

Knowledge - XML & Co.

Table of contents

1 Benutzung von Java innerhalb von XSL.....	2
2 XML Schema.....	2

1. Benutzung von Java innerhalb von XSL

Die Benutzung von Java-Befehlen innerhalb von XSL-Transformationen kann sehr nützlich sein, da sich in Java Algorithmen besser programmieren lassen und auch für viele Probleme bereits Lösungen existieren. In unserem Fall, benötigen wir eine Umwandlung der URL-Parameter in URL-konforme Syntax. Dafür wurde die encode-Methode aus der Java-Klasse `java.net.URLEncoder` verwendet. Der Zugriff auf Java Befehle innerhalb von Transformationen ist nur bei javabasierten XSLT-Prozessoren möglich. Dabei unterscheidet sich die Integration dieser wieder von Prozessor zu Prozessor. Der Standard-Prozessor unter Java (JAXP) ist Xalan.

Für die Integration von Java muss zuerst ein Namensraum für Java geschaffen werden. Dies geschieht gewöhnlich im Stylesheet-Tag.

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0" xmlns:java="http://xml.apache.org/xslt/java" >
```

Nun können innerhalb des Stylesheets alle Klassenmethoden von Java aufgerufen werden, wobei diese absolut aufgerufen werden müssen.

```
<xsl:value-of select="java:java.net.URLEncoder.encode(Name1, 'UTF-8')"/>
```

Die Parameter, die über die URL übergeben werden, müssen nun nur noch im Java-Programm decodiert werden, also wieder in den normalen Zeichensatz (Unicode) gewandelt werden.

```
URLDecoder.decode(strStreet, "UTF-8")
```

Weitere Beispiele und Informationen findet man unter folgendem Link: [hier](http://cafeconleche.org/books/xmljava/chapters/ch17s03.html) (<http://cafeconleche.org/books/xmljava/chapters/ch17s03.html>)

2. XML Schema

Mit einem [XML Schema](http://de.wikipedia.org/wiki/XML-Schema) (<http://de.wikipedia.org/wiki/XML-Schema>) (*.xsd) kann man genau bestimmen, wie die Syntax also der Aufbau eines XML-Dokumentes aussehen soll. Dies empfiehlt sich besonders, wenn man ein XML-Dokument von dritten erstellen lassen will, die von dem geforderten Aufbau nicht in Kenntnis gesetzt sind. [XML Schema](http://de.wikipedia.org/wiki/XML-Schema) (<http://de.wikipedia.org/wiki/XML-Schema>) sind die eigentlich die Nachfolger von [DTD](http://de.wikipedia.org/wiki/DTD) (<http://de.wikipedia.org/wiki/DTD>) s und haben gegenüber diesen einige Vorteile, wie einfache Syntax und Datentypunterstützung.

Hier folgt nun ein Beispiel für eine einfache Konfigurationsdatei in XML, welche ich einmal im Rahmen eines AXIS-Webservices definiert hatte. Damit Systemadministratoren oder sonstige Systemverwalter das Konfigurationsfile einfach anpassen können und dabei kein ungültiges XML-Dokument entsteht, hatte ich ein XML-Schema angelegt, dass das XML-Konfigurationsfile validiert.

Knowledge - XML & Co.

```
<?xml version="1.0" encoding="utf-8"?>
<config system="test" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="config.xsd">
  <test>
    <property name="voname">vo</property>
    <property name="vopass">vo_</property>
    <property
name="jdbc_url">jdbc:oracle:thin:@192.168.2.11:1521:cmsp0</property>
    <property name="jdbc_class">oracle.jdbc.driver.OracleDriver</property>
  </test>
  <live>
    <property name="voname">vo</property>
    <property name="vopass">vo_</property>
    <property
name="jdbc_url">jdbc:oracle:thin:@192.168.2.11:1521:cmsp0</property>
    <property name="jdbc_class">oracle.jdbc.driver.OracleDriver</property>
  </live>
</config>
```

XML-Schema:

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:annotation>
    <xsd:documentation>Dieses Schema dient zur Validierung des Config-Files
für den Address-WebServices des Mobile Office.</xsd:documentation>
  </xsd:annotation>

  <xsd:element name="config" type="configType"/>
  <xsd:element name="test" type="propListType"/>
  <xsd:element name="live" type="propListType"/>

  <xsd:element name="property">
    <xsd:complexType>
      <xsd:simpleContent>
        <xsd:extension base="xsd:string">
          <xsd:attribute name="name" type="xsd:string" use="required"/>
        </xsd:extension>
      </xsd:simpleContent>
    </xsd:complexType>
  </xsd:element>

  <xsd:complexType name="configType">
    <xsd:all>
      <xsd:element ref="test" minOccurs="1" maxOccurs="1"/>
      <xsd:element ref="live" minOccurs="1" maxOccurs="1"/>
    </xsd:all>
    <xsd:attribute name="system" type="systemType" use="required"/>
  </xsd:complexType>

  <xsd:complexType name="propListType">
    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:element ref="property"/>
    </xsd:sequence>
  </xsd:complexType>
```

```
<xsd:simpleType name="systemType">  
  <xsd:restriction base="xsd:string">  
    <xsd:enumeration value="test"/>  
    <xsd:enumeration value="live"/>  
  </xsd:restriction>  
</xsd:simpleType>  
</xsd:schema>
```