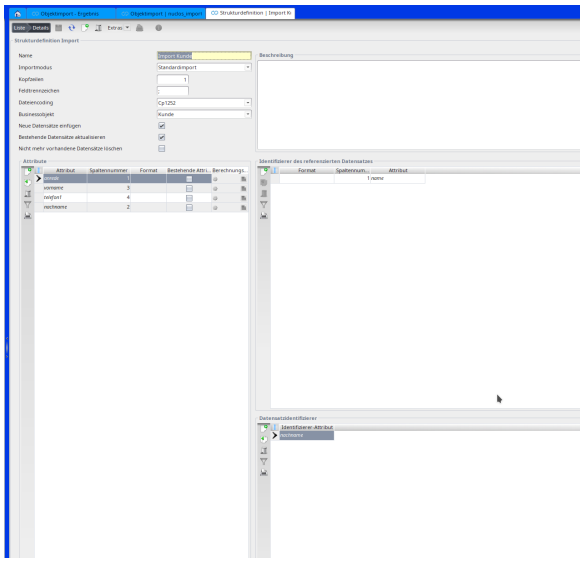
Strukturdefinition

Definition

Menüaufruf: (Konfiguration) - (Import & Export) - (Strukturdefinition)

*In der Maske **Strukturdefinition für Objektimport** wird die Zuordnung zwischen der Spalte in der Quelldatei und dem Nuclos-Feld vorgenommen.*

Konfiguration



Name Name der Strukturdefinition

Kofpzeilen gibt die Anzahl der Kopfzeilen an, die für den Import ignoriert werden sollen (z.B. Spaltenüberschriften).

Importmodus

- Beim Standardimport werden die importierten Daten über Nuclos-Speichermechanismen angelegt. D.h. es werden u.A. Regeln ausgeführt.
- Beim Direktimport werden Daten direkt in die Datenbank geschrieben, ohne dass die Nuclos Regeln durchgeführt werden.

Feldtrennzeichen ist das Trennzeichen der einzelnen Spalten in der CSV-Datei (im Normalfall ";").

Businessobjekt Zuordnung zum Importierenden Businessobjekt.

Nur Datensätze einfügen es wird grundsätzlich versucht, einen neuen Datensatz anzulegen, egal ob schon vorhanden oder nicht. Bei Eindeutigkeitsverletzungen treten Fehler im Import auf.

Bestehende Datensätze aktualisieren Es werden nachträgliche Updates durchgeführt

Nicht mehr vorhandene Datensätze löschen Wenn eines oder mehrere Felder als eindeutig erkannt werden, wird dieser Datensatz überschrieben. Datensätze, die in der Importdatei nicht mehr vorkommen, werden entfernt, wenn gesetzt ist.

Attribute

Unter **Attribut** stehen die in dem Businessobjekt zur Verfügung stehenden Felder zur Auswahl. Dieses Feld in Verbindung mit der **Spaltennummer** der Importdatei stellt die Verknüpfung dar. Mit dem Flag **Bestehende Attribute nicht überschreiben** kann ein Überschreiben auf Feldebene zusätzlich eingegrenzt werden.

Boolean Attribute

Für Boolean Attribute kann man entweder einen Berechnungsausdruck verwenden oder den Import über die Spaltennummer der Importdatei realisieren. Das Boolean in der Importdatei erkennt folgende Standards: true/false; 1/0; j/n.

Berechnungsausdruck (Groovy Script)

Über das Feld **Berechnungsausdruck** kann ein zu importierendes Attribut dynamisch berechnet werden. Über ein Groovy-Script können Sie den zu importierenden Wert für jede Zeile aus den kompletten Daten der Zeile ableiten. Dabei sehen in die Variablen **`values`** (java.lang.String[]), alle Werte der aktuellen Zeile 0-indiziert), **`line`** (java.lang.Integer, die aktuelle Zeilennummer) und **`log`** (org.nuclos.server.fileimport.ImportLogger) zur Verfügung. Wenn Sie ein Attribut dynamisch berechnen möchten, darf *keine* Spaltennummer angegeben werden.

Beispiel für Boolean-Feld:

```
if (values[2] == null || values[2].isEmpty()) {
    log.info("FALSE "+values[2]);
    return false;
} else {
    log.info("TRUE "+values[2]);
    return true;
}
```

Beispiel für Double-Feld:

```
if (values[2] == null) {
    return null;
}
else {
    return Double.parseDouble(values[2]) * 1000;
}
```

Beispiel für Referenz-Feld:

```
if (values[0] == "Frau") {
    return 40000001;
}
else if (values[0] == "Herr") {
    return 40000002;
}
return null;
```

Für Referenzfelder darf das Script natürlich nur Zahlen (Long), `null` oder leere Strings zurück liefern. Ein Leerstring wird ebenfalls als `null` behandelt.

