Statistisches Bundesamt

Release Notes zu .CORE für JAVA

Freigegebene Änderungen/Erweiterungen der Softwarebibliotheken zu .CORE

Version: 1.10
Stand: 02.01.2023
Status: Freigegeben
Kontakt: eSTATISTIK.core@destatis.de

Inhalt

Änderungsverlauf

Version	Datum	Änderung
1.0	26.03.12	Neuerstellung
1.1	21.05.12	Fehlerbehebung CORE.inspector Version 1.3.1
1.2	29.11.12	Fehlerbehebung CORE.inspector Version 1.3.1 und CORE.connect Version 1.3.2 Erweiterungen im „JAVA-Projekt mit Beispielen Version 1.1“
1.3	08.05.14	Umstellung auf JAVA 7 und die mögliche Verwendung von TLS 1.2. Optionale Erweiterung der expliziten Setzung der Prüfstufe zur Prüfung einer Datenlieferung. Zusätzliche Zertifikatsprüfung anhand der in JAVA hinterlegten Wurzelzertifikate.
1.4	03.06.15	Fehlerbehebung CORE.inspector Version 1.7.0 und CORE.connect Version 1.7.0. Aufnahme eines CSV-Prüfmoduls zur internen Prüfungen beim Einlesen von CSV-Dateien anhand der dazugehörigen Erhebungsbeschreibung. Unterstützung der Softwarebibliotheken zu JAVA 8.
1.5	11.08.15	Version der Java-Bibliothek nun zusätzlich in Manifest-Datei hinterlegt. Fehlerbehebung / Erweiterung CORE.connect Version 1.7.3. Anpassung des „JAVA-Projekt mit Beispielen Version 1.4“ an JAVA 8
1.6	28.10.15	Fehlerbehebung CORE.inspector Version 1.7.5 und CORE.connect Version 1.7.5.
1.7	12.06.18	Anpassungen für CORE.inspector und CORE.connect zur Version 1.8.0.
1.8	13.04.21	Umstellung von CORE.connect auf neue Versionen der verwendeten Bibliotheken (Aktualisierung der verwendeten Dritt-Bibliotheken)
1.9	23.12.21	Abhängigkeit zu Log4J, sowie log4j.jar-Datei entfernt Protokollierung erfolgt über Apache-Commons-Logging einbindende Software muss die konkrete Implementierung der Protokollierung vorgeben und konfigurieren, ansonsten erfolgt die Protokollierung über Standardausgabe
1.10	02.01.23	Xerces von 2.12.1 auf 2.12.2 aktualisiert Xalan entfernt (war nur eine Runtime-Dependency und kann mittlerweile durch JRE-Standardfunktionalität ersetzt werden) Korrektur im CSV-Inspector (bei wiederholten Merkmalsgruppen innerhalb von wiederholten Merkmalsgruppen wurde immer nur die erste untergeordnete Merkmalsgruppe geprüft)

1 Changelog

1.1 CORE.connect

1.1.1 CORE.connect 1.11.0

Version 1.11.0 – 02.01.2023

- **Anpassung**
  - Xerces von 2.12.1 auf 2.12.2 aktualisiert
  - Xalan entfernt (war nur eine Runtime-Dependency und kann mittlerweile durch JRE-Standardfunktionalität ersetzt werden)

- **Fehlerbehebung**
  - Korrektur im CSV-Inspector (bei wiederholten Merkmalsgruppen innerhalb von wiederholten Merkmalsgruppen wurde immer nur die erste untergeordnete Merkmalsgruppe geprüft)

1.1.2 CORE.connect 1.10.0

Version 1.10.0 – 23.12.2021

- **Anpassung**
  - Zu CORE.connect wurde die Abhängigkeit zu Log4J grundsätzlich entfernt.
  - log4j.jar-Datei wurde aus Lieferung entfernt
  - Sollte die einbindende Software weiterhin die Version 1.2.X von Log4J verwendet, bestehen keine Funktionseinschränkungen und es muss beim Austausch der Bibliothek nichts weiter beachtet werden
  - Protokollierung innerhalb der Bibliothek erfolgt nun über Apache-Commons-Logging. Somit kann die einbindende Software die konkrete Implementierung der Protokollierung vorgeben und konfigurieren. Wird keine Protokollierung vorgegeben, erfolgt die Protokollierung über die Standardausgabe
  - Hinweis: Wenn für die Protokollierung nicht Log4J in der Version 1.2.X verwendet wird, stehen die Funktionen zur programmatischen Vorgabe einer Logdatei und des Detaillierungsgrades über die CORE-Schnittstelle nicht mehr zur Verfügung

1.1.3 CORE.connect 1.9.0

Version 1.9.0 – 13.04.2021

- **Anpassung**
  - Umstellung von CORE.connect auf neue Versionen der verwendeten Bibliotheken (Aktualisierung der verwendeten Dritt-Bibliotheken)

1.1.4 CORE.connect 1.8.0

Version 1.8.0 – 12.06.2018

- **Anpassung**
  - Ignorieren von Zeilen, die nur CSV-Trenner enthalten.
  - Semester-Kodierung geändert, Prüfung auf Vorausmeldung für Semester angepasst.

