

Konnektoren

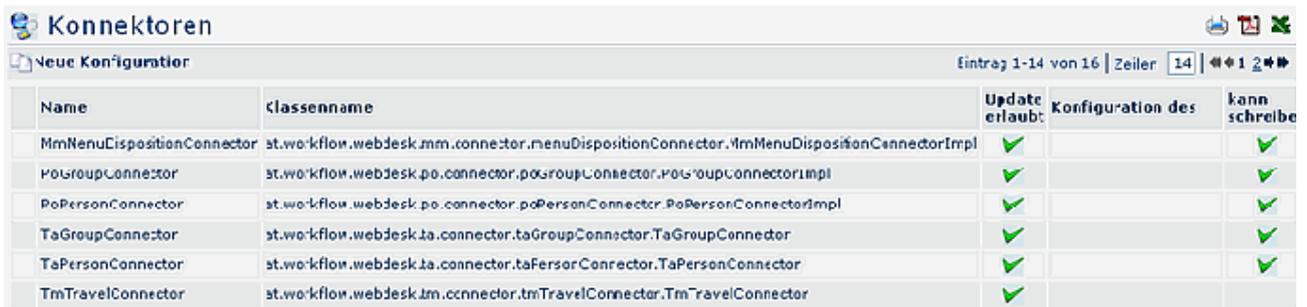
Allgemeines zu Konnektoren

Die Konnektoren stellen ein Rahmengerüst für den **Import** und **Export** von **Daten** in den Webdesk (bzw. aus dem Webdesk) zur Verfügung.

Die Definition von Quellkonnektoren (Quelle, aus der die Daten kommen) und Zielkonnektoren (wo die Daten hinkommen) erlaubt die Gestaltung des Datenflusses. So können nicht nur Daten aus dem Zeitwirtschaftssystem in den Webdesk synchronisiert werden, sondern auch Datenbanken angesteuert, Textfiles (CSV-Dateien) erstellt werden. So können auch Daten aus dem Webdesk exportiert, und in andere Systeme importiert und auf dem Weg dorthin mittels Scripting angepasst werden (Schnittstellen-Framework).

Prinzipiell gibt es folgende Arten von Konnektoren:

- Lesende Konnektoren
- Schreibende Konnektoren
- Konnektoren die sowohl lesen als auch schreiben können



The screenshot shows a window titled 'Konnektoren' with a sub-window 'Neue Konfiguration'. It displays a table with the following columns: Name, Klassenname, Update erlaubt, Konfiguration des, and kann schreiben. The table lists several connectors with their respective class names and status indicators.

Name	Klassenname	Update erlaubt	Konfiguration des	kann schreiben
MmMenuDispositionConnector	st.workflow.webdesk.mm.connector.menuDispositionConnector.MmMenuDispositionConnectorImpl	✓		✓
PosGroupConnector	st.workflow.webdesk.po.connector.posGroupConnector.PosGroupConnectorImpl	✓		✓
PoPersonConnector	st.workflow.webdesk.po.connector.poPersonConnector.PoPersonConnectorImpl	✓		✓
TaGroupConnector	st.workflow.webdesk.ta.connector.taGroupConnector.TaGroupConnector	✓		✓
TaPersonConnector	st.workflow.webdesk.ta.connector.taPersonConnector.TaPersonConnector	✓		✓
TmTravelConnector	st.workflow.webdesk.tm.connector.tmTravelConnector.TmTravelConnector	✓		

- **Name**
Name des Konnektors
- **Klassenname**
gibt den technischen Klassennamen des Konnektors an
- **Update erlaubt**
Ziel (oder rechte Seite) des Konnektors
- **Konfigurations des**
Handelt es sich bei dem Konnektor um eine Konfiguration, so wird hier der urspr. Konnektor angezeigt, von dem die Konfiguration abstammt
- **Kann schreiben**
definiert ob der Konnektor auch schreibend verwendet werden kann. Konnektoren die diese Option nicht verwenden, können bei den verknüpften Konnektoren nicht als Zielconnectoren verwendet werden.

Weiters können im Webdesk auch **verknüpfte Konnektoren**¹ definiert werden. Sie stellen eine Verknüpfung zwischen Konnektoren dar, und ermöglichen so die Definition des Datenflusses (Definition als Ziel- oder Quellkonnektoren).

Anlegen eines neuen Konnektors / Neue Konfiguration

Um einen neuen Konnektor anzulegen klicken sie in der Liste der Konnektoren auf die Schaltfläche **Neue Konfiguration**.

