IBM WebSphere Transformation Extender

Design Studio Introduction

Version 8.1

Note

Before using this information, be sure to read the general information in "Notices" on page 21.

2006

This edition of this document applies to IBM WebSphere Transformation Extender Version 8.1; and to all subsequent releases and modifications until otherwise indicated in new editions.

To send us your comments about this document, email DTX_doc_feedback@us.ibm.com. We look forward to hearing from you.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Chapter 1. Design Studio overview

The Design Studio Introduction documentation introduces the concepts, terminology, and capabilities of the Design Studio. The Design Studio provides the means for developing event-driven application-to-application (A2A) integration, business-to-business (B2B) integration, and consumer-to-business (C2B) application integration.

The Design Studio is the design component of WebSphere Transformation Extender used to model and test your e-business solutions. Use the Design Studio to:

- Define your data.
- Create maps for your data.
- Manage the systems of maps that you create.
- Develop and test maps and systems in the Microsoft Windows environment.

The Design Studio is installed on and operates on Windows platforms. The client components of the Design Studio include the Type Designer, Map Designer, Integration Flow Designer, and Database Interface Designer.

Example files are located in the *install_dir*/examples/dsgnstud directory.

WebSphere Transformation Extender product components

WebSphere Transformation Extender products provide capabilities that enable data integration and transformation. Using the Design Studio client components, you can perform design-time functions.

The Design Studio applications are used to develop maps and systems. WebSphere Transformation Extender runs these maps and systems on the server platforms of your choice, using the execution mode that best fits your integration architecture. Design work can be centralized or distributed. A single Design Studio can serve multiple servers, or multiple Design Studios can be used to create interfaces for a single production environment.

When used in production, the data transformation components can be deployed across a variety of servers on multiple platforms. The mode of execution can be command-driven, event-driven, or application-driven.

Design-time applications (development)

Use the Design Studio client applications at design-time to model, develop, and test your data transformation and application integration maps and systems. You can integrate both content transformation and content-based routing through the Design Studio graphical interface.

- Type Designer. Use to define data objects (including source and target data structures).
- Map Designer. Use to develop maps that define your data transformation logic.
- Integration Flow Designer. Use to design systems of maps that model your data transformation business processes. In addition to managing the workflow for your systems of maps, you can further define your systems by implementing map overrides, selecting server deployment options, and so forth.

- Database Interface Designer. Use to import metadata from relational databases to generate type trees for queries or tables. You can identify characteristics of objects to meet mapping and execution requirements such as update keys and database triggers.
- Command Server or Launcher. Use to test and execute maps in a development environment.

Runtime applications (production)

Use the following applications for the implementation of the systems of maps you designed.

- Command Server or Launcher. Use to run maps and systems in a runtime environment.
- Resource Adapters. Use to integrate with specific types of data sources and targets.

Client components of the Design Studio

The client components of the Design Studio include the Type Designer, Map Designer, Database Interface Designer, and Integration Flow Designer. These designers operate in a Windows environment and provide the graphical interface for managing the integration of applications and business processes.

- Type Designer

  The data object modeling component used to create and manage type trees that define properties for data structures, define containment of data, create data validation rules, and more. The Type Designer includes an importer tool for automatically generating type trees for data that is described in various formats and application structures, such as XML, COBOL, SAP R/3, Tuxedo, and PeopleSoft. The type trees are maintained in the Type Designer, although type trees can also be created using the Type Tree Maker, and the Database Interface Designer.
- Map Designer

  The modeling component used to formulate transformation and business rules. The Map Designer uses definitions of data objects created in the Type Designer as inputs and outputs. The Map Designer provides functionality to specify rules for transforming and routing data. Maps are analyzed, compiled, and tested using the Map Designer.
- Integration Flow Designer

  The modeling component used to define and manage business processes. Use the Integration Flow Designer to define interactions among maps and systems of maps, to validate the logical consistency of workflows, and to prepare systems to run.
- Database Interface Designer

  The modeling component used to import metadata about queries, tables, and stored procedures for data stored in relational databases. The Database Interface Designer identifies characteristics, such as update keys and database triggers, of those objects to meet mapping and execution requirements.

