロパティにマイグレーションします。これは、VisualAge Generator が編集メッセージを EZEC10、EZEC11、またはテーブルのみと組み合わせで使用するためです。
- *editRoutineName* に指定されたパーツがマイグレーション時に使用不可で、*editRoutineName* が 7 文字以下であり、編集メッセージが指定されていない場合、マイグレーション・ツールは *editRoutineName* が関数であると想定し、*validatorFunction* プロパティにマイ

グレーションします。メッセージ IWN.MIG.0101.e はこの場合にのみ出されます。

**ユーザーの処置:** 指定された編集ルーチンが関数でなければ、EGL *dataItem* 定義を変更し、*validatorFunction* プロパティを *validatorDataTable* プロパティに変更します。その他の考慮事項については、63 ページの『共用データ項目のマッピング編集ルーチン』にある、編集ルーチンに関する説明を参照してください。

---

**IWN.MIG.0102.w パーツ *partName* は、共用データ項目 *dataItemName* を使用しています - プリミティブ定義にマイグレーションできません。型定義を使用します。**

**説明:** レコード、テーブル、呼び出し先パラメーター・リスト、関数仮パラメーター・リスト、または関数ローカル・ストレージ内で共用データ項目が使用されているときに、VAGen の共用データ項目を EGL のプリミティブ定義にマイグレーションする設定を選択しました。*dataItemName* によって示される項目は、*partName* によって示されるパーツ内で使用されていますが、マイグレーション時にデータ項目定義が使用不可でした。マイグレーション・ツールは、マイグレーションしたコードが有効になるように、データ項目名を型定義として使用します。

**ユーザーの処置:** 型定義を使用する場合は、アクションは必要ありません。プリミティブ定義を使用する必要がある場合は、正しい項目特性を使用するように、指定されたパーツを変更します。あるいは、マイグレーション・セット (または、単一ファイル・モードでマイグレーションしている場合は、外部ソース形式ファイル) に共用データ項目を組み込んで、再度マイグレーションを行います。

---

**IWN.MIG.0103.w データ項目 *dataItemName* -- 設定が原因で *evensql=y* が無視されます。**

**説明:** 指定されたデータ項目パーツは *evensql=y* の VAGen PACK (EGL 10 進数) 項目です。*evensql=y* を指定した VAGen マイグレーション設定は受け入れられません。この設定に基づいて、マイグレーション・ツールは PACK 項目を次に大きい奇数精度 (最大 18) に変換します。EGL 精度の違いは以下のとおりです。

- *evensql=n* が項目に指定されると、EGL 精度は常に (VAGen バイト \* 2) - 1 と計算され、最大値は 18 です。
- *evensql=y* が項目に指定されると、EGL 精度は設定に基づいて計算されます。
  - 設定が選択されている (*evensql=y* を受け入れない) と、EGL 精度は、(VAGen バイト \* 2) - 1 と計



算され、最大値は 18 です。メッセージ  
IWN.MIG.0103.w はこの場合にのみ出されます。

- 設定が選択されていない (evensql=y を受け入れる) と、EGL 精度は、(VAGen バイト \* 2) - 2 と計算され、最大値は 18 になります。メッセージ  
IWN.MIG.0103.w は、出されません。

**ユーザーの処置:** なし。ただし、この項目を SQL WHERE 文節または EGL 準備ステートメントで使用することを検討してください。この項目の定義が、SQL 表定義と正確に一致しないと、パフォーマンスに影響が出る可能性があります。詳しくは、216 ページの『VisualAge Generator 互換モードの使用を中止する場合』および 59 ページの『偶数の長さの PACK データ項目』にある EVENSQL に関する情報を参照してください。

---

**IWN.MIG.0201.i** レコード *recordName* はレベル 77 項目を含んでいます。追加レコード *level77RecordName* を作成します。

**説明:** VisualAge Generator は、作業用ストレージ・レコード内のレベル 77 項目をサポートします。EGL はレベル 77 項目をサポートしません。EGL は、独立データ項目の定義を許可しません。マイグレーション・ツールは、レベル 77 項目を含む作業用ストレージ・レコードを 2 つの分離した basicRecord (1 つは非レベル 77 項目を収容し、もう 1 つはレベル 77 項目を収容) に分割します。作業用ストレージ・レコードがレベル 77 項目のみを含んでいる場合、マイグレーション・ツールはレベル 77 basicRecord のみを作成します。プログラムがレベル 77 項目を含む 1 次作業用ストレージ・レコードを指定している場合、マイグレーション・ツールは、オリジナルの basicRecord とレベル 77 basicRecord の両方の宣言をプログラム定義に組み込みます。

**ユーザーの処置:** なし。プログラムとステートメントのマイグレーション時に *recordName* が使用できない場合の影響など、その他の考慮事項については、66 ページの『レコード内のレベル 77 項目』にある、レコード内のレベル 77 項目に関する説明を参照してください。

---

**IWN.MIG.0202.i** レコード *recordName* は、*redefinedRecordName* を再定義します。

**説明:** *recordName* は、再定義されるレコードとして *redefinedRecordName* を指定する VAGen 再定義レコードです。*recordName* は、*redefinedRecordName* によって使用されているものと同じ物理ストレージに対して、異なる項目レイアウトを指定しています。EGL は、レコード・パーツ内の再定義情報を保持しません。この情報は、プログラム内でのみ保持されます。マイグレーション・ツールは、オリジナルの VAGen

*redefinedRecordName* の情報を示すコメントを *recordName* に組み込みます。プログラムのマイグレーション時に、*recordName* が使用可能であり、その結果として VisualAge Generator 内でオーバーレイ定義が行われる場合、マイグレーション・ツールは *recordName* 宣言に *redefines* プロパティを組み込みます。

**ユーザーの処置:** なし。プログラムのマイグレーション時に *recordName* が使用できない場合の影響など、その他の考慮事項については、65 ページの『再定義レコード』にある、再定義レコードに関する説明を参照してください。

---

**IWN.MIG.0203.e** レコード *recordName* には、項目が含まれていません。

**説明:** VisualAge Generator は、項目を含まないレコード・パーツの保管を許容します。ただし、このレコードは無効なので、プログラム内では使用できません。EGL は、項目を含まないレコード・パーツを許容しません。マイグレーション・ツールは、このレコードをマイグレーションします。

**ユーザーの処置:** このレコードが引き続き必要かどうか判断してください。必要な場合は、レコードを編集して 1 つ以上のデータ項目を追加します。必要でなければ、レコードを削除します。

---

**IWN.MIG.0204.e** レコード *recordName* - 代替仕様レコード *altspecRecord* は使用不可です。SQL 表名を判別できません。

**説明:** レコード *recordName* は、代替仕様レコード *altspecRecord* を指定しており、このレコードは *recordName* の項目構造を指定します。VisualAge Generator では、SQL レコードの代替仕様レコードは SQL 表名も指定します。EGL では、代替仕様レコードは *embed* ステートメントを使用して構造のみを指定します。表名は、それぞれの SQL レコード・パーツ定義内で指定する必要があります。*recordName* のマイグレーション時に、代替仕様レコードとして指定されたレコードがマイグレーション中に使用できませんでした。マイグレーション・ツールは正しい表名を判別できず、*tableNames* プロパティを *###TABLES\_NOT\_FOUND###* に設定します。*recordName* の定義は無効です。

**ユーザーの処置:** *recordName* を編集し、VAGen 代替仕様レコード (*altspecRecord*) から *tableNames* または *tableNameVariables* のどちらかまたは両方のプロパティをコピーします。*tableNames* プロパティは、実際の SQL 表名を指定します。*tableNameVariables* プロパティは、表名ホスト変数を指定します。*recordName* が実際の SQL 表名と SQL 表名ホスト変数を混合して参照

する場合は、`tableNames` と `tableNameVariables` の両プロパティを使用できます。その他の考慮事項については、68 ページの『代替仕様レコード』を参照してください。

---

**IWN.MIG.0205.e** レコード `recordName` - 代替仕様レコード `altspecRecord` は使用不可です。SQL キー項目を判別できません。

**説明:** レコード `recordName` は、代替仕様レコード `altspecRecord` を指定しており、このレコードは `recordName` の項目構造を指定します。VisualAge Generator が SQL レコードのデフォルト選択条件を決定する際に、VisualAge Generator は代替仕様レコード内で `key=yes` を指定する項目を、`recordName` 内で指定されたキー項目 (存在する場合) と組み合わせます。キーは、レコード構造内で項目がリストされている順に組み合わせられます。EGL では、代替仕様レコードは `embed` ステートメントを使用して構造のみを指定します。すべてのキー項目が、それぞれの SQL レコード・パーツ定義内で指定されている必要があります。`recordName` のマイグレーション時に、代替仕様レコードとして指定されたレコードがマイグレーション中に使用できませんでした。マイグレーション・ツールは正しいキー項目を判別できず、`keyItems` プロパティを `###KEYS_NOT_FOUND###` に設定し、その後に `recordName` からのキー項目 (存在する場合) を付けます。 `recordName` の定義は無効です。

**ユーザーの処置:** `recordName` を編集し、`keyItems` プロパティを変更して、`###KEYS_NOT_FOUND###` を項目名のリストに置き換えます。これらの項目名は、代替仕様レコード (`altspecRecord`) 内で `key=yes` を指定したものです。代替仕様レコードのキー項目を、`recordName` 内の VAGen 定義に指定されたキー項目と組み合わせ、`keyItems` プロパティがこれらの項目をレコード構造内での出現順と同じ順序でリストするようにします。代替仕様レコード内で項目が `key=yes` として指定され、かつ `recordName` 内でキー項目として指定されている場合は、`recordName` の `keyItems` の組み合わせりリストにその項目を 1 回だけ組み込みます。その他の考慮事項については、68 ページの『代替仕様レコード』を参照してください。

---

**IWN.MIG.0206.i** SQL レコード `recordName` — 代替仕様レコードを指定しないキー項目 `keyItem` を含んでいます。

**説明:** VisualAge Generator 上では、代替仕様レコードを指定しなくても、キー項目を指定する SQL レコードを保管できます。ただし、この状態では、テストおよび生成時に VisualAge Generator はキー項目を無視しま

す。キー項目は、代替仕様レコードも存在する場合に限って意味を持ちます。

**ユーザーの処置:** なし。このキー項目は、VisualAge Generator 内で無視されていました。マイグレーション・ツールは、マイグレーション時にこの項目を除去します。

---

**IWN.MIG.0207.i** レコード `recordName` - レベル 77 項目のみを含む代替仕様レコード `altspecRecord` を指定しています。`embed` ステートメントは省略されます。

**説明:** `altspecRecord` によって示されたレコードは、レベル 77 項目のみを含む作業用ストレージ・レコードです。`recordName` が作業用ストレージ・レコードを代替仕様として指定する場合、VisualAge Generator は `altspecRecord` からの構造 (非レベル 77 項目) ののみを使用します。`altspecRecord` の構造内に項目が存在しないので、マイグレーション・ツールは `embed` ステートメントを省略します。

**ユーザーの処置:** なし。ただし、`recordName` は空のレコードなので削除できます。プログラム内で、`recordName` の参照を必ずすべて削除してください。

---

**IWN.MIG.0208.e** レコード `recordName` - 代替仕様レコード `altspecRecord` が使用できません。`!itemColumnName` 変数の SQL 列名を判別できません。

**説明:** レコード `recordName` は、代替仕様レコード `altspecRecord` を指定しており、このレコードは `recordName` の項目構造を指定します。VisualAge Generator が SQL レコードのデフォルト選択条件を決定する際に、VisualAge Generator は `!itemColumnName` 変数を対応する SQL 列名に変換します。EGL の場合、`!itemColumnName` 変数はサポートされません。それぞれの SQL レコード・パーツ定義ごとに、デフォルト選択条件の中で SQL 列名を明示的に指定する必要があります。`recordName` のマイグレーション時に、代替仕様レコードとして指定されたレコードがマイグレーション中に使用できませんでした。マイグレーション・ツールは、デフォルト選択条件の中で、1 つ以上の `!itemColumnName` 変数に対応する正しい SQL 列名を判別できませんでした。マイグレーション・ツールは、EGL のデフォルト選択条件の中で `!itemColumnName` を使用します。`recordName` の定義は無効です。

**ユーザーの処置:** `recordName` を編集し、`defaultSelectCondition` プロパティを変更して、`!itemColumnName` 変数を VAGen 代替仕様レコード (`altspecRecord`) からの対応する SQL 列名に置き換えます。

その他の考慮事項については、68 ページの『代替仕様レコード』にある、!itemColumnName 変数に関する説明を参照してください。

---

**IWN.MIG.0209.e** レコード *recordName* - 代替仕様レコード *altspecRecord* に項目がありません。  
**embed** ステートメントが省略されます。

**説明:** レコード *recordName* は、代替仕様レコード *altspecRecord* を指定しており、このレコードは *recordName* の項目構造を指定します。ただし、この代替仕様レコードにはデータ項目がありません。マイグレーション・ツールは、*recordName* の定義から **embed** ステートメントを省略します。

**ユーザーの処置:** なし。ただし、*recordName* と *altspecRecord* を検討して、データ項目を組み込む必要があるか、またはこれら 2 つのレコードを削除できるかを判断する必要があります。プログラム内で、これらのレコードの参照を必ずすべて削除してください。

---

**IWN.MIG.0210.e** レコード *recordName* -  
**!itemColumnName** 変数の列名を判別できません。

**説明:** 指定されたレコードのデフォルト選択条件は、1 つ以上の VAGen !itemColumnName 変数を使用しています。VAGen !itemColumnName 変数は、実際の SQL 列名に対応する SQL レコード定義内の項目名を指定します。VisualAge Generator は、テストおよび生成時に、!itemColumnName 変数の実際の SQL 列名を SQL レコードから判別します。EGL は、!itemColumnName 変数をサポートしません。代わりに、EGL の場合は、変更された SQL ステートメントの中で実際の SQL 列名を使用する必要があります。*recordName* によって示されたレコードが VisualAge Generator では無効な場合は、メッセージ IWN.MIG.0210.e が出されます。この場合、そのレコードは、レコードまたはその代替仕様レコード内で定義されていない !itemColumnName 変数を 1 つ以上使用しています。マイグレーション・ツールは、実際の SQL 列名への置換を実行できません。

**ユーザーの処置:** レコードを編集し、!itemColumnName 変数を正しい SQL 列名に変更します。

---

**IWN.MIG.0211.w** レコード *recordName*、データ項目 *dataItemName* - 設定が原因で **evensql=y** が無視されます。

**説明:** 指定されたレコードに、指定された非共用データ項目が含まれています。レコード定義は、この非共用データ項目が、**evensql=y** の VAGen PACK (EGL 10 進数) 項目であることを指定しています。VAGen マイグレーション設定は、**evensql=y** を受け入れないことを指

定しています。この設定に基づいて、マイグレーション・ツールは PACK 項目を次に大きい奇数精度 (最大 18) に変換します。EGL 精度の違いは以下のとおりです。

- **evensql=n** が項目に指定されると、EGL 精度は常に (VAGen バイト \* 2) - 1 と計算され、最大値は 18 です。
- **evensql=y** が項目に指定されると、EGL 精度は設定に基づいて計算されます。
  - 設定が選択されている (**evensql=y** を受け入れない) と、EGL 精度は、(VAGen バイト \* 2) - 1 と計算され、最大値は 18 です。メッセージ IWN.MIG.0211.w はこの場合にのみ出されます。
  - 設定が選択されていない (**evensql=y** を受け入れる) と、EGL 精度は、(VAGen バイト \* 2) - 2 と計算され、最大値は 18 になります。メッセージ IWN.MIG.0211.w は出されません。

**ユーザーの処置:** なし。ただし、この項目を SQL WHERE 文節または EGL 準備ステートメントで使用することを検討してください。この項目の定義が、SQL 表定義と正確に一致しないと、パフォーマンスに影響が出る可能性があります。詳しくは、216 ページの『VisualAge Generator 互換モードの使用を中止する場合』および 59 ページの『偶数の長さの PACK データ項目』にある EVENSQL に関する情報を参照してください。

---

**IWN.MIG.0212.e** レコード *recordName* - 代替仕様レコード *altspecRecord* が使用できません。なんらかの項目名で **dliFieldName** プロパティに関する指定変更が必要であるのかどうかを判別できません。

**説明:** レコード *recordName* は、代替仕様レコード *altspecRecord* を指定しており、このレコードは *recordName* の項目構造を指定します。VisualAge Generator では、DL/I セグメント・レコード内のフィールド名は DL/I PSB 内の名前に一致しなければなりません。EGL では、フィールド名は予約語にすることはできず、また、# または @ シンボルで始めることもできません。EGL ではさらに、DL/I セグメント・レコードのフィールド名を、DL/I PSB で決められている 8 文字の制限より長くできます。DL/I PSB のフィールド名が EGL フィールド名に一致しない場合、EGL は、対応する EGL フィールド名のために DL/I PSB で使用される名前を提供するために **dliFieldName** プロパティを使用します。代替仕様のレコードが利用できないために、マイグレーション・ツールは、EGL 命名規則によりいずれかのフィールド名が名前変更されるかどうかを判別できません。



**ユーザーの処置:** EGL 命名規則によりレコード内のフィールド名の名前変更が必要であるかどうかを決定するために、*altspecRecord* のレコード定義を検討します。フィールド名が名前変更された場合、対応する *dliFieldName* プロパティを指定するため、*recordName* を編集し、各フィールド名変更用の *embed* ステートメントに対するオーバーライドを追加します。*dliFieldName* プロパティの値はオリジナルの VAGen フィールド名である必要があります。

フィールドに対するオーバーライドで組み込みステートメントをコード化する方法に関する例については、265 ページの表 76 を参照してください。

---

**IWN.MIG.0251.w** UI レコード *recordName* は、予約語であるか、# または @ シンボルで始まっています。これは、*newRecordName* に名前変更されました。

**説明:** UI レコード *recordName* は、EGL 予約語と競合しているか、# または @ シンボルで始まっています。ステージ 2 マイグレーション・ツールは、マイグレーションの設定で指定された接頭部の名前変更に基づいて、UI レコードを *newRecordName* に名前変更しました。ステージ 2 ツールには、VGUI レコード用の *alias* プロパティも含まれているため、EGL 生成の出力内の名前は、VisualAge Generator の出力内の名前と同一になります。EGL が VGUI レコード名とファイル名との一致を必要とするため、Stage 3 マイグレーション・ツールは、VGUI レコードのファイル名を *newRecordName.egl* に変更しました。単一ファイル・モードでマイグレーションを行う場合、マイグレーション・ツールは同じ変更を行います。

**ユーザーの処置:** なし。

---

**IWN.MIG.0301.e** テーブル名 *tableName* は予約語です。名前を変更する必要があります。

**説明:** マイグレーション・ツールは、テーブルの名前を自動的に変更しません。

**ユーザーの処置:** テーブルの名前とその参照すべてを変更する必要があります。この対象には、次の場所での参照が含まれます。

- プログラムの使用宣言ステートメント
- プログラムおよび関数のロジック・ステートメント
- データ項目の *validatorDataTable* プロパティ
- 書式フィールドの *validatorDataTable* プロパティ
- VGUI レコード・フィールド *validatorDataTable* プロパティ

生成される *dataTable* の名前としてオリジナルのテーブル名を保持する必要がある場合は、オリジナルの

*dataTable* 名に *alias* プロパティを設定します。*alias* プロパティを指定しない場合は、CICS プログラム定義など、EGL 以外による *dataTable* 名の参照すべてを必ず変更してください。

---

**IWN.MIG.0401.e** マップ・グループ (*formGroup*) 名 *mapGroupName* は予約語です。名前を変更する必要があります。

**説明:** マイグレーション・ツールは、マップ・グループ (*formGroup*) の名前を自動的に変更しません。

**ユーザーの処置:** *formGroup* の名前と、その参照すべて (プログラムの使用宣言ステートメント内での参照など) を変更する必要があります。生成される *formGroup* の名前としてオリジナルのマップ・グループ名を保持する必要がある場合は、オリジナルのマップ・グループ (*formGroup*) 名に *alias* プロパティを設定します。*alias* プロパティを指定しない場合は、CICS プログラム定義など、EGL 以外による *formGroup* 名の参照すべてを必ず変更してください。

---

**IWN.MIG.0402.e** マップ・グループ *mapGroupName* -- 複数の装置が同じ縦の長さと同幅を指定していますが、異なる浮動域を指定していません。装置: *devicesList*

**説明:** VisualAge Generator は、同じ装置サイズをもつ装置タイプに対して、異なる浮動域サイズを指定することを許容します (ただし推奨されません)。EGL の場合は、それぞれの装置サイズに対して浮動域を 1 つだけ指定できます。マイグレーション・ツールは、それぞれの装置サイズごとに浮動域サイズをマイグレーションします。複数の VAGen 装置が変換によって同じ EGL 装置のサイズ仕様およびマージン仕様を持つようになった場合、マイグレーション・ツールは EGL に関して 1 つのエントリーのみを組み込みます。*formGroup* 定義は無効です。このメッセージは、VisualAge Generator では異なる浮動域情報を指定していた、同サイズの装置タイプのグループごとに繰り返されます。

**ユーザーの処置:** *formGroup* を編集して、同サイズの装置のグループごとに 1 つを除く浮動域仕様をすべて削除します。

---

**IWN.MIG.0403.e** *formGroup formGroupName* - *formGroup* 内で書式をネストするための編集が必要です。

**説明:** 単一ファイル・モードでマイグレーションを行う場合、マイグレーション・ツールは *formGroup* 内で書式をネストしません。代わりにマイグレーション・ツールは、*formGroup* に属する書式の名前を示す EGL の *use* ステートメントを挿入します。マイグレーション・

ツールは、それぞれの書式の先頭と末尾に、所属する `formGroup` を示すコメントを挿入します。

**ユーザーの処置:** `formGroup` を含むファイルを編集して、書式が `formGroup` 内でネストされるように書式を移動します。 `formGroup` 内の `use` ステートメントが、書式を移動する先を示しています。書式を `formGroup` 内でネストした後、`use` 宣言ステートメントを除去します。

---

**IWN.MIG.0404.w** マップ・グループ `mapGroupName` - 使用している装置 `deviceName` とサイズ `depth,width` は、サポートされなくなりました。変更する必要があります。

**説明:** EGL の COBOL 生成機能は、浮動域とテキスト書式に対して、VisualAge Generator がサポートする装置タイプの一部をサポートしなくなりました。マイグレーション・ツールは、`ScreenFloatingArea` プロパティ内の `screenSize` プロパティに、オリジナルの縦の長さと同幅を組み込みます。ただし、この `screenSize` は EGL の COBOL 生成機能によってサポートされなくなりました。COBOL の生成を行う場合は、生成機能からエラー・メッセージが出されます。

**ユーザーの処置:** COBOL の生成を予定している場合は、EGL の `formGroup` を編集し、この縦の長さと同幅の `ScreenFloatingArea` プロパティを除去するか、縦の長さと同幅をサポートされるサイズに変更します。また、`formGroup` 内の `textForm` を変更して、新しい縦の長さと同幅に正しく適合するように変数と定数を再配置する必要があります。

---

**IWN.MIG.0501.e** ヘルプ・マップ・グループ `mapGroupName` は、変数フィールドのあるマップ `mapName` を含んでいます - `mapName` は、プログラムのメインのマップ・グループにある同じマップ名と競合しています。

**説明:** VisualAge Generator の場合は、プログラムのメインのマップ・グループとヘルプ・マップ・グループ内で同じマップ名を使用できます。EGL の場合は、プログラムの 2 つの `formGroup` 内で重複する書式名は使用できません。重複する名前をもつ書式がプログラムによって使用されていなくても、この制限が適用されます。プログラムのヘルプ・マップ・グループ内のマップが、プログラムのメインのマップ・グループと競合し、定数フィールドのみを含んでいる場合、マイグレーション・ツールはこれらのマップの名前を変更します。プログラムのヘルプ・マップ・グループ内のマップが変数フィールドを含んでいる場合は、その名前がプログラムのメインのマップ・グループ内にあるマップ名と競合していても、マイグレーション・ツールはマップの名前を変更し

ません。これは、そのプログラムのメインのマップ・グループとしてそのヘルプ・マップ・グループを指定する他のプログラムが、そのマップを使用している可能性があるからです。

**ユーザーの処置:** ヘルプ `formGroup` を編集し、書式の名前を変更します。また、この `formGroup` を使用するすべてのプログラム内で、書式定義とこの書式の参照すべてを必ず変更します。その他の考慮事項については、77 ページの『マップ名とヘルプ・マップ名』にある、マップ名に関する説明を参照してください。

---

**IWN.MIG.0502.e** マップ・グループ `mapGroupName`、マップ `mapName`、および変数フィールド `mapItemName` - `editRoutineName` の編集ルーチン・タイプを判別できません。関数が想定されます。

**説明:** VisualAge Generator は、マップ変数のマップ編集ルーチンとして、EZEC10、EZEC11、関数、またはテーブルをサポートします。EGL は、書式フィールドに対して `validatorFunction` 関数と `validatorDataTable` プロパティの両方をサポートします。マイグレーション・ツールは、マップ編集ルーチンを次のように変換します。

- EZEC10 と EZEC11 は、`validatorFunction` プロパティにマイグレーションされます。
- `editRoutineName` に指定されたパーツがマイグレーション時に使用可能で、このパーツが関数ならば、`editRoutineName` は `validatorFunction` プロパティにマイグレーションされます。また、`editRoutineName` が 7 文字より長い場合も、マイグレーション・ツールは編集ルーチンを `validatorFunction` プロパティにマイグレーションします。これは、テーブル名が VisualAge Generator では 7 文字に制限されているからです。
- `editRoutineName` に指定されたパーツがマイグレーション時に使用可能で、このパーツがテーブルならば、`editRoutineName` は `validatorDataTable` プロパティにマイグレーションされます。また、編集メッセージが書式フィールドに対して指定されている場合も、マイグレーション・ツールは編集ルーチンを `validatorDataTable` プロパティにマイグレーションします。これは、VisualAge Generator が編集メッセージを EZEC10、EZEC11、またはテーブルのみと組み合わせるからです。
- `editRoutineName` に指定されたパーツがマイグレーション時に使用不可で、`editRoutineName` が 7 文字以下であり、編集メッセージが指定されていない場合、マイグレーション・ツールは `editRoutineName` が関数であると想定し、`validatorFunction` プロパティにマイ

