

*IBM SPSS Modeler
17.1 Solution Publisher*

IBM

Hinweis

Vor Verwendung dieser Informationen und des darin beschriebenen Produkts sollten die Informationen unter „Bemerkungen“ auf Seite 39 gelesen werden.

Produktinformation

Diese Ausgabe bezieht sich auf Version 17, Release 1, Modifikation 0 von IBM(r) SPSS(r) Modeler und alle nachfolgenden Releases und Modifikationen, bis dieser Hinweis in einer Neuausgabe geändert wird.

Diese Veröffentlichung ist eine Übersetzung des Handbuchs
IBM SPSS Modeler 17.1 Solution Publisher,
herausgegeben von International Business Machines Corporation, USA

© Copyright International Business Machines Corporation 2015

Informationen, die nur für bestimmte Länder Gültigkeit haben und für Deutschland, Österreich und die Schweiz nicht zutreffen, wurden in dieser Veröffentlichung im Originaltext übernommen.

Möglicherweise sind nicht alle in dieser Übersetzung aufgeführten Produkte in Deutschland angekündigt und verfügbar; vor Entscheidungen empfiehlt sich der Kontakt mit der zuständigen IBM Geschäftsstelle.

Änderung des Textes bleibt vorbehalten.

Herausgegeben von:
TSC Germany
Kst. 2877
August 2015

Inhaltsverzeichnis

Vorwort v

Kapitel 1. Informationen zu IBM SPSS

Modeler 1

IBM SPSS Modeler-Produkte 1

IBM SPSS Modeler 1

IBM SPSS Modeler Server 1

IBM SPSS Modeler Administration Console 2

IBM SPSS Modeler Batch 2

IBM SPSS Modeler Solution Publisher 2

IBM SPSS Modeler Server-Adapter für IBM SPSS

Collaboration and Deployment Services 2

IBM SPSS Modeler-Editionen 2

IBM SPSS Modeler-Dokumentation 3

SPSS Modeler Professional-Dokumentation 4

SPSS Modeler Premium-Dokumentation 4

Anwendungsbeispiele 5

Ordner "Demos" 5

Kapitel 2. IBM SPSS Modeler Solution

Publisher 7

Funktionsweise von IBM SPSS Modeler Solution Pub-

lisher 7

Veröffentlichen von Streams 8

Kapitel 3. Ausführen veröffentlichter

Streams 11

Verwenden der Parameterdatei zur Anpassung der

Streamausführung 12

Einbetten von IBM SPSS Modeler Solution Publisher

in Anwendungen 13

Beispielanwendung von IBM SPSS Modeler Solu-

tion Publisher 14

Kapitel 4. Installieren von IBM SPSS

Modeler Solution Publisher Runtime . . . 15

Installation von Runtime 15

Systemanforderungen 15

Installation unter Windows 16

Installation unter UNIX 16

Fehlerbehebung bei Installationen 17

Starten von Runtime 18

Ändern des temporären Verzeichnisses 18

Deinstallation von IBM SPSS Modeler Solution Pub-

lisher Runtime 19

Windows-Deinstallationsprozedur 19

UNIX-Deinstallationsprozedur 19

Kapitel 5. IBM SPSS Embedded Predic-

tive Modeling API 21

Kapitel 6. IBM SPSS Modeler Solution

Publisher Runtime Library - API-Referenz 23

Einführung 23

Überblick über API-Prozesse. 23

API-Funktionen 24

Bemerkungen. 39

Marken. 40

Index 41

Vorwort

IBM® SPSS Modeler ist die auf Unternehmensebene einsetzbare Data-Mining-Workbench von IBM. Mit SPSS Modeler können Unternehmen und Organisationen die Beziehungen zu ihren Kunden bzw. zu den Bürgern durch ein tief greifendes Verständnis der Daten verbessern. Organisationen verwenden die mithilfe von SPSS Modeler gewonnenen Erkenntnisse zur Bindung profitabler Kunden, zur Ermittlung von Cross-Selling-Möglichkeiten, zur Gewinnung neuer Kunden, zur Ermittlung von Betrugsfällen, zur Reduzierung von Risiken und zur Verbesserung der Verfügbarkeit öffentlicher Dienstleistungen.

Die grafische Schnittstelle von SPSS Modeler erleichtert Benutzern die Anwendung ihres spezifischen Fachwissens, was zu leistungsfähigeren Vorhersagemodellen führt und die Zeit bis zur Lösungserstellung verkürzt. SPSS Modeler bietet zahlreiche Modellierungsverfahren, beispielsweise Algorithmen für Vorhersage, Klassifizierung, Segmentierung und Assoziationserkennung. Nach der Modellerstellung ermöglicht IBM SPSS Modeler Solution Publisher die unternehmensweite Bereitstellung des Modells für Entscheidungsträger oder in einer Datenbank.

Informationen zu IBM Business Analytics

Die Software IBM Business Analytics liefert umfassende, einheitliche und korrekte Informationen, mit denen Entscheidungsträger die Unternehmensleistung verbessern können. Ein umfassendes Portfolio aus Anwendungen für Business Intelligence, Vorhersageanalyse, Finanz- und Strategiemangement sowie Analysen bietet Ihnen sofort klare und umsetzbare Einblicke in die aktuelle Leistung und gibt Ihnen die Möglichkeit, zukünftige Ergebnisse vorherzusagen. Durch umfassende Branchenlösungen, bewährte Vorgehensweisen und professionellen Service können Unternehmen jeder Größe die Produktivität maximieren, Entscheidungen automatisieren und bessere Ergebnisse erzielen.

Als Teil dieses Portfolios unterstützt IBM SPSS Predictive Analytics-Software Unternehmen dabei, zukünftige Ereignisse vorherzusagen und proaktiv Maßnahmen zu ergreifen, um bessere Geschäftsergebnisse zu erzielen. Kunden aus Wirtschaft, öffentlichem Dienst und staatlichen Lehr- und Forschungseinrichtungen weltweit nutzen IBM SPSS-Technologie als Wettbewerbsvorteil für die Kundengewinnung, Kundenbindung und Erhöhung der Kundenumsätze bei gleichzeitiger Eindämmung der Betrugsmöglichkeiten und Minderung von Risiken. Durch die Einbindung von IBM SPSS-Software in ihre täglichen Operationen wandeln sich Organisationen zu "Predictive Enterprises", die Entscheidungen auf Geschäftsziele ausrichten und automatisieren und einen messbaren Wettbewerbsvorteil erzielen können. Wenn Sie weitere Informationen wünschen oder Kontakt zu einem Mitarbeiter aufnehmen möchten, besuchen Sie die Seite <http://www.ibm.com/spss>.

Technical Support

Kunden mit Wartungsvertrag können den Technical Support in Anspruch nehmen. Kunden können sich an den Technical Support wenden, wenn sie Hilfe bei der Arbeit mit IBM Produkten oder bei der Installation in einer der unterstützten Hardwareumgebungen benötigen. Zur Kontaktaufnahme mit dem Technical Support besuchen Sie die IBM Website unter <http://www.ibm.com/support>. Sie müssen bei der Kontaktaufnahme Ihren Namen, Ihre Organisation und Ihre Supportvereinbarung angeben.

Kapitel 1. Informationen zu IBM SPSS Modeler

IBM SPSS Modeler ist ein Set von Data-Mining-Tools, mit dem Sie auf der Grundlage Ihres Fachwissens schnell und einfach Vorhersagemodelle erstellen und zur Erleichterung der Entscheidungsfindung in die Betriebsabläufe einbinden können. Das Produkt IBM SPSS Modeler, das auf der Grundlage des den Industrienormen entsprechenden Modells CRISP-DM entwickelt wurde, unterstützt den gesamten Data Mining-Prozess, von den Daten bis hin zu besseren Geschäftsergebnissen.

IBM SPSS Modeler bietet eine Vielzahl von Modellbildungsmethoden, die aus dem maschinellen Lernen, der künstlichen Intelligenz und der Statistik stammen. Mit den in der Modellierungspalette verfügbaren Methoden können Sie aus Ihren Daten neue Informationen ableiten und Vorhersagemodelle erstellen. Jede Methode hat ihre speziellen Stärken und eignet sich besonders für bestimmte Problemtypen.

SPSS Modeler kann als Standalone-Produkt oder als Client in Verbindung mit SPSS Modeler Server erworben werden. Außerdem ist eine Reihe von Zusatzoptionen verfügbar, die in den folgenden Abschnitten kurz zusammengefasst werden. Weitere Informationen finden Sie unter <http://www.ibm.com/software/analytics/spss/products/modeler/>.

IBM SPSS Modeler-Produkte

Zur IBM SPSS Modeler-Produktfamilie und der zugehörigen Software gehören folgende Elemente.

- IBM SPSS Modeler
- IBM SPSS Modeler Server
- IBM SPSS Modeler Administration Console
- IBM SPSS Modeler Batch
- IBM SPSS Modeler Solution Publisher
- IBM SPSS Modeler Server-Adapter für IBM SPSS Collaboration and Deployment Services

IBM SPSS Modeler

SPSS Modeler ist eine funktionell in sich abgeschlossene Produktversion, die Sie auf Ihrem PC installieren und ausführen können. Sie können SPSS Modeler im lokalen Modus als Standalone-Produkt oder im verteilten Modus zusammen mit IBM SPSS Modeler Server verwenden, um bei Datasets die Leistung zu verbessern.

Mit SPSS Modeler können Sie schnell und intuitiv genaue Vorhersagemodelle erstellen, und das ohne Programmierung. Mithilfe der speziellen visuellen Benutzerschnittstelle können Sie den Data Mining-Prozess auf einfache Weise visualisieren. Mit der Unterstützung der in das Produkt eingebetteten erweiterten Analyseprozesse können Sie zuvor verborgene Muster und Trends in Ihren Daten aufdecken. Sie können Ergebnisse modellieren und Einblick in die Faktoren gewinnen, die Einfluss auf diese Ergebnisse haben, wodurch Sie in die Lage versetzt werden, Geschäftschancen zu nutzen und Risiken zu mindern.

SPSS Modeler ist in zwei Editionen erhältlich: SPSS Modeler Professional und SPSS Modeler Premium. Weitere Informationen finden Sie im Thema „IBM SPSS Modeler-Editionen“ auf Seite 2.

IBM SPSS Modeler Server

SPSS Modeler verwendet eine Client/Server-Architektur zur Verteilung von Anforderungen für ressourcenintensive Vorgänge an leistungsstarke Serversoftware, wodurch bei größeren Datasets eine höhere Leistung erzielt werden kann.

SPSS Modeler Server ist ein separat lizenziertes Produkt, das durchgehend im verteilten Analysemodus auf einem Server-Host in Verbindung mit einer oder mehreren IBM SPSS Modeler-Installationen ausgeführt wird. Auf diese Weise bietet SPSS Modeler Server eine herausragende Leistung bei großen Datensets, da speicherintensive Vorgänge auf dem Server ausgeführt werden können, ohne Daten auf den Client-Computer herunterladen zu müssen. IBM SPSS Modeler Server bietet außerdem Unterstützung für SQL-Optimierung sowie Möglichkeiten zur Modellierung innerhalb der Datenbank, was weitere Vorteile hinsichtlich Leistung und Automatisierung mit sich bringt.

IBM SPSS Modeler Administration Console

Modeler Administration Console ist eine grafische Anwendung zur Verwaltung einer Vielzahl der SPSS Modeler Server-Konfigurationsoptionen, die auch mithilfe einer Optionsdatei konfiguriert werden können. Die Anwendung bietet eine Konsolenbenutzerschnittstelle zur Überwachung und Konfiguration der SPSS Modeler Server-Installationen und steht aktuellen SPSS Modeler Server-Kunden kostenlos zur Verfügung. Die Anwendung kann nur unter Windows installiert werden. Der von ihr verwaltete Server kann jedoch auf einer beliebigen unterstützten Plattform installiert sein.

IBM SPSS Modeler Batch

Das Data-Mining ist zwar in der Regel ein interaktiver Vorgang, es ist jedoch auch möglich, SPSS Modeler über eine Befehlszeile auszuführen, ohne dass die grafische Benutzerschnittstelle verwendet werden muss. Beispielsweise kann es sinnvoll sein, langwierige oder sich wiederholende Aufgaben ohne Eingreifen des Benutzers durchzuführen. SPSS Modeler Batch ist eine spezielle Version des Produkts, die die vollständigen Analysefunktionen von SPSS Modeler ohne Zugriff auf die reguläre Benutzerschnittstelle bietet. SPSS Modeler Server ist für die Verwendung von SPSS Modeler Batch erforderlich.

IBM SPSS Modeler Solution Publisher

SPSS Modeler Solution Publisher ist ein Tool, mit dem Sie eine gepackte Version eines SPSS Modeler-Streams erstellen können, der durch eine externe Runtime-Engine ausgeführt oder in eine externe Anwendung eingebettet werden kann. Auf diese Weise können Sie vollständige SPSS Modeler-Streams für die Verwendung in Umgebungen veröffentlichen und bereitstellen, in denen SPSS Modeler nicht installiert ist. SPSS Modeler Solution Publisher wird als Teil des Diensts für IBM SPSS Collaboration and Deployment Services - Scoring verteilt, für den eine separate Lizenz erforderlich ist. Mit dieser Lizenz erhalten Sie SPSS Modeler Solution Publisher Runtime, womit Sie die veröffentlichten Streams ausführen können.

Weitere Informationen zu SPSS Modeler Solution Publisher finden Sie in der Dokumentation zu IBM SPSS Collaboration and Deployment Services. Das IBM SPSS Collaboration and Deployment Services Knowledge Center enthält die Abschnitte "IBM SPSS Modeler Solution Publisher" und "IBM SPSS Analytics Toolkit".

IBM SPSS Modeler Server-Adapter für IBM SPSS Collaboration and Deployment Services

Für IBM SPSS Collaboration and Deployment Services ist eine Reihe von Adaptern verfügbar, mit denen SPSS Modeler und SPSS Modeler Server mit einem IBM SPSS Collaboration and Deployment Services-Repository interagieren können. Auf diese Weise kann ein im Repository bereitgestellter SPSS Modeler-Stream von mehreren Benutzern gemeinsam verwendet werden. Auch der Zugriff über die Thin-Client-Anwendung IBM SPSS Modeler Advantage ist möglich. Sie installieren den Adapter auf dem System, das als Host für das Repository fungiert.

IBM SPSS Modeler-Editionen

SPSS Modeler ist in den folgenden Editionen erhältlich.