The Design Studio also includes the Type Tree Maker, a scripting tool that automates the capture of metadata from machine-processible sources to create graphical metadata for use in WebSphere Transformation Extender products. The use of the Type Tree Maker is not discussed in this document; refer to the Type Tree Maker documentation for more information.

Chapter 2. Type Designer

The Type Designer is the design component used to specify, define, and manage type definitions in the form of type trees. A type tree is a data dictionary that defines how types are classified. The type tree files you create in the Type Designer are data definition files. Each object identified in your data is defined as a type in your type tree.

The Type Designer is used to create and maintain type trees, although type trees can also be created using the Type Tree Maker, the Importer Wizard, and the Database Interface Designer.

Use the Type Designer to define:

- Properties for data structures.
- Data validation rules.
- Data as text or binary.
- Different character sets.

A type tree describes the syntax, structure, and semantics of your data. The syntax of data refers to its format. The format of data includes tags, delimiters, terminators, and other characters that separate or identify data breaks. The structure of data refers to its composition, including repeating substructures, nested groupings, sequences, and choices. The semantics of data refer to the meaning of the data, including rules for data values, relationships among parts of a large data object, and error detection and recovery.

The Type Designer includes an importer tool for automatically generating type trees for data described in XML, COBOL, and other formats or application data structures.

Unique key combinations in the Type Designer

There are some unique key combinations available in the Type Designer in addition to the standard Microsoft® Windows® navigation keys.

For enhanced accessibility, the following key combinations are available in the Type Designer.

Navigation description	Key combination
To toggle between the type tree window and the properties window	Shift+F2
To toggle between the type tree window and the open type detail windows for categories, groups, and items	Ctrl+Tab
To navigate to each pane of the Type Tree Differences window	Shift+down arrow
To navigate to each tab in the Type Tree Differences window	F8

About type trees

A type tree is like a data dictionary because it contains a collection of type definitions. Type trees have a file name extension of .mtt. Each icon in a type tree identifies a data type. The tree has a root type and other types are connected to the tree through branches. The root type is the base type from which all other types stem, representing the data objects of all the types in the tree.

In the type tree, the data content is not identified as the source or the target data. The input and output designations of the data are specified in the Map Designer. The Type Designer defines the specific properties of the data.

Chapter 3. Map Designer

The Map Designer is the application in the Design Studio that is used to specify data transformation logic in the form of map rules.

The Map Designer uses the definition of data objects created in the Type Designer as inputs and outputs and provides the functionality to specify rules for the transformation and routing of data.

Maps are analyzed, compiled, and tested using the Map Designer. Each map defines input and output specifications. Map rules entered in the Map Designer operate on input data objects and build output data objects.

Unique key combinations in the Map Designer

There are some unique key combinations available in the Map Designer in addition to the standard Microsoft Windows navigation keys.

For enhanced accessibility, the following key combinations are available in the Map Designer.

Navigation description	**Key combination**
To move the focus to the applicable map in the Navigator	Shift+F7
To toggle between the input and output cards that correspond to the object selected in the Navigator	Ctrl+Tab
To navigate to each tab of the Organizer window	F8
To select the first selection button in the Map Reference Selection window	Ctrl+A
To select the second selection button in the Map Reference Selection window	Ctrl+D
To select the third selection button in the Map Reference Selection window	Ctrl+L
To select the fourth selection button in the Map Reference Selection window	Ctrl+E
To navigate to each pane of the Map Source File Differences window.	Shift+down arrow
To navigate to each tab in the Map Source File Differences window	F8
To view the color palette for setting a breakpoint color for the Map Debugger	Move the focus to the Breakpoint Color option and press the spacebar

(**Tools** → **Options** → **Debug**)

Map deployment for different platforms

You build and run maps using the Map Designer on a Windows platform. You can also use the Map Designer to build a map to be run on another platform. Each map can be built for a specific platform, and then executed on that platform to perform the transformation of the data.