グレーションします。メッセージ IWN.MIG.0502.e はこの場合にのみ出されます。

**ユーザーの処置:** 指定された編集ルーチンが関数でなければ、書式フィールドを変更し、*validatorFunction* プロパティを *validatorDataTable* プロパティに変更します。その他の考慮事項については、80 ページの『マップ変数フィールドと編集ルーチン』にある、編集ルーチンに関する説明を参照してください。

---

**IWN.MIG.0503.w** マップ・グループ *mapGroupName*、マップ *mapName* - 名前なし変数フィールドが、位置 (*row,column*) で定数フィールドに変換されました。

**説明:** VisualAge Generator は、名前なし変数フィールドがマップに存在することを許容します (ただし推奨されません)。プログラムは、これらの名前なし変数フィールドにアクセスできません。テストおよび生成時に、名前なし変数フィールドは定数に変換されます。1 つ以上のプロパティがデフォルト以外の値なので、マイグレーション・ツールはこの名前なし変数フィールドを定数に変換しました。

**ユーザーの処置:** 書式定義を検討し、定数フィールドがこのフィールドの正しいマイグレーションであることを確認してください。その他の考慮事項については、83 ページの『名前なしマップ変数フィールド』にある、名前なし変数フィールドに関する説明を参照してください。

---

**IWN.MIG.0504.w** マップ・グループ *mapGroupName*、マップ *mapName* - 名前なし変数フィールドが、位置 (*row,column*) から除去されました。

**説明:** VisualAge Generator は、名前なし変数フィールドがマップに存在することを許容します (ただし推奨されません)。プログラムは、これらの名前なし変数フィールドにアクセスできません。テストおよび生成時に、名前なし変数フィールドは定数に変換されます。すべてのプロパティが定数フィールドのデフォルト値を指定しているので、マイグレーション・ツールはこの名前なし変数フィールドを除去しました。EGL の場合、デフォルトのプロパティを使用する定数を書式に対して明示的に定義する必要はありません。

**ユーザーの処置:** 書式定義を検討し、このフィールドを除去することが正しいマイグレーションであることを確認してください。その他の考慮事項については、83 ページの『名前なしマップ変数フィールド』にある、名前なし変数フィールドに関する説明を参照してください。

---

**IWN.MIG.0505.w** マップ・グループ *mapGroupName*、マップ *mapName* - 使用している装置 *deviceName* とサイズ *depth,width* は、サポートされなくなりました。変更する必要があります。

**説明:** EGL の COBOL 生成機能は、*textForm* に対して、VisualAge Generator がサポートする装置タイプの一部をサポートしなくなりました。マイグレーション・ツールは、マイグレーションした *textForm* の *screenSizes* プロパティにオリジナルの縦の長さ and 幅を組み込みます。ただし、この画面サイズは EGL の COBOL 生成機能によってサポートされなくなりました。COBOL の生成を行う場合は、生成機能からエラー・メッセージが出されます。

**ユーザーの処置:** COBOL の生成を予定している場合は、EGL の *textForm* を編集し、この縦の長さ and 幅を *screenSizes* プロパティから除去するか、縦の長さ and 幅をサポートされるサイズに変更します。また、*textForm* を変更して、新しい縦の長さ and 幅に正しく適合するように変数と定数を再配置する必要が生じることもあります。

---

**IWN.MIG.0506.e** マップ・グループ *mapGroupName*、マップ *mapName* - *row*, *column* に無保護定数があります。**protect=skip** に変更しました。

**説明:** VisualAge Generator の場合、表示マップとプリンター・マップの両方に無保護定数を使用できます。EGL の場合、定数は **protect=skip** または **protect=no** のどちらかとして指定する必要があります。マイグレーション・ツールは、このフィールドに対して **protect=skip** を設定します。

**ユーザーの処置:** **protect=skip** が受け入れ可能ならば、アクションは必要ありません。**protect=skip** を指定すると、エンド・ユーザーは直前の変数フィールドの末尾から入力を継続でき、追加された文字は次の無保護変数フィールドに配置されます。**Protect=no** を指定すると、エンド・ユーザーは直前の変数フィールドの末尾から入力を継続できません。エンド・ユーザーが入力を継続するには、次の変数フィールドまでタブ・キーで移動する必要があります。

---

**IWN.MIG.0507.w** マップ・グループ *mapGroupName*、マップ *mapName* - *row=0*, *column=0* にある定数が、*row=1*, *column=1* に変更されました。

**説明:** VisualAge Generator は、マップ上で位置 *row=0*, *column=0* にある定数を許容します (ただし完全にはサポートしません)。*row=0*, *column=0* にあるフィールド



は、属性情報を指定できません。 EGL は、row=0, column=0 にあるフィールドをサポートしません。 row=0, column=0 にあるフィールドは定数で、第 1 バイトはブランクに初期化されます。マイグレーション・ツールは、位置を row=1, column=1 に変更し、定数値の第 1 バイトを削除します。マイグレーション・ツールは、フィールドの色や強調表示などのフィールド表示プロパティを組み込みません。これは、この情報が外部ソース形式ファイルに記録されていないからです。

**ユーザーの処置:** この書式を使用するプログラムをテストして、表示の外観に変化があるかどうか判別してください。変化がある場合は、書式を編集して、望ましい外観が得られるようにフィールド表示プロパティを設定します。

---

**IWN.MIG.0508.e** マップ・グループ *mapGroupName*、マップ *mapName* - row=0, column=0 にある定数を変更できません。

**説明:** VisualAge Generator は、マップ上で位置 row=0, column=0 にある定数を許容します (ただし完全にはサポートしません)。row=0, column=0 にあるフィールドは、属性情報を指定できません。 EGL は、row=0, column=0 にあるフィールドをサポートしません。 row=0, column=0 にあるフィールドは定数で、第 1 バイトはブランクに初期化されません。フィールドの位置を変更すると、定数が移動したり書式から除去されたりして外観が変化するので、マイグレーション・ツールはフィールドの位置を変更しません。マイグレーション・ツールは、フィールドの色や強調表示などのフィールド表示プロパティを組み込みません。これは、この情報が外部ソース形式ファイルに記録されていないからです。「問題」ビューにエラーが表示されます。

**ユーザーの処置:** 書式を編集し、定数フィールドを変更して、フィールドを row=1, column = 1 に配置します。属性バイトが row=1, column=1 を占めるようになるので、必要に応じて、その補正のために定数フィールドを変更して 1 バイトを除去します。必ず、この書式を使用するプログラムをテストして、表示の外観に変化があるかどうか判別してください。変化がある場合は、書式を編集して、望ましい外観が得られるようにフィールド表示プロパティを設定します。

---

**IWN.MIG.0509.e** マップ・グループ *mapGroupName*、マップ *mapName* - row=0, column=0 にある変数を変更できません。

**説明:** このマップは、システム共通プロダクトまたは VisualAge Generator の古いバージョンで作成された可能性があります。VisualAge Generator 4.5 は、row=0, column=0 にある変数をサポートしません。row=0, column=0 にあるフィールドは、属性情報を指定できま

せん。 EGL は、row=0, column=0 にあるフィールドをサポートしません。row=0, column=0 にあるフィールドは、変数フィールドです。フィールドの位置を変更すると、フィールドが移動したり、データの第 1 バイトが失われたりするので、マイグレーション・ツールはフィールドの位置を変更しません。マイグレーション・ツールは、フィールドの色や強調表示などの表示プロパティを組み込みません。これは、この情報が外部ソース形式ファイルに記録されていないからです。「問題」ビューにエラーが表示されます。

**ユーザーの処置:** 書式を編集し、フィールドを変更して、フィールドを row=1, column = 1 に配置します。必要に応じて、row=1, column=1 に移動した属性バイトが原因でデータが失われないように、変数フィールドの近くにある他のフィールドを変更してください。必ず、この書式を使用するプログラムをテストして、表示の外観に変化があるかどうか判別してください。変化がある場合は、書式を編集して、望ましい外観が得られるようにフィールド表示プロパティを設定します。

---

**IWN.MIG.0510.e** マップ・グループ *mapGroupName*、マップ *mapName* - *mapName* がプログラム名と競合します。

**説明:** プログラムが、そのプログラムと同名のマップを含むマップ・グループまたはヘルプ・マップ・グループを使用しています。 VisualAge Generator は、マップ名がプログラム名と同じであることを許容します。 EGL は、書式名がプログラム名と同じであることを許容しません。マップ名がプログラム名と同じであり、マップに変数フィールドがない場合、マイグレーション・ツールはプログラムのヘルプ・マップ・グループ内のマップ名を変更します。ただし、次の場合、マイグレーション・ツールはマップの名前を変更しません。

- マップが、プログラムのヘルプ・マップ・グループ内にある変数フィールドを含むマップである。
- マップが、プログラムのメインのマップ・グループ内にあるいずれかのマップである。

**ユーザーの処置:** formGroup を編集し、書式の名前を変更します。また、この formGroup を使用するすべてのプログラム内で、書式定義とこの書式の参照すべてを必ず変更します。その他の考慮事項については、77 ページの『マップ名とヘルプ・マップ名』にある、マップ名に関する説明を参照してください。

---

**IWN.MIG.0601.w** 関数 *functionName*、入出力オブジェクト *recordName* - UPDATE オプションのレコード・タイプを判別できません。非 SQL レコードが想定されます。

**説明:** SQL の場合、プログラム内に複数の UPDATE

ステートメントまたは SETUPD ステートメントがあれば、VisualAge Generator の REPLACE 関数に、対応する UPDATE ステートメントまたは SETUPD ステートメントの名前を指定する必要があります。EGL は、SQL ステートメントの resultSetID を使用して、replace ステートメントと、それに対応する get ステートメントまたは open ステートメントの関係を指定します。

recordName によって示されたレコードが、マイグレーション時に使用不可でした。マイグレーション・ツールは、UPDATE 関数が非 SQL レコードを対象にしていると想定し、resultSetID を組み込みません。

**ユーザーの処置:** プログラム内の同じレコードに関して複数の get ステートメントまたは open ステートメントが存在するために、検証または生成時にエラーのフラグが立てられた場合は、関数を編集して、get forUpdate ステートメントに resultSetID を追加します。resultSetID は、プログラム内で固有であることが必要です。推奨される resultSetID は、関数名の後に、マイグレーション時に使用した結果セット接尾部の設定値を付けたものです。その他の考慮事項については、100 ページの『複数の更新を行う SQL 入出力』を参照してください。

---

**IWN.MIG.0602.w 関数 *functionName* - 入出力オブジェクト *mapName*; のマップ・タイプを判別できません。表示マップが想定されます。**

**説明:** VisualAge Generator は、表示マップとプリンター・マップの両方に DISPLAY 入出力オプションを使用します。EGL は、display ステートメントをテキスト書式のみに対して使用し、print ステートメントを印刷書式に対して使用します。VisualAge Generator 互換モードでは、display ステートメントを印刷書式に対して使用することもできます。*mapName* として指定されたマップが、マイグレーション時に使用不可でした。マイグレーション・ツールは、マップが表示マップであると想定し、EGL の display ステートメントにマイグレーションします。

**ユーザーの処置:** VisualAge Generator 互換モードを使用し続けるか、またはマップが表示マップであれば、アクションは必要ありません。マップが印刷マップであり、VisualAge Generator 互換モードの使用を中止する場合は、print ステートメントを使用するように関数を変更する必要があります。その他の考慮事項については、92 ページの『マップの DISPLAY ステートメント』を参照してください。

---

**IWN.MIG.0603.e 関数 *functionName*、SQL 入出力オブジェクト *recordName* - SQL 表名を判別できません。**

**説明:** VisualAge Generator は、テストおよび生成時に SQL レコードから SQL 表名を判別します。EGL の場

合は、変更されたすべての SQL ステートメントに表名が組み込まれている必要があります。*recordName* によって示されたレコードが、マイグレーション時に使用不可でした。マイグレーション・ツールは、検証および生成時にエラーのフラグが立てられるように、表名の代わりに EZE\_UNKNOWN\_SQLTABLE を使用します。さらにマイグレーション・ツールは、ステートメントの表ラベルを TI に設定します。

**ユーザーの処置:** 関数を編集し、レコード定義に基づいて正しい表名と表ラベルを指定します。表名は、EGL レコード定義の *tableNames* および *tableNameVariables* のどちらか、または両方のプロパティにあります。その他の考慮事項については、427 ページの『付録 D.

「問題」ビューのメッセージ』にある、EZE\_UNKNOWN\_TABLE に関する説明を参照してください。

---

**IWN.MIG.0604.e 関数 *functionName*、SQL 入出力オブジェクト *recordName* - !itemColumnName 変数の列名を判別できません。**

**説明:** 変更された SQL ステートメントが、1 つ以上の VAGen !itemColumnName 変数を使用していました。

VAGen !itemColumnName 変数は、実際の SQL 列名に対応する SQL レコード定義内の項目名を指定します。

VisualAge Generator は、テストおよび生成時に、!itemColumnName 変数の実際の SQL 列名を SQL レコードから判別します。EGL は、!itemColumnName 変数をサポートしません。代わりに、EGL の場合は、変更された SQL ステートメントの中で実際の SQL 列名を使用する必要があります。*recordName* によって示されたレコードまたは代替仕様のレコードが、マイグレーション時に使用不可でした。マイグレーション・ツールは、変更された SQL ステートメント内で !itemColumnNames を使用して、可能なかぎり多くの情報を提供します。

**ユーザーの処置:** 関数を編集し、レコード定義に基づいて SQL 列名を指定します。それぞれの !itemColumnName ごとに、対応する項目を SQL レコード定義の中で見付けます。その項目の列名が、EGL の入出力ステートメント内で使用する必要がある列名です。その他の考慮事項については、99 ページの『SQL 入出力と !itemColumnName』を参照してください。

---

**IWN.MIG.0605.w 関数 *functionName*、SQL 入出力オブジェクト *recordName* - SQLEXEC は model=None を指定し、SQL 文節を指定していません。**

**説明:** SQLEXEC ステートメントはモデル型として None を指定していますが、SQL 文節を含んでいません。実際には、VisualAge Generator はこのステートメン

トに対してノーオペレーションを生成します。マイグレーション・ツールは、EGL のノーオペレーション・ステートメント (セミコロンのみ) を生成し、モデル型が none であったことを示す VAGen の情報コメントを組み込みます。VAGen 関数がエラー・ルーチンを指定している場合、マイグレーション・ツールは、そのエラー・ルーチンに適した *try*、*onException*、および *end* の各ステートメントを組み込みます。

**ユーザーの処置:** 関数を検討して、入出力ステートメントを除去するか、拡張するかを判断してください。

---

**IWN.MIG.0607.e 関数 *functionName*、SQL 入出力オブジェクト *recordName* - SQL 入出力文節 *clauseName* を判別できません。**

**説明:** ある時期の VisualAge Generator では、変更された SQL 文節のみが関数とともに保管されていました。この場合、残りの文節は、関数の入出力オブジェクトとして指定されたレコード定義から VisualAge Generator によって作成されます。指定された *recordName* が、マイグレーション時に使用不可でした。マイグレーション・ツールは、SQL 文節を作成できません。このメッセージにリストされる可能性がある *clauseNames* は、SELECT、INTO、INSERTCOLNAME、VALUES、および FORUPDATEOF です。マイグレーション・ツールは、スケルトン文節を作成し、EZE\_UNKNOWN\_SQL\_CLAUSENAME を組み込みます。

**ユーザーの処置:** メッセージに示されているレコードを見付けます。関数を編集して、欠落している SQL 文節を組み込みます。欠落している SQL 文節をどのように指定する必要があるか判別するには、VAGen SQL ステートメント・エディターを使用して SQL 文節を表示します。詳細と起こりうる問題については、97 ページの『SQL 入出力と必須 SQL 文節の欠落』を参照してください。427 ページの『付録 D. 「問題」ビューのメッセージ』にある、EZE\_UNKNOWN\_SQL\_CLAUSENAME に関する説明を参照してください。

---

**IWN.MIG.0608.e 関数 *functionName*、SQL 入出力オブジェクト *recordName* - 代替仕様 *altspecRecordName* の SQL 入出力文節 *clauseName* を判別できません。**

**説明:** ある時期の VisualAge Generator では、変更された SQL 文節のみが関数とともに保管されていました。この場合、残りの文節は、関数の入出力オブジェクトとして指定されたレコード定義から VisualAge Generator によって作成されます。指定された *recordName* は、マイグレーション時に使用可能です。ただし、*recordName* は代替仕様レコード *altspecRecordName* を指定しており、このレコードがマイグレーション時に使用不可でし

た。マイグレーション・ツールは、SQL 文節を作成できません。このメッセージにリストされる可能性がある *clauseNames* は、SELECT、INTO、INSERTCOLNAME、VALUES、および FORUPDATEOF です。マイグレーション・ツールは、スケルトン文節を作成し、EZE\_UNKNOWN\_SQL\_CLAUSENAME を組み込みます。

**ユーザーの処置:** メッセージに示されている代替仕様レコードを見付けます。関数を編集して、欠落している SQL 文節を組み込みます。欠落している SQL 文節をどのように指定する必要があるか判別するには、VAGen SQL ステートメント・エディターを使用して SQL 文節を表示します。詳細と起こりうる問題については、97 ページの『SQL 入出力と必須 SQL 文節の欠落』を参照してください。詳細と起こりうる問題については、97 ページの『SQL 入出力と必須 SQL 文節の欠落』を参照してください。427 ページの『付録 D. 「問題」ビューのメッセージ』にある、EZE\_UNKNOWN\_SQL\_CLAUSENAME に関する説明を参照してください。

---

**IWN.MIG.0609.e 関数 *functionName* - SSA 内のレコード *recordName* は使用できません。比較値項目 *itemName* の修飾を判別できません。**

**説明:** 変更された DL/I ステートメントは非修飾の比較値項目を使用していました。デフォルトで、VisualAge Generator は最初に現行の SSA と関連する DL/I セグメント・レコード内の項目を検索します。現行の SSA に関連する指定された DL/I セグメント・レコード *recordName* は利用できません。このため、マイグレーション・ツールは、比較値項目の修飾を判別できませんでした。

**ユーザーの処置:** メッセージに示されているレコードを見付け、検討します。この項目がレコード内にある場合、関数を編集し、欠落した比較値項目の修飾を組み込むようにします。項目がレコード内がない場合、使用する正しい修飾を判別するためにプログラム・ロジックを検討します。さらに、プログラムを生成した最後の時刻から、生成済み COBOL のソース・コードを検討できます。VisualAge Generator では、ある時点で比較値項目の修飾規則が変わりました。従って、比較値項目の修飾が変動しているため、プログラムを生成した最後の時刻以来リリースが変更されていないことが明らかでない限り、VisualAge Generator の現行リリースを使用するプログラムは再生成しないでください。



---

**IWN.MIG.0610.e** 関数 *functionName* - SSA のレコード *recordName* には、使用できない代替仕様レコード *altspecRecordName* がありません。比較値項目 *itemName* の修飾を判別できません。

**説明:** 変更された DL/I ステートメントは非修飾の比較値項目を使用していました。デフォルトで、VisualAge Generator は最初に現行の SSA と関連する DL/I セグメント・レコードの項目を検索します。現行の SSA に関連する指定された DL/I セグメント・レコード *recordName* は、マイグレーション中に使用できます。ただし、*recordName* は、マイグレーション時に使用できない代替仕様レコード *altspecName* を指定しています。このため、マイグレーション・ツールは、比較値項目の修飾を判別できませんでした。

**ユーザーの処置:** メッセージに示されているレコードを見付け、検討します。項目が代替仕様レコード内にある場合、関数を編集し、欠落した比較値項目の修飾を組み込むようにします。項目がレコード内にない場合、使用する正しい修飾を判別するためにプログラム・ロジックを検討します。さらに、プログラムを生成した最後の時刻から、生成済み COBOL のソース・コードを検討できます。VisualAge Generator では、ある時点で比較値項目の修飾規則が変わりました。従って、比較値項目の修飾が変動しているため、プログラムを生成した最後の時刻以来リリースが変更されていないことが明らかでない限り、VisualAge Generator の現行リリースを使用するプログラムは再生成しないでください。

---

**IWN.MIG.0611.e** 関数 *functionName* - 比較値項目 *itemName* が、レコード *recordName* ありません。修飾を判別できません。

**説明:** 変更された DL/I ステートメントは非修飾の比較値項目を使用していました。デフォルトで、VisualAge Generator は最初に現行の SSA と関連する DL/I セグメント・レコードの項目を検索します。マイグレーション・ツールは、現行の SSA に関連した DL/I セグメント・レコードを検索しましたが、項目を検出できませんでした。VisualAge Generator では、ある時点で比較値項目の修飾規則が変わりました。マイグレーション・ツールは、比較を修飾するためにどのレコードを使用すべきかを決定できません。

**ユーザーの処置:** 使用する正しい修飾を判別するためにプログラム・ロジックを検討します。さらに、プログラムを生成した最後の時刻から、生成済み COBOL のソース・コードを検討できます。比較値項目の修飾が変動しているため、プログラムを生成した最後の時刻以来リリースが変更されていないことが明らかでない限り、VisualAge Generator の現行リリースを使用するプログラ

ムを生成しないでください。

---

**IWN.MIG.0612.e** 関数 *functionName* - SSA に関して無効な関係演算子です。正しい演算子を判別できません。

**説明:** 変更済み DL/I は、無効な関係演算子を使用していました。これは、不正な比較演算子の値が保管されるという、VisualAge Generator 内の問題によって発生します。マイグレーション・ツールは正しい関係演算子を判別できません。マイグレーション・ツールは、関係演算子として EZE\_UNKNOWN\_RELOP を使用します。

**ユーザーの処置:** VisualAge Generator で DL/I 呼び出しエディターを使用し、指定済み関数に関する SSA を検討します。演算子が関数の外部形式ファイルに誤って保管されていても、正しい演算子が DL/I Call Editor に表示されます。EGL で関数を編集し、EZE\_UNKNOWN\_RELOP を正しい値に変更します。

**注:** 問題を引き起こす可能性の最も高い演算子は、不等記号です。EGL SSA の不等記号は、!= です。

---

**IWN.MIG.0701.e** 関数 *functionName* - SET map *PAGE* ステートメントで使用されている *mapName* のマップ・タイプを判別できません。converseLib.EZE\_SETPAGE(); が使用されました。

**説明:** VisualAge Generator は、SET map *PAGE* を使用して表示マップの場合に画面を消去することや、印刷マップの場合にページ替えを行うことを示します。EGL は、converseLib.clearScreen() ステートメントをテキスト書式のみに対して使用し、converseLib.pageEject ステートメントを印刷書式に対して使用します。*mapName* として指定されたマップが、マイグレーション時に使用不可でした。マイグレーション・ツールは、マップ・タイプに関する推測は行いません。代わりにマイグレーション・ツールは、検証および生成時にエラーのフラグが立てられるように、converseLib.EZE\_SETPAGE() ステートメントを使用します。マイグレーション・ツールは、オリジナルのマップ名をコメントとして組み込みます。

**ユーザーの処置:** 関数を検討し、clearScreen() または pageEject() のどちらが正しい選択か判別します。その他の考慮事項については、106 ページの『SET map *PAGE* ステートメント』を参照してください。

---

**IWN.MIG.0702.e** 関数 *functionName* - テーブル *tableName* が欠落しているために、RETR ステートメントの戻す列名を判別できません。

**説明:** 戻す列が RETR ステートメントに指定されてい

ない場合は、指定されたテーブルの 2 列目に基づいて、VisualAge Generator が戻す列名を自動的に判別します。RETR に対する EGL の置換表現は、*if* ステートメントと、その後に続く代入ステートメントです。戻す列名は、代入ステートメントの中で明示的に指定されている必要があります。*tableName* によって示されたテーブルが、マイグレーション時に使用不可でした。マイグレーション・ツールは、検証および生成時にエラーのフラグを立てられるように、

EZE\_UNKNOWN\_RETURN\_COLUMN を使用します。この問題がプログラムのフロー・ステートメント内で発生した場合は、関数名の代わりにプログラム名がメッセージに出力されます。

**ユーザーの処置:** 関数を編集し、戻す列をテーブル定義に基づいて正しく指定します。テーブルの 2 列目が、VisualAge Generator 内で使用されるデフォルトの戻す列です。その他の考慮事項については、105 ページの『RETR ステートメント』を参照してください。

---

**IWN.MIG.0703.e 関数 *functionName* - テーブル *tableName* が欠落しているために、RETR ステートメントの検索列名を判別できません。**

**説明:** 検索列が RETR ステートメントに指定されていない場合は、指定されたテーブルの 1 列目に基づいて、VisualAge Generator が検索列名を自動的に判別します。RETR に対する EGL の置換表現は、*if* ステートメントと、その後に続く代入ステートメントです。検索列名は、*if* ステートメントの中で明示的に指定されている必要があります。*tableName* によって示されたテーブルが、マイグレーション時に使用不可でした。マイグレーション・ツールは、検証および生成時にエラーのフラグを立てられるように、

EZE\_UNKNOWN\_SEARCH\_COLUMN を使用します。この問題がプログラムのフロー・ステートメント内で発生した場合は、関数名の代わりにプログラム名がメッセージに出力されます。

**ユーザーの処置:** 関数を編集し、検索列をテーブル定義に基づいて正しく指定します。テーブルの 1 列目が、VisualAge Generator 内で使用されるデフォルトの検索列です。その他の考慮事項については、105 ページの『RETR ステートメント』を参照してください。