SPSS Modeler Professional

SPSS Modeler Professional bietet sämtliche Tools, die Sie für die Arbeit mit den meisten Typen von strukturierten Daten benötigen, beispielsweise in CRM-Systemen erfasste Verhaltensweisen und Interaktionen, demografische Daten, Kaufverhalten und Umsatzdaten.

SPSS Modeler Premium

SPSS Modeler Premium ist ein separat lizenziertes Produkt, das SPSS Modeler Professional für die Arbeit mit spezialisierten Daten, wie beispielsweise Daten, die für Entitätsanalysen oder soziale Netze verwendet werden, sowie für die Arbeit mit unstrukturierten Textdaten erweitert. SPSS Modeler Premium umfasst die folgenden Komponenten.

IBM SPSS Modeler Entity Analytics fügt den IBM SPSS Modeler-Vorhersageanalysen eine weitere Dimension hinzu. Während bei Vorhersageanalysen versucht wird, zukünftiges Verhalten aus früheren Daten vorherzusagen, liegt der Schwerpunkt bei der Entitätsanalyse auf der Verbesserung von Kohärenz und Konsistenz der aktuellen Daten, indem Identitätskonflikte innerhalb der Datensätze selbst aufgelöst werden. Bei der Identität kann es sich um die Identität einer Person, einer Organisation, eines Objekts oder einer anderen Entität handeln, bei der Unklarheiten bestehen könnten. Die Identitätsauflösung kann in einer Reihe von Bereichen entscheidend sein, darunter Customer Relationship Management, Betrugserkennung, Bekämpfung der Geldwäsche sowie nationale und internationale Sicherheit.

IBM SPSS Modeler Social Network Analysis transformiert Informationen zu Beziehungen in Felder, die das Sozialverhalten von Einzelpersonen und Gruppen charakterisieren. Durch die Verwendung von Daten, die die Beziehungen beschreiben, die sozialen Netzen zugrunde liegen, ermittelt IBM SPSS Modeler Social Network Analysis Führungskräfte in sozialen Netzen, die das Verhalten anderer Personen im Netz beeinflussen. Außerdem können Sie feststellen, welche Personen am meisten durch andere Teilnehmer im Netz beeinflusst werden. Durch die Kombination dieser Ergebnisse mit anderen Maßen können Sie aussagekräftige Profile für Einzelpersonen erstellen, die Sie als Grundlage für Ihre Vorhersagemodelle verwenden können. Modelle, die diese sozialen Informationen berücksichtigen, sind leistungsstärker als Modelle, die dies nicht tun.

IBM SPSS Modeler Text Analytics verwendet hoch entwickelte linguistische Technologien und die Verarbeitung natürlicher Sprache (Natural Language Processing, NLP), um eine schnelle Verarbeitung einer großen Vielfalt an unstrukturierten Textdaten zu ermöglichen, um die Schlüsselkonzepte zu extrahieren und zu ordnen und um diese Konzepte in Kategorien zusammenzufassen. Extrahierte Konzepte und Kategorien können mit bestehenden strukturierten Daten, beispielsweise demografischen Informationen, kombiniert und mithilfe der vollständigen Suite der Data-Mining-Tools von IBM SPSS Modeler auf die Modellierung angewendet werden, um bessere und fokussiertere Entscheidungen zu ermöglichen.

IBM SPSS Modeler-Dokumentation

Dokumentation im OnlinehilfefORMAT finden Sie im Hilfemenü von SPSS Modeler. Diese umfasst die Dokumentation für SPSS Modeler und SPSS Modeler Server sowie das Anwendungshandbuch (auch als Lernprogramm bezeichnet) und weiteres Material zur Unterstützung.

Die vollständige Dokumentation für die einzelnen Produkte (einschließlich Installationsanweisungen) steht im PDF-Format im Ordner *\Documentation* auf der jeweiligen Produkt-DVD zur Verfügung. Installationsdokumente können auch über das Web unter <http://www.ibm.com/support/docview.wss?uid=swg27043831> heruntergeladen werden.

Dokumentation in beiden Formaten ist auch über das SPSS Modeler Knowledge Center unter http://www-01.ibm.com/support/knowledgecenter/SS3RA7_17.0.0.0 verfügbar.

SPSS Modeler Professional-Dokumentation

Die SPSS Modeler Professional-Dokumentationssuite (ohne Installationsanweisungen) umfasst folgende Dokumente:

- **IBM SPSS Modeler Benutzerhandbuch.** Allgemeine Einführung in die Verwendung von SPSS Modeler, in der u. a. die Erstellung von Datenstreams, der Umgang mit fehlenden Werten, die Erstellung von CLEM-Ausdrücken, die Arbeit mit Projekten und Berichten sowie das Packen von Streams für die Bereitstellung in IBM SPSS Collaboration and Deployment Services, Predictive Applications (Vorhersageanwendungen) oder IBM SPSS Modeler Advantage beschrieben werden.
- **IBM SPSS Modeler Quellen-, Prozess- und Ausgabeknoten.** Beschreibung aller Knoten, die zum Lesen, zum Verarbeiten und zur Ausgabe von Daten in verschiedenen Formaten verwendet werden. Im Grunde sind sie alle Knoten, mit Ausnahme der Modellierungsknoten.
- **IBM SPSS Modeler Modellierungsknoten.** Beschreibungen sämtlicher für die Erstellung von Data-Mining-Modellen verwendeter Knoten. IBM SPSS Modeler bietet eine Vielzahl von Modellbildungsmethoden, die aus dem maschinellen Lernen, der künstlichen Intelligenz und der Statistik stammen.
- **IBM SPSS Modeler Algorithms Guide.** Beschreibung der mathematischen Grundlagen der in IBM SPSS Modeler verwendeten Modellierungsmethoden. Dieses Handbuch steht nur im PDF-Format zur Verfügung.
- **IBM SPSS Modeler Anwendungshandbuch.** Die Beispiele in diesem Handbuch bieten eine kurze, gezielte Einführung in bestimmte Modellierungsmethoden und -verfahren. Eine Online-Version dieses Handbuchs kann auch über das Hilfemenü aufgerufen werden. Weitere Informationen finden Sie im Thema „Anwendungsbeispiele“ auf Seite 5.
- **IBM SPSS Modeler Python Handbuch für Scripterstellung und Automatisierung.** Informationen zur Automatisierung des Systems über Python-Scripterstellung, einschließlich der Eigenschaften, die zur Bearbeitung von Knoten und Streams verwendet werden können.
- **IBM SPSS Modeler Bereitstellungshandbuch.** Informationen zum Ausführen von IBM SPSS Modeler-Streams und -Szenarios als Schritte bei der Verarbeitung von Jobs im IBM SPSS Collaboration and Deployment Services Deployment Manager.
- **IBM SPSS Modeler CLEF-Entwicklerhandbuch.** CLEF bietet die Möglichkeit, Drittanbieterprogramme, wie Datenverarbeitungsroutinen oder Modellierungsalgorithmen, als Knoten in IBM SPSS Modeler zu integrieren.
- **IBM SPSS Modeler Datenbankinternes Mining.** Informationen darüber, wie Sie Ihre Datenbank dazu einsetzen, die Leistung zu verbessern, und wie Sie die Palette der Analysefunktionen über Drittanbieteralgorithmen erweitern.
- **IBM SPSS Modeler Server Verwaltungs- und Leistungshandbuch.** Informationen zur Konfiguration und Verwaltung von IBM SPSS Modeler Server.
- **IBM SPSS Modeler Administration Console Benutzerhandbuch.** Informationen zur Installation und Nutzung der Konsolenbenutzerschnittstelle zur Überwachung und Konfiguration von IBM SPSS Modeler Server. Die Konsole ist als Plug-in für die Deployment Manager-Anwendung implementiert.
- **IBM SPSS Modeler CRISP-DM Handbuch.** Schritt-für-Schritt-Anleitung für das Data Mining mit SPSS Modeler unter Verwendung der CRISP-DM-Methode.
- **IBM SPSS Modeler Batch Benutzerhandbuch.** Vollständiges Handbuch für die Verwendung von IBM SPSS Modeler im Stapelmodus, einschließlich Details zur Ausführung des Stapelmodus und zu Befehlszeilenargumenten. Dieses Handbuch steht nur im PDF-Format zur Verfügung.

SPSS Modeler Premium-Dokumentation

Die SPSS Modeler Premium-Dokumentationssuite (ohne Installationsanweisungen) umfasst folgende Dokumente:

- **IBM SPSS Modeler Entity Analytics Benutzerhandbuch.** Informationen zur Verwendung von Entitätsanalysen mit SPSS Modeler, unter Behandlung der Repository-Installation und -Konfiguration, Entity Analytics-Knoten und Verwaltungsaufgaben.

- **IBM SPSS Modeler Social Network Analysis Benutzerhandbuch.** Ein Handbuch zur Durchführung einer sozialen Netzanalyse mit SPSS Modeler, einschließlich einer Gruppenanalyse und Diffusionsanalyse.
- **SPSS Modeler Text Analytics Benutzerhandbuch.** Informationen zur Verwendung von Textanalysen mit SPSS Modeler, unter Behandlung der Text Mining-Knoten, der interaktiven Workbench sowie von Vorlagen und anderen Ressourcen.

Anwendungsbeispiele

Mit den Data-Mining-Tools in SPSS Modeler kann eine große Bandbreite an geschäfts- und unternehmensbezogenen Problemen gelöst werden; die Anwendungsbeispiele dagegen bieten jeweils eine kurze, gezielte Einführung in spezielle Modellierungsmethoden und -verfahren. Die hier verwendeten Datasets sind viel kleiner als die großen Datenbestände, die von einigen Data-Mining-Experten verwaltet werden müssen, die zugrunde liegenden Konzepte und Methoden sollten sich jedoch auch auf reale Anwendungen übertragen lassen.

Sie können auf die Beispiele zugreifen, indem Sie im Menü "Hilfe" in SPSS Modeler auf die Option **Anwendungsbeispiele** klicken. Die Datendateien und Beispielstreams wurden im Ordner *Demos*, einem Unterordner des Produktinstallationsverzeichnis, installiert. Weitere Informationen finden Sie im Thema „Ordner "Demos"“.

Beispiele für die Datenbankmodellierung. Die Beispiele finden Sie im Handbuch *IBM SPSS Modeler Datenbankinternes Mining*.

Scriptbeispiele. Die Beispiele finden Sie im *IBM SPSS Modeler Handbuch für Scripterstellung und Automatisierung*.

Ordner "Demos"

Die in den Anwendungsbeispielen verwendeten Datendateien und Beispielstreams wurden im Ordner *Demos*, einem Unterordner des Produktinstallationsverzeichnis, installiert. Auf diesen Ordner können Sie auch über die Programmgruppe IBM SPSS Modeler im Windows-Startmenü oder durch Klicken auf *Demos* in der Liste der zuletzt angezeigten Verzeichnisse im Dialogfeld "Datei öffnen" zugreifen.

Kapitel 2. IBM SPSS Modeler Solution Publisher

IBM SPSS Modeler Solution Publisher ist ein leistungsstarkes Tool, mit dem Sie Data-Mining-Ergebnisse in Ihre Geschäftsprozesse integrieren und auf diese Weise praxisbezogene Probleme lösen können. Mithilfe von IBM SPSS Modeler Solution Publisher können Sie eine "verpackte" Version eines Streams erstellen, der von einer externen Runtime-Engine ausgeführt oder in eine externe Anwendung eingebettet werden kann. Dadurch können Sie Ihre Datenmodellierungsstreams in einer Produktionsumgebung verwenden, um die alltäglichen Geschäftsprozesse zu unterstützen und den Entscheidungsträgern Ihres Unternehmens die aus dem Data-Mining gewonnenen Erkenntnisse zur Verfügung zu stellen.

Die Verwendung von IBM SPSS Modeler Solution Publisher bietet mehr Potenzial als nur den Export des Modells (als PMML), denn Sie können damit *vollständige IBM SPSS Modeler-Streams* veröffentlichen und verwenden. Sie können also Datenaufbereitungsaufgaben sowie Datensatz- und Feldoperationen, wie das Aggregieren von Daten, das Auswählen von Datensätzen oder das Ableiten neuer Felder, durchführen, bevor Sie Vorhersagen auf der Grundlage eines Modells erstellen. Sie können die Modellergebnisse danach weiter verarbeiten, bevor Sie die Daten speichern - und das alles einfach, indem Sie den veröffentlichten Stream ausführen.

Hinweis: Veröffentlichung wird mithilfe der Registerkarte "Veröffentlichen" des entsprechenden Exportknotens erreicht. Weitere Informationen finden Sie im Thema „Veröffentlichen von Streams“ auf Seite 8. Für Streams, die in Versionen vor Version 11.0 des Produkts gespeichert wurden, wurde ein gesonderter Publisher-Knoten verwendet. Dieser Knoten wird zur Gewährleistung von Abwärtskompatibilität weiterhin unterstützt, steht jedoch auf der Knotenpalette nicht mehr zur Verfügung.

Lizenzierung von IBM SPSS Modeler Solution Publisher

Hinweis: IBM SPSS Modeler Solution Publisher wird als Teil des Scoring-Service für IBM SPSS Collaboration and Deployment Services vertrieben, für den eine gesonderte Lizenz erforderlich ist. <http://www.ibm.com/software/analytics/spss/products/deployment/cds/>

Wenn Sie eine Lizenz beziehen, erhalten Sie eine separate Instanz von IBM SPSS Modeler Solution Publisher Runtime, mit der Sie die veröffentlichten Streams ausführen können. Weitere Informationen finden Sie im Thema „Installation von Runtime“ auf Seite 15.

Funktionsweise von IBM SPSS Modeler Solution Publisher

Die Bereitstellung einer Lösung mithilfe von IBM SPSS Modeler Solution Publisher gliedert sich in zwei Phasen: *Veröffentlichen* eines Streams und *Ausführen* eines Streams.

Veröffentlichen. Wenn Sie den Data-Mining-Prozess abgearbeitet haben, erhalten Sie schließlich ein Modell, das eine gute Lösung für Ihr Geschäftsproblem darstellt. Nun können Sie das Modell auf Ihren Geschäftsprozess anwenden. Wenn Sie einen Stream veröffentlichen, wird eine detaillierte Beschreibung des Streams auf den Datenträger geschrieben (als *Bilddatei* und *Parameterdatei*). Weitere Informationen finden Sie im Thema „Veröffentlichen von Streams“ auf Seite 8.