- **Fehlerbehebung**
  - Fehler beim Einlesen der Hilfsmerkmale korrigiert: Die Satznummer muss -1 statt 0 sein.
  - Fehlerkorrektur bei Merkmalen vom Typ „Normalized String" und „Token" mit Wertebereichen.

1.1.5 CORE.connect 1.7.5

Version 1.7.5 – 28.10.2015

- **Fehlerbehebung**
  - Leere Merkmalsgruppen werden jetzt im DatML/RAW-Generator (konfigurierbar) ignoriert.

1.1.6 CORE.connect 1.7.3

Version 1.7.3 – 03.08.2015

- **Anpassung**
  - Version der Java-Bibliothek nun zusätzlich in Manifest-Datei hinterlegt.

- **Fehlerbehebung**
  - Fehler beim Interpretieren der minOccurs-Angabe bei mmgrs mit Lazy-Bedingungen behoben.
  - Fehlerbehebung im PL-Generator (Leerstring mit Kommazahlmaske)

1.1.7 CORE.connect 1.7.0

Version 1.7.0 – 16.06.2015

- **Anpassung**
  - DataProvider um #close() erweitert und im DocumentBuilderInterpreter eingebunden.
    - CSVDataProvider und MapDataProvider betroffen!
  - DataProvider#requiresMapping() entfernt.
    - CSVDataProvider und MapDataProvider betroffen!
  - SurveyDataDocumentGenerator um MappingProvider (Mapping pro Nachricht) und SDFMetaResourceProvider erweitert.
  - der internen Prüfungen beim Einlesen von CSV-Dateien anhand der dazugehörigen Erhebungsbeschreibung mittels eines CSV-Prüfmoduls in der CORE.inspector-Komponente.

- **Fehlerbehebung**
  - beim Einlesen von Feldgruppen aus CSV korrigiert. Auch bei leeren Merkmalen werden jetzt die Positionen angezeigt.

- beim Einlesen von dimensionierten Merkmalsgruppen. Am Ende einer CSV-Datei kann nun die Anzahl der Merkmalsgruppen <= der vorgegebenen Dimension der Merkmalsgruppe sein.
- beim Einlesen von Feldgruppen aus CSV korrigiert. Auch bei leeren Merkmalen werden jetzt die Positionen angezeigt.

- **JAVA-Laufzeitumgebung**
  - Verwendung der JAVA-Laufzeitumgebung Version 7 und Version 8

1.1.8 CORE.connect 1.5.1

Version 1.5.1 – 08.05.2014

- **Änderung der Zertifikatsprüfung**
  - Zertifikatsprüfung anhand der in JAVA hinterlegten Wurzelzertifikate (auch bekannt als Root-Zertifikat oder Stammzertifikat) - die Vorgabe des Zertifikat-Verzeichnisses ist alternativ weiterhin möglich.
- **Änderung der TLS-Verschlüsselung**
  - Umstellung auf JAVA 7 und die mögliche Verwendung von TLS 1.2.
- **Erweiterung**
  - Optionale Erweiterung um die Möglichkeit der expliziten Setzung der Prüfstufe für Prüfung einer Datenlieferung an den gemeinsamen Dateneingang von .CORE
    ➔ siehe Klasse **de.destatis.core.connect.CheckDeliveryRequest**
- **Anpassung**
  - Anpassung der Längenprüfung unter Verwendung von Trenn- und / oder Vorzeichen.

1.1.9 CORE.connect 1.3.1

Version 1.3.1 – 29.11.2012

- **Fehlerbehebung**
  - Bei der Generierung von DatML/RAW-Dokumenten und aktivierter ‚Strenger SDF-Validierung' wird die Überprüfung der Merkmalswerte nun auch gegen eine hinterlegte Ausprägungsgruppe durchgeführt.

1.1.10 CORE.connect 1.3.0

Version 1.3.0 – 26.03.2012

- **Auftrennung der Bibliothek CORE.connect in die Bibliotheken CORE.connect und CORE.inspector**
- **Änderungen für große Datenlieferungen (Speicherplatzproblemen)**

Da es in der Vergangenheit bei größeren Datenlieferungen zu Speicherplatzproblemen gekommen ist, wurde eine neue Methode setDocument(DocumentSource) in der Klasse DatMLRawDocument aufgenommen, welche dafür sorgt, dass das DatML/RAW-Dokument in einer Datei vorgehalten wird und nicht komplett in den Speicher geladen wird.