---

**IWN.MIG.0704.e 関数 *functionName* - テーブル *tableName* が欠落しているために、FIND ステートメントの検索列名を判別できません。**

**説明:** 検索列が FIND ステートメントに指定されていない場合は、指定されたテーブルの 1 列目に基づいて、VisualAge Generator が検索列名を自動的に判別しま

す。FIND に対する EGL の置換表現は、*if* ステートメントと、その後に続く関数呼び出しステートメントです。検索列名は、*if* ステートメントの中で明示的に指定されている必要があります。*tableName* によって示されたテーブルが、マイグレーション時に使用不可でした。マイグレーション・ツールは、検証および生成時にエラーのフラグを立てられるように、

EZE\_UNKNOWN\_SEARCH\_COLUMN を使用します。この問題がプログラムのフロー・ステートメント内で発生した場合は、関数名の代わりにプログラム名がメッセージに出力されます。

**ユーザーの処置:** 関数を編集し、検索列をテーブル定義に基づいて正しく指定します。テーブルの 1 列目が、VisualAge Generator 内で使用されるデフォルトの検索列です。その他の考慮事項については、104 ページの『FIND ステートメント』を参照してください。

---

**IWN.MIG.0706.e 関数 *functionName* - IF、WHILE、または TEST DUP ステートメント内で使用されている *recordName* のレコード・タイプを判別できません。  
EZE\_DUPLICATE を使用しました。**

**説明:** VisualAge Generator は、非 SQL レコードと SQL レコードの両方に対して、DUP と UNQ の両方の検査をサポートします。SQL レコードの場合、DUP と UNQ は同一です。非 SQL レコードの場合、EGL は、duplicate と unique の両方をサポートします。SQL レコードの場合、EGL は unique のみをサポートします。*recordName* によって示されたレコードが、マイグレーション時に使用不可でした。マイグレーション・ツールは、検証および生成時にエラーのフラグを立てられるように、DUP を EZE\_DUPLICATE にマイグレーションします。

**注:** マイグレーション・ツールは、TEST ステートメントを *if* ステートメントにマイグレーションします。

**ユーザーの処置:** 関数を編集し、EZE\_DUPLICATE を次のいずれかに変更します。

- SQL レコードの場合は *unique*
- 非 SQL レコードの場合は *duplicate*

その他の考慮事項については、110 ページの『入出力エラー値 UNQ および DUP』を参照してください。

---

**IWN.MIG.0707.e** 関数 *functionName* - IF、WHILE、または TEST NULL ステートメント内で項目 *itemName* が使用されるときに、この項目がレコードまたはマップのどちらにあるか判別できません。EZE\_NULL を使用しました。

**説明:** VisualAge Generator は、マップ項目と SQL 項目の両方に対して、NULL の検査をサポートします。マップ項目の NULL の検査は、ブランクの検査と同等です。SQL 項目の NULL の検査を行うと、NULL 標識変数が検査され、データベース内で列が NULL かどうか判別されます。同等な EGL ステートメントは、書式フィールドがブランクであるかどうか、および SQL フィールドが NULL であるかどうかを検査します。 *itemName* に示された項目が、マイグレーション時に使用不可でした。マイグレーション・ツールは、検証および生成時にエラーのフラグが立てられるように、NULL を EZE\_NULL にマイグレーションします。

**注:** マイグレーション・ツールは、TEST ステートメントを *if* ステートメントにマイグレーションします。

**ユーザーの処置:** 関数を編集し、EZE\_NULL を次のいずれかに変更します。

- 書式フィールドの場合は *blanks*
- SQL フィールドの場合は *null*

その他の考慮事項については、108 ページの『SQL 項目とマップ項目が NULL かどうかの検査』を参照してください。

---

**IWN.MIG.0708.w** 関数 *functionName* - IF、WHILE、または TEST 以外のステートメント内で EZESYS を使用しています。以前の VAGen 値が使用されます。

**説明:** VisualAge Generator は、IF、WHILE、および TEST 以外のステートメント内で EZESYS の使用をサポートします。マイグレーション・ツールは、ステートメント・タイプに基づいて EZESYS をマイグレーションします。IF、WHILE、および TEST の各ステートメントの場合、マイグレーション・ツールは EZESYS を *sysVar.systemType* に変換し、さらに値を新しい EGL 値に変換します。IF、WHILE、TEST 以外のステートメントの場合は、マイグレーション・ツールは *custPrefixEZESYS* に変換します。ここで *custPrefix* はマイグレーション用に設定した「名前変更のための接頭部」設定です。プログラムをマイグレーションするとき、VAGen マイグレーション設定「以前の EZESYS 値を初期化しない (*Do not initialize old EZESYS values*)」が選択されていないと、マイグレーション・ツールは、

*custPrefixEZESYS* の宣言および *custPrefixEZESYS* をオリジナルの VAGen 値に初期化するステートメントを組み込みます。オリジナルの VAGen 値がこのステートメントに使用されます。

**ユーザーの処置:** 関数を検討し、オリジナルの VAGen 値を使用するか、新しい EGL 値を使用するかを判断してください。新しい EGL 値を使用する場合は、*custPrefixEZESYS* を *sysVar.systemType* に変更してください。

オリジナルの VAGen 値を使用する予定で、マイグレーション中に、VAGen マイグレーション設定「以前の EZESYS を初期化しない (*Do not initialize old EZESYS values*)」を選択した場合は、指定された関数を使用するすべてのプログラムに *custPrefixEZESYS* の宣言および初期化ステートメントを追加する必要があります。オリジナルの VAGen 値を使用する予定で、VAGen マイグレーション設定「以前の EZESYS を初期化しない (*Do not initialize old EZESYS values*)」を選択しなかった場合は、変更は必要ありません。*custPrefixEZESYS* の宣言および初期化ステートメントは、既にすべてのマイグレーション済みプログラムに組み込まれています。

---

**IWN.MIG.0801.e** プログラム名 *programName* は予約語です。名前を変更する必要があります。

**説明:** マイグレーション・ツールは、プログラムの名前を自動的に変更しません。

**ユーザーの処置:** プログラムの名前と、それに対するすべての参照 (*call*、*transfer*、および *show* の各ステートメントでの参照、およびリンクエッジ・オプションのパーツでの参照など) を変更する必要があります。また、このプログラムに対応するバインド制御パーツ、またはリンク・エディット・パーツの名前も変更します。生成されるプログラムの名前としてオリジナルのプログラム名を保持する必要がある場合は、*alias* プロパティを指定できます。*alias* プロパティを指定しない場合は、CICS プログラム定義など、EGL 以外によるプログラム名の参照すべてを必ず変更してください。

---

**IWN.MIG.0802.w** プログラム *programName* — 暗黙項目を許容しています。暗黙項目の定義はマイグレーションによって作成されません。

**説明:** VisualAge Generator のプログラムは、暗黙データ項目の許容を指定できます。暗黙データ項目を許容するプログラムが、未定義の項目を実際に使用すると、テストおよび生成時に VisualAge Generator が定義を自動的に作成します。EGL は、暗黙項目を許容しません。マイグレーション・ツールは、暗黙定義を自動的に作成しません。

**ユーザーの処置:** VisualAge Generator 内でプログラム



を検証して、暗黙項目が使用されているかどうか判別してください。使用されている場合、VisualAge Generator は暗黙項目の定義を検証メッセージに示します。EGL でプログラム定義を編集し、対応する EGL プリミティブ・フィールド宣言を追加します。フィールドを格納するためのレコードを作成する必要はありません。プリミティブ・フィールド宣言は、プログラムに直接追加できます。

---

**IWN.MIG.0804.w プログラム *programName* - CLOSE**  
入出力オプションに対して使用される入出力オブジェクト *partName* のパーツ型を判別できません。レコードが想定されます。

**説明:** VisualAge Generator では、テストまたは生成時に入出力オブジェクトが自動的に組み込まれます。CLOSE 入出力オプションは、レコードと印刷マップの両方に使用できます。EGL では、入出力ステートメント内で使用されるレコードは、プログラム内で明示的に宣言されている必要があります。書式は明示的に宣言されませんが、formGroup の使用 宣言が必要です。指定されたプログラム内で CLOSE 入出力オプションが使用されており、指定された *partName* が CLOSE の入出力オブジェクトとして使用されています。ただし、指定された *partName* がマイグレーション時に使用不可でした。マイグレーション・ツールは、パーツがレコードであると想定し、データ宣言を組み込みます。

**ユーザーの処置:** マイグレーション・ツールの予測が誤っている場合は、「問題ビュー」にエラーが示されます。プログラムを編集し、printForm の宣言を除去します。

---

**IWN.MIG.0805.w プログラム *programName* - 実行モードが指定されていません。非セグメント化が想定されます。**

**説明:** ある時期の VisualAge Generator では、実行モードがプログラム・パーツとともに保管されていませんでした。実行モードは、メインのトランザクション・プログラムのみに適用されます。指定された *programName* はメインのトランザクション・プログラムですが、実行モードが外部ソース形式に含まれていません。マイグレーション・ツールは、実行モードが非セグメント化であると想定し、segmented=no プロパティを EGL ソースに組み込みます。

**ユーザーの処置:** プログラムを非セグメント化モードで実行する場合は、アクションは必要ありません。プログラムをセグメント化モードで実行する必要がある場合は、プログラムを編集し、segmented プロパティを segmented=yes に変更します。

---

**IWN.MIG.0806.w プログラム *programName*、  
*tableName* の使用宣言 - 設定が原因で  
deleteAfterUse=yes が省略されます。**

**説明:** 指定されたプログラムに、指定されたテーブルの使用宣言が含まれています。システム共通プロダクトおよび VisualAge Generator のいくつかのリリースでは、keep after use フラグによって、テーブルのメモリーがプログラムによって解放される時期が決定されていません。VisualAge Generator の keep after use フラグは、通常、テーブルの使用宣言の EGL deleteAfterUse プロパティにマイグレーションされます。しかし、ご使用の VAGen マイグレーション設定は、マイグレーション・ツールが、deleteAfterUse プロパティを組み込まないことを指定しています。

**ユーザーの処置:** なし。ご使用の VAGen プログラムが、VisualAge Generator 4.5 フィックスバック 4 を使用して生成されている場合は、振る舞いに差はありません。詳しくは、216 ページの『VisualAge Generator 互換モードの使用を中止する場合』にある deleteAfterUse に関する情報を参照してください。

---

**IWN.MIG.0807.e プログラム *programName* - PSB  
*psbName* は使用できません。PSB の DLI  
セグメント・レコードが判別できません。**

**説明:** VisualAge Generator では、プログラムの PSB で指定された DLI セグメント・レコードはテスト時、または生成時にすべて自動的に組み込まれます。これは、DLI セグメント・レコードがデフォルトの SSA を作成する時に使用されるか、または変更された SSA で明示的に使用される可能性があるからです。EGL では、DLI セグメント・レコードは、デフォルトの SSA 使用されるか、または変更された SSA で明示的に使用される場合、プログラムで明示的に宣言される必要があります。指定された *psbName* は、マイグレーション時に使用できません。従って、マイグレーション・ツールは、なんらかの DLI セグメント・レコードがプログラムのデータ宣言のリストに追加される必要があるかどうかを判別することができません。

**ユーザーの処置:** この関数またはプログラムの未確定または未解決の DLI セグメント・レコードに関するメッセージがないかどうか、EGL 「問題」リストを調べます。メッセージがない場合、DLI セグメント・レコードのための宣言が既にプログラムに (最も高い可能性として DLI セグメント・レコードに対する I/O ディレクトリーの結果として) 組み込まれています。メッセージがある場合、プログラムを編集して、DLI セグメント・レコードのための宣言を追加します。

---

**IWN.MIG.0808.e** 呼び出し先プログラム *programName*  
- PSB *psbName* は使用できません。PCB  
タイプが判別できません。

**説明:** VisualAge Generator では、パラメーターとしてプログラムに引き渡される PCB を示すために、EZEDLPCB[n] (*n* は数値リテラル) が使用されます。PCB が I/O、代替、データベース、または GSAM PCB であるかどうかに関係なく、VisualAge Generator は PCB を使用します。EZEDLPCB[0] は常に I/O PCB であり、VAGen PSB パーツには明示的にリストされません。PCB が実行時に使用されるときに、DL/I PCB が正しいタイプでない場合にはエラーが発生します。EGL では、プログラムの PSB からの PCB 名がパラメーター名として使用されます。PCB タイプは、レコード名 (IO\_PCBRecord、ALT\_PCBRecord、DB\_PCBRecord、または GSAM\_PCBRecord) を使用して適切な型定義を与えて、各プログラム・パラメーターごとに明示的に指定しなければなりません。マイグレーション・ツールは EZEDLPCB[0] の型定義として常に IO\_PCBRecord を使用します。プログラム *programName* は、*n* が 0 より大きい 1 つまたは複数の EZEDLPCB[n] パラメーターを指定します。ただし、指定された *psbName* は、マイグレーション時に使用できません。したがって、マイグレーション・ツールはパラメーター・リストの PCB 用に組み込むタイプを判別することができません。マイグレーション・ツールは、パラメーター・リストのすべての PCB 用に EZE\_UNKNOWN\_PCB\_TYPE を使用します。

**ユーザーの処置:** 指定された PSB の位置指定をします。プログラムを編集して型定義を変更し、指定された EGL PSB パーツで、対応する PCB タイプに基づく正しい xxxx\_PCBRecord を指定します。

---

**IWN.MIG.0809.e** 呼び出し先プログラム *programName*  
- PSB *psbName* は使用できません。PCB  
マッピングが判別できません。

**説明:** VisualAge Generator では、パラメーターとしてプログラムに引き渡される PCB を示すために、EZEDLPCB[n] (*n* は数値リテラル) が使用されます。VisualAge Generator は VAGen PSB パーツの対応する PCB に EZEDLPCB[n] を自動的に関連付けます。EZEDLPCB[0] は常に I/O PCB であり、VAGen PSB パーツには明示的にリストされません。DL/I PSB が実行時に予想通りの数の PCB を持っていないと、エラーが発生します。EGL では、プログラムの PSB からの PCB 名はパラメーター名として使用されます。EGL PSB パーツ内の対応する位置とパラメーター・リスト内の各 PCB を明示的に関連させるために、*pcbParms* プロパティが使用されます。プログラム *programName*

は、1 つまたは複数の EZEDLPCB[n] パラメーターを指定しますが、指定された *psbName* はマイグレーション中は使用できません。したがって、マイグレーション・ツールは *pcbParms* プロパティに組み込む PCB の数を判別できません。マイグレーション・ツールは、*pcbParms* プロパティの値として EZE\_UNKNOWN\_PCB\_MAPPING を使用します。

**ユーザーの処置:** 指定された PSB の位置指定をします。プログラムを編集して、*pcbParms* プロパティを変更し、PCB パラメーターと指定された EGL PSB パーツの PCB との間のマッピングを提供します。

---

**IWN.MIG.0810.e** 呼び出し先プログラム *programName*  
- パラメーター・リストが、PSB  
*psbName* に存在する PCB 数値よりも大  
きな数値を参照しています。PCB タイ  
プ、および PCB マッピングが完了しま  
せん。

**説明:** VisualAge Generator では、パラメーターとしてプログラムに引き渡される PCB を示すために、EZEDLPCB[n] (*n* は数値リテラル) が使用されます。VisualAge Generator は VAGen PSB パーツの対応する PCB に EZEDLPCB[n] を自動的に関連付けます。関連付けは、PCB が I/O、代替、データベース、または GSAM PCB であるかどうかに関係なく行われます。EZEDLPCB[0] は常に I/O PCB であり、VAGen PSB パーツには明示的にリストされません。DL/I PSB が予想通りの数の PCB を持っていないか、または DL/I PCB が正しいタイプでない場合、ランタイム・エラーが発生します。EGL では、プログラムの PSB からの PCB 名はパラメーター名として使用されます。PCB タイプは、レコード名 (IO\_PCBRecord、ALT\_PCBRecord、DB\_PCBRecord、または GSAM\_PCBRecord) を使用して適切な型定義を与えて、各プログラム・パラメーターごとに明示的に指定しなければなりません。マイグレーション・ツールは EZEDLPCB[0] の型定義として常に IO\_PCBRecord を使用します。さらに、EGL PSB パーツ内の対応する位置とパラメーター・リストにある各 PCB を明示的に関連させるために、*pcbParms* プロパティが使用されます。プログラム *programName* は、1 つまたは複数の EZEDLPCB[n] パラメーターを指定していますが、*n* の値のいくつかは、指定された *psbName* の PCB の数よりも大きくなっています。したがって、マイグレーション・ツールはパラメーター・リスト内のいくつかの PCB をインクルードする型定義を判別できません。マイグレーション・ツールは、指定された *psbName* にある PCB に対応しないパラメーター・リスト内の PCB については、EZE\_UNKNOWN\_PCB\_TYPE を使用します。さらに、マイグレーション・ツールは *pcbParms* プロパティに組み込む PCB の数を判別できません。マ

イグレーション・ツールは、`n` の最も高い値までで、その値を含む、すべての `EZEDLPCB[n]` パラメーターのための PCB マッピング情報を作成します。しかし、このリストは指定済み `psbName` の使用可能な PCB に一致しません。

**ユーザーの処置:** 指定された PSB の位置指定をします。どちらが正しいか判断するために、PSB とプログラム・ロジックを検討します。追加の PCB を `PSBRecord` に追加します。プログラムを編集して、パラメーターの型定義を変更し、指定された EGL PSB パーツ内の対応する PCB タイプに基づいて正しい `xxxx_PCBRecord` を指定します。同様に、`pcbParms` プロパティーを変更し、PCB パラメーターと指定された EGL PSB パーツ内の PCB との間の正しいマッピングを提供します。

---

**IWN.MIG.1001.e 生成オプション・パーツ `partName` は予約語です。名前を変更する必要があります。**

**説明:** マイグレーション・ツールは、プログラムの名前を自動的に変更しません。プログラムは `programName.opt` という名前の特殊な生成オプション・パーツを使用する可能性があるため、マイグレーション・ツールは生成オプション・パーツの名前も変更しません。

**ユーザーの処置:** プログラム名を変更する場合は、対応する生成オプション・パーツの名前を必ず変更してください。

---

**IWN.MIG.1002.w 生成オプション・パーツ `partName` - `/dbms=odbc` は `dbms="DB2"` にマイグレーションされます。**

**説明:** 指定された生成オプション・パーツは、VAGen 生成オプション `/dbms=odbc` を指定しています。EGL は、DB2 または Oracle のみをサポートします。マイグレーション・ツールは、EGL ビルド記述子パーツ内で、`/dbms=odbc` を `dbms="DB2"` に変換します。EGL は、JDBC ドライバーを使用して DB2 のサポートを提供します。ご使用のデータベース用の JDBC ドライバーが存在すれば、ビルド記述子オプション `dbms="DB2"` をデータベース・タイプとして使用できる可能性があります。

**ユーザーの処置:** ODBC サポートを使用していたさまざまな VAGen プログラムをマイグレーション、生成、およびテストして、ご使用の JDBC ドライバーの環境で必要な機能すべてが正しく動作することを確認してください。

---

**IWN.MIG.1003.e 生成オプション・パーツ `partName` - `/system=systemType` はサポートされていません。**

**説明:** 指定された生成オプション・パーツは、VAGen の `/system` 生成オプションを含んでおり、EGL によってサポートされないランタイム環境を指定しています。マイグレーション・ツールは、EGL ビルド記述子パーツ内で、`/system` 生成オプションをコメントに変換します。

**ユーザーの処置:** このビルド記述子パーツが他のビルド記述子パーツによって使用されているかどうか判断してください。使用されていない場合は、このビルド記述子パーツを削除できます。また、EGL がこのランタイム環境を将来サポートする予定がある場合は、ビルド記述子パーツを参照用に保持しておくこともできます。

---

**IWN.MIG.1004.w 生成オプション・パーツ `partName` - `/system=systemType` を使用するには、`destPort` が設定されている必要があります。**

**説明:** 示された生成オプション・パーツは `/system` 生成オプションを含み、COBOL ランタイム環境を指定しています。EGL ビルド・プロセスに対して、`destPort` ビルド記述子オプションを使用して宛先ポートを指定する必要があります。

**ユーザーの処置:** 生成オプション・パーツに対応するビルド記述子パーツを変更して、`destPort` ビルド記述子オプションを組み込みます。`destPort` の値を指定するときは、以下を考慮します。

- z/OS 環境の場合、`destPort` のデフォルト値はありません。`destPort` ビルド記述子オプションを追加し、その値が z/OS ビルド・サーバーを開始する JCL (ジョブ制御言語) で使用する値と一致しなければなりません。z/OS ビルド・サーバーを開始するサンプル JCL (ジョブ制御言語) はポート 5555 を使用します。
- iSeries 環境の場合、`destPort` のデフォルト値はありません。`destPort` ビルド記述子オプションを追加する必要があります。値は、iSeries ビルド・サーバーによって使用される値と一致しなければなりません。
- VSE 環境の場合、`destPort` のデフォルト値は 21 です。`destPort` ビルド記述子オプションを指定する必要があるのは、値が 21 とは異なる場合のみです。

---

**IWN.MIG.1005.e 生成オプション・パーツ `partName` - `/system=systemType` は現在サポートされていません。**

**説明:** 指定された生成オプション・パーツは、VAGen の `/system` 生成オプションを含んでおり、EGL によ



て現在サポートされないランタイム環境を指定しています。マイグレーション・ツールは、将来の利用のために /system 生成オプションを EGL ビルド記述子オプションに変換します。ただし、このランタイム環境は現在 EGL によってサポートされていないので、値はビルド記述子パーツ・エディターに表示されません。テキスト・エディターを使用して、この値を表示できます。

**ユーザーの処置:** なし。EGL の将来のリリースで使用する可能性があるため、このビルド記述子パーツは保持しておく必要があります。

---

**IWN.MIG.1099.e 制御パーツ** *partName - symparm*  
*symparmName* はサポートされません。

**説明:** 指定された制御パーツは、EGL によってサポートされない symparm を使用または設定しています。マイグレーション・ツールは、オリジナルの VAGen symparm 名を使用して、symparm を「現状のまま」マイグレーションします。ただし、この symparm は生成時に設定されません。

**ユーザーの処置:** 制御パーツを変更して、symparm のデフォルト値を設定します。あるいは、指定された symparm を今後は使用しないように制御パーツを変更します。

---

**IWN.MIG.1101.e リンケージ・テーブル・パーツ**  
*partName* は予約語です。名前を変更する必要があります。

**説明:** マイグレーション・ツールは、制御パーツの名前を自動的に変更しません。

**ユーザーの処置:** リンケージ・オプション・パーツ名を予約語でないものに変更します。リンケージ・オプション・パーツ名を変更する場合は、このリンケージ・オプション・パーツを参照するビルド記述子パーツすべてを必ず変更してください。

---

**IWN.MIG.1102.e リンケージ・テーブル** *partName -*  
*/contable=BINARY* はサポートされません。変更する必要があります。

**説明:** VisualAge Generator は、リンケージ・テーブル・パーツ内で */contable=BINARY* をサポートします。EGL はこの値をサポートしません。マイグレーション・ツールは、EGL のリンケージ・オプション・パーツに *conversionTable="BINARY"* という値を組み込みます。この値は無効ですが、生成時まで検出されません。

**ユーザーの処置:** *conversionTable* の値を、EGL によってサポートされる値に変更する必要があります。EGL の *conversionTable* 属性、および使用可能なオプションについて詳しくは、オンライン・ヘルプのリンケージ・

オプション・パーツに関する情報を参照してください。

---

**IWN.MIG.1103.e リンケージ・テーブル・パーツ**  
*partName -*

*/remotecomtype=CICSCLIENT* はサポートされません。デフォルトで *CICSECI* に設定されます。

**説明:** VisualAge Generator は、リンケージ・テーブル・パーツ内で */remotecomtype=CICSCLIENT* をサポートします。EGL はこの値をサポートしません。マイグレーション・ツールは、EGL のリンケージ・オプション・パーツに *remoteComType="CICSECI"* を組み込みます。この値は有効ですが、使用する予定の値とは異なる可能性があります。CICSECI を使用する場合は、*ctgPort* と *ctgLocation* を設定する必要があります。

**ユーザーの処置:** CICSECI を使用する場合は、リンケージ・オプション・パーツを変更して、*remoteComType* として CICSECI を指定するエントリーの *ctgPort* と *ctgLocation* の値を設定します。CICSECI を使用しない場合、EGL の *remoteComType* 属性、および EGL で使用できるオプションについて詳しくは、オンライン・ヘルプのリンケージ・オプション・パーツに関する情報を参照してください。

---

**IWN.MIG.1104.e リンケージ・テーブル・パーツ**  
*partName -*

*/remotecomtype=communicationType* はサポートされません。変更する必要があります。

**説明:** VisualAge Generator は、リンケージ・テーブル・パーツ内で */remotecomtype=communicationType* をサポートします。EGL は、この通信プロトコルをサポートしません。マイグレーション・ツールは、EGL のリンケージ・オプション・パーツに *remoteComType="communicationType"* を組み込みます。この値は無効であり、変更する必要があります。

**ユーザーの処置:** 使用する通信プロトコルを決定します。その後、パーツを編集し、*remoteComType* を EGL によってサポートされる値に変更します。EGL の *remoteComType* 属性、および EGL で使用可能なオプションについて詳しくは、オンライン・ヘルプのリンケージ・オプション・パーツに関する情報を参照してください。