Es müssen nun folgende Felder befüllt werden. Achtung erst nach Ausfüllen des Vaterkonnektors und anschliessendem speichern können die für diesen Vaterkonnektor spezifischen Informationen eingegeben werden!

- **Vaterkonnektor**
Selektion des Konnektors, von welchem sich die neue Konfiguration ableiten soll. Zur Auswahl stehen alle konfigurierbaren Konnektoren im Webdesk System (z.b. dbConnector, seperatorFileConnector, templateFileConnector, etc.).
- **Name**
Name des neuen Konnektors
- **Konnektor kann schreiben**
Durch Anhaken des Parameters wird definiert, ob dieser Konnektor schreiben kann (Quellkonnektor); Ist der Parameter nicht angehakt, kann der Konnektor nur lesen (Zielkonnektor). Ist ein Vaterkonnektor grundsätzlich nicht in der Lage, zu schreiben, so ist dieser Parameter wirkungslos (Beispiel: summarizeConnector)

Konfigurierbare Konnektoren

Die folgende Konnektoren können konfiguriert werden und müssen daher vor der Verwendung als neue Konnektor-Konfiguration angelegt werden. Im Gegensatz dazu können nicht konfigurierbare Konnektoren ohne Anlage einer Konfiguration sofort verwendet werden.

Name	Beschreibung	Möglichkeiten
dbConnector	lesender und schreibender Zugriff auf relationale Datenbanken	lesen, schreiben
seperatorFileConnector	Lesen und schreiben von Textfiles. Die Spalten in den Textfiles haben entweder fixe Längen oder sind mit Seperator getrennt (csv). Die Speicherung der Textfiles kann im Filesystem oder im DocumentManagement-Modul erfolgen.	lesen, schreiben
templateFileConnector	Schreiben von individuell erstellten Textfiles nach einem Template. Mit diesem Konnektor können mittels Velocity benutzerdefinierte Files erzeugt werden (z.b. XML, EDIFACT, etc..) Speicherung erfolgt im Filesystem oder im DocumentManagement-Modul.	schreiben
summarizeConnector	Liest Daten aus einem andern Quellkonnektor und gruppiert	lesen

	die Daten und summiert sie. Ein typischer Anwendungsfall wäre z.b., Reisekosten von Mitarbeitern flexibel nach Lohnarten zu gruppieren.	
notesConnector	Lesen von Daten aus Lotus Notes Datenbanken.	lesen
travelConnector	Lesen von Reiseabrechnungen, hierbei ist entspricht jeder Datensatz genau einer Reiseabrechnung. Kann auch dazu verwendet werden, um nach einer Synchronisation den Reisetstatus in der Reiseabrechnung zu verändern.	lesen
PoPersonConnector	Lesen und Schreiben von Personen-Stammdaten und Rollenzuordnungen (d.h. wer ist zuständiger Rolleninhaber für eine bestimmte Person)	lesen, schreiben
PoGroupConnector	Lesen und Schreiben von Gruppen-Stammdaten (z.b. OEs) und Rollenzuordnungen (d.h. wer ist zuständiger Rolleninhaber/ Vorgesetzter für eine bestimmte Abteilung)	lesen, schreiben

DB Konnektor - Datenbank-Konnektor

Diese Konnektordefinition ermöglicht eine Datenbank im Webdesk in einem verknüpften Konnektor verwenden zu können. Damit können über den Webdesk Daten aus einer Datenbank gelesen bzw. in eine Datenbank geschrieben werden.

Konfiguration von Konnektoren

Speichern Speichern & Schließen Zurück Löschen

Vaterkonnektor: Teste Verbindung
 Name: *
 Connector kann schreiben:

Datenbank Eigenschaften

Benutzer: + ?
 Passwort: ?
 URL: + ?

Teile des SQL Queries

Select Statement: ?
 From Statement: + ?
 Where Statement: ?
 dbconnector_escapeCharacter: ?

Einschränkungen

Haupttabelle (update/delete): ?
 In Haupttabelle darf geschrieben werden: ?