Using the Map Designer

To use the Map Designer, you must already have created or generated the type tree(s) that define your data. The Map Designer uses the data object definitions that are stored in those type trees.

After defining data objects and their properties in the Type Designer, define the map in the Map Designer to specify the source of the input and the target of the output. Sources and targets are specified with map cards. Each map can have none or more input cards, and one or more output cards.

The Map Designer is used to:

- Create maps to specify the logic necessary to transform the input data to the desired output data.
- Identify the source and data objects of the input data.
- Identify the target and data objects of the output data.
- Specify and build the output data according to the map rules.
- Provide information about data validation by generating trace files.
- View the run results of the map execution.

Chapter 4. Database Interface Designer

The Database Interface Designer is used to import metadata about queries, tables, and stored procedures for data stored in relational databases. Use the Database Interface Designer to identify characteristics of those objects to meet mapping and execution requirements such as update keys and database triggers.

Use the Database Interface Designer to:

- Specify the databases to use for a source or target.
- Define query statements.
- Automatically generate type trees for queries or tables.

Unique key combinations for the Database Interface Designer

There are some unique key combinations available in the Database Interface Designer in addition to the standard Microsoft Windows navigation keys.

For enhanced accessibility, the following key combinations are available in the Database Interface Designer.

Navigation description	Key combination
To navigate to each pane of the Database Differences window	Shift+down arrow
To navigate to each tab in the Database Differences window	F8
To set the focus on the first item on the left tree view in the Database Differences window	Shift+F7
To select the first selection button in the Set Table Update Key Columns window	Ctrl+A
To select the second selection button in the Set Table Update Key Columns window	Ctrl+D
To select the third selection button in the Set Table Update Key Columns window	Ctrl+B
To select the fourth selection button in the Set Table Update Key Columns window	Ctrl+L
To select the fifth selection button in the Set Table Update Key Columns window	Ctrl+E

Using the Database Interface Designer

After defining database query files and database table connectivity in the Database Interface Designer, define your map in the Map Designer. In map cards, you can specify an input source as either a query or a stored procedure and an output target as either a table or a stored procedure.

The Database Interface Designer is used to support business processes that read from and write to databases under control of a Relational Database Management System (RDBMS). The Database Interface Designer provides runtime database connectivity to the Transformation Server. This connectivity is supported with database-specific adapters or open database connectivity (ODBC). ODBC is a programming interface that enables programs to access data in database management systems that use structured query language (SQL) as a data access standard.

The Database Interface Designer:

- Provides connectivity to existing resources.
- Integrates multiple resources.
- Provides transactional control when resources support transactions.
- Combines transformation capabilities and resource connectivity.

All database access is transactional for resources that support transactions.

Database adapters

The WebSphere Transformation Extender database adapters provide the connectivity to the RDBMS for retrieving input data and routing output data. Database adapters provide the option of using a driver to connect to the platform of your choice so you can automatically create type trees for database queries and tables.

The Windows-based database adapters are installed with the Database Interface Designer as part of the Design Studio. You can also install database adapters on additional systems to provide remote database connectivity.

Note: Non-Windows database adapters do not require the Database Interface Designer if you plan to only use **mtsmaker** without a database/query file (**.mdq**) to generate each type tree. For information about using **mtsmaker**, refer to the Resource Adapters documentation.

Chapter 5. Integration Flow Designer

The Integration Flow Designer is the graphical process-modeling element of the Design Studio. Use the Integration Flow Designer as a modeling component for defining and managing business processes.

The Integration Flow Designer manages collections of maps at design time. Other product components, such as the Command Server or Launcher, manage the execution of maps at run time.

Collections of maps are called systems. Systems can be used as a focal point for interfacing to other development components and for preparing maps for execution.

Maps are created in the Map Designer. The systems created and managed in the Integration Flow Designer can be defined with existing maps, or they can be designed with maps that have not yet been created (pseudo map components). Systems can also be designed with subsystems.