---

**IWN.MIG.1201.e リソース関連パーツ** *partName* は予約語です。名前を変更する必要があります。

**説明:** マイグレーション・ツールは、制御パーツの名前を自動的に変更しません。

**ユーザーの処置:** リソース関連パーツ名を予約語でないものに変更します。リソース関連パーツ名を変更する場合は、このリソース関連パーツを参照するビルド記述子パーツすべてを必ず変更してください。

---

**IWN.MIG.1202.e** リソース関連パーツ *partName* -  
*/filetype=fileType* はサポートされません。  
変更する必要があります。

**説明:** VisualAge Generator は、一部のワークステーション環境では */filetype=BTRIEVE* と */filetype=MFCOBOL* をサポートします。EGL は、これらのファイル・タイプをサポートしません。マイグレーション・ツールは、EGL リソース関連パーツに *filetype* の情報を組み込みます。この値は無効で、「問題」ビューにエラーが示されます。

**ユーザーの処置:** *filetype* の値を、EGL によってサポートされる値に変更する必要があります。EGL の *filetype* 属性、および使用可能なオプションについて詳しくは、オンライン・ヘルプのリソース関連パーツに関する情報を参照してください。

---

**IWN.MIG.1203.e** リソース関連パーツ *partName* -  
*/system* は *targetSystem* ですが、これはサポートされません。*/filetype fileType* の情報に基づいてマイグレーションしました。

**説明:** リソース関連パーツには、指定された *targetSystem* を使用するエントリーが含まれています。このターゲット・システムは、EGL ではサポートされません。マイグレーション・ツールは、*fileType* に基づいてリソース関連エントリーをマイグレーションします。例えば、*targetSystem* が *mvs\** で、*fileType* が *transient* である場合、マイグレーション・ツールは

EGL リソース関連エントリーを作成し、EGL システムを *mvs\** に設定します。これは無効であり、「問題」ビューにエラーが示されます。有効な EGL システム (例: *zosscics*) を指定して、エントリーを訂正できます。

**ユーザーの処置:** 「問題」ビューにエラーがある場合は、有効なターゲット・システムを指定して、リソース関連パーツのエントリーを訂正します。

---

**IWN.MIG.1301.e** リンク・エディット・パーツ  
*partName* は予約語です。名前を変更する必要があります。

**説明:** マイグレーション・ツールは、プログラムの名前を自動的に変更しません。プログラム名は対応するリンク・エディット・パーツと一致している必要があるので、マイグレーション・ツールはリンク・エディット・パーツの名前変更も行いません。

**ユーザーの処置:** プログラム名を変更する場合は、対応するリンク・エディット・パーツの名前を必ず変更してください。

---

**IWN.MIG.1401.e** バインド制御パーツ *partName* は予約語です。名前を変更する必要があります。

**説明:** マイグレーション・ツールは、プログラムの名前を自動的に変更しません。プログラム名は対応するバインド制御パーツと一致している必要があるので、マイグレーション・ツールはバインド制御パーツの名前変更も行いません。

**ユーザーの処置:** プログラム名を変更する場合は、対応するバインド制御パーツの名前を必ず変更してください。

---

## VisualAge Generator から EGL マイグレーション・ツールへのメッセージ — ステージ 3

ステージ 3 で生成されるメッセージは、トレース・メッセージのみです。

## 付録 D. 「問題」ビューのメッセージ

未確定状態では、マイグレーション・ツールがマイグレーション時に作成する正しい EGL 構文を常に判別できるとは限りません。未確定状態は通常、関連パーツがマイグレーション時に使用できない場合に起こります。この場合、エラーが「問題」ビューに表示されるように、マイグレーション・ツールは意図的に無効な EGL 構文を作成することがあります。次の表に、EGL の検証時にエラーを引き起こす特定のテキスト・ストリングをリストします。具体的な EGL エラー・メッセージは異なる場合がありますが、左側の欄にリストされているテキスト・ストリングが、エラーとしてフラグを立てられた EGL ステートメントの近くに表示されます。マイグレーション・ツールがこれらのテキスト・ストリングを組み込むとき、ツールはマイグレーション・ログにもメッセージを出します。

表 149. EGL 構文エラーまたは検証エラーを引き起こす VAGen マイグレーション・テキスト

EGL 構文内の VAGen マイグレーション・テキスト	問題と解決策
###KEYS_NOT_FOUND###	<p><b>問題:</b> 現行 SQL レコードが別のレコードの構造を埋め込んでいます。マイグレーション時に、<code>embed</code> ステートメントに指定されたレコードが使用不可でした。VAGen 内で現行 SQL レコードに対して指定されたキー項目は <code>keyItems</code> プロパティに含まれていますが、埋め込まれるレコードからのキーが欠落しています。</p> <p><b>解決策:</b> <code>embed</code> ステートメントに指定されている EGL レコードを見付けます。###KEYS_NOT_FOUND### のテキストを、埋め込まれる SQL レコードにリストされているキーに置き換えます。埋め込まれるレコードのキーを現行レコードのキー項目と組み合わせ、埋め込まれるレコードのレコード構造内での出現順にこれらのフィールドが並ぶようにします。現行レコードのキー項目が、埋め込まれるレコードの <code>keyItems</code> プロパティの中でも指定されている場合は、EGL の <code>keyItems</code> プロパティにこのフィールドを 1 回だけ組み込みます。</p>
###TABLES_NOT_FOUND###	<p><b>問題:</b> 現行 SQL レコードが別のレコードの構造を埋め込んでいます。マイグレーション時に、<code>embed</code> ステートメントに指定されたレコードが使用不可でした。</p> <p><b>解決策:</b> <code>embed</code> ステートメントに指定されている EGL レコードを見付けて、<code>tableNames</code> と <code>tableNameVariables</code> の両プロパティを現行 SQL レコードにコピーします。</p>
EZE_DUPLICATE	<p><b>問題:</b> VAGen の IF、WHILE、または TEST の各ステートメントに指定されているレコードが、マイグレーション時に使用不可でした。</p> <p><b>解決策:</b> EGL の <code>if</code> ステートメント、または <code>while</code> ステートメントに指定されている EGL レコードを見付けます。EZE_DUPLICATE を次のいずれかに変更します。</p> <ul style="list-style-type: none"><li>• 非 SQL レコードの場合は <code>duplicate</code></li><li>• SQL レコードの場合は <code>unique</code></li></ul>

表 149. EGL 構文エラーまたは検証エラーを引き起こす VAGen マイグレーション・テキスト (続き)

EGL 構文内の VAGen マイグレーション・テキスト	問題と解決策
EZE_NULL	<p><b>問題:</b> マイグレーション・ツールは、VAGen の IF、WHILE、または TEST の各ステートメントに指定されている項目が、SQL レコードまたはマップのどちらにあるか判別できません。</p> <p><b>解決策:</b> プログラムを検討して、フィールドが SQL レコードまたは書式のどちらにあるか判別します。SQL フィールドの場合は EZE_NULL を <i>null</i> に置き換え、書式フィールドの場合は <i>blanks</i> に置き換えます。</p>
EZE_SETPAGE();	<p><b>問題:</b> マイグレーション中に、VAGen SET map PAGE ステートメントに指定されたマップが使用できませんでした。</p> <p><b>解決策:</b> EZE_SETPAGE() ステートメントに付随する、// VAGen Info のコメントに指定されている書式を見付けます。EZE_SETPAGE を次のいずれかに変更します。</p> <ul style="list-style-type: none"> <li>• テキスト書式の場合は <i>clearScreen()</i></li> <li>• 印刷書式の場合は <i>pageEject()</i></li> </ul>
EZE_UNKNOWN_PARTTYPE	<p><b>問題:</b> マイグレーション・データベースに格納されている外部ソース形式が無効です。マイグレーション・ツールは、パーツ型を判別できず、パーツを EGL 構文に変換できませんでした。</p> <p><b>解決策:</b> EZE_UNKNOWN_PARTTYPE ステートメントに指定されているパーツは無効です。この問題が少数のパーツのみに対して起こる場合は、VisualAge Generator から外部ソース形式をエクスポートし、単一ファイル・モードでこれらのパーツのマイグレーションを試行してください。</p> <p>ユーザー独自のツールを作成してマイグレーション・データベースをロードしている場合は、ツールが外部ソース形式のコードをマイグレーション・データベースにロードする方法に問題があることが考えられます。問題の原因の判別に役立つ照会については、441 ページの『付録 G. マイグレーション・データベース』を参照してください。</p>
EZE_UNKNOWN_PCB_MAPPING	<p><b>問題:</b> マイグレーション中に、プログラム用に指定された PSB パーツが使用できませんでした。マイグレーション・ツールは、pcbParms プロパティに指定する値を判別できませんでした。</p> <p><b>解決策:</b> プログラム内で、<i>psb</i> という名前の変数に関するデータ宣言を見付けてください。<i>psb</i> に関して指定される型定義は、プログラムの EGL PSBRecord パーツの名前です。プログラムの pcbParms プロパティを変更し、入力 PCB パラメーターを、EGL PSBRecord パーツ内の対応する PCB にマップします。詳しくは、(423 ページの) メッセージ IWN.MIG.0809.e を参照してください。</p>

表 149. EGL 構文エラーまたは検証エラーを引き起こす VAGen マイグレーション・テキスト (続き)

EGL 構文内の VAGen マイグレーション・テキスト	問題と解決策
EZE_UNKNOWN_PCB_TYPE	<p><b>問題:</b> マイグレーション中に、プログラムに指定された PSB パーツが使用不可であったか、または含まれていた PCB の数がプログラムのパラメーター・リストで指定された数を下回っていました。マイグレーション・ツールは、プログラムの PCB パラメーターに使用する型定義を判別できませんでした。パラメーター・リスト内のそれぞれの PCB は、<code>pcbn</code> です。ここで、<i>n</i> は、PSB パーツ内の PCB に対応する数値リテラルです。</p> <p><b>解決策:</b> プログラム内で、<code>psb</code> という名前の変数のデータ宣言を見つけてください。<code>psb</code> に関して指定される型定義は、プログラムの EGL PSBRecord パーツの名前です。プログラムの PCB パラメーターを変更し、PSB パーツ内の対応する PCB の型定義に基づいて正しい型定義レコード (IO_PCBRecord、ALT_PCBRecord、DB_PCBRecord、または GSAM_PCBRecord) を指定します。詳しくは、(423 ページの) メッセージ IWN.MIG.0808.e を参照してください。</p>
EZE_UNKNOWN_QUALIFIER	<p><b>問題:</b> 現在のセグメント検索指数 (SSA) に、修飾されていない EGL ホスト変数 (VAGen 比較値項目) が含まれています。SSA の DL/I セグメント・レコードまたはその代替仕様レコードが、マイグレーション中に使用できませんでした。あるいは、レコードは使用可能でしたが、比較値項目が含まれていませんでした。マイグレーション・ツールは、EGL ホスト変数の修飾子を判別できませんでした。</p> <p><b>解決策:</b> 現在の SSA の DL/I セグメント・レコードまたはその代替仕様を見つけてください。ホスト変数とそのレコード内の項目であるかどうかを判別します。レコード内の項目である場合は、ホスト変数の修飾子を DL/I セグメント・レコード名に変更します。レコード内の項目でない場合は、使用する正しい修飾子を決定してください。正しい修飾子の判別方法についての詳細は、(419 ページの) メッセージ IWN.MIG.0611.e を参照してください。</p>
EZE_UNKNOWN_RELOP	<p><b>問題:</b> 変更された DL/I ステートメントで、無効な比較演算子を使用されました。これは、不正な比較演算子の値が保管されるという、VisualAge Generator 内の問題によって発生します。マイグレーション・ツールは、正しい比較演算子を判別できませんでした。</p> <p><b>解決策:</b> VisualAge Generator で DL/I Call Editor を使用して、指定された関数の SSA を検討します。演算子が関数の外部形式ファイルに誤って保管されていても、正しい演算子が DL/I Call Editor に表示されます。EGL で関数を編集し、EZE_UNKNOWN_RELOP を正しい値に変更します。</p> <p><b>注:</b> 問題を引き起こす可能性の最も高い演算子は、不等 記号です。EGL SSA の不等 記号は <code>!=</code> です。</p>
EZE_UNKNOWN_RETURN_COLUMN	<p><b>問題:</b> VAGen の RETR ステートメントに指定されている VAGen テーブルが、マイグレーション時に使用不可でした。</p> <p><b>解決策:</b> 代入ステートメントに指定されている EGL dataTable を見付け、EZE_UNKNOWN_RETURN_COLUMN を dataTable の 2 列目の名前に置き換えます。</p>



表 149. EGL 構文エラーまたは検証エラーを引き起こす VAGen マイグレーション・テキスト (続き)

EGL 構文内の VAGen マイグレーション・テキスト	問題と解決策
EZE_UNKNOWN_SEARCH_COLUMN	<p><b>問題:</b> VAGen の FIND ステートメントまたは RETR ステートメントに指定されている VAGen テーブルが、マイグレーション時に使用不可でした。</p> <p><b>解決策:</b> <i>if</i> ステートメントに指定されている EGL dataTable を見付け、EZE_UNKNOWN_SEARCH_COLUMN を dataTable の 1 列目の名前に置き換えます。</p>
EZE_UNKNOWN_SQLTABLE	<p><b>問題:</b> 入出力オブジェクトとして指定されている SQL レコードが、マイグレーション時に使用不可でした。マイグレーション・ツールは、EGL 入出力ステートメントの正しい tables 文節を判別できませんでした。</p> <p><b>解決策:</b> 入出力ステートメントに指定されているレコードを見付けて、レコードの <i>tableNames</i> プロパティまたは <i>tableNameVariables</i> プロパティ (あるいはその両方) から正しい tables 文節を判別します。</p>
EZE_UNKNOWN_SQL_FORUPDATEOF	<p><b>問題:</b> VisualAge Generator は、SQL の入出力オプション UPDATE または SETUPD に対して、デフォルトの <i>for update of</i> 文節を作成しました。入出力オブジェクトとして指定されている SQL レコードが、マイグレーション時に使用不可でした。このため、マイグレーション・ツールは、EGL の入出力ステートメントの正しい <i>for update of</i> 文節を判別できませんでした。</p> <p><b>解決策:</b> 入出力ステートメントに指定されているレコードを見付けて、レコードのフィールド・リストから正しい <i>for update of</i> 文節を判別します。VisualAge Generator のデフォルトの <i>for update of</i> 文節は、レコード内でのフィールドのリスト順と同じ順序でレコードの列名を並べたリストですが、次の項目を省略しています。</p> <ul style="list-style-type: none"> <li>レコードに対する EGL の <i>keyItems</i> プロパティにリストされている列名。</li> <li>EGL の <i>isReadOnly=yes</i> プロパティを使用して指定された列名。</li> </ul> <p>入出力ステートメントに指定されているレコードが別の SQL レコードを埋め込む場合は、次のことを行ってください。</p> <ul style="list-style-type: none"> <li><i>embed</i> ステートメントに指定されているレコードを使用して、列の順序と <i>isReadOnly=yes</i> プロパティを判別する。</li> <li>入出力ステートメントに指定されているレコード (埋め込み側レコード) を使用して、<i>keyItems</i> プロパティを判別する。</li> </ul> <p><i>for update of</i> 文節を EGL の <i>prepare</i> ステートメント内で使用する場合は、列名のリストを二重引用符で囲みます。</p>



表 149. EGL 構文エラーまたは検証エラーを引き起こす VAGen マイグレーション・テキスト (続き)

EGL 構文内の VAGen マイグレーション・テキスト	問題と解決策
EZE_UNKNOWN_SQL_INSERTCOLNAME	<p><b>問題:</b> VisualAge Generator は、SQL の ADD 入出力オプションに対してデフォルトの列のリストを作成しました。入出力オブジェクトとして指定されている SQL レコードが、マイグレーション時に使用不可でした。このため、マイグレーション・ツールは EGL の <i>add</i> ステートメントの正しい列名リストを判別できませんでした。</p> <p><b>解決策:</b> 入出力ステートメントに指定されているレコードを見付けて、レコードのフィールド・リストから正しい列のリストを判別します。VisualAge Generator のデフォルトの列名リストは、レコード内でのフィールドのリスト順と同じ順序でレコードの列名を並べたリストですが、EGL の <i>isReadOnly=yes</i> プロパティを使用して指定された列名を省略しています。入出力ステートメントに指定されたレコードが別のレコードを埋め込む場合は、<i>embed</i> ステートメントに指定されたレコードを使用して、列の順序と <i>isReadOnly=yes</i> プロパティを判別してください。列名のリストは、EGL の <i>prepare</i> ステートメント内で使用されることはありません。</p>
EZE_UNKNOWN_SQL_INT0	<p><b>問題:</b> VisualAge Generator は、SQL の入出力オプション INQUIRY、SETINQ、UPDATE、または SETUPD の <i>into</i> 文節に対して、デフォルトのデータ項目のリストを作成しました。入出力オブジェクトとして指定されている SQL レコードが、マイグレーション時に使用不可でした。このため、マイグレーション・ツールは、EGL の入出力ステートメントの正しい <i>into</i> 文節を判別できませんでした。</p> <p><b>解決策:</b> 入出力ステートメントに指定されているレコードを見付けて、<i>into</i> 文節の正しいフィールドのリストを判別します。VisualAge Generator のデフォルトのフィールド・リストは、レコード内でのフィールドのリスト順と同じ順序でレコードのフィールドを並べたリストです。入出力ステートメントに指定されているレコードが別のレコードを埋め込む場合は、<i>embed</i> ステートメントに指定されたレコードを使用して、フィールドの順序を判別してください。<i>into</i> 文節は、EGL の <i>prepare</i> ステートメント内で使用されることはありません。</p>
EZE_UNKNOWN_SQL_SELECT	<p><b>問題:</b> VisualAge Generator は、SQL の入出力オプション INQUIRY、SETINQ、UPDATE、または SETUPD の <i>select</i> 文節に対して、デフォルトの列のリストを作成しました。入出力オブジェクトとして指定されている SQL レコードが、マイグレーション時に使用不可でした。このため、マイグレーション・ツールは、EGL の入出力ステートメントの正しい <i>select</i> 文節を判別できませんでした。</p> <p><b>解決策:</b> 入出力ステートメントに指定されているレコードを見付けて、<i>select</i> 文節の正しい列名のリストを判別します。VisualAge Generator のデフォルトの列名リストは、レコード内でのフィールドのリスト順と同じ順序でレコードの列名を並べたリストです。入出力ステートメントに指定されているレコードが別のレコードを埋め込む場合は、<i>embed</i> ステートメントに指定されたレコードを使用して、列の順序を判別してください。<i>select</i> 文節を EGL の <i>prepare</i> ステートメント内で使用する場合は、列名のリストを二重引用符で囲みます。</p>

表 149. EGL 構文エラーまたは検証エラーを引き起こす VAGen マイグレーション・テキスト (続き)

EGL 構文内の VAGen マイグレーション・テキスト	問題と解決策
EZE_UNKNOWN_SQL_VALUES	<p><b>問題:</b> VisualAge Generator は、SQL の ADD 入出力オプションの値を指定するために、デフォルトのデータ項目のリストを作成しました。入出力オブジェクトとして指定されている SQL レコードが、マイグレーション時に使用不可でした。このため、マイグレーション・ツールは EGL の <i>add</i> ステートメントの <i>values</i> 文節の正しいフィールド名リストを判別できませんでした。</p> <p><b>解決策:</b> 入出力ステートメントに指定されているレコードを見つけて、レコードのフィールド・リストから正しいフィールドのリストを判別します。VisualAge Generator のデフォルトのフィールド名リストは、レコード内でのフィールドのリスト順と同じ順序でレコードのフィールドを並べたリストですが、EGL の <i>isReadOnly=yes</i> プロパティを使用して指定されたフィールドを省略しています。入出力ステートメントに指定されたレコードが別のレコードを埋め込む場合は、<i>embed</i> ステートメントに指定されたレコードを使用して、フィールドの順序と <i>isReadOnly=yes</i> プロパティを判別してください。<i>values</i> 文節は、EGL の <i>prepare</i> ステートメント内で使用されることはありません。</p>

## 付録 E. 「問題」ビューの IWN.xxx メッセージ

IWN.SYN、IWN.VAL、および IWN.XML のメッセージの中には、完全に EGL 内で開発したコードの場合より、VisualAge Generator からマイグレーションした EGL ソース・コードの場合に出される可能性が高いものがあります。ここでは、マイグレーションされたコードに対して特別な意味のあるメッセージをリストします。

### IWN.VAL メッセージ

**IWN.VAL.3012.e** 同じ *recordName* という名前が、関数、プログラム、またはライブラリー *programName* で変数宣言、パラメーター宣言、使用宣言、または定数宣言としても使用されています。

**説明:** VisualAge Generator で、パラメーター・リストとテーブルおよび追加レコードのリストに、同じレコード名を指定することが許容されました。この状態で、VisualAge Generator はテーブルおよび追加のレコードのリスト内のレコードを無視しました。

**ユーザーの処置:** プログラムを編集し、レコードの宣言を除去します。

**IWN.VAL.3256.e** 長さ *m* の配列リテラルは、長さ *n* の配列リテラルに割り当てるには大きすぎます。

**説明:** VGUI レコードの場合、実行ボタンの値のリストが配列のサイズとしては大きすぎます。VisualAge Generator では、生成時に余分な値は無視されました。

**ユーザーの処置:** VGUI レコードを編集し、追加の値を除去します。

**IWN.VAL.4300.e** パーツ *partName* は、次のタイプのいずれかに解決できなかったか、または解決されませんでした: *partTypeList*

**説明:** このメッセージが出される可能性がある状況は、いくつかあります。

- 指定された制御パーツが存在しない。
- 指定された制御パーツが、エラーのある制御パーツとは異なるプロジェクトまたはパッケージ内にある。VisualAge Generator 内に制御パーツの関連パーツがないので、マイグレーション・ツールは制御パーツの import ステートメントを作成しません。
- XML パーサーが .eglbld ファイルを完全に処理できなかった。この場合、指定された制御パーツは、エラ

ーのある制御パーツと同じファイル内に存在する可能性があります。メッセージ「IWN.XML.3999.e XML 検証エラー - 属性 "xxxxx" はエレメント "yyyyy" に対してすでに指定されています」があるかどうか調べてください。このメッセージは、属性 xxxxx が同じ制御パーツに対して複数回指定されていることを示しています。

**ユーザーの処置:** 指定された制御パーツが存在しない場合は、そのパーツを作成するか、そのパーツの参照を除去します。

指定された制御パーツが別のプロジェクト内にある場合は、現行プロジェクトのプロパティの EGL ビルド・パスを更新して、指定された制御パーツが存在するプロジェクトを組み込みます。指定された制御パーツが別のパッケージ内にある場合は、現行の .eglbld ファイルを編集して、指定された制御パーツが存在するパッケージの import ステートメントを組み込みます。

属性 xxxxx がすでに指定されていることを示すメッセージ IWN.XML.3999.e が出された場合は、現行の .eglbld ファイルを編集して、制御パーツ内で xxxxx が 1 回だけ指定されるようにします。eglbld ファイルを保管すると、「問題」ビューのメッセージが更新されます。

XML パーサーは、.eglbld ファイル内で最初に重複する属性の箇所では処理を停止するので、ファイル全体の構文解析を行うには、その前にいくつかのエラーを解決する必要があります。すべての

IWN.XML.3999.e メッセージが解決された後、指定された制御パーツが参照側の制御パーツと同じ .eglbld ファイル内にあれば、制御パーツが使用可能になります。

**IWN.VAL.4925.e** *programName* の変数宣言 *recordName* を解決できませんでした。

**説明:** プログラムが DL/I を使用する場合、指定された *recordName* は、プログラムの PSBRecord で指定された DL/I セグメントの名前であることがあります。

VisualAge Generator では、プログラムの PSB で指定される DL/I セグメントは、すべてプログラムの関連付け

と見なされます。これにより、デフォルト SSA で使用される DL/I セグメントは、すべてプログラムで使用可能になります。ただし、VisualAge Generator は、セグメントが、実際に DL/I I/O 用に使用されていない場合、あるいはより低いレベルのセグメント用にデフォルト SSA をビルドするために使用されていない場合には、欠落している DL/I セグメント・レコードに関してエラー・メッセージを発行しません。マイグレーション・ツールは、プログラムの PSB 内のそれぞれの DL/I セグメント・レコードごとに、プログラムに宣言を自動的に追加します。

**ユーザーの処置:** プログラムを検討して、DL/I セグメント・レコードが必要であるかどうかを判別します。レコードが使用されておらず、より低いレベルのセグメントのデフォルト SSA のビルドに必要なでない場合は、プログラムを編集して、レコードの宣言を除去します。

---

**IWN.VAL.4929.e** プログラム *programName* 内の *formGroupName* に対する **use** ステートメントが解決しません。

**説明:** プログラムは VisualAge Generator 内でマップ・グループを指定していましたが、プログラム内に display または converse のどちらの入出力オプションも存在せず、また呼び出し先パラメーターとしてマップが指定されていません。別の可能性としては、プログラムがヘルプ・マップ・グループを指定していて、プログラムの使用するマップのどれもがヘルプ・マップを指定していないことが考えられます。

**ユーザーの処置:** プログラムを変更して、formGroup の use ステートメントを除去するか、formGroup を含むパッケージの import ステートメントを追加するかのどちらかを行います。VisualAge Generator に formGroup が存在していなかった場合は、formGroup を作成する必要があります。この場合、formGroup パーツを作成し、フォームは作成しないでください。

---

**IWN.VAL.5100.e** *xxxxx* は、*yyyyy* システム・ワードに対して無効な修飾子です。

**説明:** VisualAge Generator では有効であった EZE ワードが、現在 EGL によってサポートされていません。マイグレーション・ツールは、将来 EGL の置換表現になると考えられるものについて「最適予測」を行って、EZE ワードをマイグレーションします。これにより、プログラム・ロジックが保持されます。

**ユーザーの処置:** 関数を編集して、このシステム・ワードが使用されなくなるようにロジックの変更を行います。あるいは、新規プロジェクトを作成して、現在使用できないマイグレーション済み関数を保持します。現在サポートされていない VAGen 値を含む関数すべてを、