Beispiel für Datums-Feld:

```
java.text.SimpleDateFormat dateFormat = new java.text.SimpleDateFormat("dd.MM.yyyy");
java.util.Date date = dateFormat.parse("25.02.2020");

return date;
```

Format (Parsestring)

Das Feld Format hat, abhängig von dem Datentypen, in die die Eingabe umgewandelt wird, verschiedene Bedeutungen.

> Die Implementierung erfolgt in org.nuclos.common2.fileimport.parser.CsvFileImportParserFactory. Dieser Klasse können Entwickler weitere Details entnehmen.

> Format wird nur verwendet, wenn *kein* Script hinterlegt ist.

String

Das Format hat die Form '<match>#<replacement>'. Dabei ist <match> eine Regex, die auf die Eingabe angewendet wird. Das initiale Ergebnis ist <replacement>.

Für jeden Group Match auf der Eingabe wird im Ergebnis jeder Match des reguläre Ausdrucks '$<group_number>$' durch den Match der Gruppe ersetzt.

Im Normalfall wird dies wohl verwendet, um *einen* Gruppenmatch durch in das Ergebnis zu übertragen. Beispiel: Format 'Nr(\\d)+#Nr $1', Eingabe 'Nr9' -> Ergebnis 'Nr 9', Eingabe 'nomatch' -> Ergebnis 'nomatch'.

Boolean

In diesem Fall wird das Format ***nicht*** verwendet!

Erlaubte `true` Werte: ja, yes, true, wahr, y, j, t, 1

Erlaubte `false` Werte: nein, no, false, falsch, n, f, 0

Werte, die nicht *match*en, werden zu `false`.

Date

Format wird als SimpleDateFormat verwendet. Wird kein Format angegeben, wird 'dd.MM.yyyy' verwendet.

BigDecimal

In diesem Fall wird das Format ***nicht*** verwendet!

Number (außer BigDecimal)

Wie unter **String** beschrieben. Am Ende wird nur-Leerzeichen zu `null`. Bei allem anderen wird versucht, es in den entsprechenden Zahlentyp umzuwandeln.

Datensatzidentifizierer

Das Feld **Identifizierer-Attribut** stellt eine Kombination aus Attributen dar, mit welchen ein Datensatz als eindeutig gekennzeichnet wird (wichtig für die Aktualisiserung von Daten).

Identifizierer des referenzierten Datensatzes

In diesem Bereich können Sie die Referenzen von Feldern festlegen. Wird unter den Attributen ein Referenzfeld markiert, so sind unter **Attribut Name** alle verfügbaren Felder das Referenz-Businessobjekt aufgelistet. Wählen Sie hier das Feld auf, über das das Mapping auf das Referen-Businessobjekt stattfinden soll. Zusätzlich müssen Sie die entsprechende **Spaltennummer** angeben. Beispiel: Es soll ein Attribut "Standort" importiert werden. Der Standort stellt ein Referenzfeld dar. In der CSV Datei steht in der Spalte "Standort" die Standortnummer, die für das Mapping verwendet werden kann. Also wählen Sie im "Identifizierer des referenzierten Datensatzes" die Nummer als Mapping Feld aus.

CSV Dateiimport

Aktuell wird der Import von CSV Formaten unterstützt. Die zu importierende Datei muss daher im CSV Format vorliegen.

Valides CSV Format

Da es unterschiedliche Definitionen von CSV-Formaten gibt, ist es notwendig sich an folgende Regeln zu halten damit Nuclos die Daten korrekt importieren kann:

- jedes Feld kann in doppelte Anführungsstriche gesetzt werden

```
"testA",testB,"87",900
```

- Felder mit Zeilenumbruch müssen in Anführungsstriche gesetzt werden

```
"testA",testB,"87","900
Stück"
```

- falls Felder in Anführungsstriche gesetzt werden, müssen Anführungsstriche innerhalb des Feldwerts mit einem vorangestellten Anführungsstrich maskiert werden

```
"TestA","24"" Monitor",87,900
```

- soll das Feldtrennzeichen im Feldwert verwendet werden, muss dieser in Anführungsstriche gesetzt werden

```
"TestA","TestB,",87,900
```

Siehe hierzu auch https://tools.ietf.org/html/rfc4180