Variante 1: setDocument(javax.xml.transform.Source source)

Nur bei ***kleineren Datenlieferungen*** sinnvoll, da der Inhalt der Datenquelle durch den XML- Parser eingelesen und ***vollständig im Speicher*** abgelegt wird.
Bei Speicherplatzproblemen (OutOfMemory) sollte stattdessen die Variante 2 verwendet werden, welche es ermöglicht, das XML-Dokument in einer Datei vorzuhalten.

```
DatMLRawDocument doc = new DatMLRawDocument();
doc.setDocument(new StreamSource(new File("TestDatenlieferung.xml")));
DataDelivery delivery = new DataDelivery(doc);
ResponseToDataDelivery response = client.send(delivery);
```

Variante 2: setDocument(DocumentSource source)

Das DatML/RAW-Dokument wird in einer Datei vorgehalten und nicht komplett in den Speicher geladen.

```
DatMLRawDocument doc = new DatMLRawDocument();
doc.setDocument(DocumentSourceFactory.createDocumentSource(new File("TestDatenlieferung.xml")));
DataDelivery delivery = new DataDelivery(doc);
ResponseToDataDelivery response = client.send(delivery);
```

siehe JAVA-Beispiel (Kap. 3.3) ‚ExampleCheckDatenlieferung.java' und ‚ExampleSendDatenlieferung.java'

- **Automatisierte Versorgung der Bibliothek CORE.connect mit den benötigten aktuellen Ressourcen (z.B. DatML/SDF) zu einer Erhebung.**
- **Testen der Verbindung zum gemeinsamen Dateneingang von .CORE möglich**
- **Generierung von Meldungen im Lieferdatenformat der amtlichen Statistik DatML/RAW**

  (Das Lieferdatenformat DatML/RAW ist Teil des XÖV-zertifizierten Nachrichtenformats XStatistik.)

  - über Abfrage-Schnittstelle (Spezifische Implementierung innerhalb des Statistikmoduls).
  - über bereits vorliegende Standardimplementierung, die die Vorgabe der Daten im CSV- bzw. im Properties-Format (Feldname-Wert-Zuordnung) ermöglichen.
  - durch Bereitstellung einer graphischen Benutzeroberfläche zur Definition der Abbildung der Erhebungsmerkmale auf Position in der CSV-Datei oder eigene Feldbezeichnung.

- **Erweiterte Antwortinformationen zu einer Datenlieferung**

Das Senden einer DatML/RAW-Datenlieferung an den gemeinsamen Dateneingang von eSTATISTIK.core liefert neben dem Eingangsstempel auch das entsprechende Prüfprotokoll zurück (siehe Klasse ResponseToDataDelivery).
Das entsprechende Prüfprotokoll kann auch weiterhin mittels des Eingangsstempels abgerufen werden.

- **Bessere Lesbarkeit der Prüfprotokolle**

Zur besseren Lesbarkeit der im XML-Format (DatML/RES) vorliegenden Prüfprotokolle, wurden entsprechende Stylesheets zur Aufbereitung des Prüfprotokolles im Prüfprotokoll eingefügt.

  - für DatML/RAW 2.0 ‚<?xml-stylesheet type="text/xsl" href="./res1html.xsl"?>'
  - für DatML/RAW 2.1 ‚<?xml-stylesheet type="text/xsl" href="./res2html.xsl"?>'

  Zur Browser-Anzeige sind die Stylesheet-Dateien res1html.xsl und res2html.xsl im gleichen Verzeichnis abzuspeichern, in welchem auch die Protokolle abgespeichert werden. Die Stylesheets stehen

- über die öffentliche Erhebungsdatenbank unter https://erhebungsdatenbank.estatistik.de/eid/show.html?genresid=04

oder

- über die Webanwendung zu eSTATISTIK.core (https://core.estatistik.de/core/) unter Lieferhistorie anzeigen

zur Verfügung.

- **Prüfung von DatML/RAW-Datenlieferungen mittels des gemeinsamen Dateneingangs**

CORE.connect stellt die Möglichkeit zur Verfügung, DatML/RAW-Datenlieferungen über den gemeinsamen Dateneingang von eSTATISTIK.core prüfen zu lassen.

siehe JAVA-Beispiel (Kap. 3.3) ‚ExampleCheckDatenlieferung.java'

- **Abrufen von zusätzlichen Informationen mittels des gemeinsamen Dateneingangs**

CORE.connect stellt die Möglichkeit zur Verfügung, zusätzliche Informationen über den gemeinsamen Dateneingang von eSTATISTIK.core zu erhalten.