この新規プロジェクトに移動します。

---

**IWN.VAL.5101.e** *mainFunctionName* このステートメントの位置では、*xxxxx* システム・ワードの使用は無効です。

**説明:** 指定された main 関数を使用するプログラムが、さらに他の関数を呼び出しています。関数呼び出しチェーンに含まれる関数の 1 つが、指定されたシステム・ワードをステートメント内で使用しています。マイグレーション・ツールは、VAGen の EZE ワードを置換する EGL システム・ワードを常に修飾します。*xxxxx* システム・ワードが sysLib、sysVar、mathLib、strLib、VGLib、VGVar、converseLib、converseVar、dliLib、dliVar、または javaLib によって修飾されていない場合、最も可能性が高い原因は次のとおりです。

- VAGen プログラムが暗黙データ項目を許容しており、*xxxxx* の定義がマイグレーション時に自動的に作成された。EGL は、暗黙データ項目を許容しません。マイグレーション・ツールはまた、暗黙データ項目の定義を自動的に作成しません。
- レコード、マップ、またはテーブルがマイグレーション・セットに含まれていないため、マイグレーション・ツールがプログラムに必要な import ステートメントを組み込むことができなかった。

**ユーザーの処置:** VAGen プログラムが暗黙項目を許容していたかどうか確認します。許容していた場合は、VisualAge Generator 内でプログラムを検証します。VAGen の「メッセージの表示 (View Messages)」リストに、暗黙データ項目の正しい定義を示すメッセージが表示されます。プログラムの宣言セクションに、データ項目の定義を追加します。VAGen プログラムが暗黙項目を許容していなかった場合は、VisualAge Generator 内でプログラムの関連リストを作成します。この関連リストから、VAGen 参照ツールを使用して、指定されたデータ項目を検索します。「参照 (References)」ツールの結果は、どのレコード、マップ、またはテーブルがマイグレーション・セットから欠落しているかを知る手掛かりになります。

---

**IWN.VAL.5168.e** *xxxxx* を Is/Not 式の中で使用することは無効です。

**説明:** VisualAge Generator では、指定された値は EZESYS に対して有効な値でした。EGL には、この値に対応する値がありません。マイグレーション・ツールは、プログラム・ロジックを保持するために VisualAge Generator 値をマイグレーションします。

**ユーザーの処置:** 関数を編集して、この値が使用されなくなるようにロジックの変更を行います。あるいは、新規プロジェクトを作成して、現在使用できないマイグレ



ーション済み関数を保持します。現在サポートされていない VAGen 値を含む関数すべてを、この新規プロジェクトに移動します。

---

**IWN.VAL.5411.e** プロパティ "*segmentRecord*" によって参照されるパーツ "*segmentName*" を解決できません。

**説明:** VisualAge Generator では、PSB で指定される DL/I セグメントは、すべて PSB の関連付けと見なされます。ただし、VisualAge Generator は、セグメントが、実際にプログラムで DL/I I/O 用に使用されていない場合、あるいはより低いレベルのセグメント用にデフォルト SSA をビルドするために使用されていない場合には、欠落している DL/I セグメント・レコードに関してエラー・メッセージを発行しません。EGL では、PSB で参照される DL/I セグメント・レコードは、すべて必須です。

**ユーザーの処置:** PSB を使用するすべてのプログラムの PSBRecord およびロジックを検討します。欠落している DL/I セグメント・レコードを指定する PCB をブレースホルダーとしてのみ使用する場合、あるいは非 EGL プログラムに移動する場合は、PCB を変更して、階層情報を除去することを考慮してください。

---

**IWN.VAL.6503.e** *xxxxx* SQL 入出力ステートメントに、適切でない文節があります。*yyyyy* は、*zzzzz* 文節の前に表示されます。

**説明:** メッセージ IWN.VAL.6506.e を参照してください。

**ユーザーの処置:** メッセージ IWN.VAL.6506.e を参照してください。

---

**IWN.VAL.6506.e** *xxxxx* SQL 入出力ステートメントには、*yyyyy* 文節をただ 1 つ指定できません。

**説明:** このメッセージが出される可能性がある状況は、いくつかあります。

- VisualAge Generator の場合は、SQL キーワードを列名として使用できます。EGL の場合は、一部の SQL キーワードを使用できません。
- VisualAge Generator では、ある SQL 文節は括弧で囲みません。EGL では、これらの文節を括弧で囲む必要があります。例えば、副選択およびそれに関連した FROM、WHERE、GROUP BY、および ORDER BY 文節は、括弧で囲まなければなりません。

**ユーザーの処置:** 副選択を使用していない場合は、244 ページの『特殊な処理を必要とする SQL 予約語』で、SQL キーワードのリスト、および列名に関する問題を

解決するために使用できる手法について参照してください。

副選択を使用している場合は、副選択とそれに関連する FROM、WHERE、GROUP BY、および ORDER BY 文節の周囲に括弧を追加します。例えば、以下の SQL ステートメントについて考えてみます。

```
with #sql{
  SELECT EMPNO, COUNT(*) WORKDEPT
    FROM EMPLOYEE T1
    WHERE WORKDEPT LIKE '%E%'
    GROUP BY WORKDEPT
  UNION ALL
  SELECT EMPNO
    FROM EMP_ACT
    WHERE PROJNO IN('MA2100', 'MA2112')
}
```

SQL ステートメントを変更して、以下のように 2 番目の SELECT の周囲に括弧を追加します。

```
with #sql{
  SELECT EMPNO, COUNT(*) WORKDEPT
    FROM EMPLOYEE T1
    WHERE WORKDEPT LIKE '%E%'
    GROUP BY WORKDEPT
  UNION ALL
  ( SELECT EMPNO
    FROM EMP_ACT
    WHERE PROJNO IN('MA2100', 'MA2112'))
}
```

---

**IWN.VAL.6507.e** *xxxxx* SQL 入出力ステートメントは、*yyyyy* 文節を許可しません。

**説明:** メッセージ IWN.VAL.6506.e を参照してください。

**ユーザーの処置:** メッセージ IWN.VAL.6506.e を参照してください。

---

**IWN.VAL.6619.e** *functionName* - *recordName.fieldName* は解決できません。

**説明:** プログラムが DL/I を使用する場合、指定された *recordName* は、プログラムの PSBRecord で指定された DL/I セグメントの名前であることがあります。メッセージが add または get などの DL/I I/O ステートメントをポイントする場合、より低いレベルのセグメント用にデフォルト SSA をビルドするために DL/I セグメント・レコードが必要です。

**ユーザーの処置:** プログラム・ロジックを検討して、デフォルト SSA をビルドするために DL/I セグメント・レコードが必要かどうかを判別します。必要である場合は、プログラムを編集して、指定されたレコード用に宣言を追加します。

---

**IWN.VAL.6620.e** *functionName* - 変数アクセス *xxxxx* は未確定です。

**説明:** call ステートメントに関して問題が起こっているのかどうか、また *xxxxx* が非修飾データ項目かどうかを判別します。問題が call ステートメントに関連している場合は、データ項目がプログラムの *inputRecord* プロパティに関連したレベル 77 レコード内にあるかどうか確認します。非修飾項目が CALL ステートメントに使用されている場合、VisualAge Generator はプログラムの 1 次作業用ストレージ・レコード内のレベル 77 項目を優先します。一方、EGL は call ステートメントに対してこれと同じ優先順位を設定しません。

**ユーザーの処置:** 問題が call ステートメントに関連している場合は、この項目の修飾としてレベル 77 レコード名を使用できることがあります。ただし、この関数を呼び出すプログラムすべてが、同じレベル 77 レコードを使用していることを確認する必要があります。

---

**IWN.VAL.6681.e** *functionName* - 関数 *systemFunctionName(parameterTypeList)* は、引数 (*argumentTypeList*) に適用できません。

**説明:** VisualAge Generator は、一部の EZE ストリング関数の文字引数として数値項目 (数値、バイナリー、パック) を許容していました。EGL は、システム・ストリング (*strLib*) 関数の文字引数として数値項目をサポートしません。

**ユーザーの処置:** 文字データを使用するようにプログラムを変更してください。例えば、以前 *systemFunctionName* で使用したフィールドとタイプと長さが同じである数値フィールドで副構造化された文字フィールドを持つレコードを作成できます。 *functionName* によって指定された関数を変更して、数値項目を副構造化された数値フィールドに移動し、次に文字項目を *systemFunctionName* への引数として使用します。

---

**IWN.VAL.6695.e** *functionName* - この項目参照では、状態 *XXXXX* は許可されていません。

**説明:** 状態が PROTECT、SKIP、INVISIBLE、BLINK、または色である場合、このデータ項目は印刷書式に存在します。VisualAge Generator は、プリンター書式に対してこれらの属性の設定を許容していましたが、EGL では許容しません。

状態が EMPTY である場合は、書式またはレコードがマイグレーション時に使用不可であったかどうか判別してください。EGL は、独立データ項目の定義を許可し、型定義が見付からない場合は、定義が項目に対する

ものであると想定します。EGL は、データ項目に対する *set empty* の使用をサポートしません。

**ユーザーの処置:** 問題が印刷書式の属性の設定に関連している場合は、関数を変更してステートメントを除去します。また、テキスト書式と印刷書式の両方に同じ関数を使用されている場合は、印刷書式に使用するための関数のコピーを作成する必要があります。

問題が *set empty* ステートメントに関連している場合は、欠落しているレコードまたは書式を定義またはマイグレーションします。

---

**IWN.VAL.6716.e** *functionName* - 引数 *xxxxx* を、*inOut* パラメーター *yyyy* へパスすることはできません。引数は、*inOut* パラメーターと互換性のある参照である必要があります。

**説明:** *functionName* は、呼び出す関数の名前です。*inOut* パラメーターにパスされた引数は、参照に互換性がある必要があります。参照に互換性があるとは、引数タイプとパラメーター・タイプが完全に一致する必要がありますことを意味します。VisualAge Generator はサポートしていませんが、場合によっては、引数とパラメーターのタイプが異なっても許可されることがあります。

**ユーザーの処置:** 呼び出された関数が EGL システム関数の 1 つである場合、必要なパラメーター・タイプに一致するように、呼び出し側の関数を変更する必要があります。呼び出された関数がユーザー関数である場合、その呼び出された関数を変更して、より幅広い引数のタイプおよび長さを許可する NUMBER などの関数パラメーター・タイプを使用できる場合があります。それ以外には、呼び出された関数と、その関数が呼び出されたすべての場所を検討する方法があります。呼び出された関数を、それぞれが *inOut* パラメーターの異なる定義を持つ複数の関数に分割しなければならない場合があります。

---

**IWN.VAL.7553.e** *functionName* - *systemFunctionName* の引数 *n* は、ストリング項目、ストリング定数、またはストリング・リテラルであることが必要です。

**説明:** メッセージ IWN.VAL.6681.e を参照してください。

**ユーザーの処置:** メッセージ IWN.VAL.6681.e を参照してください。

---



---

**IWN.VAL.7789.e** numElementsItem 項目 *fieldName*  
は、乗算が発生する項目にすることはできません。

**説明:** UI レコードについて、VisualAge Generator は発生項目を配列として許容していましたが、その項目を配列としては処理しませんでした。 EGL は、

---

## IWN.XML メッセージ

---

**IWN.XML.3997.e** XML 検証エラー - エlement・タイプ "xxxxx" に対して、属性 "yyyyy" を宣言する必要があります。

**説明:** VisualAge Generator では、Element xxxxx に対してオプション yyyyy が有効です。この組み合わせは、EGL によってサポートされていません。マイグレーション・ツールは、無効であってもこの値をマイグレーションするので、ユーザーに問題の解決を促すためのエラーが「問題」ビューに示されます。

**ユーザーの処置:** xxxxx に対して有効なオプションについて、EGL のオンライン・ヘルプを確認してください。使用するオプションを決定したら、必要な変更を行うために、テキスト・エディターでビルド記述子ファイルを開く必要が生じることがあります。

---

**IWN.XML.3998.e** XML 検証エラー - 値 "xxxxx" が指定された属性 "system" には、リスト "environmentList" からの値が必要です。

**説明:** VisualAge Generator では、値 xxxxx は有効なターゲット環境です。この環境は、現在 EGL によってサポートされていません。マイグレーション・ツールは、一部のターゲット環境に関する情報をマイグレーションして、将来考えられる利用のために情報を保持します。

**ユーザーの処置:** 新規プロジェクトを作成して、現在使用できないマイグレーション済みビルド記述子を保持します。現在サポートされていない VAGen 値を含むビルド記述子パーツすべてを、この新規プロジェクトに移動します。パーツを移動するには、テキスト・エディターを使用してビルド記述子ファイルを開き、エラーのあるパーツをコピーして貼り付けます。

---

**IWN.XML.3999.e** (4 つの代替メッセージのうち 1 つ目)  
**XML 検証エラー - 属性 "xxxxx" はElement "yyyyy" に対してすでに指定されています。**

**説明:** 属性 xxxxx は、同じ制御パーツ内で複数回指定されています。XML パーサーは、.eglbld ファイルの処理を停止します。この結果、このエラーによって他の

numElementsItem プロパティで乗算が発生する項目を指定することを許可しません。

**ユーザーの処置:** VGUI レコードを編集して、指定された fieldName を変更し、配列にならないようにします。乗算が発生する親フィールドの外にフィールドを移動しなければならない場合があります。

エラーが隠される可能性があります。このエラーによって .eglbld ファイルの処理が終了するため、.eglbld ファイル内の、エラーの時点の後で定義される未解決のパーツに関してエラーが発生することがあります。

**ユーザーの処置:** 制御パーツ内で xxxxx がただ 1 つ指定されるように、テキスト・エディターを使用して現行 .eglbld ファイルを編集します。 .eglbld ファイルを保管すると、「問題」ビューのメッセージが更新されます。XML パーサーは、.eglbld ファイル内で最初に重複する属性の箇所で処理を停止するので、ファイル全体の構文解析を行うには、その前にいくつかのエラーを解決する必要があります。

---

**IWN.XML.3999.e** (4 つの代替メッセージのうち 2 つ目)  
**XML 検証エラー - Element・タイプ "xxxxx" の内容は、"(listOfValues)" と一致している必要があります。**

**説明:** VisualAge Generator 内では、リソース関連に対して指定された値のうち 1 つ以上が有効な値です。この値は、EGL によってサポートされていません。マイグレーション・ツールは、無効であってもこの値をマイグレーションするので、ユーザーに問題の解決を促すためのエラーが「問題」ビューに示されます。

**ユーザーの処置:** xxxxx に対して有効なオプションについて、EGL のオンライン・ヘルプを確認してください。使用するオプションを決定したら、必要な変更を行うために、テキスト・エディターでビルド記述子ファイルを開く必要が生じることがあります。

---

**IWN.XML.3999.e** (4 つの代替メッセージのうち 3 つ目)  
**XML 検証エラー - 値 yyyyy を持つ属性 xxxxx には、リスト zzzzz から値が必要です。**

**説明:** VAGen 値すべてが、EGL に対応する置換表現を持っているわけではありません。

**ユーザーの処置:** マイグレーションによって値がどのように変換されるかについて詳しくは、247 ページの『付録 B. VisualAge Generator と EGL の言語エレメントの関係』のリンクージ・テーブルおよびリソース関連のテーブルを参照してください。 EGL で使用可能な選

扱については、EGL 文書を参照してください。

---

#### IWN.XML.3999.e (4 つの代替メッセージのうち 4 つ目)

##### XML 検証エラー - エLEMENT・タイプ

"yyyyy" に関して、属性 "xxxxx" を宣言する必要があります。

**説明:** VAGen 値すべてが、EGL に対応する置換表現を持っているわけではありません。VisualAge Generator で有効な値の組み合わせが EGL では無効になる場合があります。例えば、次のようになります。

- リンケージ・テーブル・パーツでは、ライブラリー属性は localCall エントリーについては無効です。EGL

では、生成された Java コードからネイティブ C++ DLL への呼び出しは、同じマシン上で実行されている場合であっても、リモート呼び出しと見なされず。

**ユーザーの処置:** マイグレーションによって値がどのように変換されるかについて詳しくは、247 ページの『付録 B. VisualAge Generator と EGL の言語エレメントの関係』のリンケージ・テーブルおよびリソース関連のテーブルを参照してください。EGL で使用可能な選択については、EGL 文書を参照してください。

---

## JSP の Java メッセージ

文字定数が無効です。(Invalid character constant)

**説明:** このエラーは以下の状態のときに発生します。

- VGUI レコード用に生成された xxxxxBean.java に関して、setFillCharacter メソッドが値「nullFill」を使用しているとき。VisualAge Generator では、共用データ項目に 2 つのプロパティ・セットがあります。1 つはマップ用で、もう 1 つは UI レコード用です。EGL では、dataItem パーツに 1 つのプロパティ・セットがあります。マイグレーション・ツールは、2 つのプロパティ・セットをマージし、UI プロパティを優先します。ただし、マイグレーション・ツールは、データ項目パーツがマップ、UI レコード、あるいはその両方のいずれで使われるかを予測できません。データ項目で UI のデフォルト充てん文字が指定されなかったため、マイグレーション・ツールはマップのデフォルト文字 (Null) を使用しました。UI レコードで共用データ項目が使用された場合、UI 充てん文字が指定されていなかったため、VisualAge Generator はブランクをデフォルト充てん文字として使用していました。データ項目パーツを VGUI レコードの型定義として使用する場合、EGL は Null の充てん文字を使用して生成しますが、これは無効です。

**ユーザーの処置:** setFillCharacter メソッドに値「nullFill」を指定したために問題が発生した場合は、VGUI レコードを編集して、項目の指定変更プロパティを追加し、fillCharacter=" " を設定してください。これにより、nullFill は、EGL フォームで使用するためのデフォルトの fillCharacter として保存されます。

---

## 付録 F. 外部ソース形式の誤りが原因で EGL の作成に問題が生じる状態

VisualAge Generator 4.5 フィックスパック 5 には、以下に記載される内容を含む、マイグレーション・ツールに必要なすべての APAR が含まれる予定です。

ステージ 1 マイグレーション・ツールの実行後に作成された外部ソース形式が原因で、EGL を作成するマイグレーション・ツールの実行時に問題が生じる可能性があります。こうした状態が起こることは非常にまれですが、ここではこのような状態について説明します。

- データ項目パーツ

- 無効な外部ソース形式が原因で、マップ範囲編集 (最小値と最大値) がステージ 2 で例外を引き起こすことがあります。この問題は、外部ソース形式が VisualAge Generator にインポートされ、データ項目が編集されなかった場合に発生します。この外部ソース形式がマイグレーションのステージ 1 でエクスポートされたとき、マップ範囲編集は無効な形式になっています。VisualAge Generator 内でデータ項目を変更して、項目を保管します。例えば、項目記述のブランクの追加や除去を行います。あるいは、**マイグレーションのステージ 1 を実行する前に**、VisualAge Generator の APAR PQ75621 または APAR PQ79914 の修正をインストールします。
- VisualAge Generator on Java の場合は、名前のいずれかに # 記号が使用されていると、類似したパーツ名をもつデータ項目が原因で、予測不能な結果が生じることがあります。例えば、DATAITEM#A、DATAITEM@A、および DATAITEM\$A が存在する場合は、DATAITEM#A に対して、誤った外部ソース形式コードがマイグレーション・データベースに格納される可能性があります。データ項目名の中で # 記号を使用している場合は、作成された EGL ソース・コードを検討して、情報が正しくマイグレーションされたことを確認してください。あるいは、**ステージ 1 を実行する前に**、VisualAge Generator の APAR PQ85794 の修正をインストールします。

- レコード・パーツ

- SQL レコードの場合、デフォルト・キー項目のフィールドに無効な (または印刷不能) 文字があると、問題が生じる可能性があります。ステージ 2 の実行中に「NoSuchElementException」というメッセージが出された場合は、VisualAge Generator 内でレコードを変更し、デフォルト・キー項目フィールドを選択して、印刷不能文字を削除してください。レコードを保管して、ステージ 1 を再実行します。あるいは、**マイグレーションのステージ 1 を実行する前に**、VisualAge Generator の APAR PQ89390 の修正をインストールします。



---

## 付録 G. マイグレーション・データベース

---

### DB2 マイグレーション・データベースの作成

注記のない限り、次の説明は、Java または Smalltalk のどちらからマイグレーションする場合にも関係なく適用されます。Smalltalk からマイグレーションを行う場合であっても、マイグレーションのステージ 2 とステージ 3 を実行する予定のマシン上で、JDBC ドライバー・レベルを設定する必要があります。

#### DB2 7.2 に対する JDBC レベルの設定

マイグレーション・ツールを使用するには、db2java.zip が JDBC 2.0 レベルであることが必要です。DB2 7.2 には、2 つの db2java.zip ファイルが付属しており、1 つは JDBC 1.1 レベル、もう 1 つは JDBC 2.0 レベルです。DB2 7.2 を JDBC 2.0 用に構成するには、次の手順で行います。

1. すべての DB2 プロセスを停止する。
  - a. 「コントロール パネル」にナビゲートし、「管理ツール」->「サービス」を選択する。
  - b. DB2 を停止する必要がある場合があります。最初に停止を試みたときに停止しなかった DB2 プロセスは、存続しています。
2. DOS コマンド・プロンプト・ウィンドウを開き、`usejdbc2.bat` ファイルを含むディレクトリーにナビゲートする。DB2 7.2 のインストール時にデフォルトのインストール・ディレクトリーを使用した場合、ファイルは `%SQLLIB%java12` ディレクトリーにあります。
3. `usejdbc2.bat` ファイルを実行する。
4. 最初のステップで停止したものをすべて始動する。

#### DB2 8.1 以上に対する JDBC レベルの設定

DB2 8.1 以上がインストールされている場合は、db2java.zip ファイルはすでに正しいレベルです。

#### Windows XP 上での DB2 の使用

マイグレーション・ツールは、次のことを必要とします。

- マイグレーション・データベースへのアクセスに使用されるユーザー ID は、ブランクを含んではなりません。
- ステージ 2 および 3 のマイグレーション・ツール・ウィザード内でマイグレーション・セットが表示されるようにするには、Windows のユーザー ID が限定権限でなく管理者権限を持っている必要があります。

#### マイグレーション・データベースの作成

マイグレーション・データベースを作成するには、次の手順で行います。

1. DB2、および DB2 を使用するアプリケーションすべてがシャットダウンされていることを確認する。例えば、VisualAge Generator および EGL 開発環境をシャットダウンします。
2. DB2 コマンド・ウィンドウを開く。
  - Java からマイグレーションを行う場合は、  
`VisualAgeJava-installation-directory\bin\vgmigration` ディレクトリーにナビゲートする。
  - Smalltalk からマイグレーションを行う場合は、  
`VisualAge-Smalltalk-installation-directory` にナビゲートする。
3. ファイル `SetupDatabase.bat` を実行する。このファイルは、同じディレクトリーにあるファイル `createdatabase.sql` を実行し、同じディレクトリー内のファイル `createdatabase.out` に出力を保管します。このファイルは、VGMIG という名前の DB2 データベースを作成し、データベースに接続して、データベース・パラメーターを構成します。データベースの作成には長くて 1 分かかることがあります。必ず、すべてのコマンドの実行が完了するまで待ってください。

注:

- コンソールに最初に表示されるコマンドが実行された結果、エラー・メッセージが出る場合があります。このメッセージは無視できます。このメッセージは、単に VGMIG データベースがまだ存在しないことを示しています。
  - VGMIG 以外の名前を指定してデータベースを作成する場合は、`createdatabase.sql` 内で VGMIG が出現する箇所すべてを、目的のデータベース名に変更する必要があります。また、ステージ 1 から 3 のマイグレーション・ツール設定の中でも、VGMIG を忘れずに変更する必要があります。
  - デフォルトでは、VGMIG データベースはパスワードによって保護されていません。パスワード保護が必要な場合は、データベースがパスワードによって保護されるように変更する必要があります。
4. ファイル `SetupTables.bat` を実行する。このファイルは、同じディレクトリーにあるファイル `createtables.sql` を実行し、同じディレクトリー内のファイル `createtables.out` に出力を保管します。このファイルは、マイグレーション・ツールが必要とするすべての表とビューをマイグレーション・データベース内で作成します。表は、MIGSCHEMA という名前の高位修飾子 (スキーマ) を使用して作成されます。データベースの作成には長くて 1 分かかることがあります。必ず、すべてのコマンドの実行が完了するまで待ってください。

注:

- コンソールに最初に表示されるコマンドが実行された結果、エラー・メッセージが出る場合があります。これらのメッセージは無視できます。これらのメッセージは、単に表とビューがまだ存在しないことを示しています。
- MIGSCHEMA 以外の名前を指定してスキーマを作成する場合は、`createtables.sql` 内で MIGSCHEMA が出現する箇所すべてを、目的のスキ



ーマ名に変更する必要があります。ステージ 1 から 3 のマイグレーション・ツール設定の中で、MIGSCHEMA をすべて忘れずに変更する必要があります。

- マイグレーション・データベースを完全に消去する必要がある場合は、SetupTables.bat ファイルを DB2 コマンド・ウィンドウから再実行できます。

#### 5. DB2 コマンド・ウィンドウを閉じる。

この時点で、マイグレーション・データベース、スキーマ、表、およびビューが作成されました。ここで、ステージ 1 マイグレーション・ツールが使用する設定ファイルを作成できます。Java からマイグレーションを行う場合は、123 ページの『ステージ 1 設定の実行』を参照してください。Smalltalk からマイグレーションを行う場合は、147 ページの『ステージ 1 設定の実行』を参照してください。

---

## マイグレーション・データベースのリセット

マイグレーション・データベースをリセットする必要がある場合は (例えば、名前変更規則を変更したために)、次のいずれかの手法を使用します。

- マイグレーション・データベース内の表をすべて削除して再作成するツールを使用する。

このツールは、次の状況で使用します。

- マイグレーション・プランをすべて削除する必要がある場合。
- Java プロジェクトの複数のバージョンをマイグレーションした場合。
- Smalltalk 構成マップの複数のバージョンをマイグレーションした場合。