Datenbank Eigenschaften

- **Benutzer**
Benutzername, der zum Einloggen verwendet werden soll
- **Passwort**
Passwort, welches zum Einloggen verwendet werden soll
- **URL**
Eingabe der URL der Datenbank, z.B.: jdbc:sqlserver://servername:port;databaseName=db_name;):
 - Beispiel für MySql:
jdbc:mysql://192.168.3.12:3306/TestDB
 - Beispiel für MSSql:
jdbc:mssql://192.168.3.12:1433/TestDB

Teile des SQL Queries

- **Select Statement**
Der Select-Teil des SQL Statements ist optional. Ist es leer, so wird 'select *' verwendet.
- **From Statement**
Der From-Teil des SQL Statements. (z.b.: from Tabellenname_aus_dem_Schema). Dieser Parameter muss angegeben werden.
- **Where statement**
Der Where-Teil des SQL Statements ist optional. Ist er leer so ist keine Einschränkung vorgesehen.
- **dbConnector_escapeCharacter**
damit kann eine Maskierung für Sonderzeichen angegeben werden, um diese in SQL- Abfragen verwenden zu können (meistens /)

Einschränkungen

- **Haupttabelle (update/delete)**

Die Haupttabelle, in die schlussendlich geschrieben wird. Diese Information wird nur dann verwendet, wenn der Konnektor das Ziel darstellt.

- **In Haupttabelle darf geschrieben werden**

Der Parameter definiert, ob es erlaubt ist in die Haupttabelle zu schreiben.

Teste Verbindung

Dieser Button erlaubt zu überprüfen, ob mit den eingestellten Parametern die Verbindung funktioniert.

Seperator File Connector

Ein Filekonnektor kann prinzipiell lesen und schreiben, kann aber auch so definiert werden, dass er nur liest oder nur schreibt.

Kann ein File Konnektor schreiben, so muss dies hier eingegeben werden, damit er auch in die Auswahlliste der möglichen Konnektoren aufscheint.

Verwendete Datei / Lesen



- **Wähle Datei**

Aktuell von dem Konnektor verwendete Datei zum Einlesen. Ist dieses Feld befüllt, so wurde die Datei bereits in der Datenbank gespeichert.

- **Wähle Dateipfad am Server**

Die Datei kann mit "Durchsuchen" auf dem Server ausgewählt werden.

- **Dateiupload und extrahieren der Spalten**

Datei wird auf den Server geladen und analysiert. Dabei wird versucht die Spaltennamen aus der ersten Zeile auszulesen oder die Spaltennamen werden automatisch generiert.

Verwendete Datei / Schreiben



- **Wähle Datei am Server**

Speichert die Datei mit dem angegebenen Namen in die Datenbank. Folgende Platzhalter stehen zur Verfügung: \$d (=Tag), \$M (=Monat), \$y (=Jahr), \$h (=Stunde), \$m (=Minute) und \$s (=Sekunde)

- **Als Dokument speichern / Formatierungsmuster angeben**

Schreibt die Datei unter dem angegebenen Pfad am Server. Folgende Platzhalter stehen zur Verfügung: \$d (=Tag), \$M (=Monat), \$y (=Jahr), \$h (=Stunde), \$m (=Minute) und \$s (=Sekunde)

Dateiaufbau

Dateiaufbau

- Ein Trennzeichen wird benutzt, um die Spalten zu kennzeichnen
- Spalten haben eine fixe Länge

Erste Zeile definiert Namen ?

Kommentarkennzeichner ?

Trennzeichen ?

Stringkennzeichner ?

- **Ein Trennzeichen wird benutzt, um die Spalten zu kennzeichnen**
Die Daten in der Datei sind so aufgebaut, dass ein Trennzeichen das Spaltenende kennzeichnet. Meistens wird hierfür ein ; verwendet (z.B. in CSV- Files)
- **Spalten haben eine fixe Länge**
Spalten werden nicht von einem Trennzeichen gekennzeichnet sondern haben immer eine bestimmte Länge z.B. 15 Zeichen.
- **Erste Zeile definiert Namen**
Wenn gewählt, sollte die erste Zeile der Datei die Namen der Spalten beinhalten.
- **Kommentarkennzeichner**
Definiert das Muster, das verwendet wird, um eine Zeile als Kommentar zu kennzeichnen.
- **Trennzeichen**
Das Trennzeichen wird vom Konnektor benutzt, um Spalten in der Datei zu kennzeichnen.
- **Stringkennzeichner**
Der Stringkennzeichner muss vor und nach der Zeichenfolge stehen und wird benutzt um diesen zu kennzeichnen. Dies kann durchaus nützlich sein, z.B. wenn ein Trennzeichen innerhalb der Zeichenfolge vorkommt.

Definition der Spalten

Definition der Spalten

Spaltenname	Offset	Länge	Bündigkeit	Auffüllen mit	Vorgabe	Prefix	Postfix
<input type="text" value="PERSNR"/>							<input type="text" value="▲▼"/> <input type="checkbox"/>
<input type="text" value="NAME"/>							<input type="text" value="▲▼"/> <input type="checkbox"/>
<input type="text" value="Richtige OE"/>							<input type="text" value="▲▼"/> <input type="checkbox"/>
<input "falsch"""="" type="text" value="*Orga "/>							<input type="text" value="▲▼"/> <input type="checkbox"/>
<input type="button" value="Zeile hinzufügen"/>		<input type="button" value="ausgewählte Zeile(n) entfernen"/>					

Zeige erstellte Dateien (nur Dateien bei denen der Konnektor als Ziel diente werden angezeigt.)