- **Liste aller freigegebenen Erhebungen**

  Liefert eine Liste aller freigegebenen Erhebungen vom Server des gemeinsamen Dateneingangs von eSTATISTIK.core herunter.

  siehe JAVA-Beispiel (Kap. 3.3) ‚ExampleGetSurveyIDList.java'

- **Herunterladen von Ressourcen**

  Lädt eine Ressource (SDF - Erhebungsbeschreibungen, ASK - Elektr. Fragebogen, EDT - PL-Metadaten und Liefervereinbarungen) vom Server des gemeinsamen Dateneingangs von eSTATISTIK.core herunter.

  siehe JAVA-Beispiel (Kap. 3.3) ‚ExampleGetRes.java'

- **Herunterladen von allgemeinen Ressourcen**

  Lädt eine allgemeine Ressource (SDFMeta und DatML/RAW-Schema) in Abhängigkeit der DatML/RAW-Version und einer evtl. bereits vorliegenden ("aktuellen") Ressource vom Server des gemeinsamen Dateneingangs von eSTATISTIK.core herunter.

  siehe JAVA-Beispiel (Kap. 3.3) ‚ExampleGetComRes.java'

- **Protokollierung in einem permanent laufendem Service mit Protokollierung**

Wird die CORE-connect-Bibliothek in einem permanent laufendem Service integriert, welcher bereits eine Protokollierung mit log4j beinhaltet, darf das SystemProperty client.log.file nicht gesetzt werden. In diesen Fall muss die reguläre log4j-Konfiguration des aufrufenden Service verwendet werden.

Ursache: Unter Verwendung der SystemProperty client.log.file für die CORE-connect-Bibliothek wird nach Beenden der CORE-connect-Aufrufe (CORE-connect-Bibliothek wird nicht mehr verwendet) die Protokollierung innerhalb der JAVA VM abgeschaltet.

1.2 CORE.inspector

1.2.1 CORE.inspector 1.11.0

Version 1.11.0 – 02.01.2023

- **Anpassung**
  - Umstellung von CORE.connect auf neue Versionen der verwendeten Bibliotheken (Aktualisierung der verwendeten Dritt-Bibliotheken)

1.2.2 CORE.inspector 1.10.0

Version 1.10.0 – 23.12.2021

- **Anpassung**
  - Zu CORE.connect wurde die Abhängigkeit zu Log4J grundsätzlich entfernt.
  - log4j.jar-Datei wurde aus Lieferung entfernt
  - Sollte die einbindende Software weiterhin die Version 1.2.X von Log4J verwendet, bestehen keine Funktionseinschränkungen und es muss beim Austausch der Bibliothek nichts weiter beachtet werden
  - Protokollierung innerhalb der Bibliothek erfolgt nun über Apache-Commons-Logging. Somit kann die einbindende Software die konkrete Implementierung der Protokollierung vorgeben und konfigurieren. Wird keine Protokollierung vorgegeben, erfolgt die Protokollierung über die Standardausgabe.
  - Hinweis: Wenn für die Protokollierung nicht Log4J in der Version 1.2.X verwendet wird, stehen die Funktionen zur programmatischen Vorgabe einer Logdatei und des Detaillierungsgrades über die CORE-Schnittstelle nicht mehr zur Verfügung

1.2.3 CORE.inspector 1.9.0

Version 1.9.0 – 13.04.2021

- **Anpassung**
  - Umstellung von CORE.connect auf neue Versionen der verwendeten Bibliotheken (Aktualisierung der verwendeten Dritt-Bibliotheken)

1.2.4 CORE.inspector 1.8.0

Version 1.8.0 – 12.06.2018

- **Anpassung**
  - Ignorieren von Zeilen, die nur CSV-Trenner enthalten.
  - Semester-Kodierung geändert, Prüfung auf Vorausmeldung für Semester angepasst.

1.2.5 CORE.inspector 1.7.5

Version 1.7.5 – 28.10.2015

- **Fehlerbehebung**
  - Leere Merkmalsgruppen werden im CSV-Inspector ignoriert.

1.2.6 CORE.inspector 1.7.3

Version 1.7.3 – 03.08.2015

- **Anpassung**
  - Version der JAVA-Bibliothek nun zusätzlich in Manifest-Datei hinterlegt.

1.2.7 CORE.inspector 1.7.0

Version 1.7.0 – 16.06.2015

- **Anpassung**
  - Anpassung des Fehlertextes zur Fehlermeldung bei fehlendem MetaMerkmal "Absender.Kennung". Statt '*kein Datum*' wird nun der Fehlertext '*kein Wert*' zur Fehlermeldung ausgegeben.
  - Aufnahme eines CSV-Prüfmoduls zur internen Prüfung von CSV-Dateien anhand der dazugehörigen Erhebungsbeschreibung.
- **Fehlerbehebung**
  - bei Prüfung der Eindeutigkeit der Indizes in verschachtelten Merkmalsgruppen.
- **JAVA-Laufzeitumgebung**
  - Verwendung der JAVA-Laufzeitumgebung Version 7 und Version 8

1.2.8 CORE.inspector 1.5.1

Version 1.5.1 – 09.04.2014

- **Umstellung von JAVA 7**
  - Zu CORE.connect äquivalente Umstellung auf JAVA 7.
- **Anpassung**
  - Anpassung der Längenprüfung unter Verwendung von Trenn- und / oder Vorzeichen.