表をすべて削除して再作成するツールを実行するには、次の手順で行います。

1. DB2 コマンド・ウィンドウから、**SetupTables.bat** があるディレクトリーにナビゲートする。
  - Java の場合、このディレクトリーはご使用の *VisualAge-for-Java-install-directory¥ide¥vgmigration* です。
  - Smalltalk の場合、このディレクトリーはご使用の *VisualAge-Smalltalk-install-directory* です。

#### 2. SetupTables.bat を実行する。

- 指定したマイグレーション・セットを削除するツールを使用する。少数のマイグレーション・セットのみを削除する必要がある場合は、このツールを使用します。指定したマイグレーション・セットを削除するツールを実行するには、次の手順で行います。
  1. 次のようにして、マイグレーション・データベースから削除する必要があるマイグレーション・セット ID を判別する。
    - a. DB2 コントロール・センターまたは SQL 照会を使用して、CONFIGPLAN 表を検索する。
    - b. 削除する CONFIGPLANNAME を見付ける。
    - c. 指定する必要があるマイグレーション・セット ID は、対応する CONFIGPLANID 列にあります。

2. DB2 コマンド・ウィンドウから、**deletemigsets.bat** があるディレクトリーにナビゲートする。

- Java の場合、このディレクトリーはご使用の *VisualAge-for-Java-install-directory¥ide¥vgmigration* です。
- Smalltalk の場合、このディレクトリーはご使用の *VisualAge-Smalltalk-install-directory* です。

3. 次のいずれかの形式を使用して、**deletemigsets.bat** ファイルを実行する。

- ただ 1 つのマイグレーション・セットを削除する場合は、次の形式を使用する。

```
deletemigsets n
```

ここで、*n* は削除するマイグレーション・セット ID です。

- 複数のマイグレーション・セットを削除する場合は、次の形式を使用する。

```
deletemigsets "n1,n2"
```

ここで、*n1* と *n2* は削除するマイグレーション・セット ID です。

---

## DB2 を使用したリモート・データベースのカatalog

マイグレーション・ツールでは、ローカル DB2 データベースを使用するとパフォーマンスが向上します。しかし、リモート・データベースを使用する場合は、このセクションの情報が役立ちます。DB2 上でリモート・データベースをカatalogするには、次の情報が必要です。

- データベースがあるリモート・マシンのホスト名または IP アドレス
- クライアントのポート番号とプロトコル (例: 60000/tcp)
- ノード名 (リモート・マシンを記述する別名。例: db2node)
- データベース名
- データベース別名 (オプション)

DB2 を使用してリモート・データベースへの TCP/IP 接続を確立するには、次の手順で行います。

1. Windows の場合はコマンド・プロンプト・ウィンドウを起動する。
2. ノードをカatalogするには、次のコマンドをすべて 1 行に入力する。

```
db2 catalog tcpip node nodeName remote [ hostName | ipAddress ]  
server [ svcname | portNumber ]
```

*hostName* または *ipAddress* のどちらも入力できます。例えば、ノード db2node 上にある、IP アドレス 9.10.11.123 をもち、ポート番号 60000 を使用するリモート・サーバーをカatalogするには、次のコマンドを入力します。

```
db2 catalog tcpip node db2node remote 9.10.11.123 server 60000
```

3. データベースをカatalogするには、次のコマンドをすべて 1 行に入力する。

```
db2 catalog database databaseName  
[ as databaseAlias ] at node nodeName
```

「as *databaseAlias*」はオプションです。 *databaseAlias* を指定しない場合、別名はデータベース名と同じものになります。 *nodeName* は、ステップ 2 で使用した *nodeName* と同じである必要があります。

例えば、ノード db2node 上での別名が sam1 になるように SAMPLE という名前のリモート・データベースをカタログするには、次のコマンドを入力します。

```
db2 catalog database sample as sam1 at node db2node
```

4. データベースへの接続をテストするには、次のコマンドをすべて 1 行に入力する。

```
db2 connect to databaseAlias use userName using password
```

ステップ 3 で別名 (*databaseAlias*) を指定しなかった場合は、データベース名を使用します。例えば、パスワード db2password を使用するユーザー db2user の別名 sam1 を使用して、データベース SAMPLE に接続するには、次のコマンドを入力します。

```
db2 connect to sam1 user db2user using db2password
```

ステップ 2 でデータベース別名として sam1 を指定しなかった場合は、次のコマンドを入力します。

```
db2 connect to SAMPLE user db2user using db2password
```

5. データベース接続情報が表示されます。

その他の支援情報については、次の Web サイトをご覧ください。

<https://aurora.vcu.edu/db2help/db2i4/frame3.htm#idx>

---

## DB2 を使用したリモート・データベースのアンカタログ

DB2 上でリモート・データベースをアンカタログするには、次の情報が必要です。

- データベース別名、またはデータベース名 (データベースをカタログしたときに別名を指定しなかった場合)。

DB2 を使用してリモート・データベースをアンカタログするには、次の手順で行います。

1. Windows の場合はコマンド・プロンプト・ウィンドウを起動する。.
2. データベースをアンカタログするには、次のコマンドを入力する (ここで、*databaseAlias* はデータベース別名)。

```
db2 uncatalog database databaseAlias
```

例えば、データベース SAMPLE (別名 sam1 が指定されている) をアンカタログするには、次のコマンドを入力します。

```
db2 uncatalog database sam1
```

データベースをカタログしたときにデータベース別名を指定しなかった場合は、データベースの名前を使用します。例えば、データベース別名として sam1 を指定しなかった場合は、次のコマンドを入力します。

```
db2 uncatalog database SAMPLE
```

その他の支援情報については、次の Web サイトをご覧ください。

## 便利な照会

サンプルのステージ 1 マイグレーション・ツールを変更した場合、またはユーザー独自のステージ 1 マイグレーション・ツールを開発した場合は、次の SQL 照会が変更内容の検証に役立つことがあります。

注:

- これらの例は、DB2 コマンド・ウィンドウから実行できます。
- これらの例は、デフォルトのマイグレーション・データベース名 (VGMIG) とデフォルト・スキーマ (MIGSCHEMA) の使用を前提としています。
- 特に注記のない限り、DB2 コマンド全体を 1 行に入力する必要があります。本書で後述するコマンドは、スペースの制約によって複数の行に示されている場合があります。
- これらの例を使用するには、まずデータベースに接続する必要があります。データベースに接続するには、次のコマンドを実行します。

```
db2 connect to VGMIG
```

ステージ 1 マイグレーション・ツールが正しく実行されたかどうか判別するための支援情報を得るには、Java の場合は *VAGen-installation* ディレクトリ、Smalltalk の場合はご使用の *VAGen-installation* ディレクトリにある、次に示す .bat ファイルを実行します。

checkStage1.bat

この .bat ファイルは、次のようにいくつかの照会を実行します。

- データベース内のマイグレーション・プラン名のリスト
- データベース内のパーツの総数
- 外部ソース形式の妥当性、および EGL ファイルへのパーツの配置を検査するその他の照会

最初の 2 つの照会結果は 0 より多く、その他の照会結果は 0 であることが必要です。結果が異なる場合は、ステージ 1 マイグレーション時に問題が発生しており、IBM サポートに連絡する必要があります。

VAGen パーツがマイグレーションされたかどうか判別するには、次のコマンドを実行します。

```
db2 select configplanname, configplanversion, vgpartname, vgparttime, is_migrated  
from migschema.vgpart where vgpartname = 'yourPartName'
```

マイグレーション・データベース内にあるパーツすべての外部ソース形式の先頭数文字を検査するには、次のコマンドを実行します。

```
db2 select vgpartname, cast(vgesfssource as char(n)) from migschema.vgpart
```

*n* は、1 から 256 までの数値で、表示する文字の数です。

マイグレーション・データベース内のパーツが、有効な外部ソース形式タグで始まっていないことを判別するには、次のコマンドを実行します。

```
db2 select vpartname, cast(vgesfsorce as char(n))
      from migschema.vgpart where vgesfsorce not like ':%'
```

*n* は、1 から 256 までの数値で、表示する文字の数です。

ステージ 1 を再実行せずに、マイグレーションのステージ 2 と 3 を再実行する場合に、マイグレーション・データベース内のパーツをすべてリセットするには、次のコマンドを実行します。

```
db2 update migschema.vgpart set is_migrated = 'N', eglsource = NULL,
      eglpartname = NULL
db2 delete from migschema.translation_msgs
```

マイグレーション・データベースをバックアップするには、次のコマンドを実行します。

```
db2 backup database vgmig to x:¥mybackups¥backupName
```

*x:¥mybackups¥backupName* は、バックアップを配置するドライブとディレクトリーです。 *x:¥mybackups¥backupName* の下にいくつかのサブディレクトリーが作成されます。

前にバックアップしたマイグレーション・データベースを復元するには、次のコマンドを実行します。

```
db2 restore database vgmig from x:¥mybackups¥backupName REPLACE EXISTING
```

*x:¥mybackups¥backupName* は、バックアップを配置するドライブとディレクトリーです。





---

## 付録 H. マイグレーション・ツールのパフォーマンス

マイグレーション・ツールのパフォーマンスに影響する要因はさまざまです。例えば、次のような要因があります。

- ステージ 1、2、および 3 の全般的なパフォーマンス
  - メモリー。
  - プロセッサ速度。
  - ローカルまたはリモートの DB2 データベース。リモート・データベースの場合は、ネットワーク接続の速度が重要です。
  - Java プロジェクトとパッケージの数、または Smalltalk 構成マップとアプリケーションの数。
  - パーツの数と、パーツ型別の分布。
  - マイグレーション・セットの数。
  - 演算部の行数。
- ステージ 1 のパフォーマンス
  - マイグレーションを開始する前に Java ワークスペースまたは Smalltalk イメージが新規のものであること。
  - ローカルまたはリモートの Java リポジトリ、または Smalltalk ライブラリー。リモート・リポジトリまたはリモート・ライブラリーの場合は、ネットワーク接続の速度が重要です。マイグレーションを実行するマシンにリポジトリをコピーすることにより、処理速度が向上したお客様の例があります。
  - 名前変更規則の複雑さ。
  - マイグレーション・データベースにマイグレーション・セットがすでに存在するかどうか。別の名前変更規則を使用してマイグレーション・セットを再作成する場合は、ステージ 1 マイグレーション・ツールを使用してオリジナルのマイグレーション・セットを消去するよりも、setuptables.bat を実行して SQL 表を再作成する方が効率的です。SQL 表の再作成は、マイグレーション・データベースに他のマイグレーション・セットがない場合に限り有効です。
- ステージ 2 および 3 のパフォーマンス
  - .ini ファイルの VMArgs オプションの設定。VMArgs オプションが設定されていない場合、ステージ 3 で信頼性のあるワークスペースの更新と再ビルドを実行できません。このオプションを設定する必要があります。このオプションを設定する方法については、169 ページの『始動パラメーター』を参照してください。
  - マイグレーション・オプション。

数多くの要因が関係するため、マイグレーションの実行時間を予測できる具体的な公式は存在しませんが、次に示すセクションでは、マイグレーションのさまざまなステージの所要時間を予測するためのガイダンスを示します。

- プロジェクト、パッケージ、パーツ、およびプログラムの数
- マイグレーション・セットおよびその他のマイグレーション・オプションの数。

- プロセッサ速度
- 演算部の行数
- ステージ 1 の新規 Java ワークスペース

さらに、ディスク・スペース要件の計画に役立つ情報が記載されたセクションがあります。

## プロジェクト、パッケージ、パーツ、およびプログラムの数

次の表に、マイグレーションのさまざまなステージにかかる時間についての情報を示します。これらのテストは、1.0 ギガバイトのメモリと 1.1 ギガヘルツのプロセッサ速度の環境で、Windows 2000 を使用して実行されました。測定は、EGL 6.0.0.1 を対象としたものであり、時間はすべて分単位で表記されています。

表 150. マイグレーション時間に対するマイグレーション・セットのサイズの影響

テスト・ケース	プロジェクトの数	パッケージの数	パーツの数	プログラムの数	ステージ 1 の時間	ステージ 2 の時間	ステージ 3 のファイル書き込みにかかった時間	ステージ 3 の更新またはビルドにかかった時間
1	1	98	18,403	1,246	53	22	177	17
2	1	10	2,771	147	23	5	5	8
3	3	29	2,660	33	8	1	4	1
4	3	44	7003	25	62	9	66	22
5	4	92	10,601	25	16	5	41	8
6	5	35	8,238	100	23	4	29	9
7	474	570	11,280	162	46	10	58	36
8	6	118	15,250	71	91	16	110	31

次に、表 150 に基づく全般的な観察結果を示します。

- テスト・ケース 2 と 3 は、パーツ数は類似していますが、テスト・ケース 2 ではステージ 1 の実行に 3 倍の時間がかかり、ステージ 2 の実行には 5 倍の時間がかかっています。テスト・ケース 2 のステージ 1 と 2 の実行時に、関連パーツの分析を行う対象のプログラム数は 4.5 倍です。このため、プログラム数が実行時間に影響を及ぼしています。
- テスト・ケース 5 と 7 は、パーツ数は類似していますが、テスト・ケース 7 ではステージ 1 に約 3 倍の時間がかかっています。テスト・ケース 7 のステージ 1 では、関連パーツの分析を行う対象のプログラムがはるかに多く存在します。さらに、テスト・ケース 7 のデータ項目数は、テスト・ケース 5 に比べて少数です。VAGen では、データ項目には関連パーツが存在しないので、ステージ 1 マイグレーション・ツールはデータ項目の関連パーツの判別を試みません。また、テスト・ケース 7 のステージ 3 では、ファイルを書き込むために約 1.5 倍の時間がかかっています。関連パーツを分析する対象になる生成可能パーツは、このテスト・ケースの方が多く存在しています。それぞれの生成可能パーツは専用のファイルに保管されるので、書き込むファイルの数が多くなります。

CommonParts.egl ファイルと UnusedParts.egl ファイルに保管されるパーツ数は、テスト・ケース 5 の場合よりも比較的少数です。ファイルは小さく、ファイル数は多くなります。

- テスト・ケース 1 のパーツ数は、テスト・ケース 7 を 50% 上回っていますが、テスト・ケース 7 ではワークスペースの更新と再ビルドにかかる時間が 2 倍長くなっています。テスト・ケース 7 には、ビルド時に分析するプロジェクトとパッケージがはるかに多く存在します。また、テスト・ケース 7 では、ビルド・プロセス中に 162 のプロジェクト・サイクルが検出されているので、このこともビルド時間が長くなる原因となっています。

450 ページの表 150 の情報に基づき、数百あるいは 1000 のプログラムがある場合には、これらのプログラムが複数のサブシステムにわたっていても、これらを単一のマイグレーション・セットとしてマイグレーションすることをお勧めします。このためには、すべてのサブシステムが共通プロジェクトの同じバージョンを使用していて、サブシステムに重複するパーツ名がないことが前提になります。

## マイグレーション・セットおよびその他のマイグレーション・オプションの数

表 151 に、より少ない数のマイグレーション・セットに統合した場合の影響を示します。ステージ 2 の「残りの VAGen パーツをマイグレーションする」オプションと、ステージ 3 の「既存ファイルを上書き」オプションの影響も示しています。表には、3 つの異なる技法を使用してマイグレーションされた VAGen プロジェクトの同じセットが示されています。10 のプロジェクト、228 のパッケージ、19255 のパーツ、および 225 のプログラムがあります。2 つの共通プロジェクトには、7734 のパーツと 41 のプログラムが含まれています。これは、パーツの 40% に相当します。8 つのマイグレーション・セットは、それぞれ 1 つのサブシステムを表し、2 つの共通プロジェクトを含んでいます。8 つのサブシステムには、重複するパーツはありません。したがって、TestCase 10C に示すように、すべてのサブシステムを単一のマイグレーション・セットとしてマイグレーションすることができます。測定は EGL 6.0.1 を対象としたものであり、時間はすべて分単位で表記されています。

表 151. マイグレーション・セットおよびマイグレーション・オプションの数の影響

テスト・ケース	マイグレーション・セットの数	残りの VAGen パーツをマイグレーションする	既存ファイルを上書き	ステージ 1 の時間	ステージ 2 の時間	ステージ 3 のファイル書き込みにかかった時間	ステージ 3 の更新またはビルドにかかった時間
10A	8	なし	なし	575	54	780	
10B	8	あり	あり	575	55	880	
10C	1	あり	あり	219	34	159	

次に、表 151 に基づく全般的な観察結果を示します。

- テスト・ケース 10A および 10B は、同じステージ 1 データベースをステージ 2 とステージ 3 の開始点として使用しています。これら 2 つのテスト・ケースの違いは、ステージ 2 とステージ 3 用に選択されたオプションのみです。
  - ステージ 2 では、「残りの VAGen パーツをマイグレーションする」オプションの変更による影響はほとんどありません。これは、恐らく、未使用パーツの数が比較的少ない (250) ためであると考えられます。
  - ステージ 3 では、新規にマイグレーションされたパーツを既存のファイルにマージする時間は、ファイルをすべて再書き込みする時間より短くなります。「既存ファイルを上書き」オプションを選択した場合、共通プロジェクト内の各ファイルを毎回書き込まなければなりません。「既存ファイルを上書き」オプションを選択解除した場合、マイグレーション・ツールは、新規にマイグレーションされた、既存ファイルにマージすべきパーツがない場合、そのファイルを再書き込みしません。この特定のテスト・ケースでは、ほとんどの共通パーツは最初の 2 つのマイグレーション・セット・セットで使用されます。したがって、追加のパーツをマイグレーションする時間は、各マイグレーション・セットの既存ファイルを再書き込みする時間より短くなります。
  - ステージ 3 でも、一度 import ステートメントが決定されると、未使用パーツを分析する必要がないため、分析時間への影響はわずかです。
- テスト・ケース 10C により、その他の 10 のプロジェクトを示す、ハイレベルの PLP パーツを含む 1 つのプロジェクトが追加されました。この技法により、8 つのサブシステムすべてを単一のマイグレーション・セットとしてマイグレーションできるようになります。
  - ステージ 1 では、各マイグレーション・セットに対して、共通パーツをロードして分析する必要がないため、時間が大幅に節約されます。
  - ステージ 2 では、複数のマイグレーション・セットにおけるパーツ間マイグレーションに対して、共通パーツを分析する必要がないため、ある程度節約されます。ただし、マイグレーション・ツールはすべてのパーツを変換するため、ステージ 1 やステージ 3 の場合ほど大幅には節約されません。
  - ステージ 3 では、以下のいくつかの要素の組み合わせにより、時間が大幅に節約されます。
    - 共通パーツは、その import ステートメントを判別するために 1 度しか分析されない。
    - EGL ファイルは、1 度しか書き込まれない。
    - すべてのパーツが同時にマイグレーションされるため、ファイルに関するマージ・ロジックが必要がない。

---

## プロセッサ速度

453 ページの表 152 は、プロセッサ速度を 1.1 ギガヘルツから 1.6 ギガヘルツに変更した場合の影響を示しています。どちらのマシンも、1.0 ギガバイトのメモリーを搭載しています。1.1 ギガヘルツのマシンは Windows 2000 を使用し、1.6 ギガヘルツのマシンは Windows XP を使用しました。プロセッサ速度は、特にステージ 1 と 2 の処理時間に大きな影響を及ぼします。測定は、EGL 5.1.2 を対象としています。

表 152. プロセッサ速度がマイグレーション時間に及ぼす影響

テスト・ケース	1.1 ギガヘルツ				1.6 ギガヘルツ			
	ステージ 1 の時間 (時)	ステージ 2 の時間 (時)	ステージ 3 のファイル書き込みにかかった時間 (時)	ステージ 3 のファイルの更新またはビルドにかかった時間 (時)	ステージ 1 の時間 (時)	ステージ 2 の時間 (時)	ステージ 3 のファイル書き込みにかかった時間 (時)	ステージ 3 のファイルの更新またはビルドにかかった時間 (時)
10	1.2	1.0	0.4	0.2	0.4	0.3	0.3	0.1
11	3.0	1.6	1.8	0.9	1.6	0.6	0.9	0.9

表 152 に基づき、マイグレーション時にはプロセッサ速度の高いマシンを使用することをお勧めします。

## 演算部の行数

ある一時期、VisualAge Generator には一連のブランク行が関数に挿入されるという問題がありました。場合によっては、32,000 行ものブランク行が挿入されていました。これらの余分なブランク行は、パフォーマンスに重大な影響を及ぼします。表 153 に、関数の行数がマイグレーション時間に及ぼす影響を示します。このテスト・ケースのプロジェクト数は 2、パッケージ数は 17、パーツ数は 919、プログラム数は 87 です。497 の関数が存在し、そのうち 8 つに大量のブランク行が含まれていました (30,000 行以内と考えられる)。測定は、EGL 5.1.2, を対象としたものであり、時間はすべて分単位で表記されています。

表 153. 関数の行数がマイグレーション時間に及ぼす影響

テスト・ケース	VAGen 内でブランク行を除去する前				VAGen 内でブランク行を除去した後			
	ステージ 1 の時間	ステージ 2 の時間	ステージ 3 のファイル書き込みにかかった時間	ステージ 3 のファイルの更新またはビルドにかかった時間	ステージ 1 の時間	ステージ 2 の時間	ステージ 3 のファイル書き込みにかかった時間	ステージ 3 のファイルの更新またはビルドにかかった時間
12	40	25	1	3	12	11	1	3

マイグレーションの開始前に余分なブランク行を除去しただけで、ステージ 1 と 2 の処理時間には劇的な違いが生じます。多数のブランク行がある関数が分かっている場合は、マイグレーション前にブランク行を除去する必要があります。ただし、VisualAge Generator 内でこの問題が発生することはまれであるため、マイグレーションの前にこうした関数を探すことはコスト面で効率的でないことが考えられます。連続するブランク行が 3 行を超える場合、マイグレーション・ツールはステージ 2 マイグレーション時に余分なブランク行を自動的に除去します。このため、ステージ 3 では処理時間に変化はありません。これらのブランク行が除去されるので、EGL ではパフォーマンスが向上します。

## ステージ 1 の新規 Java ワークスペース

次の表に、ステージ 1 マイグレーションの開始時に新規ワークスペースを使用した場合の影響を示します。テスト・ケース 13 には、3 つのプロジェクト、29 のパッケージ、2660 のパーツ、および 33 のプログラムが含まれています。テスト・ケース 14 には、マイグレーション・セットのバージョンが 7 つあります。最初のバージョンのプロジェクト数は 3、パッケージ数は 4、パーツ数は 30、プログラムはありません。最後のバージョンのプロジェクト数は 6、パッケージ数は 11、パーツ数は 66、プログラム数は 7 です。測定は、EGL 5.1.2, を対象としたものであり、時間はすべて分単位で表記されています。

表 154. 新規 Java ワークスペースがマイグレーションに及ぼす影響

テスト・ケース	新規ワークスペースを使用しない場合のステージ 1 の時間	新規ワークスペースを使用する場合のステージ 1 の時間
13	17	12
14	11	1

表 154 に基づき、VisualAge Java からマイグレーションを行う場合は、新規ワークスペースから始めることを検討してください。新規 Java ワークスペースからの開始方法について詳しくは、135 ページの『パフォーマンスの向上』を参照してください。

VisualAge Smalltalk の場合も、同様に時間が節約される可能性があります。このため、VisualAge Smalltalk からマイグレーションを行う場合も、新規イメージから始めることを検討してください。新規 Smalltalk イメージからの開始方法について詳しくは、158 ページの『パフォーマンスの向上』を参照してください。

## ディスク・スペース要件

VisualAge Generator アプリケーションのディスク・スペース要件と、対応する EGL アプリケーションのディスク・スペース要件との間に、直接の関係はありません。

次のテーブルには、450 ページの表 150 で示したものと同一テスト・ケースのディスク・スペース要件が記載されています。EGL 測定が行われたのは、すべてマイグレーション直後であり、「問題」ビューのメッセージの収集や生成を実行する前です。測定は、EGL 6.0.0.1 を対象としたものであり、サイズはすべてメガバイト (MB) 単位で表記しています。

表 155. ディスク・スペース要件

テスト・ケース	VAGen .dat ファイル・サイズ	ステージ 1 の DB2 パックアップ・ファイル・サイズ	EGL プロジェクト交換ファイル	EGL ワークスペース・サイズ	EGL .metadata ディレクトリー・サイズ	EGL ワークスペース・サイズの合計
1	25.5	124.0	6.4	40.1	12.9	53.0
2	3.5	48.0	0.5	3.6	9.3	12.9
3	3.0	36.0	0.5	3.3	8.5	11.8



表 155. ディスク・スペース要件 (続き)

テスト・ケース	VAGen .dat ファイル・サイズ	ステージ 1 の DB2 バックアップ・ファイル・サイズ	EGL プロジェクト交換ファイル	EGL ワークスペース・サイズ	EGL .metadata ディレクトリー・サイズ	EGL ワークスペース・サイズの合計
4	14.9	100.0	2.1	15.6	14.1	29.7
5	11.1	60.0	1.0	8.0	10.2	18.2
6	8.0	60.0	1.0	6.6	9.9	16.5
7	33.5	12.0	6.0	37.5	58.0	95.5
8	13.2	132.0	1.5	12.8	10.7	23.5