Ausführung. Nach der Veröffentlichung des Streams können Sie den im Stream implementierten Prozess neu erstellen, indem Sie den veröffentlichten Stream ausführen. Dies geschieht entweder über die eigenständige Instanz von IBM SPSS Modeler Runtime (*modelerrun.exe*) oder durch Entwicklung einer Anwendung, die IBM SPSS Modeler Runtime Library zur Ausführung des Streams verwendet. Um Streams außerhalb von IBM SPSS Modeler auszuführen (mithilfe von Runtime oder über eine benutzerdefinierte Anwendung), müssen Sie zunächst IBM SPSS Modeler Solution Publisher Runtime installieren. Weitere Informationen finden Sie im Thema „Installation von Runtime“ auf Seite 15.

Veröffentlichen von Streams

Die Veröffentlichung der Streams erfolgt direkt über IBM SPSS Modeler unter Verwendung eines der Standardexportknoten wie dem Datenbank-, Flatfile-, Statistics-, IBM SPSS Data Collection-, SAS-, Excel- und XML-Exportknoten. Der Typ des Exportknotens bestimmt das Format der Ergebnisse, die bei jeder Ausführung des veröffentlichten Streams mithilfe von IBM SPSS Modeler Solution Publisher Runtime oder einer externen Anwendung geschrieben werden sollen. Beispiel: Wenn die Ergebnisse bei jeder Ausführung des veröffentlichten Streams in eine Datenbank geschrieben werden sollen, müssen Sie einen Datenbankexportknoten verwenden.

Veröffentlichen eines Streams

1. Öffnen bzw. erstellen Sie einen Stream auf normale Art und Weise und fügen Sie am Ende einen Exportknoten an.
2. Geben Sie auf der Registerkarte "Veröffentlichen" im Exportknoten einen Stammnamen für die veröffentlichten Dateien an (d. h. den Dateinamen, an den die verschiedenen Erweiterungen *.pim*, *.par* und *.xml* angefügt werden).
3. Klicken Sie auf die Schaltfläche **Veröffentlichen**, um den Stream zu veröffentlichen, oder wählen Sie **Stream veröffentlichen** aus, um den Stream bei jeder Ausführung des Knotens automatisch veröffentlichen zu lassen.

Veröffentlichter Name. Geben Sie den Stammnamen für die veröffentlichten Bild- und Parameterdateien an.

- Die **Bilddatei** (**.pim*) enthält alle Informationen, die Runtime benötigt, um den veröffentlichten Stream genau in dem Zustand auszuführen, in dem er sich zum Zeitpunkt des Exports befand. Wenn Sie sich sicher sind, dass Sie keine der Einstellungen für den Stream ändern müssen (beispielsweise die Eingangsdatenquelle oder die Ausgabedatendatei) können Sie auch nur die Bilddatei verwenden.
- Die **Parameterdatei** (**.par*) enthält konfigurierbare Informationen zu Datenquellen, Ausgabedateien und Ausführungsoptionen. Wenn Sie die Eingabe bzw. Ausgabe des Streams steuern möchten, ohne den Stream erneut veröffentlichen zu müssen, benötigen Sie neben der Bilddatei auch die Parameterdatei.
- Die **Metadatendatei** (**.xml*) beschreibt die Ein- und Ausgaben des Bilds und der zugehörigen Datenmodelle. Sie ist für Anwendungen gedacht, die die Runtime-Bibliothek einbetten und denen die Struktur der Ein- und Ausgabedaten bekannt sein muss.

Hinweis: Diese Datei wird nur erstellt, wenn Sie das Kontrollkästchen **Metadaten veröffentlichen** auswählen.

Parameter veröffentlichen. Falls erforderlich, können Sie Streamparameter in die **.par*-Datei aufnehmen. Sie können die Werte dieser Streamparameter beim Ausführen des Bilds ändern, entweder, indem Sie die **.par*-Datei bearbeiten oder alternativ über die Runtime-API.

Mit diesem Kontrollkästchen aktivieren Sie die Schaltfläche **Parameter**. Beim Klicken auf die Schaltfläche wird das Dialogfeld "Parameter veröffentlichen" angezeigt.

Wählen Sie die Parameter aus, die Sie in das veröffentlichte Bild aufnehmen möchten, indem Sie in der Spalte **Veröffentlichen** das entsprechende Kontrollkästchen aktivieren.

Bei Streamausführung. Gibt an, ob der Stream bei der Ausführung des Knotens automatisch veröffentlicht werden soll.

- **Daten exportieren.** Führt den Exportknoten in der standardmäßigen Art und Weise aus, ohne Veröffentlichung des Streams. (Im Grunde wird der Knoten in IBM SPSS Modeler genau so ausgeführt, als ob IBM SPSS Modeler Solution Publisher nicht verfügbar wäre.) Bei Auswahl dieser Option wird der Stream nur veröffentlicht, wenn Sie die Veröffentlichung explizit durch Klicken auf die Schaltfläche **Veröffentlichen** im Dialogfeld des Exportknotens auslösen. Alternativ können Sie den aktuellen Stream mithilfe des Tools "Veröffentlichen" in der Symbolleiste oder unter Verwendung eines Scripts veröffentlichen.

- **Stream veröffentlichen.** Veröffentlicht den Stream für die Verwendung mithilfe von IBM SPSS Modeler Solution Publisher. Wählen Sie diese Option aus, wenn der Stream bei jeder Ausführung automatisch veröffentlicht werden soll.

Kommentare

- Wenn Sie vorhaben, den veröffentlichten Stream mit neuen oder aktualisierten Daten auszuführen, dürfen Sie nicht vergessen, dass die Reihenfolge der Felder in der Eingabedatei mit der Reihenfolge der Felder in der im veröffentlichten Stream angegebenen Quellenknoten-Eingabedatei übereinstimmen muss.
- Bei der Veröffentlichung in externe Anwendungen sollten Sie die Filterung überflüssiger Felder bzw. das Umbenennen von Feldern in Erwägung ziehen, um die Eingabeanforderungen zu erfüllen. Beides lässt sich durch Verwendung eines Filterknotens vor dem Exportknoten erreichen.

Kapitel 3. Ausführen veröffentlichter Streams

Das wichtigste Ergebnis bei der Ausführung eines veröffentlichten Streams ist stets die Erzeugung eines Datasets. Die Daten können in einer Datenträgerdatei gespeichert oder in eine Datenbank geschrieben werden. Veröffentlichte Streams können keine Diagramme, Tabellen, generierten Modelle oder Ausgaben, bei denen es sich nicht um Daten handelt, erzeugen.

Über IBM SPSS Modeler Solution Publisher veröffentlichte Streams können mithilfe des Programms IBM SPSS Modeler Solution Publisher Runtime ausgeführt werden. Das Runtime-Programm, *modelerrun.exe*, wird über die Befehlszeile gestartet. Dabei werden die auszuführende Bilddatei und (optional) die Parameterdatei durch Optionen angegeben. Der Runtime-Befehl lautet wie folgt:

```
modelerrun -nobanner -p <Parameterdatei> -o <Optionen> <Bilddatei>
```

Die Befehlseingaben werden in der folgenden Tabelle angezeigt.

Tabelle 1. Runtime-Befehlseingaben

Befehlseingabe	Beschreibung
-nobanner	unterdrückt die Starttextnachricht für Runtime (optional).
<Parameterdatei>	ist der Dateiname der Parameterdatei (optional).
<Optionen>	ist eine durch Kommas getrennte Liste von Option=Wert-Paaren, die Ausführungsoptionen für Runtime angeben. Die Optionen werden weiter unten detailliert beschrieben (optional). Hinweis: Auf ein Komma müssen Werte folgen.
<Bilddatei>	ist der Dateiname der auszuführenden Bilddatei (erforderlich).

In der folgenden Tabelle sind die verfügbaren Ausführungsoptionen aufgeführt.

Tabelle 2. Gültige Runtime-Optionen.

Name	Standardwert	Beschreibung
max_file_size	-1	Maximale Größe der gespeicherten Dateien. Der Wert -1 bedeutet, dass keinerlei Beschränkung vorliegt.
max_sql_string_length	2.048	Maximale Länge für Zeichenfolgen, die aus einer Datenbank mit SQL importiert werden. Zeichenfolgewerte, die länger sind als dieser Wert, werden ohne Warnung rechts gekürzt. Der zulässige Bereich liegt zwischen 1 und 65.535 Zeichen.
memory_usage	100	Multiplikator für die dynamische Arbeitsspeicherzuordnung. Passen Sie diesen Wert nach oben oder unten an, um den Gesamtwert für die Arbeitsspeicherauslastung des Servers zu regulieren.
temp_directory	"	Gibt das für temporäre Dateien zu verwendende Verzeichnis an.
request_passwords	Y	Regelt, ob Sie bei der Ausführung eines veröffentlichten Streams, für den Datenbankzugriff erforderlich ist, aufgefordert werden, ein Datenbankkennwort einzugeben (Y/N).

Verwenden der Parameterdatei zur Anpassung der Streamausführung

Wenn ein Stream veröffentlicht wird, enthält die Bilddatei Informationen zur Datenquelle und zum Ziel der im ursprünglichen Stream gesammelten Ausgabedaten. Es ist jedoch häufig zweckmäßig, einen Stream mit Daten aus einer anderen Datenquelle auszuführen oder die Ausgabedaten an ein anderes Ziel zu leiten. Dies ist durch Bearbeitung der Parameterdatei des Streams möglich.

Dies ist besonders wichtig, wenn Sie den veröffentlichten Stream auf einem anderen Computer ausführen als dem, auf dem er erstellt wurde. Sie müssen die Speicherorte der Ein- und Ausgabedateien so aktualisieren, dass sie die Struktur des Zielcomputers berücksichtigen. Beachten Sie, dass veröffentlichte Streams keine relativen Pfade unterstützen, sodass es besonders wichtig ist, die in der Parameterdatei angegebenen Datenspeicherorte zu überprüfen. (Wenn Sie beim Veröffentlichlichen des Streams in den Optionen des Exportknotens relative Pfade angeben, wandelt IBM SPSS Modeler diese vor der Erstellung der Bild- und Parameterdatei für den veröffentlichten Stream in absolute Pfade um.)

Außerdem können Sie mithilfe der Parameterdatei Ausführungsoptionen festlegen und die Werte von Streamparametern ändern.

Die Parameterwerte werden in der Parameterdatei als <Name.Attribut>=<Wert>-Paare angegeben. Bei einigen Parametern enthält der Parameternamen eine ID-Stelle, mit der Knoten desselben Typs im Stream unterschieden werden. Beispiel: In einem Stream, in dem sich sowohl die Eingabe als auch die Ausgabe auf eine Datenbankverbindung beziehen, könnten die Parameter, die die Eingabeverbindung beschreiben, dbconn0.datasource, dbconn0.user usw. lauten und die der Ausgabeverbindung zugeordneten Parameter könnten die Bezeichnungen dbconn1.datasource, dbconn1.user usw. tragen. Werte werden stets in doppelte Anführungszeichen gesetzt. Die Parameterdatei enthält folgende Parameter, von denen sich alle zur Anpassung der Streamausführung ändern lassen.

Tabelle 3. Ausführungsparameter.

Parameter	Zweck
dbconnx.datasource	Datenquellennamen (DSN).
dbconnx.user	Benutzername für Datenbanken mit eingeschränktem Zugriff.
dbconnx.password	Kennwort für Datenbanken mit eingeschränktem Zugriff.
dbconnx.epassword	Codiertes Kennwort für Datenbanken mit eingeschränktem Zugriff. Zur Erzeugung eines verschlüsselten Kennworts wählen Sie an der IBM SPSS Modeler-Benutzerschnittstelle im Menü "Extras" die Option Kennwort verschlüsseln aus. Sie können das codierte Kennwort nach Bedarf kopieren und einfügen.
dbconnx.password_required	Flag, das angibt, ob zur Eingabe eines Kennworts für diese Datenbankverbindung aufgefordert wird oder nicht. (Die möglichen Werte sind "Y" (Ja) oder "N" (Nein).) Dieser Wert ist automatisch auf "N" gesetzt, wenn zum Veröffentlichungszeitpunkt kein Kennwort angegeben wurde.
dbconnx.catalog	Der der Datenbankverbindung zugeordnete Katalog.
dbtablex.name	Name der Datenbanktabelle. <i>Hinweis:</i> Die Reihenfolge der Felder in einer neuen Tabelle muss mit der Reihenfolge der Felder im ursprünglich veröffentlichten Stream übereinstimmen.
dbtablex.schema	Name des Schemas.
dbtablex.catalog	Der der aktuellen Tabelle zugeordnete Katalog.
filex.name	Dateiname für die Datendatei. <i>Hinweis:</i> Die Reihenfolge der Felder in einer neuen Eingabedatei muss mit der Reihenfolge der Felder im ursprünglich veröffentlichten Stream übereinstimmen.
filex.path	Speicherort der Datei (Verzeichnisname).

Tabelle 3. Ausführungsparameter (Forts.).

Parameter	Zweck
filex.field_names_included	Bei Dateien im Textformat ist dies das Flag, das angibt, ob die Feldnamen als erste Zeile der Datei mit aufgenommen wurden. (Die möglichen Werte sind "Y" (Ja) oder "N" (Nein).)
filex.field_separator	Bei Dateien im Textformat ist dies das zum Trennen der Feldwerte verwendete Zeichen (auch mehrere Zeichen möglich).
filex.decimal_separator	Bei Dateien im Textformat ist dies das als Dezimaltrennzeichen verwendete Zeichen.
options.angle_in_radians	Flag, das angibt, ob Radianten als Maßeinheit in trigonometrischen CLEM-Ausdrücken verwendet werden. (Die möglichen Werte sind "Y" (Ja) oder "N" (Nein).)
options.date_2digit_baseline	Legt das Jahrhundert für Datumsangaben mit zweistelliger Jahreszahl fest.
options.date_baseline	Das Basisjahr (immer 1. Jan.), das von den CLEM-Datumsfunktionen verwendet wird, die mit einem einzelnen Datum arbeiten.
options.time_rollover	Flag, das angibt, ob sich negative Zeitdifferenzen auf die Vergangenheit beziehen.
options.decimal_separator	Das Standarddezimaltrennzeichen in Textdateien.
options.time_format	Zeitformat, das verwendet wird, wenn Zeichenfolgen von CLEM-Zeitfunktionen als Zeiten interpretiert werden.
options.date_format	Datumsformat, das verwendet wird, wenn Zeichenfolgen von CLEM-Datumsfunktionen als Datumsangaben interpretiert werden.
options.timestamp_format	Format, das beim Einlesen von Zeitmarkenfeldern als Zeichenfolgen aus ODBC-Datenquellen verwendet wird.
paramx.value	Wert des Streamparameters. <i>Hinweis:</i> Name, Beschreibung und Speichertyp des Parameters sind als Referenz angegeben, können jedoch nicht geändert werden. Der Wert muss für den Typ gültig sein.