Use the Integration Flow Designer to:

- Define interactions among maps and systems of maps.
- Specify integration workflow.
- Display data flow relationships among system components.
- Validate the logical consistency of workflows.
- Prepare systems to run.
- Verify component relationships.
- Generate process control information.
- Experiment at design time.
- Coordinate multiple data sources and targets.
- Coordinate events.
- Model a variety of what-if scenarios.
- Visualize the scenario results.
- Create and manage collections of logically related maps.
- Specify system deployment servers: Use one of three predefined server definitions or create server definitions that describe the servers in your enterprise.

Unique key combinations in the Integration Flow Designer

There are some unique key combinations available in the Integration Flow Designer in addition to the standard Microsoft Windows navigation keys.

For enhanced accessibility, the following key combinations are available in the Integration Flow Designer.

Navigation description	Key combination
To move the focus to the Navigator	Shift+F7
To navigate to each pane of the System File Differences window	Shift+down arrow

Navigation description	Key combination
To navigate to each tab in the System File Differences window	F8

Tip: In the More Find/Replace window, press `Alt+i` twice to move the focus to the **Find All** button.

Benefits of using the Integration Flow Designer

The graphical representation of systems in the Integration Flow Designer helps you to visualize system designs, what-if scenarios, and system execution behavior. The interface provides easy management of system definition diagrams that allow concentration on the system design. Error reduction is achieved by automation of repetitive tasks and the use of system hierarchy.

Map and system components

The Integration Flow Designer enables you to overwrite settings compiled into maps. Map components of a system provide:

- Visual indication of audit log generation.
- Identification of the source or target type: message, database, file, or application.
- Ability to override map and card settings compiled into a map.
- A method of recording your system design with linked documents.

A map component is an Integration Flow Designer object that represents an executable map and displays its relationship with other components in the same system.

Systems can contain one or more map components. To keep the system manageable, you may create smaller systems and then include them as a subsystem of a larger system. Map components and system components are executable units.

Execution settings specify how the map or subsystem executes. The execution settings for system components are maintained in the Integration Flow Designer and are eventually used on designated servers to execute the system components. Each component in a system has separate execution settings. The referenced files (maps or subsystems) are static and are not modified directly with this information.

A map component that references an executable map has a default set of sources and targets defined within the referenced map. Map settings and card settings for sources and targets are defined in the Map Designer, and may be overwritten in the Integration Flow Designer.

The sources and targets defined in the map are the interface points of the map component used within a system. The sources and targets of executable map components are displayed in the main window of the Integration Flow Designer.

In the Integration Flow Designer, the expanded view of a map component distinguishes its sources from its targets. By default, the sources are displayed to the left of a map component and the targets are displayed to the right.

Chapter 6. Design and execution

The Design Studio components are the design-time applications that are used to define, manage, and test maps and systems in a development environment.

While the Design Studio components are primarily used during the design phase of your transformation project, the Launcher and Command Server are used to execute maps and systems in a runtime environment.

Design-time configuration

Design-time tasks include:

- Creating/generating type trees.
- Modifying type properties.
- Configuring the map settings, including SourceRule settings in input cards and TargetRule settings in output cards. (These configuration settings are compiled into maps.)

Systems can be designed using the Integration Flow Designer before, during, or after the type tree and map design phase.

These design-time tasks are performed on the client components of the Design Studio. Changing these settings at the time of execution are runtime tasks.

Use a platform-specific Launcher or Command Server to execute maps and systems in a runtime environment.

Runtime configuration

Map settings and card settings can be overwritten in two ways:

- Modifying the execution settings of a system component in the Integration Flow Designer.
- At execution time with execution commands.

Execution commands override the map settings or card settings of the compiled map when that map is run. See the Execution Commands documentation for a complete description of the override commands.

Execution settings

Execution settings in the Integration Flow Designer are either Launcher or Command Server settings that are generated for the system. Command Server settings are used when the system is to run by means of the Command Server. Launcher settings are used when the system is to run by means of the Launcher.