次に、454 ページの表 155 に基づく全般的な観察結果を示します。

- VAGen .dat ファイルのサイズは、エクスポートされた VisualAge Java リポジトリ・ファイルのサイズです。このリポジトリ・ファイルには、マイグレーション・セットからの VAGen プロジェクトとパッケージのみが含まれています。
- DB2 バックアップ・ファイル・サイズは、ステージ 1 終了時のマイグレーション・データベース用バックアップ・ファイルのサイズです。ステージ 2 のサイズは、データベースに EGL ソース・コードを追加した結果、さらに大きくなります。
- EGL プロジェクト交換ファイルのサイズは、VAGen .dat ファイル・サイズよりもかなり小さくなります。
- テスト・ケース 2、3、4、7、および 8 の VAGen .dat ファイル・サイズおよび EGL ワークスペース・サイズは、ほぼ同じです。したがって、EGL ワークスペースのサイズを計画する際、1:1 の比率が、経験法則から有効と考えられます。ただし、テスト・ケース 1 の EGL ワークスペース・サイズは、VAGen .dat ファイル・サイズの約 1.5 倍となります。テスト・ケース 1 にはさらに多くのプログラムがありますが、他のテスト・ケースに比べて全般的にファイルが小さくなっています。したがって、大量のファイルを持つと予想される場合、スペースに余裕を持たせる必要があります。
- ほとんどのテスト・ケースの EGL .metadata ディレクトリー・サイズは、15 MB 未満です。ただし、テスト・ケース 7 では、.metadata ディレクトリーのサイズはそれより大きくなっています。これはおそらく、テスト・ケース 7 の場合、プロジェクトとパッケージがさらに大量に存在したためと思われます。したがって、大量のプロジェクトおよびパッケージを持つと予想される場合、.metadata ディレクトリー用のスペースに余裕を持たせる必要があります。例えば、VAGen で 1 つのプロジェクト当たり 1 つのプログラムでソース・コードを編成した場合、これに相当するさらに大きなスペースを .metadata ディレクトリー用に計画する必要があります。



---

## 付録 I. 前のバージョンのマイグレーション・ツールを使用してマイグレーションを行った場合に必要な変更

前のバージョンのマイグレーション・ツールを使用して VisualAge Generator から EGL にマイグレーションを行った場合、手動で変更を行わなければならない場合があります。以下のセクションでは、必要な変更について説明します。

- 一般的な変更
- IMS および DL/I サポートによる変更
- Web トランザクション・サポートによる変更

---

### 一般的な変更

EGL 6.0.1 では、以下の一般的な変更が行われています。

- @ 記号による変更。
- 一部の EZE スtring関数用の追加 EGL 置換表現。
- 必要となる可能性がある追加の変更について、オンライン・ヘルプ・トピック の EGL 対 EGL マイグレーションも参照してください。

#### @ 記号による変更

**VisualAge Generator:** パーツ名および非共用項目名は、先頭を @ 記号にすることができます。

**6.0.1 より前の EGL:** VisualAge Generator 互換モードでは、パーツ名および変数名の先頭を @ 記号にすることができます。

**EGL 6.0.1:** VisualAge Generator 互換モードでも、パーツ名および変数名の先頭を @ 記号にすることはできません。

**必要な変更:** 先頭が @ 記号であるすべてのパーツ名または変数名を変更します。

#### 一部の EZE スtring関数用の追加 EGL 置換表現

**VisualAge Generator:** VisualAge Generator は EZE スtring関数をサポートします。VisualAge Generator は、一部の EZE スtring関数で、文字引数としての数値項目の使用を許容します。

**6.0.1 より前の EGL:** マイグレーション・ツールは、EZE スtring関数を同等の EGL システム関数に変換します。ただし、これらの関数は文字データをサポートすることのみを意図しています。

**EGL 6.0.1:** EGL は 3 つの新規関数を追加して、文字引数としての数値項目の使用をサポートします。これらの関数により、VAGen カスタマーのマイグレーションを対象とするサポートが改善されています。オリジナルの EGL 関数も、依然とし

て使用可能です。マイグレーション・ツールは、常に新規 EGL 関数への変換を行います。これは、これらの関数により VAGen の振る舞いに関する最良のマッチングが提供されるためです。

**以前にマイグレーションしたパーツに必要な変更:** 影響を受けるオリジナルの VisualAge Generator EZE スtring関数、EGL 6.0.1 より前のマイグレーション・ツールによって作成される EGL ソース、および EGL 6.0.1 用のマイグレーション・ツールによって作成される EGL ソースに関しては、458 ページの表 156 を参照してください。EGL 関数名は、影響を受ける EGL 関数の 1 つに関する無効な引数について「問題」ビューにメッセージが表示される場合にのみ変更する必要があります。例えば、以下のようなメッセージが表示される場合があります (ここで、*eglFunctionName* は、458 ページの表 156 の中央列にリストされる関数の 1 つです)。

- IWN.VAL.6681.e *functionName* - 関数 *eglFunctionName(parameterTypeList)* は引数 (*argumentTypeList*) に適用されません。(The function *eglFunctionName(parameterTypeList)* is not applicable for the arguments (*argumentTypeList*).)
- IWN.VAL.7553.e *functionName* - *eglFunctionName* の引数 *n* は、String 項目、String 定数、または String・リテラルであることが必要です。(Argument *n* for *eglFunctionName* must be a string item, string constant, or string literal.)

表 156. EGL 6.0.1 で変更された EZE String関数

VAGen 言語エレメント	6.0.1 より前の EGL	EGL 6.0.1
EZESCMPR	strLib.compareStr	VGLib.compareBytes
EZESCNCT	strLib.concatenate	VGLib.concatenateBytes
EZESCOPY	strLib.copyStr	VGLib.copyBytes

## IMS および DL/I サポートによる変更

EGL は、IMS ランタイム環境および DL/I 入出力をサポートするようになりました。IMS または DL/I に関連する言語エレメントを持つパーツを以前にマイグレーションした場合、以下に対して変更を行う必要がある場合があります。

- プログラム・パーツ
- PSB パーツおよび DL/I セグメント・レコード
- 関数入出力 - PSB 名、データベース ID、親のスキャン、更新のスキャン、および SSA
- EZEDL\* 特殊機能語および CSPTDLI サービス・ルーチン
- 生成オプション・パーツ
- リンケージ・テーブル・パーツ
- リソース関連パーツ

## プログラム・パーツ

**VisualAge Generator:** VisualAge Generator は、IMS ランタイム環境で使用するため、あるいは、MVS または VSE ランタイム環境で DL/I データベースとともに使

用するために、PSB を指定するプログラムをサポートします。VisualAge Generator は、プログラムの PSB で指定されているすべての DL/I セグメント・レコードを自動的にプログラムの関連と見なします。VisualAge Generator は、呼び出し先プログラムに対する、PSB (EZEDLPSB) または PCB (EZEDLPCB[n] (ここで  $n$  は数値リテラル)) の引き渡しをサポートします。

**6.0.1 より前の EGL:** マイグレーション・ツールは、プログラムにコメントとして PSB 情報を組み込みます。このツールは、PSB からの DL/I セグメント・レコードをプログラムのレコード宣言リストに追加しません。このツールは、EZEDLPSB または EZEDLPCB[n] を参照するプログラム・パラメーターをコメント化します。

**EGL 6.0.1:** EGL は、IMS ランタイム環境で使用するため、あるいは z/OS ランタイム環境で DL/I データベースとともに使用するために、PSB を指定するプログラムをサポートします。EGL では、プログラムの PSB で指定されるすべての DL/I セグメント・レコードが定義される必要があります。EGL は、PSB または PCB を受け取るためのプログラム・パラメーターもサポートします。マイグレーション・ツールは、EGL 構文への変換を行います。

**以前にマイグレーションしたパーツに必要な変更:** プログラムの PSB の指定に関するヘルプについては、300 ページの表 100 を参照してください。EZEDLPSB および EZEDLPCB[n] を正しい EGL 構文に手動で変換するためのヘルプについては、298 ページの表 99 を参照してください。プログラムの PSB で参照されるすべての DL/I セグメント・レコードに関する宣言を必ず追加してください。または、プログラムとその関連を再度マイグレーションします。

## PSB パーツおよび DL/I セグメント・レコード

**VisualAge Generator:** VisualAge Generator は、PSB パーツおよび DL/I セグメント・レコードをサポートします。

**6.0.1 より前の EGL:** マイグレーション・ツールは、PSB パーツおよび DL/I セグメント・レコードを無視します。

**EGL 6.0.1:** EGL は、PSB パーツおよび DL/I セグメント・レコードをサポートします。マイグレーション・ツールは、これらのパーツを変換します。

**以前にマイグレーションしたパーツに必要な変更:** なし。これらのパーツは、以前にマイグレーションされていません。EGL 6.0.1 で提供されているマイグレーション・ツールを使用して、パーツをマイグレーションする必要があります。

## 関数入出力 - PSB 名、データベース ID、親のスキャン、更新のスキャン、および SSA

**VisualAge Generator:** VisualAge Generator は、DL/I セグメント・レコードを関数の入出力オブジェクトとしてサポートします。VisualAge Generator により、ユーザーは DL/I 呼び出しの PSB およびデータベース ID を指定することができます。SCAN 入出力オプションの場合、親のスキャン (GNP) 呼び出しまたは更新のスキャン (GHN) 呼び出しが必要であることを指定することができます。デフォルト SSA を変更することもできます。

**6.0.1 より前の EGL:** マイグレーション・ツールは、PSB、データベース ID、親のスキャン、更新のスキャン、および SSA が指定されていない場合と同じように、DL/I 入出力が含まれる関数をマイグレーションします。これにより、関数ロジックが可能な限り保持されます。変換済み EGL は、VAGen DL/I Call Editor が関数に関して使用されなかった場合に有効です。マイグレーション・ツールは、PSB、データベース ID、親のスキャン、更新のスキャン、または変更済み SSA をコメントとして組み込みません。

**EGL 6.0.1:** EGL は、変更済み SSA の他に、usingPCB、inParent、および forUpdate オプションを含む、関数の入出力用の DL/I セグメント・レコードをサポートします。マイグレーション・ツールは DL/I 入出力関数を同等の EGL 関数に変換します。

**以前にマイグレーションしたパーツに必要な変更:** 6.0.1 より前の EGL リリースでは DL/I 関数を使用することができませんでした。EGL 6.0.1 で提供されているマイグレーション・ツールを使用して、DL/I 関数を再度マイグレーションする必要があります。DL/I セグメント・レコードを必ず組み込んでください。

## EZEDL\* 特殊機能語および CSPTDLI サービス・ルーチン

**VisualAge Generator:** VisualAge Generator は、EZEDL\* 特殊機能語および CSPTDLI サービス・ルーチンをサポートします。

**6.0.1 より前の EGL:** マイグレーション・ツールは、EZEDL\* 特殊機能語および CSPTDLI サービス・ルーチンを、結果として生じる EGL 構文に関する「最適予測」に変換します。これにより、関数ロジックが可能な限り保持されます。この「最適予測」は、EGL の各リリースによって異なります。

**EGL 6.0.1:** EGL は、EZEDL\* 特殊機能語および CSPTDLI サービス・ルーチンの置換表現を提供します。マイグレーション・ツールは現在、EGL 6.0.1 置換表現を使用します。

**以前にマイグレーションしたパーツに必要な変更:** オリジナルの VisualAge Generator EZEDL\* 特殊機能語および CSPTDLI サービス・ルーチン、EGL 6.0.0.1 のマイグレーション・ツールによって作成された EGL ソース、および EGL 6.0.1 に必要な EGL 言語エレメントに関しては、表 157 を参照してください。EZEDL\* 特殊機能語または CSPTDLI の間違った置換表現を参照するすべての関数を変更する必要があります。

表 157. EZEDL\* 特殊機能語および CSPTDLI サービス・ルーチン

VAGen 言語エレメント	マイグレーション・ツールによって生成される EGL 6.0.0.1 言語エレメント	EGL 6.0.1 言語エレメントおよび必要な変更
EZEDLCER	dliVar.dliCicsErrorCode	dliVar.cicsError
EZEDLCON	dliVar.dliCicsConditionCode	dliVar.cicsCondition
EZEDLDBD	dliVar.dliDbdName	dliVar.dbName
EZEDLERR	dliVar.handleHardDLIErrors	dliVar.handleHardDLIErrors
EZEDLKEY	dliVar.dliKey	dliVar.keyArea[1:dliVar.keyAreaLen]
EZEDLKYL	dliVar.dliKeyLength	dliVar.keyAreaLen
EZEDLLEV	dliVar.dliLevel	dliVar.segmentLevel



表 157. EZEDL\* 特殊機能語および CSPTDLI サービス・ルーチン (続き)

VAGen 言語エレメント	マイグレーション・ツールによって生成される EGL 6.0.0.1 言語エレメント	EGL 6.0.1 言語エレメントおよび必要な変更
EZEDLPCB[n] ここで <i>n</i> は数値リテラルです。	ステートメントでは、マイグレーション・ツールが dliVar.dliPCB[n] への変換を行います。プログラムの呼び出し先パラメーター・リストでは、マイグレーション・ツールは、 dliVar.dliPCB[n] という型定義を持つ dliVar.dliPCB[n] を指定するコメントへの変換を行います。	マイグレーション・ツールは、プログラムの PSB を <i>psb</i> に宣言する変数を常にセットします。このため、ステートメントでは、EZEDLPCB[n] は以下のように変換されます。 <ul style="list-style-type: none"> <li>• EZEDLPCB[0] は psb.iopcb に変換されます。</li> <li>• 1 はデフォルトの添え字であるため、EZEDLPCB は psb.pcb1 に変換されます。</li> <li>• EZEDLPCB[n] (<i>n</i> は数値リテラル) は、psb.pcbn に変換されます。</li> </ul> プログラムの呼び出し先パラメーター・リストでは、特別な考慮事項が適用されます。詳しくは、298 ページの表 99 を参照してください。
EZEDLPRO	dliVar.dliPcbOptions	dliVar.procOptions
EZEDLPSB	ステートメントでは、マイグレーション・ツールは dliVar.dliPsbName への変換を行います。  プログラムの呼び出し先パラメーター・リストでは、マイグレーション・ツールは、dliVar.dliPsbName の型定義を持つ dliVar.dliPsbName を指定するコメントへの変換を行います。	CALL ステートメントを除くステートメントでは、EZEDLPSB は dliLib.psbData.psbName への変換を行います。CALL ステートメントでは、EZEDLPSB は dliLib.psbData への変換を行います。  プログラムの呼び出し先パラメーター・リストでは、特別な考慮事項が適用されます。詳しくは、298 ページの表 99 を参照してください。
EZEDLRST	dliVar.dliCicsProgramRestarted	dliVar.cicsRestart
EZEDLSEG	dliVar.dliSegmentName	dliVar.segmentName
EZEDLSSG	dliVar.dliSegmentCount	dliVar.numSensitiveSegs
EZEDLSTC	dliVar.dliStatusCode	dliVar.statusCode
EZEDLTRM	converseVar.commitOnConverse	converseVar.commitOnConverse
サービス・ルーチン: • CSPTDLI	同等な EGL ルーチン: • dliLib.callDLI	同等な EGL ルーチン: • VGLib.VGTDLI

## 生成オプション・パーツ

**VisualAge Generator:** VisualAge Generator は、IMS 環境に関して生成オプションをサポートします。

- /system=IMSVS および IMSBMP
- /mfsdev
- /spa=size,ADF,bytePosition (ここで、size および bytePosition は数値リテラルです)

- /workdb=DLI および SQL
- 他の IMS 関連生成オプション

**6.0.1 より前の EGL:** マイグレーション・ツールは、ほとんどの IMS 関連生成オプションを、EGL 6.0.1 で有効な EGL ビルド記述子オプションに変換します。これらのオプションはテキスト・エディターで表示することができますが、EGL ビルド・パーツ・エディターでは表示することができません。マイグレーション・ツールは、/mfsdev 生成オプション、および DLI および SQL の /workdb 値をコメント化します。このツールは、/spa オプションを spaSize="size,ADF,bytePosition" に変換します。

**EGL 6.0.1:** EGL は、IMS 関連生成オプションの置換表現を含む、IMSVS および IMSBMP ランタイム環境をサポートします。マイグレーション・ツールは、これらの生成オプションの変換をサポートします。

**以前にマイグレーションしたパーツに必要な変更:** /mfsdev および /workdb 生成オプションに関するコメントの EGL ビルド記述子オプションへの手動での変換についてのヘルプは、359 ページの表 137 を参照してください。さらに、spaSize ビルド記述子オプションを 3 つの新規オプション spaSize="size"、spaADF="YES" または "NO"、および spaStatusBytePosition="bytePosition" に変更する必要があります。または、生成オプション・パーツを再度マイグレーションし、その結果を現行の EGL パーツ定義と比較します。

## リンケージ・テーブル・パーツ

**VisualAge Generator:** VisualAge Generator は、IMS 環境用のリンケージ・テーブル内にある :calllink エントリーの /remoteComType に関して以下の値をサポートします。

- appcims

**6.0.1 より前の EGL:** マイグレーション・ツールは、remoteComType="appcims" への変換を行います。

**EGL 6.0.1:** EGL は remoteComType="APPCIMS" をサポートしません。置換表現の値は IMSJ2C および IMSTCP です。マイグレーション・ツールは、remoteComType="appcims" への変換を行い、可能なかぎり多くのリンケージ・テーブル・エントリーを保持します。

**以前にマイグレーションしたパーツに必要な変更:** IMSJ2C または IMSTCP を使用するかどうかを決定します。EGL remoteComType 属性、および EGL で使用可能なオプションについて詳しくは、オンライン・ヘルプを参照してください。

## リソース関連パーツ

**VisualAge Generator:** VisualAge Generator は、IMS 環境で以下のリソース関連オプションをサポートします。

- /system=imsvs または imsbmp
- /filetype=gsam、mmsgq、smsgq、または seq。gsam は MVSBatch に場合に有効で、seq は IMSBMP の場合に有効です。
- /pcbno=*n* (ここで *n* は数値リテラルです)。

**6.0.1 より前の EGL:** マイグレーション・ツールは、サポートされないランタイム環境のリソース関連エントリー、サポートされないファイル・タイプのリソース関連エントリー、または /pcbno が含まれるリソース関連エントリーをコメント化します。

**EGL 6.0.1:** EGL は IMSVS および IMSBMP ランタイム環境をサポートします。EGL は、ファイル・タイプ GSAM、MMSGQ、および SMSGQ もサポートします。GSAM は zosBatch でサポートされています。SEQ は IMSBMP の場合に有効です。マイグレーション・ツールは、新規 /system 値、/filetype 値、および /pcbno オプションの変換をサポートしています。

**以前にマイグレーションしたパーツに必要な変更:** リソース関連コメントを EGL 関連エントリーに手動で変換するためのヘルプは、388 ページの表 143 を参照してください。または、リソース関連パーツを再度マイグレーションし、その結果を現行の EGL パーツ定義と比較します。

---

## Web トランザクション・サポートによる変更

Web トランザクション・サポートの EGL への追加により、以下の変更が行われました。Web トランザクションに関連した言語エレメントを持つパーツを以前にマイグレーションした場合、以下に対して変更を行う必要がある場合があります。

- DataItem パーツ - ヘルプおよびラベルのテキスト
- Web トランザクション・プログラムおよび UI レコード・パーツ
- XFER ステートメント
- 生成オプション・パーツ

### DataItem パーツ - ヘルプおよびラベルのテキスト

**VisualAge Generator:** VisualAge Generator は、UI レコードで使用されるデータ項目のヘルプおよび UI ラベルのテキストをサポートします。

**6.0.1 より前の EGL:** マイグレーション・ツールは、ヘルプおよびラベルのテキストを、EGL ヘルプおよび displayName プロパティの「最適予測」に変換します。ヘルプまたはラベルのテキストに複数の行がある場合、マイグレーション・ツールはその行を連結し、テキストの最終行を除く各行に、復帰および改行用の ¥r¥n を組み込みます。

**EGL 6.0.1:** EGL は、DataItem パーツのヘルプおよび displayName テキストをサポートします。ヘルプ・テキストに複数の行がある場合、マイグレーション・ツールはその行を連結しますが、テキストの最終行を除く各行に復帰および改行用の ¥r¥n を組み込みません。これは、ヘルプ・テキスト内の JavaScript に関するサポートを提供します。UI ラベルのテキストに複数の行がある場合、マイグレーション・ツールはその行を連結しますが、テキストの最終行を除く各行に改行用の ¥n を組み込みます。

**以前にマイグレーションしたパーツに必要な変更:** DataItem パーツを変更します。ヘルプ・プロパティに関して指定された行が複数ある場合、最終行を除く各行末にある ¥r¥n を削除します。displayName プロパティに関して指定された行が複数ある場合、最終行を除く各行について、¥r¥n を ¥n に変更します。

## Web トランザクション・プログラムおよび UI レコード・パーツ

**VisualAge Generator:** VisualAge Generator は、Web トランザクション・プログラムおよびユーザー・インターフェース (UI) レコードをサポートします。

**6.0.1 より前の EGL:** マイグレーション・ツールは、Web トランザクション・プログラムまたは UI レコードを無視します。

**EGL 6.0.1:** EGL は、VGWebTransaction プログラムおよび VGUI レコードをサポートします。マイグレーション・ツールは、これらのパーツを変換します。

**以前にマイグレーションしたパーツに必要な変更:** なし。これらのパーツは、以前にマイグレーションされていません。EGL 6.0.1 で提供されているマイグレーション・ツールを使用して、パーツをマイグレーションする必要があります。

## XFER ステートメント

**VisualAge Generator:** VisualAge Generator は、マップまたは UI レコード付きの XFER をサポートします。

**6.0.1 より前の EGL:** マイグレーション・ツールは、マップ付きの XFER を EGL `show` ステートメントに変換し、(「最適予測」として) UI レコード付きの XFER を EGL `forward` ステートメントに変換します。これにより、ロジックが可能なかぎり保持されます。

**EGL 6.0.1:** EGL は、フォーマットと VGUI レコードの両方について `show` ステートメントを使用します。マイグレーション・ツールは、マップまたは UI レコード付きの XFER を `show` ステートメントに変換します。

**以前にマイグレーションしたパーツに必要な変更:** 関数を編集し、`forward` ステートメントを `show` ステートメントに変更します。

## 生成オプション・パーツ

**VisualAge Generator:** VisualAge Generator は、Web トランザクション・プログラムの以下の生成オプションをサポートします。

- /genuirecords
- /javadestdir、/javadesthost、/javadestpassword、/javadestuid、および /javasystem。
- /genout、/genresourcebundle、/messageTablePrefix、/resourceBundleLocale、および /targnls。

**6.0.1 より前の EGL:** マイグレーション・ツールは、/genuirecords 生成オプションを EGL `genUIRecords="YES"` ビルド記述子オプションに変換します。このツールは、他の Web トランザクション関連の生成オプションをコメント化します。

**EGL 6.0.1:** EGL は、Web トランザクション関連生成オプションの置換表現を含む、VGWebTransaction プログラムおよび VGUI レコードをサポートします。`genVGUIRecords` は、/genuirecords の置換表現です。さらに、COBOL ランタイム環境用の VGWebTransaction プログラムを生成する場合、EGL では、オリジナルの VAGen 生成オプション・パーツを 2 つの EGL パーツに分割する必要があります。1 つは COBOL ランタイム環境での VGWebTransaction プログラムの生成用で、もう 1 つは VGUI レコードの Java 関連出力の生成用です。マイグレーション

ン・ツールは、生成オプション・パーツを 2 つの EGL ビルド記述子パーツに分割するほかに、Web トランザクション関連の生成オプションの変換をサポートします。

**以前にマイグレーションしたパーツに必要な変更:** 生成オプションのコメントを EGL ビルド記述子オプションに手動で変換するためのヘルプは、359 ページの表 137 を参照してください。この表は、VGUIRecords の Java 関連出力を生成するためにどのオプションを 2 次ターゲット・ビルド記述子パーツに分割する必要があるのかも示します。さらに、genUIRecords ビルド記述子オプションを genVGUIRecords に変更する必要があります。または、生成オプション・パーツを再度マイグレーションし、その結果を現行の EGL パーツ定義と比較します。





---

## 特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒106-0032  
東京都港区六本木 3-2-31  
IBM World Trade Asia Corporation  
Licensing

以下の保証は、国または地域の法律に沿わない場合は、適用されません。IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation  
SWS General Legal Counsel  
Department TL3 Building 062  
P. O. Box 12195  
Research Triangle Park, NC 27709-2195

IBM は、本書に含まれる情報の正確性について合理的な努力を行っています。製品と共に本書のソフトコピーが提供される場合は、そのソフトコピー版に含まれる情報が最新で正確です。ただし、本書は現存するままの状態を提供され、IBM はその内容およびここにリストされている製品に関する、または本書の完全性または正確性についてのいかなる保証も提供しません。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

IBM は、本書およびそれに記載の製品を変更する場合があります。

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者をお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があります、単に目標を示しているものです。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

#### 著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。お客様は、IBM のアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

---

## 商標

以下は、IBM Corporation の商標です。

- AIX
- CICS
- DB2
- IBM
- IMS
- iSeries
- MVS
- OS/2
- OS/400
- Rational
- VisualAge
- WebSphere
- z/OS

Intel は、Intel Corporation の米国およびその他の国における商標です。

Java およびすべての Java 関連の商標およびロゴは、Sun Microsystems, Inc. の米国およびその他の国における商標または登録商標です。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

Microsoft®、Windows、および Windows NT® は、Microsoft Corporation の米国およびその他の国における商標です。

UNIX は、The Open Group の米国およびその他の国における登録商標です。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。



# 索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

## [ア行]

暗黙項目 53, 103, 421  
    プログラム内 87  
一般的な関数の EZE ワード 350  
インポート 37, 179, 184, 187, 188, 193, 194, 405, 406  
    外部ソース形式 26  
    ステージ 3 ツール 18  
ウィザード  
    インポート 26, 197  
エクスポート 26, 193, 405, 406  
エラー・メッセージ  
    HPT 397  
    IWN.MIG 405  
    IWN.VAL 433  
    IWN.XML 437