Einbetten von IBM SPSS Modeler Solution Publisher in Anwendungen

Neben der eigenständigen Runtime-Engine bietet IBM SPSS Modeler Solution Publisher Solution Publisher eine Runtime-Programmierbibliothek (CLEMRTL), mit der andere Programme die Ausführung des Streams von IBM SPSS Modeler Solution Publisher steuern können. Sie können CLEMRTL-Verfahren in Client-Programmen aufrufen, die in C, C++ geschrieben wurden. Zur Verwendung von CLEMRTL müssen Sie die Headerdatei *clemrtl.h* (verfügbar im Ordner *..\installation\clemrtl\include*) einfügen und die entsprechende Bibliotheksdatei für Ihre Entwicklungsplattform bei der Erstellung mit Ihrer Anwendung verknüpfen. Die Datei "clemrtl.dll" befindet sich im Ordner *..\installation\bin*.

Alle Quellendateien, die Bibliotheksprozeduren referenzieren, müssen den Header *clemrtl.h* enthalten. Diese Headerdatei bietet ANSI-C-Prototypen für die Bibliotheksprozeduren und definiert nützliche Makros. Es erfordert keine weiteren Header als die für Ihr Programm benötigten. Als Schutz vor Namenskonflikten beginnen alle Namen von Bibliothekstypen und -funktionen mit *clemrtl_* und alle Makronamen ist *CLEMRTL_* vorangestellt.

CLEMRTL erstellt Berichte mit nützlichen Informationen, die wieder an die Anwendung zurückgemeldet werden sollten. CLEMRTL bietet drei Mechanismen für den Umgang mit solchen Nachrichten:

- Nachrichten können in ein mit der Funktion *clemrtl_setLogFile()* angegebenes Protokoll geschrieben werden.

- Die Anwendung kann mit der Funktion `clemrtl_getErrorDetail()` Einzelheiten des letzten Fehlers abrufen.
- Mit der Funktion `clemrtl_setReportHandler()` kann die Anwendung ihr eigenes Verfahren für den Umgang mit Berichten bereitstellen.

Das Runtime-System verfügt über einen lokalisierten Nachrichtenkatalog für Berichte. Jeder an die Anwendung weitergegebene Bericht enthält die Zeichenfolge der lokalisierten Nachricht. Berichte enthalten außerdem den Berichtscode, sodass die Anwendung die Nachricht auch anders interpretieren und darstellen kann. Zum Abrufen detaillierter (lokalisierter) Nachrichten, muss die Anwendung die entsprechende `messages.cfg`-Datei in ein `config`-Verzeichnis transferieren. Verschiedene Versionen von `messages.cfg` für verschiedene Ländereinstellungen finden Sie im Verzeichnis `<Installationsverzeichnis>/config/<Ländereinstellung>`. Dabei ist `<Installationsverzeichnis>` das Verzeichnis, in dem IBM SPSS Modeler Solution Publisher installiert wurde, und `<Ländereinstellung>` die gewünschte Ländereinstellung. Wählen Sie die entsprechende Version von `messages.cfg` und fügen Sie sie zu einem `config`-Unterverzeichnis in Ihrem Projektverzeichnis hinzu.

Das Runtime-System benötigt für den korrekten Betrieb keine speziellen Umgebungs- oder Registrierungseinstellungen. Abhängige Bibliotheken müssen mit einer mit der CLEMRTL verknüpften Anwendung verteilt werden. Diese Dateien finden Sie im Verzeichnis `<Installationsverzeichnis>/bin`.

Voraussetzungen

In der folgenden Tabelle sind die Plattformen und Compiler aufgeführt, die mit der Runtime-Programmierbibliothek getestet wurden.

Tabelle 4. Plattformen und Compiler.

Plattform	Vorgeschlagene(r) Compiler
AIX	IBM XL C/C++ Enterprise Edition V8.0 für AIX
Solaris (64-Bit)	Sun Studio 11 5.8 Patch 121017-06
Linux (64-Bit)	gcc 3.2.3, Intel C++-Compiler 9.1.043
Windows	Visual Studio 2005

Beispielanwendung von IBM SPSS Modeler Solution Publisher

Eine Beispielanwendung, `dlltest.c`, finden Sie im Ordner `\clemrtl\demo` im Installationsverzeichnis von IBM SPSS Modeler Solution Publisher. Verwenden Sie die Beispielanwendung, um sich mit der Einbettung der Runtime-Bibliothek in Ihre eigenen Anwendungen vertraut zu machen.

Die von der Beispielanwendung verwendeten Informationen finden Sie im Ordner `\clemrtl\demo`. Anweisungen dazu entnehmen Sie bitte der Datei `README.txt`. Die Kompilierung von Anwendungen ist je nach Plattform unterschiedlich, weshalb auch Beispielprojektdateien enthalten sind.

Kapitel 4. Installieren von IBM SPSS Modeler Solution Publisher Runtime

Installation von Runtime

Die Installation von IBM SPSS Modeler Solution Publisher Runtime ermöglicht Ihnen die Ausführung von veröffentlichten Streams. Veröffentlichte Streams sind vom Betriebssystem unabhängig. Sie können also jeden veröffentlichten Stream auf jeder von Runtime unterstützten Plattform ausführen.

Die Funktionalität, mit der Sie Streams aus IBM SPSS Modeler veröffentlichen können, wird mit IBM SPSS Modeler installiert. Beachten Sie, dass die veröffentlichten Dateien bei jedem Release von IBM SPSS Modeler erneut exportiert werden müssen. So funktioniert beispielsweise eine Datei, die mit einer früheren Version exportiert wurde, nicht in Version 17 von IBM SPSS Modeler Solution Publisher Runtime.

Systemanforderungen

Folgende Systemanforderungen gelten für die Installation von IBM SPSS Modeler Solution Publisher Runtime:

- **Betriebssystem.** Windows Server 2008 Standard Edition für 32-Bit x86-Systeme; Windows Server 2008 Enterprise Edition für 64-Bit x64-Systeme; Windows Server 2008 Enterprise Edition R2 für 64-Bit x64-Systeme; Windows Server 2003 Standard Edition R2 für 32-Bit x86-Systeme oder 64-Bit x64-Systeme; IBM AIX[®] 6.1 oder 7.1 für 64-Bit POWER-Systeme; Oracle Solaris[™] 9.x oder 10 für 64-Bit SPARC-Systeme; Red Hat Enterprise Linux 5.x für 32-Bit x86-Systeme; Red Hat Enterprise Linux 5.x oder 6.x für 64-Bit x64-Systeme oder IBM System z-Systeme; Red Hat Enterprise Linux Advanced Platform 5.x 32-Bit x86-Systeme; Red Hat Enterprise Linux Advanced Platform 5.x oder 6.x für 64-Bit x64-Systeme oder IBM System z-Systeme; SuSE Linux Enterprise Server 10 oder 11 für 64-Bit x64-Systeme oder IBM System z-Systeme
- **Hardware.** Pentium[®]-Prozessor oder Prozessor der Pentium-Klasse oder höher (für 32-Bit-Windows); x64-Prozessorfamilie (AMD 64 und EM64T) (für 64-Bit-Windows) mit einer Taktfrequenz von mindestens 1 GHz; PowerPC-Prozessor mit mindestens 233 MHz und IBM System p für IBM AIX; UltraSPARC II (oder höher) für Solaris; Pentium-Prozessor oder Prozessor der Pentium-Klasse oder höher für 32-Bit-Linux; x64-Prozessorfamilie (AMD 64 und EM64T) oder IBM s390x System z für 64-Bit-Linux
- **Mindestens erforderlicher freier Plattenspeicherplatz.** 2 GB empfohlen.
- **RAM.** 4 GB empfohlen.
- **Virtuelle Umgebung.** IBM SPSS Modeler Server wird von folgenden virtuellen Umgebungen unterstützt:
 - IBM PowerVM Hypervisor für IBM AIX
 - IBM PR/SM oder z/VM für Linux für IBM System z
 - VMWare ESX Server 4.1
 - VMWare vSphere 4.0
- **Clusterumgebung.** IBM SPSS Modeler Server wird von folgenden virtuellen Umgebungen unterstützt:
 - IBM HACMP/RSCT für IBM AIX
 - IBM WebSphere App Server (64-Bit)
 - Microsoft Cluster Service für Windows
 - Oracle Solaris Cluster
 - Oracle WebLogic App Server (64-Bit)
 - Red Hat Cluster Suite für Red Hat Enterprise Linux

Installation unter Windows

Im folgenden Abschnitt wird das Verfahren für die Installation von SPSS Modeler Solution Publisher Runtime unter dem Betriebssystem Windows beschrieben.

1. Legen Sie den Installationsdatenträger für IBM SPSS Modeler Solution Publisher in das Laufwerk ein oder greifen Sie auf andere Art auf das Installationsmedium zu.
2. Wählen Sie im Installationsmenü die Option **IBM SPSS Modeler Solution Publisher Runtime installieren** oder **IBM SPSS Modeler Solution Publisher Runtime for Win64 installieren** aus, je nachdem, auf welchem System die Installation vorgenommen wird.
3. Klicken Sie zum Starten im Installationsassistenten auf **Weiter**.
4. Befolgen Sie die auf dem Bildschirm angezeigten Anweisungen. Klicken Sie zum Fortfahren auf **Weiter**.
5. Sobald Sie alle Optionen angegeben haben, kann die Installation gestartet werden. Klicken Sie auf **Installieren**, um mit der Übertragung der Dateien zu beginnen.
6. Wenn alle Dateien installiert wurden, klicken Sie auf **Fertig stellen**.

Installation unter UNIX

Im folgenden Abschnitt werden die zusätzlichen Systemanforderungen und die Prozedur für die Installation von IBM SPSS Modeler Solution Publisher Runtime unter dem Betriebssystem UNIX beschrieben.

Zusätzliche Anforderungen für UNIX

Sie müssen sicherstellen, dass Kernel-Limits im System für den Betrieb von IBM SPSS Modeler Server ausreichen. Die "ulimits" für Daten, Arbeitsspeicher und Dateien sind besonders wichtig und sollten in der IBM SPSS Modeler Server-Umgebung auf unbegrenzt (unlimited) eingestellt werden. Gehen Sie dazu wie folgt vor:

1. Fügen Sie *modelersrv.sh* die folgenden Befehle hinzu:

```
ulimit -d unlimited
ulimit -m unlimited
ulimit -f unlimited
```

Setzen Sie außerdem den Stackgrenzwert auf das für Ihr System zulässige Maximum (ulimit -s XXXX).
Beispiel:

```
ulimit -s 64000
```

2. Starten Sie IBM SPSS Modeler Server neu.

Darüber hinaus müssen *GNU-cpio* und das Komprimierungsdienstprogramm *gzip* installiert und in PATH angegeben sein, damit das Installationsprogramm die Installationsdateien dekomprimieren kann. Zusätzlich sollten Sie auf dem Computer, auf dem SPSS Modeler Server ausgeführt wird, die Ländereinstellung auf DE_DE.UTF-8 setzen.

UNIX-Installationsprozedur

Die folgenden Installationsanweisungen gelten für IBM SPSS Modeler Solution Publisher Runtime für UNIX. Sie benötigen Lese- und Schreibberechtigungen für das Zielinstallationsverzeichnis. Melden Sie sich daher mit einem Konto an, das über ausreichende Berechtigungen verfügt.

Anmerkung: Sie müssen für die Installation als *Root* angemeldet sein.

1. Legen Sie den Installationsdatenträger in das Laufwerk ein und stellen Sie sicher, dass Sie auf das Laufwerk zugreifen können. Unter Solaris muss der Volume Manager (vold-Dämon) laufen.
2. Wechseln Sie in das Verzeichnis mit dem DVD-Mountpoint.
3. Wechseln Sie in das Verzeichnis *modelrun*.
4. Ändern Sie das entsprechende Plattformverzeichnis.
5. Führen Sie das Installationsscript *.bin* aus. Beispiel:

- ```
./modelersolutionpublisherhpie64.bin -i console
```
6. Es werden detaillierte Anweisungen angezeigt. Drücken Sie zum Fortfahren die Eingabetaste.
  7. Die Lizenzinformationen werden angezeigt. Lesen Sie die Lizenzvereinbarung, geben Sie 1 ein, um sie zu akzeptieren, und drücken Sie zum Fortfahren die Eingabetaste.
  8. Es wird eine Liste der verfügbaren Sprachen angezeigt. Geben Sie die Nummer für die zu installierende Sprachversion ein und drücken Sie zum Fortfahren die Eingabetaste.
  9. Sie werden aufgefordert, den Installationsort einzugeben. Um das Standardverzeichnis `/usr/IBM/SPSS/ModelerSolutionPublisher<nn>` zu verwenden (wobei `<nn>` die Versionsnummer ist), drücken Sie die Eingabetaste. Runtime wird im angegebenen Verzeichnis installiert.
  10. Sie werden aufgefordert, den Installationsort zu bestätigen. Wenn er korrekt ist, geben Sie `y` ein und drücken Sie die Eingabetaste.
  11. Es wird eine Vorinstallationsübersicht angezeigt, damit Sie Ihre bisherigen Eingaben bestätigen können. Drücken Sie zum Fortfahren die Eingabetaste.
  12. Es wird eine Nachricht angezeigt, die besagt, dass die Installationsroutine nun ausgeführt werden kann. Drücken Sie zum Fortfahren die Eingabetaste.
  13. Während die Installationsroutine ausgeführt wird, wird eine Fortschrittsleiste angezeigt. Wenn die Installation abgeschlossen ist, drücken Sie die Eingabetaste, um das Installationsprogramm zu beenden.

## Konfigurieren von ODBC unter UNIX

Standardmäßig ist der DataDirect-Treibermanager nicht für die Verwendung von ODBC durch IBM SPSS Modeler Solution Publisher Runtime auf UNIX-Systemen konfiguriert. Geben Sie folgende Befehle ein, um UNIX so zu konfigurieren, dass der DataDirect-Treibermanager geladen wird (wobei *Solution\_Publisher-Installationsverzeichnis* für das Installationsverzeichnis von Solution Publisher Runtime steht):

```
cd Solution_Publisher-Installationsverzeichnis
rm -f libspssodbc.so
ln -s libspssodbc_datadirect.so libspssodbc.so
```

## Fehlerbehebung bei Installationen

### Ungültige digitale Signatur bei Installation

IBM SPSS Modeler-Produkte verwenden von IBM ausgegebene Zertifikate für digitale Signaturen. Unter bestimmten Umständen kann beim Versuch, SPSS Modeler-Produkte zu installieren, der folgende Fehler angezeigt werden:

Fehler 1330. Eine erforderliche Datei kann nicht installiert werden, da die Kabinettdatei *Dateiname* eine ungültige Signatur hat...