- **Spaltenname**
Name der Tabellenspalte
- **Offset**
Definiert an welcher Stelle einer Zeile die Spalte beginnt.
- **Länge**
Länge der vorhergehenden Spalte
- **Bündigkeit**
Dieses Feld wird nur beim Schreiben berücksichtigt und definiert die Anordnung der Spalte beim Schreiben.

- **Auffüllen mit**
Definiert die Zeilenfolge, die zum Auffüllen des Spalteninhaltes verwendet wird.
- **Vorgabe**
Der default Wert. Wird verwendet wenn kein Wert gegeben ist.
- **Präfix**
Der retournierte Wert wird um den Präfix angereichert.
- **Postfix**
Der retournierte Wert wird um den Postfix angereichert.
- **Zeile hinzufügen / Ausgewählte Zeile entfernen**
Durch Selektion der Spalte und Anklicken der Schaltfläche werden neue Zeilen hinzugefügt bzw. entfernt. Speichern.
- **Zeige erstellte Dateien**
Es werden nur Dateien angezeigt, bei denen der Konnektor als Ziel diente.

Template File Connector

Konfiguration von Konnektoren

Speichern Speichern & Schließen Zurück Löschen

Vaterkonnektor: templateFileConnector

Name: TestTemplate

Velocity Template

Velocitytemplate

```
#foreach( $row in $rows)
  Vorname= $row.Vorname
  Nachname=$row.Nachname
#end
```

Alle Zeilenschaltungen (CRs + LFs) entfernen?

Velocity Template

- **Velocity Template**
Hier können Sie Velocity- Scripts ausführen. Eine Referenz der Velocitybefehle finden Sie auf [hier](#)²
- **Alle Zeilenschaltungen (CRs + LFs) entfernen?**
Es werden alle Carriage Returns (CR = Eingabe von Return) und Line Feeds (LF = Eingabe von Space; Leerzeichen) automatisch aus der Zieldatei entfernt

Zusätzlich kann der Template Fileconnector gleichzeitig ein zweites File schreiben:

Template FileConnector für Weiterleitung

Save Object () und postProcess () werden auf eine 2. Instanz weitergeleitet, welche bereits vom Template File Konnektor selbst erzeugt wird. Die Entitätsnamen und die UID zur Speicherung von Ergebnis-Files muss an die 2. Instanz weitergegeben werden.

Hier wird der Name des 2. Template File Konnektors angegeben, an die im Falle der Ausführung von Schreiboperationen auf der aktuellen Konfiguration diese weitergeleitet werden.

Speicherort von Output

Speicherort von Output

Wähle Dateipfad am Server

Als Dokument speichern / Formatierungsmuster angeben

Als Dokument in DB und im Filesystem speichern.

test01\$d

- **Wähle Datei am Server**
Speichert die Datei mit dem angegebenen Namen in die Datenbank. Folgende Platzhalter stehen zur Verfügung: \$d (=Tag), \$M (=Monat), \$y (=Jahr), \$h (=Stunde), \$m (=Minute) und \$s (=Sekunde)
- **Als Dokument speichern / Formatierungsmuster angeben**
Schreibt die Datei unter dem angegebenen Pfad am Server. Folgende Platzhalter stehen zur Verfügung: \$d (=Tag), \$M (=Monat), \$y (=Jahr), \$h (=Stunde), \$m (=Minute) und \$s (=Sekunde)
- **Als Dokument in DB und im Filesystem speichern**
Speichert die Datei mit dem angegebenen Namen in die Datenbank. Folgende Platzhalter stehen zur Verfügung: \$d (=Tag), \$M (=Monat), \$y (=Jahr), \$h (=Stunde), \$m (=Minute) und \$s (=Sekunde)

Spalten Definition

Spalten Definition

Spaltenname

Vorname

Nachname

Zeile hinzufügen

ausgewählte Zeile(n) entfernen

Zeige erstellte Dateien (nur Dateien bei denen der Konnektor als Ziel diente werden angezeigt.)