## [カ行]

外部ソース形式 18, 22, 24, 26, 27, 35, 47, 48, 52, 75, 96, 130, 137, 161, 169, 183, 193, 194, 195, 289, 405, 406, 407, 408, 409, 415, 416, 422, 427, 439, 446  
関数 14, 29, 30, 31, 37, 39, 40, 41, 43, 44, 45, 46, 47, 48, 52, 53, 59, 63, 67, 73, 80, 81, 83, 92, 93, 94, 98, 100, 101, 102, 103, 104, 105, 106, 109, 111, 113, 114, 115, 117, 174, 297, 298, 304, 305, 306, 308, 309, 326, 328, 406, 409, 414, 416, 417, 418, 419, 420, 421, 434, 435  
    共通 33, 46, 54  
    名前変更 42, 175, 239, 243  
    入出力 310  
    未確定状態の処理 92  
    DL/I I/O 320  
    DL/I ステートメント 322  
    SQL 52, 173  
    SQL 入出力 96, 311, 314, 318  
関連パーツ 4, 30, 31, 35, 40, 53, 59, 138, 162, 407  
    使用しないマイグレーション 48  
    マイグレーションに使用 47  
関連プログラム・パーツ 87

共通コード 7, 21, 29, 31, 32, 33, 34, 35, 47, 104, 152  
共通パーツ 127, 130, 150, 151  
結果  
    検討 130  
    ステージ 1 137  
        マイグレーション・データベース 161  
    中間 17  
    パイロット・プロジェクト 7  
    マイグレーション 408  
構成マップ 11, 14, 17, 19, 20, 21, 22, 24, 27, 32, 43, 44, 45, 46, 87, 148, 149, 150, 151, 152, 153, 158, 160, 161, 162, 163, 164, 188, 359, 443  
構文 27, 93  
    一般的な関数の例 306  
    一般的な規則  
        VisualAge Generator と EGL の違い 248  
    一般的なテーブルの例 274  
    一般的な表示マップの例 282  
    一般的なプリンター・マップの例 285  
    一般的なプログラムの例 298  
    一般的なレコードの例 257  
    サービス・ルーチンの一般的な例 353  
    ステートメントの例  
        関数呼び出し 328  
    データ項目の例 249  
    テーブル 247  
    プログラムの main 関数の例 304  
    マップ・グループの例 276  
    割り当て、MOVE、MOVEA の例 328  
EGL 4, 18, 29, 30, 35, 46, 175, 187, 203, 247, 397, 427  
    エラー 54  
    厳密さ 29  
    変換 (ステージ 2) 169  
    無効 98  
SET の例 330  
VAGen 31, 54, 175, 405, 406  
XFER の例 341

## [サ行]

サービス・ルーチン 91, 247, 353  
    一般構文 353  
    VisualAge Generator ルーチンおよび同等な EGL ルーチン 353

サブシステム 17, 31, 32, 33, 34, 35, 38, 39, 41, 48, 53, 54, 63, 67, 69, 70, 81, 96, 98, 100, 104, 105, 139, 164, 202  
システム・ライブラリー関数 30, 46, 63, 90, 114, 328, 335  
実行時の違い  
    COBOL  
        マップ 228  
        CALL 228  
        DXFR 228  
        XFER 228  
    Java  
        CALL 228  
        DXFR 228  
        XFER 228  
充てん文字 64, 252, 270, 294  
出力ファイル 27, 29  
上位 PLP プロジェクト 19, 20, 21, 126, 137, 138, 139  
    作成 138  
使用宣言 422  
シンボリック・パラメーター 359, 388, 392, 393  
    パーツ関連 393  
    ファイル関連 394  
    ユーザー定義 395  
ステージ 1 18  
    Java 121  
        実行 136  
        設定 43, 44, 123  
    Smalltalk 145  
        実行 159  
        設定 43, 44, 147  
ステージ 2 18, 169  
    実行 183  
        バッチ・モード 184  
        ユーザー・インターフェース 183  
    設定  
        設定 178  
ステージ 3 18, 187  
    実行 187  
    設定 187  
ステートメント 52, 114, 247, 298, 309, 326  
    一般規則  
        データ項目修飾と数値リテラル 327  
    関数呼び出し 328  
    使用宣言 74, 300, 413  
    入出力 174, 347, 422  
    入出力のあいまいさ 92

## ステートメント (続き)

表示 92  
フロー 297, 304, 343  
未確定状態で作成される 59  
リンク・エディット 392  
レベル 77 項目 103  
割り当て、MOVE、MOVEA 328  
CALL 340, 347  
CALL、DXFR、XFER 398  
call、transfer、show 86, 421  
DXFR 36, 340  
FIND 104  
IF、WHILE、TEST 335  
I/O 59  
print 92  
RETRIEVE、FIND 333  
SET 330  
SETUPD、UPDATE 416  
SQL 99, 100, 305  
XFER 36, 341  
使用 宣言 73  
制御パーツ 22, 27, 43, 179, 357, 358, 398, 425  
生成オプション 25, 40, 42, 43, 53, 357, 359  
バインド制御 25, 42, 43, 393, 426  
リソース関連 25, 43, 357, 388  
リンク・エディット 25, 42, 43, 392, 425, 426  
リンケージ・オプション 25  
リンケージ・テーブル 379  
Calllink 379  
Crtxlink 385  
Dxfirlink 386  
Filelink 384  
リンケージ・テーブル・オプション 357  
生成 31, 34, 41, 47, 49, 417  
テーブル 25  
プログラム 7, 19, 25, 30, 409, 414, 415, 424  
レポート 130, 132, 160, 161  
VisualAge Generator 139  
生成オプション 11, 204, 357, 359, 392, 393, 424  
変換テーブルの値 378  
VisualAge Generator 36  
生成オプション・パーツ 11  
設定 21, 26, 187, 188, 399, 403, 404  
エディター 35, 36  
サンプル・ファイル 123  
推奨 170  
ステージ 1 19, 20, 22, 43, 164  
設定 139, 141  
Java 20  
Java 上での設定 123

## 設定 (続き)

ステージ 1 (続き)  
Smalltalk 上での設定 147  
ステージ 2 23, 24, 183  
設定 178  
ステージ 3 25  
設定 193  
単一ファイル・モード 171  
名前変更 175  
必要な EGL 170  
ビルド記述子 35, 36  
ファイル名の取得 157  
リポジトリ・フィルター 162  
ワークベンチ  
設定 169  
SQL 154, 173  
VAGen 構文マイグレーションの設定 311, 314  
VAGen マイグレーション構文の設定 175  
VAGen マイグレーションの設定 42, 171, 196, 318  
設定ファイル 404  
マイグレーション 159  
Java 21  
ソース・コード 3, 7, 17, 18, 22, 23, 24, 25, 26, 29, 33, 52, 138, 162, 201, 203, 225, 353, 433  
検討 203  
パイロット・プロジェクト 4  
Java からの抽出 121, 136  
Smalltalk からの抽出 159

## 〔タ行〕

代替仕様レコード 412, 413  
対話 30, 33, 52, 54, 109, 113, 309  
単一ファイル・マイグレーション  
バッチ・モード 195  
ユーザー・インターフェース 194  
単一ファイル・モード 26, 28, 29, 42, 47, 75, 194, 195, 196, 276, 281, 407, 409, 413, 427  
セットアップ 193  
パーツの配置 193  
マイグレーション 193

### ツールの実行

ステージ 1 23, 159, 439  
Java 136  
Smalltalk 159  
ステージ 2 23, 183  
バッチ・モード 24, 184  
ユーザー・インターフェース 183  
ステージ 3 24, 187  
バッチ・モード 26

データ項目 11, 33, 37, 38, 39, 43, 53, 54, 59, 60, 65, 107, 113, 249, 326, 327, 398, 405, 406, 409, 410, 412, 413, 427, 439  
暗黙 86, 103, 421  
共用 31, 33, 34, 41, 45, 46, 59, 61, 63, 64, 70  
設定 176  
代入ステートメント 103  
名前変更 42, 175, 239, 243  
データベース 9, 17, 18, 19, 22, 33, 93, 94, 130, 138, 141, 178, 179, 182, 183, 187, 359, 398, 403, 421, 424  
データベースの更新 130, 161  
テーブル 31, 37, 42, 53, 59, 60, 69, 95, 139, 274, 413  
データベース 131, 154  
名前変更 239, 243  
FIND ステートメント 104  
RETR ステートメント 105  
テーブルおよび追加レコードのリスト 54, 88, 300, 359  
ディスプレイ 280  
デバッグ  
実行時の違い 226  
マップ 226  
SQL 227  
EZESQLCA 227  
EZESQRRM 227  
EZESQWN6 227  
トレース 359  
メッセージ 397  
レベル 132, 154, 155

## 〔ナ行〕

名前変更 21, 22, 42, 78, 124, 129, 130, 131, 148, 161, 405, 408  
ユーザー出口情報 172  
「名前変更」ページ 129, 153  
名前変更規則 130, 151, 152, 401, 403, 443  
名前変更接頭部 42, 175, 347, 421  
入出力オプション、デフォルト (未変更) の DL/I ステートメントの 322

## 〔ハ行〕

パーツ 21  
ステージ 1、2、3 18, 25  
多数 17  
配置 35, 42  
ステージ 1 から 3 42  
ステージ 1、2、3 27, 42



パーツ (続き)  
  配置 (続き)  
    単一ファイル・モード 27, 193, 194  
    プロジェクト・リスト・パーツ (PLP) 19  
パーツ間マイグレーション 4, 19, 29, 31, 35, 47, 59  
パーツ名 14, 29, 36, 38, 42, 202  
  解決 31, 40, 41  
  競合 53, 91, 173, 176, 357  
  重複 31, 41  
  名前変更 175, 243  
  無効 41, 72, 73, 86, 239, 243, 392, 393, 425, 426  
  VisualAge Generator 405  
配列 36, 270, 289, 328, 335, 344  
  多次元 4  
  動的 4  
  マップ 54  
バインド制御 206  
バインド制御パーツ 206, 210  
  テンプレートとして使用 209  
  プログラム固有 211  
パッケージ 11, 12, 14, 17, 20, 22, 23, 25, 26, 27, 35, 37, 38, 39, 41, 42, 43, 44, 53, 54, 70, 87, 88, 123, 127, 128, 129, 130, 136, 140, 150, 151, 160, 161, 184, 185, 186, 190, 194, 196, 197, 202, 245, 276, 359, 379, 385, 401, 403, 406  
  名前変更 129, 153  
  命名 151  
バッチ・モード 17, 24, 26, 169, 178, 183, 185, 195, 197  
パフォーマンス  
  マイグレーション・ツール 447  
比較値項目  
  DL/I I/O 101  
表示 29, 30, 33, 36, 46, 47, 48, 54, 76, 92, 106, 278, 280, 281, 282, 286, 309, 415, 417, 419, 427  
ビルド記述子 204, 206  
  デバッグ 207  
    デフォルト 213  
  デフォルト 205  
  EGL 210  
ビルド記述子オプション 204, 207, 210, 214, 215  
  検討  
    一般 204  
  リンク・エディット 212  
  bind 211  
  COBOL 生成  
    検討 206  
  genproject 206

ビルド記述子オプション (続き)  
  Java 生成  
    検討 206  
ビルド記述子パーツ 201  
  検討 203  
ビルド・パーツ 40, 53, 358, 359  
ビルド・パス 14, 31, 34, 35, 38, 40, 41, 190, 202  
フィルター 20, 22, 177, 399  
  構成マップ 149, 150  
  バージョン 126  
  バージョン深さ 125, 126, 149, 150  
  バージョン名 125, 127, 149, 150  
  パッケージ 128  
  プロジェクト 125, 126, 128  
  リポジトリ 19, 124, 125, 126, 130, 137, 139, 149, 162, 164  
複数の更新を行う SQL 入出力 100  
付録索引項目 239, 249, 257, 274, 276, 281, 305, 326, 342, 353, 357, 397, 427, 439  
プログラム 29, 30, 31, 33, 35, 42, 43, 44, 45, 47, 48, 54, 59, 65, 77, 88, 94, 108, 139, 202, 297, 298, 300, 304, 398, 406, 421, 422  
  暗黙データ項目 86  
  サンプル 17  
    ステージ 1 ツール 20, 23  
  単一ファイル・マイグレーション 26  
  名前変更 239, 243  
  振る舞い 17, 29  
  プロパティ 35, 37  
  マイグレーションに使用 46  
プロジェクト名 20, 44, 125, 126, 128, 129, 137, 138, 141, 151, 184, 185  
プロジェクト・リスト・パーツ 14  
プロジェクト・リスト・パーツ (PLP) 20, 138  
ヘルプ・マップ 78, 282, 285, 298  
ヘルプ・マップ名 77  
ヘルプ・マップ・グループ 54, 77, 176, 398, 414, 416  
編集関数 30  
編集テーブル 30, 48, 252, 255, 270, 271  
編集ルーチン 30, 45, 47, 48, 63, 80, 255, 294, 297, 398, 409, 414, 415

## [マ行]

マイグレーションの計画 3, 4  
マイグレーション・セット 19, 21, 22, 24, 25, 30, 31, 34, 35, 43, 44, 46, 47, 48, 52, 54, 75, 88, 125, 126, 127, 128, 135, 137, 138, 140, 141, 148, 149, 151, 152, 154, 159, 162, 179, 182, 183, 184, 185, 190, 276, 399, 406, 407, 443

マイグレーション・ツールのパフォーマンス 447  
マイグレーション・データベース 20, 21, 22, 23, 24, 25, 35, 121, 124, 125, 131, 132, 137, 140, 145, 147, 148, 154, 161, 185, 187, 190, 406, 407, 427, 441, 446  
  作成 441  
  テーブル 442, 443  
  ビュー 442, 443  
  リセット  
    テーブル 443  
マイグレーション・フィーチャー 146  
  追加 122  
  ロード 146  
マイグレーション・プラン 19, 20, 21, 22, 124, 125, 129, 130, 132, 137, 138, 141, 148, 153, 155, 160, 161, 164, 398, 443  
  手動作成 139  
  上位構成マップ 162  
  複数 21  
マップ 27, 31, 37, 45, 46, 53, 59, 61, 62, 76, 78, 92, 104, 140, 202, 239, 280, 328, 417, 419, 421, 427  
  一般情報 281  
  印刷 422  
  関数と入出力オプション 309  
  数値ハードウェア属性 81  
  スパン 151  
  代入ステートメント 103  
  定数フィールド 286, 289, 292  
  名前なし変数フィールド 83  
  名前変更 42, 175, 243  
  範囲編集 439  
  表示 30, 33, 46  
    一般構文、マップ・タイプ、およびプロパティ 282  
  プリンター 29, 30, 33, 46  
    一般構文、マップ・タイプ、およびプロパティ 285  
  変数フィールド 30, 47, 48, 80, 286, 289, 292, 294, 296  
    エラー・メッセージ 297  
  未確定状態 73  
  無保護定数 84  
  EZEMSG 347  
  XFER への指定 341  
マップ項目  
  暗黙 86  
  編集ルーチン 63  
  NULL かどうかの検査 108  
マップ編集 64  
マップ名 77  
マップ・グループ 27, 29, 32, 33, 42, 43, 59, 74, 76, 140, 276, 398, 406, 413, 414, 415, 416

マップ・グループ (続き)  
一般構文と浮動域 278  
一般情報 276  
スパン 127, 151  
装置の名前、タイプ、サイズ 280  
名前変更 239  
未確定状態 73  
マップ・グループ・パーツ 11  
マップ・パーツ 11  
マップ・プロパティ  
一般情報 252  
一般編集 252  
エラー・メッセージ 255  
数値の編集 255  
未確定状態 24, 27, 30, 34, 35, 59, 163, 203, 247  
関数 92  
その他のステートメント 103  
データ項目 59  
テーブル 72  
プログラム 86  
マップ・グループとマップ 73  
レコード 65  
EZE ワード 113  
未使用パーツ 21, 127, 130, 150, 151, 152  
メッセージ 22, 61, 69, 76, 81, 85, 98, 100, 137, 146, 161, 186, 194, 197, 252, 271, 282, 285, 286, 289  
警告 132, 155  
重大 132, 154  
情報 132, 155  
ステージ 1 共通 397  
ステージ 2 183, 405  
ステージ 3 179, 187  
トレース 426  
デバッグ 132, 155  
マイグレーション・ツールからの 397  
「問題」ビュー 25, 170, 177, 197, 427  
HPT 397  
IWN.MIG 405  
IWN.VAL 433  
IWN.XML 437  
VisualAge for Java のステージ 1 401  
VisualAge for Smalltalk のステージ 1 403  
「問題」ビュー 12, 30, 40, 42, 48, 52, 53, 54, 63, 69, 74, 76, 85, 86, 94, 96, 104, 105, 106, 107, 108, 109, 170, 195, 202, 203, 209, 359, 379, 384, 385, 407, 416, 422, 426, 427, 433, 437, 438

## [ヤ行]

用語 3  
予約語 22, 24, 27, 35, 42, 239, 269, 274, 282, 285, 298, 398  
テーブル名 72  
プログラム名 86  
EGL  
リスト 239  
formGroup 名 73  
Java  
リスト 245  
SQL 175  
リスト 243  
UI レコード名 71  
予約語リスト 41

## [ラ行]

ライブラリー 14, 148, 150, 158, 379  
管理 4, 6, 7, 8, 14, 46  
Smalltalk 18, 23, 147, 149, 159  
リソース関連 206, 359, 388, 425, 426, 437, 438  
リソース関連パーツ 201  
EGL  
検討 209  
リポジトリ 14, 17, 124, 135, 141, 179  
ソース・コード 4, 6, 7, 8, 18, 24, 25, 359  
Java 18, 21, 23, 136  
リポジトリ管理 14  
リポジトリ・エクスプローラー 12  
リポジトリ・フィルター 150  
リンク・エディット 206  
リンケージ・オプション 206  
リンケージ・オプション・パーツ 201  
検討 207  
レコード 31, 37, 53, 59, 64, 70, 71, 88, 104, 257, 410, 411, 412, 421, 422  
共通 54  
再定義 65, 66  
作業用ストレージ 23, 176  
索引付き 310  
シリアル 310  
相対 310  
代替仕様 68, 69, 259  
代入ステートメント 103  
名前変更 42, 239, 243  
入出力 309  
メッセージ・キュー 310  
ユーザー・インターフェース (UI) 22, 42, 269, 270, 271, 272, 273, 309, 398  
レベル 77 項目 66, 67  
DL/I 265  
SQL 261, 427

レポート 20, 22, 23, 33, 124, 137, 161, 398  
ステージ 1 マイグレーション 141, 154, 155, 157, 161, 179, 188  
ログ・ファイル 20, 22, 24, 27, 137, 161, 185, 186, 194, 196, 197  
ステージ 2 マイグレーション 183  
名前 132, 179  
名前の設定 157  
name 155, 187

## [ワ行]

ワークスペース 4, 6, 11, 12, 18, 24, 25, 33, 34, 36, 38, 70, 122, 134, 139, 140, 141, 170, 177, 184, 185, 186, 188, 193, 196, 359, 400, 408  
新規 135  
重複パーツ 28, 31  
復元 136, 159  
保管 135, 158  
ワークスペースにインポート 179, 182, 183, 185  
ワークベンチ  
設定  
設定 169

## A

AUDIT 230

## C

CALL AUDIT 353  
CALL COMMIT 353  
CALL CREATX 353  
CALL CSPTDLI 353  
CALL EZCHART 353  
CALL RESET 353  
CICS 7, 8, 34, 42, 73, 74, 86, 225, 359, 379, 384, 385, 388, 395, 398, 413, 421  
機能語  
サポートされない、ネイティブ環境 230  
コミットに関する違い 231  
サービス・ルーチン  
サポートされない、ネイティブ環境 230  
サポートされない機能  
ネイティブ環境 230  
リソース関連  
サポートされない、ネイティブ環境 230  
ロールバックに関する違い 231  
CALL CREATX に関する違い 231

## CICS (続き)

### EZE 特殊データ・ワードの違い

EZEAPP 231  
EZEDEST 231  
EZEDESTP 231  
EZELTERM 231  
EZERCODE 231  
EZERT8 231  
EZESEGTR 231  
EZEUSR 231  
EZEUSRID 231

### EZECONCT の違い 231

XFER、DXFR 230

## COBOL 生成

生成とテスト 214

## containerContextDependent プロパティ

14, 31, 34, 35, 40, 41, 54

## D

deleteAfterUse 422

destPort 424

## DL/I I/O

比較値項目 101

## E

evensql 409, 412

## EZE ワード 52, 59, 247, 342

### 一般的な関数

EZEBYTES 350  
EZEC10 350  
EZEC11 350  
EZECOMIT 350  
EZECONV 350  
EZEG10 350  
EZEG11 350  
EZEPURGE 350  
EZEROLLB 350  
EZEWAIT 350

### 一般的な数学関数

EZEABS 351  
EZECEIL 351  
EZEEXP 351  
EZEFLFLOOR 351  
EZEFLFREXP 351  
EZEFLDEXP 351  
EZELOG 351  
EZELOG10 351  
EZEMAX 351  
EZEMIN 351  
EZEMODF 351  
EZENCMPR 351  
EZEPOW 351  
EZEPRSCN 351

## EZE ワード (続き)

### 一般的な数学関数 (続き)

EZEROUND 351  
EZESQRT 351

### オブジェクト・スクリプト

EZESCRPT 353

### 三角関数

EZEACOS 352  
EZEASIN 352  
EZEATAN 352  
EZEATAN2 352  
EZECOS 352  
EZECOSH 352  
EZESIN 352  
EZESINH 352  
EZETAN 352  
EZETANH 352

### 数学 351

### ストリング

EZESBLKT 351  
EZESCCWS 351  
EZESCMPR 351  
EZESCNCT 351  
EZESCOPY 351  
EZESFIND 351  
EZESNULT 351  
EZESSET 351  
EZESTLEN 351  
EZESTOKN 351

### その他のデータ

EZE Aid 347  
EZEAPP 347  
EZECNVCM 347  
EZECONVT 347  
EZEDEST 347  
EZEDESTP 347  
EZEFECD 347  
EZELOC 347  
EZELTERM 347  
EZEMNO 347  
EZEMSG 347  
EZE OVER 347  
EZE OVERS 347  
EZERCODE 347  
EZEREPLY 347  
EZERT2 347  
EZERT8 347  
EZESEGM 347  
EZESEGTR 347  
EZESYS 347  
EZETST 347  
EZEUSR 347  
EZEUSRID 347

### 日時

EZEDAY 346  
EZEDAYL 346

## EZE ワード (続き)

### 日時 (続き)

EZEDAYLC 346  
EZEDTE 346  
EZEDTEL 346  
EZEDTELC 346  
EZETIM 346

### 浮動小数点数学関数

EZEFLADD 352  
EZEFLDIV 352  
EZEFLMOD 352  
EZEFLMUL 352  
EZEFLSET 352  
EZEFLSUB 352

### プログラム・フロー

EZECLOS 342  
EZEFLD 342  
EZERTN 342

### ユーザー・インターフェース

EZEUIERR 352  
EZEUILOC 352

### DL/I

EZEDLCER 345  
EZEDLCON 345  
EZEDLDBD 345  
EZEDLERR 345  
EZEDLKEY 345  
EZEDLKYL 345  
EZEDLLEV 345  
EZEDLPCB 345

### EZELTERM

未確定状態 114

### EZESYS

未確定状態 114

### EZEWAIT

未確定状態 117

### SQL

EZECONCT 344  
EZESQCOD 344  
EZESQISL 344  
EZESQLCA 344  
EZESQRD3 344  
EZESQRRM 344  
EZESQWN1 344  
EZESQWN6 344

EZEDLPCB 90

EZELOC 230

EZEPURGE 230

## F

formGroup 73, 74, 202

## I

import ステートメント 25, 27, 31, 34,  
35, 37, 38, 39, 40, 41, 53, 54, 88, 190,  
202, 276, 281, 359  
isDecimalDigit 81, 292  
IWN.MIG 405  
IWN.VAL 433  
IWN.XML 437

## J

Java 生成  
生成とテスト 215  
Java と C++ の違い  
一般 233  
マップ 234  
EZE 特殊データ・ワード  
EZECONVT 235  
EZERCODE 235  
SQL  
EZESQLCA 234  
EZESQRRM 234  
EZESQWN6 234  
JDBC レベル、設定 441  
JSP  
文字定数が無効です 438

## M

MigPreferences.xml 122, 123, 126, 129,  
136, 145, 147  
サンプル 133, 155

## P

PLN の上書き 160, 164  
PSB 300, 354

## S

SET map PAGE 106  
SQL 9, 29, 249, 257, 359, 411, 417  
項目が NULL かどうかの検査 108,  
421  
ステートメント 59  
ハード・エラー 110  
WHERE 文節 36, 59  
SQL EZE ワード 344  
SQL 行レコード 60  
SQL 照会 443, 446  
SQL ステートメント  
変更済み  
Execution Time Statement Build を  
使用しない 314

SQL ステートメント (続き)

変更済み (続き)

Execution Time Statement Build を  
使用する 318

未変更

Execution Time Statement Build を  
使用しない 311

SQL 入出力 417, 418

実行時ステートメント・ビルド 174

SQL 入出力オプション

ADD 311, 314, 318

CLOSE 311, 314, 318

DELETE 311, 314, 318

INQUIRY 311, 314, 318

REPLACE 311, 314, 318

SCAN 311, 314, 318

SETINQ 311, 314, 318

SETUPD 311, 314, 318

SQLEXEC 311, 314, 318

UPDATE 311, 314, 318

SQL 入出力ステートメント 95

SQL 入出力と SQL 文節の欠落 97

SQL 入出力と !itemColumnName 99

SQL 表 23, 96, 410

SQL 文節

FOR UPDATE OF 314, 318

GROUP BY 314, 318

HAVING 314, 318

INTO 314, 318

ORDER BY 314, 318

SELECT 314, 318

WHERE 314, 318

SQL レコード 52, 411

代替仕様 68

SQL レコード定義 32

## U

UI レコード 45  
名前変更 42

## V

VAGen マイグレーションの設定 185

## W

Windows XP  
DB2 の使用 441





プログラム番号: 5724-J19

Printed in Japan

SD88-7536-02



日本アイ・ビー・エム株式会社  
〒106-8711 東京都港区六本木3-2-12