Choose execution settings based on the server on which the system is to run. These execution settings overwrite the map and card settings, but do not alter the map source file (.mms) or compiled map file (.mmc).

Chapter 7. Customizing your environment

The Design Studio application toolbars and shortcut keys can be customized to fit your needs. Use the configuration options in each designer application to set preferences for your working environment.

Customizing toolbars

You can customize the standard toolbar or create your own. You can also delete and rearrange tools within a toolbar.

- **Delete a tool from a toolbar**. Press the **Alt** key and drag the tool off the toolbar.
- **Rearrange tools on a toolbar.** Hold down the **Alt** key and drag any tool, placing it in the toolbar where desired.
- **Add tools to a toolbar**. Select **Toolbar(s)** from the **View** menu. When the **Customize** dialog box appears, click the **Commands** tab, drag the tools to the toolbar as desired. You can also delete and rearrange tools; however, you do not need to use the **Alt** key in the **Customize** dialog box.
- **Make a floating toolbar.** Drag a toolbar and place it inside or outside the Type Designer window.
- **Cancel all changes made to a toolbar**. In the **Customize** dialog box, click **Reset**.
- **Customize the appearance of the tools.** You can select options for showing tool tips, viewing the tools as flat (**Cool Look** check box enabled) or raised, and specifying small or large tools on the toolbar (**Large Buttons** check box enabled).

To reset a toolbar to its default configuration

1. From the **View** menu, choose **Toolbar(s)**.
2. From the **Toolbars** tab, select the toolbar you want to reset and click **Reset**.

To create a new toolbar

1. From the **View** menu, choose **Toolbar(s)**.
2. On the **Customize** dialog box, click **New**.
3. Enter a name for the toolbar (for example, `Editing`) and click **OK**.
4. Select the new toolbar in the **Customize** dialog box.
5. The new toolbar appears on your screen.
6. Select the **Commands** tab.
7. Drag each tool that you want into the new toolbar.
8. Click **OK** to save the toolbar when you are finished.

For example, you can drag the **Cut**, **Copy**, and **Paste** tools to the new ″Editing″ floating toolbar.

Arrange your toolbars by dragging them into positions.

To move a toolbar

Drag the toolbar to the new location inside or outside the Type Designer window.

When positioning a toolbar inside the Type Designer window and docking is not desired, press the **Ctrl** key to prevent docking.

Shortcut keys

You can assign your own shortcut actions to keys.

To assign shortcut keys

1. From the **Tools** menu, choose **Short Cuts**.

   The Shortcut Keys dialog box is displayed.
2. In the **Select a macro list** in the **Shortcut Keys** dialog box, select a command to which you want to assign a shortcut key.
3. Click **Create Shortcut**.
4. Enter the keystroke(s) you want to assign. For example, to use the keystroke(s) **Ctrl+Shift+J**, press and hold the **Ctrl** key and the **Shift** key, then press **J**.

   A message displays that indicates whether the shortcut is currently assigned.
5. After you finish assigning shortcuts, click **OK**.

   To view the shortcut assignment for a command, select the command. A description of the command and the assigned shortcut keys are displayed.

To remove a shortcut

1. In the Shortcut Keys dialog box, highlight a shortcut in the **Select a macro** list box.
2. Highlight the shortcut to be deleted from the **Assigned shortcuts** list box.
3. Click **Remove**.
4. To save changes, click **OK**.

To restore all shortcuts to their default configuration

Click the **Reset All** button.

Chapter 8. Reference information

There are some restrictions that apply when using the Design Studio components to design your data transformation process. Certain words and symbols have a specific meaning when used within theDesign Studio applications.

Quotation marks

Although there are platform-specific limitations regarding the usage of double and single quotation marks, when adding a text string into any map rule, component rule, or command line, use the following characters for quotation marks.