### Alle Windows-Benutzer

Diese Nachricht wird angezeigt, wenn Sie versuchen, SPSS Modeler-Produkte auf einem Computer zu installieren, der nicht über eine Internetverbindung verfügt und auf dem nicht das richtige Zertifikat installiert ist. Beheben Sie dieses Problem mithilfe der folgenden Prozedur.

1. Klicken Sie auf **OK**, um die Nachricht zu bestätigen.
2. Klicken Sie auf **Abbrechen**, um das Installationsprogramm zu schließen.
3. Wenn der Computer, auf dem Sie die Installation vornehmen möchten, nicht über eine Internetverbindung verfügt, führen Sie den nächsten Schritt auf einem Computer mit Internetverbindung durch und kopieren Sie die *.cer*-Datei auf den Computer, auf dem die Installation erfolgen soll.
4. Blättern Sie auf der Seite <http://www.verisign.com/support/roots.html> nach unten, bis Sie die Rootzertifizierungsstelle **VeriSign Class 3 Primary CA - G5** gefunden haben. Laden Sie diese Rootzertifizierungsstelle herunter und speichern Sie sie als *.cer*-Datei.
5. Doppelklicken Sie auf die *.cer*-Datei.
6. Klicken Sie auf der Registerkarte "Allgemein" auf **Zertifikat installieren**.

7. Befolgen Sie die Anweisungen im Zertifikatimportassistenten. Verwenden Sie dabei jeweils die Standardoptionen und klicken Sie abschließend auf **Fertigstellen**.
8. Führen Sie den Installationsvorgang erneut durch.

---

## Starten von Runtime

Nach der Installation von Runtime können Sie damit Streams ausführen, die mithilfe eines der Exportknoten aus IBM SPSS Modeler veröffentlicht wurden. Weitere Informationen finden Sie im Thema „Veröffentlichen von Streams“ auf Seite 8. Starten Sie Runtime über die Befehlszeile unter Verwendung von Optionen, die den Namen der veröffentlichten Bilddatei und eine optionale Parameterdatei für die Ausführung angeben.

Zur Ausführung von Runtime geben Sie an der Eingabeaufforderung folgende Zeile ein:

```
modelerrun -p <Parameterdatei> -o <Optionen> <Bilddatei>
```

Die Befehlseingaben werden in der folgenden Tabelle angezeigt.

*Tabelle 5. Runtime-Befehlseingaben*

| Befehlseingabe   | Beschreibung                                                                                                                                                  |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <Parameterdatei> | ist der Dateiname der veröffentlichten Parameterdatei (optional).                                                                                             |
| <Optionen>       | ist eine durch Kommas getrennte Liste von Option=Wert-Paaren, die Ausführungsoptionen für Runtime angeben. <b>Hinweis:</b> Auf ein Komma müssen Werte folgen. |
| <Bilddatei>      | ist der Dateiname der auszuführenden veröffentlichten Bilddatei (*.pim).                                                                                      |

*Hinweis:* Bei Verwendung von Mehrbytesprachen (wie Chinesisch oder Japanisch) in einer UNIX-Umgebung müssen auch die zu verwendenden Sprachencodes angegeben werden. Das folgende Beispiel zeigt die Sprachencodes für vereinfachtes Chinesisch:

```
./modelerrun -o locale="zh_CN.GB18030",encoding="GB18030" -p
```

Dabei gibt -p den Speicherort der pim- und par-Dateien an.

Weitere Informationen finden Sie in Kapitel 3, „Ausführen veröffentlichter Streams“, auf Seite 11.

## Ändern des temporären Verzeichnisses

Für einige in IBM SPSS Modeler Solution Publisher Runtime durchgeführte Operationen müssen u. U. temporäre Dateien erstellt werden. Runtime verwendet standardmäßig das temporäre Systemverzeichnis zum Erstellen von temporären Dateien. Sie können das temporäre Verzeichnis wie folgt ändern:

*Hinweis:* Sie müssen unbedingt die richtige Syntax verwenden. Achten Sie insbesondere auf Leerzeichen und den Verzeichnispfad.

Angenommen, Sie verwenden IBM SPSS Modeler Solution Publisher Runtime mit den folgenden veröffentlichten Dateien: *webtest.pim* und *webtest.par*. Sie möchten den Speicherort für temporäre Verzeichnisse in *C:\published\temp* ändern. So können Sie den Speicherort des von der ausführbaren Datei verwendeten temporären Verzeichnisses ändern:

1. Führen Sie in dem Verzeichnis, in dem die Dateien *.pim* und *.par* gespeichert sind, folgenden Befehl aus:

```
modelerrun -o temp_directory="C:\\published\\temp" -p webtest.par webtest.pim
```

Dadurch wird Runtime angewiesen, "C:\published\temp" als temporäres Verzeichnis ausschließlich für diesen spezifischen Vorgang zu verwenden.



---

## Deinstallation von IBM SPSS Modeler Solution Publisher Runtime

Im folgenden Abschnitt werden die Vorgänge für die Deinstallation von IBM SPSS Modeler Solution Publisher Runtime beschrieben.

### Windows-Deinstallationsprozedur

1. Öffnen Sie in der Windows-Systemsteuerung das Dialogfeld **Software**.
2. Wählen Sie **IBM SPSS Modeler Solution Publisher <nn>** in der Liste aus.
3. Klicken Sie auf **Entfernen**.
4. Befolgen Sie die Anweisungen auf dem Bildschirm, um Runtime zu entfernen.

*Hinweis:* Sie müssen an Ihrem Computer mit Administratorberechtigungen angemeldet sein, um Programme entfernen zu können.

### UNIX-Deinstallationsprozedur

Löschen Sie das Installationsverzeichnis mit allen installierten Dateien.



---

## Kapitel 5. IBM SPSS Embedded Predictive Modeling API

Die IBM SPSS Embedded Predictive Modeling API definiert eine einfache Java-Schnittstelle, die Ihnen die Integration der in IBM SPSS Modeler entworfenen Artefakte in Ihre Anwendung durch Einbettung ermöglicht. Die vollständige Dokumentation im Javadoc-Format ist für Codierer, die die Bibliothek verwenden, im Lieferumfang von IBM SPSS Modeler Solution Publisher enthalten.

Beachten Sie die folgenden Einschränkungen für die Verwendung der Embedded Predictive Modeling (EPM) API:

- **Eindeutige Beschriftungen für Quellen- und Endknoten erforderlich.** Die EPM API verwendet Knotenbeschriftungen. Auch wenn die Beschriftungen immer im Kontext von Quellen- oder Endknoten verwendet werden, können sie mehrdeutig sein.
- **Superknoten als Quellen- oder Endknoten nicht unterstützt.** Superknoten sollen bei allen Operationen als allgemeine Quelle vorhanden sein und werden daher nicht als Quellen- oder Endknoten unterstützt.
- **Einschränkungen für Namen von Eingabefeldern.** Einige Zeichen können in Feldnamen nicht verwendet werden.

Nachfolgend finden Sie eine Übersicht über die API-Funktionalität. Weitere Details finden Sie in der zum Lieferumfang des Produkts gehörenden Javadoc-Dokumentation.

### Eingebettetes Scoring für Vorhersagemodelle

**Verwendung:** Wird von den Anwendungen verwendet, um beim Eingang neuer Daten bedarfsgesteuert Vorhersageanalysen zu generieren. Die Anwendung muss aktualisierte Modelle bearbeiten. Quellenknoten sowie die Endknoten der Scoring-Verzweigung werden in diesem Scoring-Modus durch Ein- und Ausgaben der Anwendung ersetzt.

**Aktion:** Bereitet die von Ihnen im Stream von IBM SPSS Modeler entworfene Scoring-Verzweigung einmal für wiederholte Aufrufe vor, die die Vorhersageanalyse über von der Anwendung verwaltete Eingabedaten schnell und effizient generieren.

### Aktualisierung von Vorhersagemodellen

**Verwendung:** Führt das einführende Training des Modellalgorithmus im Streamentwurf mithilfe von historischen Daten durch. "Aktualisiert" die konfigurierten Vorhersagemodellalgorithmen darüberhinaus regelmäßig mit den aktuellen Verlaufsdaten.

**Aktion:** Führt alle Erstellungsknoten für Vorhersagemodelle im Stream von IBM SPSS Modeler aus, wodurch der Vorhersagemodellalgorithmus wie konfiguriert mithilfe von Data-Mining-Verfahren trainiert wird. Aktualisiert die mit den einzelnen Modellerstellungsknoten verknüpften Modellanwendernuggets. Mithilfe von Streamparametern und den Eigenschaften von Quellen-/Terminalknoten kann eine gewisse Dynamik implementiert werden.

### Evaluierung von Vorhersagemodellen

**Verwendung:** Wird von einer Anwendung verwendet, um die Genauigkeit, Konfidenz oder andere andere qualitative Evaluierungen eines in einem Stream von IBM SPSS Modeler entworfenen, trainierten Vorhersagemodells zu kommunizieren. Diese Informationen werden normalerweise verwendet, um die Umstufung eines neu aktualisierten Vorhersagemodells in den Scoring-Modus in der Anwendung zu steuern.

Die Modellaktualisierung gibt in erster Linie ein neues Objekt `PredictiveModel` zurück, um den Vergleich der Evaluierungsdaten für das neu aktualisierte Modell mit den Daten des Originalmodells zu ermöglichen.

**Aktion:** Führt alle Dokumenterstellungsknoten (in erster Linie die nicht interaktiven Ausgabeknoten in den Ausgabe- und Diagrammpaletten von IBM SPSS Modeler) im Stream von IBM SPSS Modeler aus und gibt die resultierenden Dokumente zurück. Mithilfe von Streamparametern und den Eigenschaften von Quellen-/Terminalknoten kann eine gewisse Dynamik implementiert werden.

## Ausführung des Vorhersagemodellstreams

**Verwendung:** Alle Anwendungsfälle, die nicht in den Aktualisierungs- oder den Evaluierungsmodus oder die beiden Scoring-Modi passen, müssen mithilfe von Streamausführung bearbeitet werden.

**Aktion:** Über die Streamoptionen von IBM SPSS Modeler gesteuerte Ausführung. Entweder im Modus "Alle Endknoten", der dem Klicken auf die große grüne Pfeilschaltfläche in IBM SPSS Modeler Client entspricht, oder wie im Script auf Stromebene codiert. Mithilfe von Streamparametern und den Eigenschaften von Quellen-/Terminalknoten kann eine gewisse Dynamik implementiert werden.

## Vorhersagemodellscoring - externe Daten

**Verwendung:** Allgemein aufgrund der Art und Weise, in der ein vollständiger Stapel mit Eingabedaten verarbeitet wird, als "Stapelprozess" bezeichnet.

**Aktion:** Führt die Scoring-Verzweigung gemäß dem Entwurf aus. Eingaben aus konfigurierten Datenquellenknoten und generierte Vorhersageanalysen werden gemäß den Angaben des Endknotens als persistent definiert. Mithilfe von Streamparametern und den Eigenschaften von Quellen-/Terminalknoten kann eine gewisse Dynamik implementiert werden.

## Streamparameter für Vorhersagemodelle und Spezifikation der Eigenschaften von Quellen- und Endknoten

**Verwendung:** Eine allgemeine Methode ist das Definieren von in Ausdrücken referenzierten Parametern in Verzweigungen des Streams. Eine andere Steuermöglichkeit besteht im Ändern bestimmter Schlüsseleigenschaften eines Quellen- oder Endknotens, damit beispielsweise ein anderer Dateipfad verwendet wird usw.

**Aktion:** Bestimmte Aspekte der Ausführung eines Streams von IBM SPSS Modeler können von einer Anwendung dynamisch geändert werden.

**Anmerkung:** Parameteränderungen müssen vor der Erstellung von `PredictiveScorer` (siehe Beschreibung des eingebetteten Scorings oben) vorgenommen werden. Eigenschaften von Quellen- und Terminalknoten sind für dieses Objekt ohne Bedeutung.

---

# Kapitel 6. IBM SPSS Modeler Solution Publisher Runtime Library - API-Referenz

---

## Einführung

Mit der CLEMRTL-API können Sie die Ausführung veröffentlichter Streams über Ihre Anwendung steuern. In diesem Abschnitt werden die verfügbaren API-Funktionen beschrieben.

Folgende allgemeine Probleme treten bei der Verwendung der API auf:

- Die API besitzt eine C-Linkage für größtmögliche Kompatibilität, die Bibliothek weist jedoch C++-Abhängigkeiten auf. Dies kann bedeuten, dass sie auf einigen Plattformen nur mit einem C++-Linker verwendet werden kann.
- `clemrtl_initialize_ext()` muss aufgerufen werden, bevor andere Funktionen in der Bibliothek verwendet werden.
- Der Typ `clemrtl_image_handle` wird als Bild-ID verwendet.
- Jede Funktion gibt einen Indikatorcode aus. In der folgenden Tabelle sind die Werte der Statusindikatoren aufgeführt:

*Tabelle 6. Werte der Statusindikatoren*

| Ergebnis      | Beschreibung                                                                                                           |
|---------------|------------------------------------------------------------------------------------------------------------------------|
| CLEMRTL_OK    | Erfolg                                                                                                                 |
| CLEMRTL_FAIL  | Fehlgeschlagen, keine weiteren Details verfügbar                                                                       |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Informationen über den Fehler über die Funktion <code>clemrtl_getErrorDetail()</code> abrufbar |

## Überblick über API-Prozesse

Die allgemeine Gliederung einer Anwendung, die die API verwendet, lautet wie folgt:

1. Initialisieren der Bibliothek mithilfe von `clemrtl_initialize_ext()`.
2. Prüfen und Ändern der Ausführungsoptionen mithilfe von `clemrtl_getOption()` und `clemrtl_setOption()`.
3. Öffnen eines Bildes mithilfe von `clemrtl_openImage()` und Abrufen eines Bildhandle.
4. Prüfen und Ändern der Bildparameter mithilfe von `clemrtl_enumerateParameters()`, `clemrtl_getParameter()` und `clemrtl_setParameter()`.
5. Ausführen des Bildes mithilfe von `clemrtl_execute()`. Wenn dasselbe Bild mehrfach ohne Änderung der Parameter ausgeführt werden soll, verwenden Sie zunächst `clemrtl_prepare()`.
6. Schließen des Bildes mithilfe von `clemrtl_closeImage()`.
7. Zum Abbrechen einer laufenden Ausführung verwenden Sie `clemrtl_interrupt()`.
8. Um Informationen zum letzten Fehler abzurufen, verwenden Sie `clemrtl_getErrorDetail()`. Um Fehlermeldungen und andere Diagnosenachrichten bei ihrem Eintreffen abzurufen, verwenden Sie `clemrtl_setReportHandler()`.