1.2.9 CORE.inspector 1.3.2

Version 1.3.2 – 2.11.2012

- **Fehlerbehebung**
  - bei Überprüfung der Merkmalswerte gegen eine hinterlegte Ausprägungsgruppe

1.2.10 CORE.inspector 1.3.1

Version 1.3.1 – 16.05.2012

- **Fehlerbehebung**
  - bei Verwendung von Bedingungen in übergeordneten Merkmalsgruppen
  - bei XPath-Angaben in Verbindung von Merkmalen innerhalb von Merkmalsgruppen

1.2.11 CORE.inspector 1.3.0

Version 1.3.0 – 29.03.2012

- **Thread-Safety**

Der CORE.inspector ist nun thread-safe.

Die Instanzierungen und Verwendung mehrer paralleler Inspektoren in jeweils eigenen Threads wurde realisiert.

- **Aufnahme neuer Fehlerschlüssel**

Nachfolgende Fehlerschlüssel und Fehlertexte wurden neu aufgenommen:

1. 32002 (WARN) Berichtsempfänger maschinell korrigiert.
2. 33016 (ERROR) Die Identifizierung der BerichtseinheitsID ist nicht möglich (falsche oder fehlende BerichtseinheitsID). Bitte übermitteln Sie die Meldung erneut unter Angabe der ID, die Ihnen das zuständige Statistische Amt mitgeteilt hat.
3. 42006 (WARN) Der Wert des Hilfsmerkmals liegt nicht im definierten Wertebereich.
4. 42007 (WARN) Der Wert des Merkmals liegt nicht im definierten Wertebereich.
5. 42008 (WARN) Die zur Erzeugung des DatML/RAW-Dokumentes verwendeten DatML/SDF-Ressource (%VERSION%) stimmt nicht mit der zur Prüfung des DatML/RAW-Dokumentes verwendeten DatML/SDF-Ressource (%VERSION%) überein.

1.3 JAVA-Projekt mit Beispielen

1.3.1 JAVA-Projekt mit Beispielen 1.4

Version 1.4– 10.08.2015

- **Erweiterung der Beispiele**
  - Anpassung der Beispiele an JAVA 8 (siehe z.B. Klasse ExampleGetComRes -> Ermittlung der allgemeine DatML/RAW-Schema-Ressource für DatML/RAW 2.1 als zip-Archiv).
  - Ermittlung von Ressourcen nun für den BZR 2015 (statt BZR 2011), da ältere Ressourcen nicht mehr zur Verfügung stehen.
  - Generierung mittels Abfrage--Schnittstelle:
    - Daten zur StatistikID 0376 und zum aktuellen BZR ‚2015' angepasst (Merkmal ‚DatumBeendigungRSB' hinzugefügt).

1.3.2 JAVA-Projekt mit Beispielen 1.3

Version 1.3– 16.06.2015

- **Erweiterung der Beispiele**
  - Im Beispiel die Klasse ImplDataProvider um die neue Methode **void close()** erweitert und die Methode **boolean requiresMapping()** entfernt.

1.3.3 JAVA-Projekt mit Beispielen 1.2

Version 1.2– 08.05.2014

- **Erweiterung der Beispiele**
  - Im Beispiel ExampleCheckDatenlieferung die optionale Erweiterung um die Möglichkeit der expliziten Setzung der Prüfstufe für Prüfung einer Datenlieferung an den gemeinsamen Dateneingang von .CORE hinzugefügt.

1.3.4 JAVA-Projekt mit Beispielen 1.1

Version 1.1 – 29.11.2012

- **Erweiterung der Beispiele**
  - Die Beispiele zur Abfrage-Schnittstelle (Spezifische Implementierung innerhalb des Statistikmoduls), sowie bei der Standardimplementierung mittels CSV- bzw. Properties-Datei (Feldname-Wert-Zuordnung) wurden erweitert.
    - Beispiel 1 zur „Statistik über beendete Insolvenzverfahren und Restschuldbefreiung" im Package ‚de.destatis.core.generation.examples_0376'.
    - Beispiel 2 zur „Vierteljährlichen Verdiensterhebung" im Package ‚de.destatis.core.generation.examples_0001' beinhaltet Mehrdimensionale-Felder.

- Aufnahme der AutoUpdate-Funktion von SDF- und SDFMeta-Ressourcen in den Beispielen zur Abfrage-Schnittstelle (Spezifische Implementierung innerhalb des Statistikmoduls), sowie bei der Standardimplementierung mittels CSV- bzw. Properties-Datei (Feldname-Wert-Zuordnung).
- Aufnahme der Aktivierung der ‚Strengen SDF-Validierung' (es erfolgt eine Prüfung von Werten und eine Auswertung von SDF-Bedingungen) in den Beispielen zur Abfrage-Schnittstelle (Spezifische Implementierung innerhalb des Statistikmoduls), sowie bei der Standardimplementierung mittels CSV- bzw. Properties-Datei (Feldname-Wert-Zuordnung).