Symbol	ASCII Character	Text Reference in Documentation
′	039	Single quotation mark
″	034	Double quotation mark
′′	039 and 039	Two single quotation marks (for some non-Windows environments)

Using any other characters can produce unexpected results.

Reserved words and symbols

Reserved words and symbols have a special meaning and may be used only in the manner described in this chapter. The following is the list of reserved words and symbols, and their intended use.

Reserved Word or Symbol
Intended Use

$ Shorthand representation of the object name to the left of a rule cell. Used in a component rule or a map rule.

: Used to separate components.

:.: Shorthand representation of a unique component or

partition path.

For example, **A:.:B** refers to the unique path from the type **A** to the type **B**.

<> Separates a partition from its partitioned type

, Used to separate arguments of functions and maps

″ ″ Used to enclose literal text strings.

( ) Used for a component range, to enclose function or map arguments, and to block sub-expressions in a rule

@ Separates a comment from a component.

[ ] Denotes an indexed member of a series.

{ } Used to enclose a list of literals.

Names of Functions
Names of functions are reserved words. Names of functions can be used as type names, but cannot be used as map names.

For a complete list of functions, refer to the Functions and Expressions documentation.

Operators
Operators are reserved.

For a complete list of operators, refer to the Functions and Expressions documentation.

ANY Represents one or more types in a component name.

ANY cannot be used in component rules or map rules.

COMPONENT
Refers to any component preceding the current one in the same component list. Used in the Type Designer to refer to preceding components of the same component list.

This reserved word can only follow the reserved word IN. For example:

```
COUNT (Record IN COMPONENT)
```

FALSE
Represents the logical false.

IN Separates a component from an object that contains it. For example, `LineItem IN File` refers to all occurrences of the type **LineItem** in the type **File**.

LAST Special index value that refers to the previous data object of a particular series.

NONE
Stands for the null value or non-existence of data content. Can be used in a component rule to test the presence of data. Can be used in a map rule to generate no occurrences of an output.

<*space*>
Separates type names. (This is an actual space.)

The names of individual types in component names are separated by spaces. Component names themselves are separated by colons. For example,

`C:A B` is the **`C`** component of the **`A`** subtype of **`B`**;

`C:A B:X` is the **C** component of the **`A B`** component of type **`X`**; and so forth.

<*symbol*>
Symbol abbreviations for entering non-printable characters. For example, `<CR>`.

TRUE Represents the logical true.

WHERE
Can be used in the `LOOKUP` or `EXTRACT` function in place of the comma (**,**) between arguments.

Hex, decimal, and symbol values

The following table lists characters, hexadecimal values, decimal values, and symbols.

In this list, some characters in the character name are capitalized to indicate the common abbreviation for the character. For example, the <STX> abbreviation represents the **S**tart of**T**e**X**t character.

The hexadecimal values are provided as a reference. For any literal value specified in the Type Designer Properties window, hexadecimal values must be enclosed in double angle brackets. For example, `<<0A>>` represents the hexadecimal value for a line feed character.

The decimal values are provided as a reference. Decimal values can be used with the SYMBOL function. For any literal value specified in the Type Designer Properties window, decimal values must be enclosed in single angle brackets. For example, **<10>** represents the decimal value for a line feed character.

Symbols can be inserted into map rules and component rules. You can insert them from the Symbols dialog box or by typing the hexadecimal or decimal value.

Character	Hex Value	Decimal Value	Symbol
NewLine	-	-	<NL>
WhiteSPace	-	-	<WSP>
KanjiSPace (WideSpace)	-	-	<KSP>
NULL	00	0	<NULL>
StartOfHeading	01	1	<SOH>
StartofTeXt	02	2	<STX>
EndofTeXt	03	3	<ETX>
EndOfTrans.	04	4	<EOT>
ENQuiry	05	5	<ENQ>
ACKnowlege	06	6	<ACK>
BELL	07	7	<BELL>
BackSpace	08	8	<BS>
HorizTab	09	9	<HT>
LineFeed	0A	10	<LF>
VerticalTab	0B	11	<VT>
FormFeed	0C	12	<FF>
CarriageReturn	0D	13	<CR>
ShiftOut	0E	14	<SO>
ShiftIn	0F	15	<SI>
DataLinkEscape	10	16	<DLE>
DeviceControl1	11	17	<DC1>
DeviceControl2	12	18	<DC2>
DeviceControl3	13	19	<DC3>
DeviceControl4	14	20	<DC4>