- **Spaltenname**
Definiert den Namen der Spalte, der später in den verknüpften Konnektoren ersichtlich ist
- **Zeile hinzufügen / ausgewählte Zeilen entfernen**
- **Zeige erstellte Dateien**
Es werden nur Dateien angezeigt, bei denen der Konnektor als Ziel diente

Summarize Konnektor

Konfiguration von Konnektoren

Speichern Speichern & Schließen Zurück Löschen

Vaterkonnektor: summarizeConnector
Name: Summarize Konnektor
Connector kann schreiben:

Quellkonnektor: TmTravelConnectorDefault
Spalte für Personalnummer:
Überschrift der Zielgruppe:
Überschrift der Summe:
Null-Werte rausfiltern (d.h. keine Zeilen mit 0 Beträgen):

Überschriften der Spalten

Quellfeld	Javascript	Zielgruppe	Löschen
activeTravel			Löschen

Hinzufügen

- **Quellkonnektor**
Selektion des Quellkonnektors
- **Spalte für Personalnummer**
XX
- **Überschrift der Zielgruppe**
XX
- **Überschrift der Summe**
XX
- **Null-Werte rausfiltern**
Durch Anhaken des Parameters werden Zeilen mit 0-Beträgen nicht angezeigt,

Überschriften der Spalten

Die Spaltenüberschriften sind abhängig vom Quellkonnektor.

- **Quellfeld**
XX
- **Javascript**
XX
- **Zielgruppe**
XX
- **Hinzufügen / Löschen**
Durch Anklicken der Schaltfläche können neue Überschriften hinzugefügt werden, bzw. unerwünschte Überschriften gelöscht werden.

Bei Fragen zur detaillierten Funktionsweise der Parameter kontaktieren Sie bitte die Workflow EDV GmbH.

Notes Konnektor

Konfiguration von Konnektoren

Speichern Speichern & Schließen Zurück Löschen

Vaterkonnektor: notesConnector
Name: NotesKonnektor
Connector kann schreiben:

Datenbank Eigenschaften

Servername
Benutzername
Passwort
Datenbank
Such String

Spaltenname	Art des Feldwertes	Index
	Wert des Feldes lt. Index	

Zeile hinzufügen ausgewählte Zeile(n) entfernen

Datenbank Eigenschaften

- Servername
- Benutzername
- Passwort
- Datenbank
- Such String

Spaltenname	Art des Feldwertes	Index
	Wert des Feldes lt. Index	

Zeile hinzufügen ausgewählte Zeile(n) entfernen

Server- und Datenbankverbindung testen

- Spaltenname
- Art des Feldwertes
Dieser Parameter definiert, ob der Wert des Notes-Feldes laut Index, bzw. als Zeichenkette getrennt mit Strichpunkt (sofern mehrere Einträge vorhanden sind) ausgegeben wird
- Index
Wenn bei Type "Wert des Feldes laut Index" angegeben wird, muss der Index angegeben werden

TmTravel Konnektor

Die Aufgabe des TmTravel Connectors ist es Reisedaten zu liefern, welche dann in ein nachgelagertes System zur Auszahlung weitergeleitet werden. Da diese Auszahlung immer den letzten Schritt des eigentlichen Reiseabrechnungsworkflows bedeutet, werden nach einer erfolgreichen Synchronisation mit diesem Konnektor auch alle gelesenen Reisen verändert:

- Es wird ein neuer Status in der Reise gesetzt (damit der nächste Lauf der Synchronisation diese Reise nicht mehr anliefert)

- es wird (optional) das Datum der Synchronisation in den Reisedatensatz zurückgeschrieben. Auf diese Weise kann in Auswertungen leicht nachvollzogen werden, wann eine Reise übergeleitet wurde.

Natürlich kann der TravelKonnektor auch "nur" zum Lesen verwendet werden, ohne dass ein Reisestatus oder ein Überleitungsdatum gesetzt wird. Dies kann in der Konnektorkonfiguration festgelegt werden.

- **Zu setzender Reisestatus**

Dieser Parameter ermöglicht es, den Status zu bestimmen, der nach Abschluss gesetzt werden soll. Der angegebene Status wird nach erfolgreichem Auslesen und Durchführung des Syncs auf allen selektieren Dienstreisen gesetzt, Beispiel: Expense_Transfered (Abrechnung Übergeleitet)

- **Überleitungsdatum setzen**

Ist diese Option aktiviert, so wird nach erfolgreichem Auslesen und Durchführung der Synchronisation über das Connectorframework das Überleitungsdatum mit Heute befüllt!

1. </daisy/personalwolke-default/3236-dsy.html>
2. <http://www.jakarta.org/velocity/velocity-1.2/docs/vtl-reference-guide.html>