2 Migration

2.1 CORE.connect

2.1.1 CORE.connect - Version 1.10.0 zu Version 1.11.0

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.1.2 CORE.connect - Version 1.9.0 zu Version 1.10.0

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.1.3 CORE.connect - Version 1.8.0 zu Version 1.9.0

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.1.4 CORE.connect - Version 1.7.5 zu Version 1.8.0

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.1.5 CORE.connect - Version 1.7.3 zu Version 1.7.5

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.1.6 CORE.connect - Version 1.7.0 zu Version 1.7.3

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.1.7 CORE.connect - Version 1.5.1 zu Version 1.7.0

Austausch der JAR-Dateien aus dem lib-Verzeichnis.
Verwendung der Java-Laufzeitumgebung Version 7 und Version 8.

Bei Verwendung des Interfaces **DataProvider** die neue Methode **void close()** aufnehmen und die Methode **boolean requiresMapping()** entfernen.

2.1.8 CORE.connect - Version 1.3.1 zu Version 1.5.1

Austausch der JAR-Dateien aus dem lib-Verzeichnis.
Verwendung der Java-Laufzeitumgebung Version 7.

2.1.9 CORE.connect - Version 1.3.0 zu Version 1.3.1

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.1.10 CORE.connect - Version 1.2 zu Version 1.3.0

Wurde für CORE.connect 1.2 nachfolgende Implementierung zur Prüfung der DatML/RAW-Meldung verwendet (Überprüfung des in der Instanz von DatMLRAWDocument enthaltenen Objektes gemäß der aktuellen Konfigurationseinstellungen des Inspectors):

```
// Daten aus Datei lesen
DatMLRawDocument doc = new DatMLRawDocument();
doc.setDocument(new StreamSource(new File(dateiname)));
// Daten prüfen
doc.setInspectionSurveyDir("res");
doc.setInspectionLevel(4);
InspectionReport report = doc.inspect();
```

so muss Aufgrund der Trennung von CORE.connect Version 1.2 in CORE.connect und CORE.inspector Version 1.3.0 Änderungen an der Implementierung vorgenommen werden.

Eine mögliche alternative Implementierung wäre:

```
// Daten prüfen
Inspector inspector = new Inspector();
inspector.setSurveyDir("res");
inspector.setInspectionLevel(4);
InputSource inputSource = new InputSource(new FileInputStream(new File(dateiname)));
```

2.2 CORE.inspector

2.2.1 CORE.inspector - Version 1.10.0 zu Version 1.11.0

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.2.2 CORE.inspector - Version 1.9.0 zu Version 1.10.0

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.2.3 CORE.inspector - Version 1.8.0 zu Version 1.9.0

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.2.4 CORE.inspector - Version 1.7.5 zu Version 1.8.0

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.2.5 CORE.inspector - Version 1.7.3 zu Version 1.7.5

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.2.6 CORE.inspector - Version 1.7.0 zu Version 1.7.3

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.2.7 CORE.inspector - Version 1.5.1 zu Version 1.7.0

Austausch der JAR-Dateien aus dem lib-Verzeichnis.
Verwendung der Java-Laufzeitumgebung Version 7 und Version 8.

2.2.8 CORE.inspector - Version 1.3.2 zu Version 1.5.1

Austausch der JAR-Dateien aus dem lib-Verzeichnis.
Verwendung der Java-Laufzeitumgebung Version 7.

2.2.9 CORE.inspector - Version 1.3.1 zu Version 1.3.2

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

2.2.10 CORE.inspector - Version 1.3.0 zu Version 1.3.1

Austausch der JAR-Dateien aus dem lib-Verzeichnis.

3 Lieferumfang

3.1 Mit Freigabe der Version 1.11.0 von CORE.connect und CORE.inspector

3.1.1 Dokumente

- Softwarebibliotheken_zu_CORE_fuer_JAVA.pdf

3.1.2 Bibliotheken

- CORE_connect_java_1_11_0.zip
- CORE_inspector_java_1_11_0.zip

3.1.3 JAVA-Projekt mit Beispielen

- CORE_Softwarebibliotheken_Beispiele_Implementierung.zip

mit folgenden Beispielen:

CORE.connect:

- ExampleClientWithINI.java
- ExamplePingRequest.java
- ExampleGetSurveyIDList.java
- ExampleGetRes.java
- ExampleGetComRes.java
- ExampleCheckDatenlieferung.java
- ExampleSendDatenlieferung.java
- ExampleProtokollAbrufen.java

DatML/RAW-Generierung mittels CORE.connect:

Beispiel 1 im Package **de.destatis.core.generation.examples_0376**

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

Beispiel 2 mit mehrdimensionalen Feldern im Package **de.destatis.core.generation.examples_0001**