Character	Hex Value	Decimal Value	Symbol
NegativeAcK	15	21	<NAK>
SYNchron.Idle	16	22	<SYNI>
EndTransBlock	17	23	<ETB>
CANcel	18	24	<CAN>
EndofMedium	19	25	<EM>
SUBstitute	1A	26	<SUB>
ESCape	1B	27	<ESC>
FileSeparator	1C	28	<FS>
GroupSeparator	1D	29	<GS>
RecordSep.	1E	30	<RS>
UnitSeparator	1F	31	<US>
SPace	20	32	<SP>
!	21	33	-
″	22	34	-
#	23	35	-
$	24	36	-
%	25	37	-
&	26	38	-
′	27	39	-
(	28	40	-
)	29	41	-
*	2A	42	-
+	2B	43	-
,	2C	44	-
-	2D	45	-
.	2E	46	-
/	2F	47	-
0	30	48	-
1	31	49	-
2	32	50	-
3	33	51	-
4	34	52	-
5	35	53	-
6	36	54	-
7	37	55	-
8	38	56	-
9	39	57	-
:	3A	58	-
;	3B	59	-
<	3C	60	-
=	3D	61	-

Character	Hex Value	Decimal Value	Symbol
>	3E	62	-
?	3F	63	-
@	40	64	-
A	41	65	-
B	42	66	-
C	43	67	-
D	44	68	-
E	45	69	-
F	46	70	-
G	47	71	-
H	48	72	-
I	49	73	-
J	4A	74	-
K	4B	75	-
L	4C	76	-
M	4D	77	-
N	4E	78	-
O	4F	79	-
P	50	80	-
Q	51	81	-
R	52	82	-
S	53	83	-
T	54	84	-
U	55	85	-
V	56	86	-
W	57	87	-
X	58	88	-
Y	59	89	-
Z	5A	90	-
[	5B	91	-
\	5C	92	-
]	5D	93	-
^	5E	94	-
_	5F	95	-
`	60	96	-
a	61	97	-
b	62	98	-
c	63	99	-
d	64	100	-
e	65	101	-
f	66	102	-

Character	Hex Value	Decimal Value	Symbol
g	67	103	-
h	68	104	-
i	69	105	-
J	6A	106	-
k	6B	107	-
l	6C	108	-
m	6D	109	-
n	6E	110	-
o	6F	111	-
p	70	112	-
q	71	113	-
r	72	114	-
s	73	115	-
t	74	116	-
u	75	117	-
v	76	118	-
w	77	119	-
x	78	120	-
y	79	121	-
z	7A	122	-
{	7B	123	-
|	7C	124	-
}	7D	125	-
~	7E	126	-
DELete	7F	127	-

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION ″AS IS″ WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
577 Airport Blvd., Suite 800
Burlingame, CA 94010
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries, or both:

AIX
AIX 5L
AS/400
Ascential
Ascential DataStage
Ascential Enterprise Integration Suite
Ascential QualityStage
Ascential RTI
Ascential Software
Ascential
CICS
DataStage
DB2
DB2 Universal Database
developerWorks
Footprint
Hiperspace
IBM
the IBM logo
ibm.com
IMS
Informix
Lotus
Lotus Notes
MQSeries
MVS
OS/390
OS/400
Passport Advantage
Redbooks
RISC System/6000
Roma
S/390
System z
Trading Partner
Tivoli

WebSphere
z/Architecture
z/OS
zSeries

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

This product includes software developed by the Eclipse Project (http://www.eclipse.org/).

IBM WebSphere Transformation Extender, Version 8.1

Index

Printed in USA