Die Bibliothek ist unter folgenden Bedingungen für eine Verwendung mit mehreren Threads geeignet:

- Globale Funktionen, also Funktionen, die kein Bildhandle als Argument verwenden, können nicht gleichzeitig aufgerufen werden.

- Bildlokale Funktionen, also Funktionen, die ein Bildhandle als Argument verwenden, können nicht gleichzeitig aufgerufen werden, wenn sie auf dasselbe Bildhandle angewendet werden. Ausnahme: `clemrtl_interrupt()`.
- Bildlokale Funktionen können gleichzeitig aufgerufen werden, wenn sie auf verschiedenen Bildhandles angewendet werden.

Dies bedeutet insbesondere, dass separate Threads Bilder gleichzeitig vorbereiten und ausführen können, vorausgesetzt, Sie verwendet verschiedene Bildhandles. Verschiedene Bildhandles können sich auf dasselbe Bild beziehen (`clemrtl_openImage()` erstellt bei jedem Aufruf ein new-Handle, auch für dasselbe Bild), vergessen Sie aber nicht, die Bildparameter bei der Ausführung mehrerer Instanzen desselben Bildes zu ändern, um die Ausgabe nach Bedarf zu ändern. Wenn zwei gleichzeitig ausgeführte Bilder versuchen, in dieselbe Ausgabedatei oder Datenbanktabelle zu schreiben, sind die Ergebnisse nicht vorhersagbar.

## API-Funktionen

Im Folgenden finden Sie eine vollständige Liste der der API vorliegenden Funktionen.

### initialise

Diese Funktion wird nicht weiter unterstützt und entspricht der Funktion

```
initialise_ext(flags, 0, 0);
```

Neue Programme sollten *initialise\_ext* aufrufen.

### initialise\_ext

```
int clemrtl_initialize_ext(
 unsigned flags,
 int arg_count,
 const clemrtl_init_arg* args);
```

Initialisiert IBM SPSS Modeler Solution Publisher Runtime. *Diese Funktion muss vor allen anderen API-Funktionen aufgerufen werden.*

Tabelle 7. *initialise\_ext*-Parameter

| Parameter | Beschreibung                                                                                                                                                                                         |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| flags     | Steuert bestimmte Aspekte des Initialisierungsprozesses. Der Wert wird als bitweises OR der unten beschriebenen Flags konstruiert.                                                                   |
| arg_count | Die Anzahl der zusätzlichen Initialisierungsargumente. Dieser Wert muss kleiner oder Gleich der Länge des Arrays args sein.                                                                          |
| args      | Zusätzliche Initialisierungsargumente. Argumente werden durch den Typ beschrieben:<br><pre>typedef struct _clemrtl_init_arg {     const char* name;     const char* value; } clemrtl_init_arg;</pre> |

Das Feld name gibt den Namen einer Konfigurationseigenschaft an und das Feld value ihren Wert. Die in der folgenden Tabelle aufgeführten Konfigurationseigenschaften werden erkannt.

Tabelle 8. Konfigurationseigenschaften

| Name                   | Wert                                                                                                                                                                       |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| installation_directory | Der vollständige Pfad zum Installationsverzeichnis von IBM SPSS Modeler Solution Publisher. Der Installationsordner ist der Ordner, die die Ordner bin und config enthält. |

Gibt einen der in der folgenden Tabelle angezeigten Fehlercodes zurück.

*Tabelle 9. Fehlercodes*

| Ergebnis     | Beschreibung                                      |
|--------------|---------------------------------------------------|
| CLEMRTL_OK   | Erfolg.                                           |
| CLEMRTL_FAIL | Fehlgeschlagen, keine weiteren Details verfügbar. |

Mit dem Parameter `Flags` können bestimmte Aspekte des Initialisierungsprozesses gesteuert werden. Der Wert 0 (oder `CLEMRTL_INIT_DEFAULTS`) gibt das Standardverhalten wie folgt an:

- Festlegen der Standardzeitzone durch Aufrufen von `tzset()`.
- Setzen der Ländereinstellung `LC_NUMERIC` auf "C".
- Festlegen, dass der globale `new`-Handler einen Ausnahmefehler ausgibt, wenn der Arbeitsspeicher erschöpft ist.

Dieses Verhalten lässt sich durch Angabe von Kombinationen der in der folgenden Tabelle aufgeführten `Flags` ändern.

*Tabelle 10. Flags*

| Flag                                        | Beschreibung                                                                         |
|---------------------------------------------|--------------------------------------------------------------------------------------|
| <code>CLEMRTL_INIT_NO_TZ</code>             | Die Zeitzone wird nicht festgelegt.                                                  |
| <code>CLEMRTL_INIT_NO_LOCALE</code>         | Ländereinstellung wird nicht geändert.                                               |
| <code>CLEMRTL_INIT_NO_NEW_HANDLER</code>    | Es wird kein <code>new</code> -Handler festgelegt.                                   |
| <code>CLEMRTL_INIT_LOCAL_NEW_HANDLER</code> | Der <code>new</code> -Handlers wird innerhalb der einzelnen API-Aufrufe lokalisiert. |

Für die Bibliothek ist erforderlich, dass die Ländereinstellung `LC_NUMERIC` auf "C" gesetzt ist, um eine ordnungsgemäße Umwandlung zwischen Zahlen und Zeichenfolgen durchzuführen (das IBM SPSS Modeler-Zahlenformat verwendet die Regeln der Ländereinstellung "C"). Die beinhaltet auch die Umwandlung von aus Textdateien gelesenen bzw. in Textdateien geschriebenen Zahlen.

- Geben Sie `NO_LOCALE` an, wenn Ihre Anwendung von der Ländereinstellung `LC_NUMERIC` abhängt und keine Umwandlung zwischen Zahlen und Zeichenfolgen erforderlich ist.
- `NO_NEW_HANDLER` und `LOCAL_NEW_HANDLER` schließen sich gegenseitig aus. Für die Bibliothek ist es erforderlich, dass `new` einen Ausnahmefehler ausgibt, wenn der Arbeitsspeicher erschöpft ist.
- Geben Sie `NO_NEW_HANDLER` an, wenn Ihre Anwendung einen eigenen `new`-Handler festlegt oder wenn Sie sicher sind, dass der Compiler/das Runtime-System standardmäßig eine Ausnahme ausgibt.

Geben Sie `LOCAL_NEW_HANDLER` nur an, wenn es für Ihre Anwendung erforderlich ist, dass `new` einen Nullzeiger ausgibt, wenn der Arbeitsspeicher erschöpft ist. *Hinweis:* Dieser Vorgang ist für eine Multithread-Umgebung nicht sicher.

## getOption

```
int clemrtl_getOption(
 const char* name,
 char* value,
 int value_size);
```

Ruft den Wert einer Ausführungsoption ab.

*Tabelle 11. getOption-Parameter*

| Parameter         | Beschreibung     |
|-------------------|------------------|
| <code>name</code> | Der Optionsname. |

Tabelle 11. *getOption-Parameter (Forts.)*

| Parameter  | Beschreibung                                  |
|------------|-----------------------------------------------|
| value      | Ein Puffer für die Aufnahme des Optionswerts. |
| value_size | Die Größe des Wertepuffers.                   |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

Tabelle 12. *Statuscodes*

| Ergebnis      | Beschreibung                                      |
|---------------|---------------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                           |
| CLEMRTL_FAIL  | Fehlgeschlagen, keine weiteren Details verfügbar. |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar.        |

Der Vorgang schlägt fehl, wenn der Optionsname nicht zu den oben aufgeführten Ausführungsoptionsnamen gehört - beispielsweise `memory_usage` - oder wenn der Wertepuffer nicht groß genug ist, um den Wert einschließlich eines abschließenden Nullzeichens (NULL) aufzunehmen. Optionswerte sind immer Zeichenfolgen, auch wenn sie numerisch interpretiert werden. Weitere Informationen finden Sie in Kapitel 3, „Ausführen veröffentlichter Streams“, auf Seite 11.

## setOption

```
int clemrtl_setOption(
 const char* name,
 const char* value);
```

Legt den Wert einer Ausführungsoption fest.

Tabelle 13. *setOption-Parameter*

| Parameter | Beschreibung     |
|-----------|------------------|
| name      | Der Optionsname. |
| value     | Der Optionswert. |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

Tabelle 14. *Statuscodes*

| Ergebnis      | Beschreibung                                      |
|---------------|---------------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                           |
| CLEMRTL_FAIL  | Fehlgeschlagen, keine weiteren Details verfügbar. |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar.        |

Der Vorgang schlägt fehl, wenn der Optionsname nicht zu den oben aufgeführten Ausführungsoptionsnamen gehört, beispielsweise `memory_usage`. Optionswerte sind immer Zeichenfolgen, auch wenn sie numerisch interpretiert werden.

## setLogFile

```
int clemrtl_setLogFile(
 int mode,
 const char* log_file);
```

Leitet Protokollnachrichten von allen danach geöffneten Bildern um.



Tabelle 15. setLogFile-Parameter

| Parameter | Beschreibung                                                                                                                                                                                                                                                                          |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| mode      | Der Protokollierungsmodus muss einen der folgenden Werte aufweisen:<br>CLEMRTL_NULL_LOG - Nachrichten unterdrücken.<br>CLEMRTL_STDERROR_LOG - Nachrichten an die Standard-Fehlerausgabe senden.<br>CLEMRTL_FILE_LOG - Nachrichten an die im Parameter log_file genannte Datei senden. |
| log_file  | Zeichenfolge, die den Dateinamen enthält.                                                                                                                                                                                                                                             |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

Tabelle 16. Statuscodes

| Ergebnis      | Beschreibung                                      |
|---------------|---------------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                           |
| CLEMRTL_FAIL  | Fehlgeschlagen, keine weiteren Details verfügbar. |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar.        |

Der Vorgang schlägt fehl, wenn es sich beim Modus nicht um einen der erkannten Modi handelt oder wenn eine Dateiprotokollierung angefordert wurde und der Dateiname nicht angegeben wurde oder ungültig ist.

## openImage

```
int clemrtl_openImage(
 const char* image_file,
 const char* param_file,
 clemrtl_image_handle* handle);
```

Öffnet ein veröffentlichtes Streambild.

Tabelle 17. openImage-Parameter

| Parameter  | Beschreibung                 |
|------------|------------------------------|
| image_file | Der Name der Bilddatei.      |
| param_file | Der Name der Parameterdatei. |
| handle     | Empfängt das Bildhandle.     |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

Tabelle 18. Statuscodes

| Ergebnis      | Beschreibung                                      |
|---------------|---------------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                           |
| CLEMRTL_FAIL  | Fehlgeschlagen, keine weiteren Details verfügbar. |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar.        |

Die Parameterdatei kann NULL sein, wenn keine gesonderte Parameterdatei erforderlich ist. Der Vorgang schlägt fehl, wenn die Bilddatei nicht angegeben wurde, wenn der Dateiname ungültig oder der Inhalt nicht lesbar ist. Wenn der Vorgang erfolgreich ist, kann das zurückgegebene Bildhandle zur Identifikation der Bildinstanz in nachfolgenden API-Aufrufen verwendet werden.

## closeImage

```
int clemrtl_closeImage(
 clemrtl_image_handle handle);
```

Schließt ein Bildhandle und gibt seine Ressourcen frei.

*Tabelle 19. closeImage-Parameter*

| Parameter | Beschreibung    |
|-----------|-----------------|
| handle    | Das Bildhandle. |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

*Tabelle 20. Statuscodes*

| Ergebnis      | Beschreibung                                      |
|---------------|---------------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                           |
| CLEMRTL_FAIL  | Fehlgeschlagen, keine weiteren Details verfügbar. |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar.        |

Der Vorgang schlägt fehl, wenn ein anderer API-Aufruf für das Bild läuft, insbesondere, wenn das Bild noch immer ausgeführt wird. Wenn der Vorgang erfolgreich ist, wird die Bilddatei geschlossen und das Handle kann nicht in einem nachfolgenden API-Aufruf werden, außer bei einem weiteren Aufruf von `clemrtl_closeImage()`, was jedoch ohne Wirkung bleibt.

## enumerateParameters

```
int clemrtl_enumerateParameters(
 clemrtl_image_handle handle,
 clemrtl_parameter_proc proc,
 void* data);
```

Wendet auf jeden Bildparameternamen und -wert eine Callback-Prozedur an.

*Tabelle 21. enumerateParameters-Parameter*

| Parameter | Beschreibung                               |
|-----------|--------------------------------------------|
| handle    | Das Bildhandle.                            |
| proc      | Die Parameter-Callback-Prozedur.           |
| data      | Benutzerdefinierte Daten für den Callback. |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

*Tabelle 22. Statuscodes*

| Ergebnis      | Beschreibung                               |
|---------------|--------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                    |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar. |

Die Callback-Prozedur weist folgenden Typ auf:

```
typedef void (*clemrtl_parameter_proc)(
 void* data,
 const char* name,
 const char* value);
```

Tabelle 23. Parameter der Callback-Prozedur

| Parameter | Beschreibung                                                                           |
|-----------|----------------------------------------------------------------------------------------|
| data      | An <code>clemrtl_enumerateParameters()</code> weitergegebene benutzerdefinierte Daten. |
| name      | Der Parametername.                                                                     |
| value     | Der Wert des Parameters.                                                               |

Die Prozedur wird in beliebiger Reihenfolge auf jeden Parameter genau einmal angewendet.

### getParameter

```
int clemrtl_getParameter(
 clemrtl_image_handle handle,
 const char* name,
 char* value,
 int value_size);
```

Ruft den Wert eines Bildparameters ab.