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

CORE.inspector:

- ExampleInspect.java

3.2 Mit Freigabe der Version 1.10.0 von CORE.connect und CORE.inspector

3.2.1 Dokumente

- Softwarebibliotheken_zu_CORE_fuer_JAVA.pdf

3.2.2 Bibliotheken

- CORE_connect_java_1_10_0.zip
- CORE_inspector_java_1_10_0.zip

3.2.3 JAVA-Projekt mit Beispielen

- CORE_Softwarebibliotheken_Beispiele_Implementierung.zip

mit folgenden Beispielen:

CORE.connect:

- ExampleClientWithINI.java
- ExamplePingRequest.java
- ExampleGetSurveyIDList.java
- ExampleGetRes.java
- ExampleGetComRes.java
- ExampleCheckDatenlieferung.java
- ExampleSendDatenlieferung.java
- ExampleProtokollAbrufen.java

DatML/RAW-Generierung mittels CORE.connect:

Beispiel 1 im Package **de.destatis.core.generation.examples_0376**

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

Beispiel 2 mit mehrdimensionalen Feldern im Package **de.destatis.core.generation.examples_0001**

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

CORE.inspector:

- ExampleInspect.java

3.3 Mit Freigabe der Version 1.9.0 von CORE.connect und CORE.inspector

3.3.1 Dokumente

- Softwarebibliotheken_zu_CORE_fuer_JAVA.pdf

3.3.2 Bibliotheken

- CORE_connect_java_1_9_0.zip
- CORE_inspector_java_1_9_0.zip

3.3.3 JAVA-Projekt mit Beispielen

- CORE_Softwarebibliotheken_Beispiele_Implementierung.zip

mit folgenden Beispielen:

CORE.connect:

- ExampleClientWithINI.java
- ExamplePingRequest.java
- ExampleGetSurveyIDList.java
- ExampleGetRes.java
- ExampleGetComRes.java
- ExampleCheckDatenlieferung.java
- ExampleSendDatenlieferung.java
- ExampleProtokollAbrufen.java

DatML/RAW-Generierung mittels CORE.connect:

Beispiel 1 im Package **de.destatis.core.generation.examples_0376**

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

Beispiel 2 mit mehrdimensionalen Feldern im Package **de.destatis.core.generation.examples_0001**

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

CORE.inspector:

- ExampleInspect.java

3.4 Mit Freigabe der Version 1.8.0 von CORE.connect und CORE.inspector

3.4.1 Dokumente

- Softwarebibliotheken_zu_CORE_fuer_JAVA.pdf

3.4.2 Bibliotheken

- CORE_connect_java_1_8_0.zip
- CORE_inspector_java_1_8_0.zip

3.4.3 JAVA-Projekt mit Beispielen

- CORE_Softwarebibliotheken_Beispiele_Implementierung.zip

mit folgenden Beispielen:

CORE.connect:

- ExampleClientWithINI.java
- ExamplePingRequest.java
- ExampleGetSurveyIDList.java
- ExampleGetRes.java
- ExampleGetComRes.java
- ExampleCheckDatenlieferung.java
- ExampleSendDatenlieferung.java
- ExampleProtokollAbrufen.java

DatML/RAW-Generierung mittels CORE.connect:

Beispiel 1 im Package **de.destatis.core.generation.examples_0376**

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

Beispiel 2 mit mehrdimensionalen Feldern im Package **de.destatis.core.generation.examples_0001**

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

CORE.inspector:

- ExampleInspect.java

3.5 Mit Freigabe der Version 1.7.3 von CORE.connect und CORE.inspector

3.5.1 Dokumente

- Softwarebibliotheken_zu_CORE_fuer_JAVA.pdf

3.5.2 Bibliotheken

- CORE_connect_java_1_7_3.zip
- CORE_inspector_java_1_7_3.zip

3.5.3 JAVA-Projekt mit Beispielen

- CORE_Softwarebibliotheken_Beispiele_Implementierung.zip

mit folgenden Beispielen:

CORE.connect:

- ExampleClientWithINI.java
- ExamplePingRequest.java
- ExampleGetSurveyIDList.java
- ExampleGetRes.java
- ExampleGetComRes.java
- ExampleCheckDatenlieferung.java
- ExampleSendDatenlieferung.java
- ExampleProtokollAbrufen.java

DatML/RAW-Generierung mittels CORE.connect:

Beispiel 1 im Package **de.destatis.core.generation.examples_0376**

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

Beispiel 2 mit mehrdimensionalen Feldern im Package **de.destatis.core.generation.examples_0001**

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

CORE.inspector:

- ExampleInspect.java

3.6 Mit Freigabe der Version 1.7.0 von CORE.connect und CORE.inspector

3.6.1 Dokumente

- Softwarebibliotheken_zu_CORE_fuer_JAVA.pdf

3.6.2 Bibliotheken

- CORE_connect_java_1_7_0.zip
- CORE_inspector_java_1_7_0.zip

3.6.3 JAVA-Projekt mit Beispielen

- CORE_Softwarebibliotheken_Beispiele_Implementierung.zip

mit folgenden Beispielen:

CORE.connect:

- ExampleClientWithINI.java
- ExamplePingRequest.java
- ExampleGetSurveyIDList.java
- ExampleGetRes.java
- ExampleGetComRes.java
- ExampleCheckDatenlieferung.java
- ExampleSendDatenlieferung.java
- ExampleProtokollAbrufen.java

DatML/RAW-Generierung mittels CORE.connect:

Beispiel 1 im Package ‚de.destatis.core.generation.examples_0376'

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

Beispiel 2 mit mehrdimensionalen Feldern im Package ‚de.destatis.core.generation.examples_0001'

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

CORE.inspector:

- ExampleInspect.java

3.7 Mit Freigabe der Version 1.5.1 von CORE.connect und CORE.inspector

3.7.1 Dokumente

- Softwarebibliotheken_zu_CORE_fuer_JAVA.pdf

3.7.2 Bibliotheken

- CORE_connect_java_1_5_1.zip
- CORE_inspector_java_1_5_1.zip

3.7.3 JAVA-Projekt mit Beispielen

- CORE_Softwarebibliotheken_Beispiele_Implementierung.zip

mit folgenden Beispielen:

CORE.connect:

- ExampleClientWithINI.java
- ExamplePingRequest.java
- ExampleGetSurveyIDList.java
- ExampleGetRes.java
- ExampleGetComRes.java
- ExampleCheckDatenlieferung.java
- ExampleSendDatenlieferung.java
- ExampleProtokollAbrufen.java

DatML/RAW-Generierung mittels CORE.connect:

Beispiel 1 im Package ‚de.destatis.core.generation.examples_0376'

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

Beispiel 2 mit mehrdimensionalen Feldern im Package ‚de.destatis.core.generation.examples_0001'

- ExampleGenerationFromInterface.java
- ExampleGenerationFromProperties.java
- ExampleGenerationFromCSV.java

CORE.inspector:

- ExampleInspect.java

3.8 Mit Freigabe der Version 1.3.1 von CORE.connect und 1.3.2 von CORE.inspector

3.8.1 Dokumente

- Softwarebibliotheken zu .CORE für JAVA.pdf

3.8.2 Bibliotheken

- CORE_connect_java_1_3_1.zip
- CORE_inspector_java_1_3_2.zip

3.8.3 JAVA-Projekt mit Beispielen

- CORE_Softwarebibliotheken_Beispiele_Implementierung.zip

  mit folgenden Beispielen:

  CORE.connect:
  - ExampleClientWithINI.java
  - ExamplePingRequest.java
  - ExampleGetSurveyIDList.java
  - ExampleGetRes.java
  - ExampleGetComRes.java
  - ExampleCheckDatenlieferung.java
  - ExampleSendDatenlieferung.java
  - ExampleProtokollAbrufen.java

  DatML/RAW-Generierung mittels CORE.connect:

  Beispiel 1 im Package ‚de.destatis.core.generation.examples_0376'
  - ExampleGenerationFromInterface.java
  - ExampleGenerationFromProperties.java
  - ExampleGenerationFromCSV.java

  Beispiel 2 mit mehrdimensionalen Feldern im Package ‚de.destatis.core.generation.examples_0001'
  - ExampleGenerationFromInterface.java
  - ExampleGenerationFromProperties.java
  - ExampleGenerationFromCSV.java

  CORE.inspector:
  - ExampleInspect.java

3.9 Mit Freigabe der Version 1.3.1 von CORE.inspector

3.9.1 Bibliotheken

- CORE_inspector_java_1_3_1.zip

3.10 Mit Freigabe der Version 1.3.0 von CORE.connect und CORE.inspector

3.10.1 Dokumente

- Softwarebibliotheken zu .CORE für JAVA.pdf

3.10.2 Bibliotheken

- CORE_connect_java_1_3_0.zip
- CORE_inspector_java_1_3_0.zip

3.10.3 JAVA-Projekt mit Beispielen

- CORE_Softwarebibliotheken_Beispiele_Implementierung.zip

  mit folgenden Beispielen:

  CORE.connect:
  - ExampleClientWithINI.java
  - ExamplePingRequest.java
  - ExampleGetSurveyIDList.java
  - ExampleGetRes.java
  - ExampleGetComRes.java
  - ExampleCheckDatenlieferung.java
  - ExampleSendDatenlieferung.java
  - ExampleProtokollAbrufen.java
  - ExampleGenerationFromInterface.java
  - ExampleGenerationFromProperties.java
  - ExampleGenerationFromCSV.java

  CORE.inspector:
  - ExampleInspect.java