Tabelle 24. `getParameter`-Parameter

| Parameter  | Beschreibung                                    |
|------------|-------------------------------------------------|
| handle     | Das Bildhandle.                                 |
| name       | Der Parametername.                              |
| value      | Ein Puffer für die Aufnahme des Parameterwerts. |
| value_size | Größe des Wertepuffers.                         |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

Tabelle 25. Statuscodes

| Ergebnis      | Beschreibung                                      |
|---------------|---------------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                           |
| CLEMRTL_FAIL  | Fehlgeschlagen, keine weiteren Details verfügbar. |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar.        |

Der Vorgang schlägt fehl, wenn der Parametername nicht mit dem Namen eines Parameters im Bild übereinstimmt oder wenn der Wertepuffer nicht groß genug ist, um den Wert einschließlich eines abschließenden Nullzeichens aufzunehmen. Parameternamen verwenden das oben beschriebene Format `Name.Attribut`, beispielsweise `Datei0.Name`, und Parameterwerte sind immer Zeichenfolgen, auch wenn sie numerisch interpretiert werden.

### setParameter

```
int clemrtl_setParameter(
 clemrtl_image_handle handle,
 const char* name,
 const char* value);
```

Legt den Wert eines Bildparameters fest.

Tabelle 26. `setParameter`-Parameter

| Parameter | Beschreibung    |
|-----------|-----------------|
| handle    | Das Bildhandle. |

Tabelle 26. setParameter-Parameter (Forts.)

| Parameter | Beschreibung             |
|-----------|--------------------------|
| name      | Der Name des Parameters. |
| value     | Der Wert des Parameters. |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

Tabelle 27. Statuscodes

| Ergebnis      | Beschreibung                               |
|---------------|--------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                    |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar. |

Der Vorgang schlägt fehl, wenn das Bildhandle ungültig ist oder wenn der Parametername nicht mit dem Namen eines Parameters im Bild übereinstimmt. Parameternamen verwenden das oben beschriebene Format Name.Attribut, beispielsweise Datei0.Name, und Parameterwerte sind immer Zeichenfolgen, auch wenn sie numerisch interpretiert werden.

### getFieldCount

```
int clemrtl_getFieldCount(
 clemrtl_image_handle handle,
 const char* key,
 size_t* field_count);
```

Ergibt die Anzahl der Felder in einer Eingabequelle oder einem Ausgabeziel.

Tabelle 28. getFieldCount-Parameter

| Parameter   | Beschreibung                                                                                                                                     |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| handle      | Das Bildhandle.                                                                                                                                  |
| key         | Der Name der zu untersuchenden Ein- bzw. Ausgabe laut Parameterdatei. Der Schlüssel (key) kann sich auf eine Datei oder eine Datenbank beziehen. |
| field_count | Ruft die Anzahl der Felder ab.                                                                                                                   |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

Tabelle 29. Statuscodes

| Ergebnis      | Beschreibung                                      |
|---------------|---------------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                           |
| CLEMRTL_FAIL  | Fehlgeschlagen, keine weiteren Details verfügbar. |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar.        |

### getFieldTypes

```
int clemrtl_getFieldTypes(
 clemrtl_image_handle handle,
 const char* key,
 size_t field_count,
 int* field_types);
```

Ergibt die Feldtypen für eine Eingabequelle oder ein Ausgabeziel.

Tabelle 30. *getFieldTypes*-Parameter

| Parameter   | Beschreibung                                                                                                                                                                                                                                                                                                                                                                        |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| handle      | Das Bildhandle.                                                                                                                                                                                                                                                                                                                                                                     |
| key         | Der Name der zu untersuchenden Ein- bzw. Ausgabe laut Parameterdatei. Der Schlüssel (key) kann sich auf eine Datei oder eine Datenbank beziehen.                                                                                                                                                                                                                                    |
| field_count | Die Anzahl der zu untersuchenden Felder. Dieser Wert muss kleiner oder Gleich der Länge des Arrays field_types sein.                                                                                                                                                                                                                                                                |
| field_types | Ein Array mit mindestens der Länge field_count, das die Feldtypen aufnimmt. Die Typen der ersten field_count-Felder in der Ein- bzw. Ausgabe werden in das Array kopiert. Wenn field_count größer ist als die tatsächliche Anzahl von Feldern bleiben die zusätzlichen Elemente im Array undefiniert. Die Werte für die Feldtypen sind unten in der Tabelle "Datentypen" angegeben. |

## Datentypen

field\_types muss einer der in der folgenden Tabelle dargestellte Datentypen sein.

Tabelle 31. *Datentypen*

| Typ       | Interpretation                                                                             | Typisches C-Deklarationssymbol |
|-----------|--------------------------------------------------------------------------------------------|--------------------------------|
| STRING    | Mit Null abgeschlossene UTF-8-Zeichenfolge.                                                | const char*                    |
| INTEGER   | Ganze Zahl mit Vorzeichen (32 Bit).                                                        | int                            |
| LONG      | Ganze Zahl mit Vorzeichen (64 Bit).                                                        | long long                      |
| REAL      | 64-Bit-Gleitkommazahl.                                                                     | double                         |
| TIME      | Ganze Zahl mit Vorzeichen (64 Bit), die die Sekunden seit 0.00 Uhr angibt.                 | long long                      |
| DATE      | Ganze Zahl mit Vorzeichen (64 Bit), die die Sekunden seit dem 01.01.1970, 0.00 Uhr angibt. | long long                      |
| TIMESTAMP | Ganze Zahl mit Vorzeichen (64 Bit), die die Sekunden seit dem 01.01.1970, 0.00 Uhr angibt. | long long                      |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

Tabelle 32. *Statuscodes*

| Ergebnis      | Beschreibung                                      |
|---------------|---------------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                           |
| CLEMRTL_FAIL  | Fehlgeschlagen, keine weiteren Details verfügbar. |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar.        |

## setAlternativeInput

```
int clemrtl_setAlternativeInput(
 clemrtl_image_handle handle,
 const char* key,
```

```

size_t field_count,
int field_types,
void** (*iterator)(void* arg),
void* arg);

```

Ersetzt eine Dateieingabequelle durch eine alternative Eingabequelle.

*Tabelle 33. setAlternativeInput-Parameter*

| Parameter   | Beschreibung                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| handle      | Das Bildhandle.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| key         | Der Name der ursprünglichen Eingabequelle laut Parameterdatei. Bei der Eingabequelle muss es sich um eine Datei handeln. Daher hat der Schlüssel (key) für eine ganze Zahl $N \geq 0$ immer das Format "DateiN".                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| field_count | Die Anzahl der Felder in der Eingabe. Der Wert muss genau mit der Anzahl der Felder in der ursprünglichen Dateieingabe übereinstimmen. Andernfalls schlägt der Aufruf fehl.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| field_types | Ein Array von Typen mit mindestens der Länge field_count. Der Wert field_types[i] ist der Typ des i-ten Felds und muss einen der unten in der Tabelle "Datentypen" aufgeführten Werte aufweisen. Die Typen müssen mit den Typen der ursprünglichen Dateieingabe kompatibel sein. Andernfalls schlägt der Aufruf fehl.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| iterator    | <p>Eine Funktion, die die alternativen Eingabedaten erstellt. Die Funktion wird wie folgt auf das Argument angewendet:</p> <pre>void** row = iterator(arg);</pre> <p>Die Funktion wird während der Ausführung aufgerufen (innerhalb eines clemrtl_execute-Aufrufs) und wird für jeden Eingabedatensatz einmal aufgerufen. Der Ausgabewert "NULL" zeigt das Ende der Eingabe an. In diesem Fall wird die Funktion nicht erneut aufgerufen und die Ausführung endet schließlich. Andernfalls ist das Ergebnis ein Datenarray mit mindestens der Länge field_count, wobei row[i] den Wert des i-ten Felds angibt. Ein Wert kann "NULL" sein. Andernfalls muss er als Zeiger auf ein Datum verweisen, dessen Typ durch die entsprechenden Werte von field_types[i] festgelegt wird. Die Zeiger müssen bis zum nächsten Aufruf des Iterators gültig sein, oder alternativ bis zum Ende der Ausführung, wenn diese vorzeitig endet.</p> |
| arg         | Ein nicht transparentes Argument, das bei jedem Aufruf an den Iterator weitergeleitet wird.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

## Datentypen

field\_types muss ein Element aus einer begrenzten Liste von Typen sein. Weitere Informationen finden Sie im Thema „getFieldTypes“ auf Seite 30.

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

*Tabelle 34. Statuscodes*

| Ergebnis      | Beschreibung                                      |
|---------------|---------------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                           |
| CLEMRTL_FAIL  | Fehlgeschlagen, keine weiteren Details verfügbar. |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar.        |

Die Operation schlägt fehl, wenn die Feldtypen nicht mit denen der ursprünglichen Dateieingabe kompatibel sind, oder wenn die Feldanzahl nicht exakt mit der Anzahl der Felder in der ursprünglichen Eingabedatei übereinstimmt.

## setAlternativeOutput

```
int clemrtl_setAlternativeOutput(
 clemrtl_image_handle handle,
 const char* key,
 size_t field_count,
 int field_types,
 void (*iterator)(void* arg, void** row),
 void* arg);
```

Ersetzt ein Dateiausgabeziel durch ein alternatives Ausgabeziel.

Tabelle 35. setAlternativeOutput-Parameter

| Parameter   | Beschreibung                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| handle      | Das Bildhandle.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| key         | Der Name des ursprünglichen Ausgabeziels laut Parameterdatei. Das Eingabeziel muss eine Datei sein. Daher hat der Schlüssel (key) für eine ganze Zahl $N \geq 0$ immer das Format "DateiN".                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| field_count | Die Anzahl der Felder in der Ausgabe. Der Wert muss genau mit der Anzahl der Felder in der ursprünglichen Dateiausgabe übereinstimmen. Andernfalls schlägt der Aufruf fehl.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| field_types | Ein Array von Typen mit mindestens der Länge field_count. Der Wert field_types[i] ist der Typ des i-ten Felds und muss einen der unten in der Tabelle "Datentypen" aufgeführten Werte aufweisen. Die Typen müssen mit den Typen der ursprünglichen Dateiausgabe kompatibel sein. Andernfalls schlägt der Aufruf fehl.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| iterator    | <p>Eine Funktion, die die Bildausgabe verarbeitet. Die Funktion wird wie folgt auf das Argument und eine Datenzeile angewendet:<br/>iterator(arg, row);</p> <p>Die Funktion wird während der Ausführung aufgerufen (innerhalb eines clemrtl_execute-Aufrufs) und wird für jede Ergebniszeile einmal aufgerufen. Der Wert "NULL" für row (Zeile) zeigt das Ende der Ausgabe an. Danach wird die Funktion nicht mehr aufgerufen. Eine Anwendung sollte sich nicht von diesem letzten Aufruf abhängig sein und nach dem Ende der Ausführung alle externen Ressourcen leeren und schließen. Andernfalls ist row ein Datenarray mit mindestens der Länge field_count, wobei row[i] den Wert des i-ten Felds angibt. Ein Wert kann "NULL" sein. Andernfalls muss er als Zeiger auf ein Datum verweisen, dessen Typ durch die entsprechenden Werte von field_types[i] festgelegt wird. Die Funktion muss alle Datenwerte, die sie benötigt, kopieren, da der Arbeitsspeicher nach Rückgabe des Aufrufs eventuell nicht beibehalten wird.</p> |
| arg         | Ein nicht transparentes Argument, das bei jedem Aufruf an den Iterator weitergeleitet wird.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

## Datentypen

field\_types muss ein Element aus einer begrenzten Liste von Typen sein. Weitere Informationen finden Sie im Thema „getFieldTypes“ auf Seite 30.

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

*Tabelle 36. Statuscodes*

| Ergebnis      | Beschreibung                                      |
|---------------|---------------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                           |
| CLEMRTL_FAIL  | Fehlgeschlagen, keine weiteren Details verfügbar. |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar.        |

Die Operation schlägt fehl, wenn die Feldtypen nicht mit denen der ursprünglichen Dateiausgabe kompatibel sind, oder wenn die Feldanzahl nicht exakt mit der Anzahl der Felder in der ursprünglichen Ausgabedatei übereinstimmt.

### **execute**

```
int clemrtl_execute(clemrtl_image_handle handle);
```

Führt ein Bild aus.

*Tabelle 37. execute-Parameter*

| Parameter | Beschreibung    |
|-----------|-----------------|
| handle    | Das Bildhandle. |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

*Tabelle 38. Statuscodes*

| Ergebnis      | Beschreibung                               |
|---------------|--------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                    |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar. |

Wenn das Bild nicht vorbereitet wurde, wird es zuerst mithilfe der aktuellen Parameterwerte vorbereitet. Der Vorgang schlägt fehl, wenn das Bildhandle ungültig ist oder wenn während der Vorbereitung oder Ausführung ein Fehler auftritt. Der Aufruf kommt nicht wieder, bis die Ausführung abgeschlossen ist.

### **prepare**

```
int clemrtl_prepare(clemrtl_image_handle handle);
```

Bereitet ein Bild für die Ausführung vor.

*Tabelle 39. prepare-Parameter*

| Parameter | Beschreibung    |
|-----------|-----------------|
| handle    | Das Bildhandle. |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

*Tabelle 40. Statuscodes*

| Ergebnis      | Beschreibung                               |
|---------------|--------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                    |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar. |



Der Vorgang schlägt fehl, wenn das Bildhandle ungültig ist oder wenn während der Vorbereitung ein Fehler auftritt.

Bilder müssen vor der Ausführung vorbereitet werden. Bei der Vorbereitung eines Bildes werden Parameterwerte im Bild fixiert. `clemrtl_execute()` bereitet bei jedem Aufruf automatisch ein Bild vor, wenn es noch nicht bereits vorbereitet wurde. Die Verwendung von `clemrtl_prepare()` zur Vorbereitung eines Bildes ist sinnvoll, wenn das Bild mehrmals mit denselben Parameterwerten ausgeführt werden soll, und kann zu einer erheblichen Leistungssteigerung führen. Nach der Vorbereitung eines Bildes werden nachfolgende Änderungen an Parameterwerten ignoriert; verwenden Sie wieder `clemrtl_prepare()`, um das Bild mit den neuen Parameterwerten zu aktualisieren.

## interrupt

```
int clemrtl_interrupt(clemrtl_image_handle handle);
```

Beendet die für ein Bild laufende Ausführung.

*Tabelle 41. interrupt-Parameter*

| Parameter | Beschreibung    |
|-----------|-----------------|
| handle    | Das Bildhandle. |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

*Tabelle 42. Statuscodes*

| Ergebnis      | Beschreibung                               |
|---------------|--------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                    |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar. |

Der Vorgang bleibt ohne Wirkung, wenn das Bildhandle ungültig ist oder das Bild nicht ausgeführt wird.

Diese Funktion kann problemlos mit einem weiteren API-Aufruf für dasselbe Bildhandle aufgerufen werden.

## getErrorDetail

```
int clemrtl_getErrorDetail(
 clemrtl_image_handle handle,
 char* severity,
 int* code,
 char* text,
 int text_size);
```

Ruft detaillierte Informationen zu dem letzten Fehler ab, der bei einem Bild auftrat.

*Tabelle 43. getErrorDetail-Parameter*

| Parameter | Beschreibung                                                                                                                        |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------|
| handle    | Das Bildhandle.                                                                                                                     |
| severity  | Ruft den Code für den Schweregrad als einzelnes Zeichen ab:<br><br>I - Information<br>W - Warnung<br>E - Fehler<br>X - Systemfehler |
| code      | Empfängt die Fehlernummer.                                                                                                          |

Tabelle 43. *getErrorDetail-Parameter (Forts.)*

| Parameter | Beschreibung                                       |
|-----------|----------------------------------------------------|
| text      | Ein Puffer für die Aufnahme des Nachrichtentextes. |
| text_size | Größe des Textpuffers.                             |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

Tabelle 44. *Statuscodes*

| Ergebnis      | Beschreibung                               |
|---------------|--------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                    |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar. |

Die Ergebnisse sind nicht verlässlich, wenn ein Bildhandle ungültig ist. Der Nachrichtentext wird, falls erforderlich gekürzt, damit er in den Textpuffer passt.

Wenn ein API-Aufruf mit CLEMRTL\_ERROR fehlschlägt und der Fehler nicht mit einem bestimmten Bildhandle in Bezug steht, werden bei Weitergabe von 0 als Bildhandle Einzelheiten des letzten nicht bildspezifischen Fehlers abgerufen, doch das Ergebnis ist in einer Umgebung mit mehreren Threads nicht verlässlich.

### setReportHandler

```
int clemrtl_setReportHandler(
 clemrtl_image_handle handle,
 clemrtl_report_proc proc,
 void* data);
```

Installiert eine Berichts-Callback-Prozedur für ein Bild.

Tabelle 45. *setReportHandler-Parameter*

| Parameter | Beschreibung                               |
|-----------|--------------------------------------------|
| handle    | Das Bildhandle.                            |
| proc      | Die Berichts-Callback-Prozedur.            |
| data      | Benutzerdefinierte Daten für den Callback. |

Gibt einen der in der folgenden Tabelle angezeigten Statuscodes zurück.

Tabelle 46. *Statuscodes*

| Ergebnis      | Beschreibung                               |
|---------------|--------------------------------------------|
| CLEMRTL_OK    | Erfolg.                                    |
| CLEMRTL_ERROR | Fehlgeschlagen, weitere Details verfügbar. |

Die Callback-Prozedur weist folgenden Typ auf:

```
typedef void (*clemrtl_report_proc)(
 void* data,
 char severity,
 int code,
 const char* text);
```

Tabelle 47. Parameter der Callback-Prozedur

| Parameter | Beschreibung                                                                                                                |
|-----------|-----------------------------------------------------------------------------------------------------------------------------|
| data      | An <code>clemrt1_setReportHandler()</code> weitergegebene benutzerdefinierte Daten.                                         |
| severity  | Der Code für den Schweregrad als einzelnes Zeichen:<br><br>I - Information<br>W - Warnung<br>E - Fehler<br>X - Systemfehler |
| code      | Nachrichtenummer.                                                                                                           |
| text      | Nachrichtentext.                                                                                                            |

Die Prozedur wird bei jedem Eintreffen auf die einzelnen Nachrichten angewendet. Wenn die Prozedur auf NULL gesetzt wird, werden alle bestehenden Handler entfernt, die auf dem Bild installiert sind.



---

## Bemerkungen

Diese Informationen wurden für weltweit angebotene Produkte und Dienstleistungen erarbeitet.

Möglicherweise bietet IBM die in dieser Dokumentation beschriebenen Produkte, Services oder Funktionen in anderen Ländern nicht an. Informationen über die gegenwärtig im jeweiligen Land verfügbaren Produkte und Services sind beim zuständigen IBM Ansprechpartner erhältlich. Hinweise auf IBM Lizenzprogramme oder andere IBM Produkte bedeuten nicht, dass nur Programme, Produkte oder Services von IBM verwendet werden können. Anstelle der IBM Produkte, Programme oder Services können auch andere, ihnen äquivalente Produkte, Programme oder Services verwendet werden, solange diese keine gewerblichen oder anderen Schutzrechte von IBM verletzen. Die Verantwortung für den Betrieb von Produkten, Programmen und Services anderer Anbieter liegt beim Kunden.

Für in diesem Handbuch beschriebene Erzeugnisse und Verfahren kann es IBM Patente oder Patentanmeldungen geben. Mit der Auslieferung dieses Handbuchs ist keine Lizenzierung dieser Patente verbunden. Lizenzanforderungen sind schriftlich an folgende Adresse zu richten (Anfragen an diese Adresse müssen auf Englisch formuliert werden):

IBM Director of Licensing  
IBM Europe, Middle East & Africa  
Tour Descartes  
2, avenue Gambetta  
92066 Paris La Defense  
France

Verweise in diesen Informationen auf Websites anderer Anbieter werden lediglich als Service für den Kunden bereitgestellt und stellen keinerlei Billigung des Inhalts dieser Websites dar. Das über diese Websites verfügbare Material ist nicht Bestandteil des Materials für dieses IBM Produkt. Die Verwendung dieser Websites geschieht auf eigene Verantwortung.

Werden an IBM Informationen eingesandt, können diese beliebig verwendet werden, ohne dass eine Verpflichtung gegenüber dem Einsender entsteht.

Lizenznehmer des Programms, die Informationen zu diesem Produkt wünschen mit der Zielsetzung: (i) den Austausch von Informationen zwischen unabhängig voneinander erstellten Programmen und anderen Programmen (einschließlich des vorliegenden Programms) sowie (ii) die gemeinsame Nutzung der ausgetauschten Informationen zu ermöglichen, wenden sich an folgende Adresse:

IBM Software Group  
ATTN: Licensing  
200 W. Madison St.  
Chicago, IL; 60606  
USA

Die Bereitstellung dieser Informationen kann unter Umständen von bestimmten Bedingungen - in einigen Fällen auch von der Zahlung einer Gebühr - abhängig sein.

Die Lieferung des in diesem Dokument beschriebenen Lizenzprogramms sowie des zugehörigen Lizenzmaterials erfolgt auf der Basis der IBM Rahmenvereinbarung bzw. der Allgemeinen Geschäftsbedingungen von IBM, der IBM Internationalen Nutzungsbedingungen für Programmpakete oder einer äquivalenten Vereinbarung.

Alle in diesem Dokument enthaltenen Leistungsdaten stammen aus einer kontrollierten Umgebung. Die Ergebnisse, die in anderen Betriebsumgebungen erzielt werden, können daher erheblich von den hier erzielten Ergebnissen abweichen. Einige Daten stammen möglicherweise von Systemen, deren Entwicklung noch nicht abgeschlossen ist. Eine Gewährleistung, dass diese Daten auch in allgemein verfügbaren Systemen erzielt werden, kann nicht gegeben werden. Darüber hinaus wurden einige Daten unter Umständen durch Extrapolation berechnet. Die tatsächlichen Ergebnisse können davon abweichen. Benutzer dieses Dokuments sollten die entsprechenden Daten in ihrer spezifischen Umgebung prüfen.

Alle Informationen zu Produkten anderer Anbieter stammen von den Anbietern der aufgeführten Produkte, deren veröffentlichten Ankündigungen oder anderen allgemein verfügbaren Quellen. IBM hat diese Produkte nicht getestet und kann daher keine Aussagen zu Leistung, Kompatibilität oder anderen Merkmalen machen. Fragen zu den Leistungsmerkmalen von Produkten anderer Anbieter sind an den jeweiligen Anbieter zu richten.

Aussagen über Pläne und Absichten von IBM unterliegen Änderungen oder können zurückgenommen werden und repräsentieren nur die Ziele von IBM.

Diese Veröffentlichung enthält Beispiele für Daten und Berichte des alltäglichen Geschäftsablaufs. Sie sollen nur die Funktionen des Lizenzprogramms illustrieren und können Namen von Personen, Firmen, Marken oder Produkten enthalten. Alle diese Namen sind frei erfunden; Ähnlichkeiten mit tatsächlichen Namen und Adressen sind rein zufällig.

---

## Marken

IBM, das IBM Logo und [ibm.com](http://ibm.com) sind Marken oder eingetragene Marken der IBM Corporation in den USA und/oder anderen Ländern. Weitere Produkt- und Servicennamen können Marken von IBM oder anderen Unternehmen sein. Eine aktuelle Liste der IBM Marken finden Sie auf der Webseite "Copyright and trademark information" unter [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

Intel, das Intel-Logo, Intel Inside, das Intel Inside-Logo, Intel Centrino, das Intel Centrino-Logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium und Pentium sind Marken oder eingetragene Marken der Intel Corporation oder der Tochtergesellschaften des Unternehmens in den USA und anderen Ländern.

Linux ist eine eingetragene Marke von Linus Torvalds in den USA, anderen Ländern oder beidem.

Microsoft, Windows, Windows NT und das Windows-Logo sind Marken der Microsoft Corporation in den USA und/oder anderen Ländern.

UNIX ist eine eingetragene Marke von The Open Group in den USA und anderen Ländern.

Java und alle auf Java basierenden Marken und Logos sind Marken oder eingetragene Marken der Oracle Corporation und/oder ihrer verbundenen Unternehmen.

Weitere Produkt- und Servicennamen können Marken von IBM oder anderen Unternehmen sein.

---

# Index

## A

- Anwendungsbeispiele 3
- API
  - alternative Ausgabe 33
  - alternative Eingabe 31
  - Ausführungsoptionen 25, 26
  - Fehlercodes 23
  - Feldanzahl 30
  - Feldtypen 30
  - Referenz 23
  - setReportHandler 36
  - Statuscodes 23
  - Übersicht 13, 23
- API-Funktionen
  - closeImage 23, 28
  - enumerateParameters 23, 28
  - execute 23, 34
  - getErrorDetail 23, 35
  - getFieldCount 23, 30
  - getFieldTypes 23, 30
  - getOption 23, 25
  - getParameter 23, 29
  - Initialisierung 23, 24
  - interrupt 23, 35
  - openImage 23, 27
  - prepare 23, 34
  - setAlternativeInput 23, 31
  - setAlternativeOutput 23, 33
  - setLogFile 23, 26
  - setOption 23, 26
  - setParameter 23, 29
  - setReportHandler 23
- Ausführen von Streams
  - mit IBM SPSS Modeler Solution Publisher 11, 12
- Ausgabeknoten 7, 21

## B

- Beispiele
  - Anwendungshandbuch 3
  - Übersicht 5
- Bereitstellung 7, 21

## C

- closeImage
  - API-Funktionsreferenz 28
  - Überblick über API-Prozesse 23

## D

- Deinstallation
  - IBM SPSS Modeler Solution Publisher Runtime 19
- Dokumentation 3

## E

- enumerateParameters
  - API-Funktionsreferenz 28
  - Überblick über API-Prozesse 23
- EPM
  - API 21
- execute
  - API-Funktionsreferenz 34
  - Überblick über API-Prozesse 23

## F

- Fehlerberichte
  - API-Funktionsreferenz 26, 35, 36
- Fehlercodes 23

## G

- getErrorDetail
  - API-Funktionsreferenz 35
  - Überblick über API-Prozesse 23
- getFieldCount
  - API-Funktionsreferenz 30
  - Überblick über API-Prozesse 23
- getFieldTypes
  - API-Funktionsreferenz 30
  - Überblick über API-Prozesse 23
- getOption
  - API-Funktionsreferenz 25
  - Überblick über API-Prozesse 23
- getParameter
  - API-Funktionsreferenz 29
  - Überblick über API-Prozesse 23

## I

- IBM SPSS Embeddable Predictive Analytics
  - API 21
- IBM SPSS Modeler 1
  - Dokumentation 3
- IBM SPSS Modeler Server 1
- IBM SPSS Modeler Solution Publisher 7, 8, 21
  - Einbettung in Anwendungen 13
  - Übersicht 7
- IBM SPSS Modeler Solution Publisher, Knoten 21
- IBM SPSS Modeler Solution Publisher Runtime
  - Deinstallation 19
  - Installation 15
  - starten 18
  - Systemanforderungen 15, 16
  - Temp-Verzeichnis 18
- initialise
  - API-Funktionsreferenz 24
  - Flags 24
  - Überblick über API-Prozesse 23

- initialise\_ext
  - API-Funktionsreferenz 24
  - Flags 24
- interrupt
  - API-Funktionsreferenz 35
  - Überblick über API-Prozesse 23

## L

- Ländereinstellung
  - über API festlegen 24

## O

- Öffnen
  - Streams 27
- openImage
  - API-Funktionsreferenz 27
  - Überblick über API-Prozesse 23

## P

- PAR-Datei 8
- Parameter
  - API-Bildparameter 28, 29
  - API-Callback-Prozeduren 28
  - API-Parameterdateien 27, 28
  - Parameter für Streamausführung 12
- PIM-Datei 8
- prepare
  - API-Funktionsreferenz 34
  - Überblick über API-Prozesse 23
- Protokolldateien 26

## R

- Runtime
  - Deinstallation 19
  - Optionen 11
  - starten 18
  - Temp-Verzeichnis 18
  - UNIX-Installation 16
  - Windows-Installation 16
- Runtime-Programmiersbibliothek (CLEM-RTL) 13

## S

- Schließen
  - Streams 28
- setAlternativeInput
  - API-Funktionsreferenz 31
  - Überblick über API-Prozesse 23
- setAlternativeOutput
  - API-Funktionsreferenz 33
  - Überblick über API-Prozesse 23
- setLogFile
  - API-Funktionsreferenz 26
  - Überblick über API-Prozesse 23

- setOption
  - API-Funktionsreferenz 26
  - Überblick über API-Prozesse 23
- setParameter
  - API-Funktionsreferenz 29
  - Überblick über API-Prozesse 23
- setReportHandler
  - API-Funktionsreferenz 36
  - Überblick über API-Prozesse 23
- Statuscodes 23

## U

- UNIX
  - IBM SPSS Modeler Solution Publisher
  - Runtime installieren 16

## V

- Veröffentlichen von Streams
  - IBM SPSS Modeler Solution Publisher 7, 8
- Veröffentlichungsknoten 21
- Verzeichnis für temporäre Dateien
  - IBM SPSS Modeler Solution Publisher
  - Runtime 18

## W

- Windows
  - IBM SPSS Modeler Solution Publisher
  - Runtime installieren 15

## Z

- Zeitzone
  - über API festlegen 24





