Listings zum Kurs

xsl-template

XML Listing:<?xml version="1.0"
encoding="UTF-8"?>
<test>

<titel>Dateiendungen</titel>
 <glossar>

<eintrag>
  <begriff>bak</begriff>

<bedeutung>Backup-Datei</bedeutung>

</eintrag>
  <eintrag>

<begriff>bmp</begriff>

<bedeutung>Bitmap-Grafik</bedeutung>

</eintrag>
 </glossar>
</test>XSLT
Listing:<?xml version="1.0"
encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >
<xsl:template
match="/">

<html><head></head>

<body>
  <xsl:apply-templates />

</body>

</html>
</xsl:template>
<xsl:template
match="titel">
 <h1><xsl:value-of select="."
/></h1>
</xsl:template>
<xsl:template
match="glossar/eintrag">
  <xsl:apply-templates
/>
</xsl:template>
<xsl:template
match="begriff">
 <b
style="color:blue"><xsl:apply-templates
/>:
</xsl:template>
<xsl:template
match="bedeutung">
  <xsl:value-of select="."
/>
</xsl:template>
</xsl:stylesheet>

xsl-apply-templates

XML Listing:<?xml version="1.0"
encoding="UTF-8"?>
<test>
  <text>

  Was du <zeit>heute</zeit>
  kannst besorgen,
  das
verschiebe nicht
  auf <zeit>morgen</zeit>.

</text>
</test>XSLT Listing:<?xml version="1.0"
encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >
<xsl:template
match="/">

<html><head></head>

<body>
  <xsl:apply-templates />

</body>

</html>
</xsl:template>
<xsl:template
match="titel">
 <h1><xsl:value-of select="."
/></h1>
</xsl:template>
<xsl:template
match="glossar/eintrag">
  <xsl:apply-templates
/>
</xsl:template>
<xsl:template
match="begriff">
 <b
style="color:blue"><xsl:apply-templates
/>:
</xsl:template>
<xsl:template
match="bedeutung">
  <xsl:value-of select="."
/>
</xsl:template>
</xsl:stylesheet>

xsl-with-param

XML Listing:<?xml version="1.0"
encoding="UTF-8"?>
<test>

<start>10</start>

<ende>20</ende>
</test>XSLT Listing Version

1.0:<?xml version="1.0"
encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template
match="/">

<html><head></head><body>

<table border="1">
 <xsl:call-template
name="schleife">
 <xsl:with-param name="zaehler"
select="number(/test/start)"/>
 </xsl:call-template>

</table>

</body></html>
</xsl:template>
<xsl:t
emplate name="schleife">
 <xsl:param name="zaehler"
/>
 <xsl:choose>
 <xsl:when test="$zaehler
<= number(/test/ende)">
 <tr>

<td><xsl:value-of select="$zaehler"
/></td>
 <td><xsl:value-of
select="$zaehler * $zaehler" /></td>

</tr>
 <xsl:call-template name="schleife">

<xsl:with-param name="zaehler" select="$zaehler + 1" />

</xsl:call-template>
 </xsl:when>

<xsl:otherwise>
 <xsl:call-template name="Abbruch"
/>
 </xsl:otherwise>

</xsl:choose>
</xsl:template>
<xsl:template
name="Abbruch">
 <xsl:comment>Schleife
beendet!</xsl:comment>
</xsl:template>
</xsl:styles
heet>XSLT Listing Version 2.0:<?xml version="1.0"
encoding="UTF-8"?>
<xsl:stylesheet
version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"&gt
;
<xsl:template
match="/">
<html>
<head></head
>
<body>
<table border="1">

<xsl:variable name="from" select="xs:integer(/test/start)"/>

<xsl:variable name="to" select="xs:integer(/test/ende)"/>

<xsl:for-each select="$from to $to">

<tr><td><xsl:value-of
select="."/></td><td><xsl:value-of
select=".*2"/></td></tr>

</xsl:for-each>
</table>
</body>
&
lt;/html>
</xsl:template>
</xsl:stylesheet>

tunnel

XSLT Listing:<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="2.0">
 <xsl:output omit-xml-declaration="yes"
/>
 <xsl:template match="/">
 <xsl:call-template
name="template1">
 <xsl:with-param name="p1"
tunnel="yes">
 pink
 </xsl:with-param>

</xsl:call-template>
 </xsl:template>
 <xsl:template
name="template1">
 <xsl:param
name="p1">
 orange
 </xsl:param>
 1. p1 in
template1:
 <xsl:value-of select="$p1" />

<xsl:call-template name="template2" />

</xsl:template>
 <xsl:template
name="template2">
 <xsl:call-template name="template3"
/>
 </xsl:template>
 <xsl:template
name="template3">
 <xsl:param name="p1"
tunnel="yes">
 blue
 </xsl:param>
 2. p1 in
template3:
 <xsl:value-of select="$p1" />

</xsl:template>
</xsl:stylesheet>

rekursiv

Ein XML Dokument zu diesem Use Case findet man unter xfsdump.rekursiv.xml.XSLT
Listing:<?xml version="1.0"
encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:template
match="/">
 <html>
 <head>

<title>rekursiv</title>
 </head>

<body>
 <xsl:apply-templates
select="/filesystem/body/folder"/>

</body>
 </html>
 </xsl:template>

<xsl:template match="folder">

<xsl:value-of select="@name"/>
 <xsl:if
test="folder">
 <xsl:apply-templates
select="folder"/>

</xsl:if>

</xsl:template>
</xsl:stylesheet>

copying

XML Listing:<?xml version="1.0"
encoding="UTF-8"?>
<filesystem>

<head>
 <path>K:\xslkurs\rekursiv\</path>

<date>2004-11-05T14:00:00</date>

</head>
</filesystem>XSLT Listing:<?xml
version="1.0" encoding="UTF-8"?>
<xsl:stylesheet
version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:template
match="/">
 <filesystem>

<head>
 <xsl:for-each
select="/filesystem/head/*">
 <xsl:value-of
disable-output-escaping="yes"
select="concat('<',local-name(),'>')"/>

<xsl:value-of select="."/>
 <xsl:value-of
disable-output-escaping="yes"
select="concat('</',local-name(),'>')"/>

</xsl:for-each>
 </head>

</filesystem>

</xsl:template>
</xsl:stylesheet>

xsl-copy

XML Listing:<?xml version="1.0"
encoding="UTF-8"?>
<filesystem>

<head>
 <path>K:\xslkurs\rekursiv\</path>

<date>2004-11-05T14:00:00</date>

</head>
</filesystem>XSLT Listing:<?xml
version="1.0" encoding="UTF-8"?>
<xsl:stylesheet
version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output
method="xml" />
 <xsl:template
match="/"><xsl:apply-templates
/></xsl:template>
 <xsl:template
match="@*|node()">
 <xsl:copy><xsl:apply-templates
select="@*|node()"/></xsl:copy>

</xsl:template>
</xsl:stylesheet>

xsl-copy-of

XML Listing:<?xml version="1.0"
encoding="UTF-8"?>
<filesystem>

<head>
 <path>K:\xslkurs\rekursiv\</path>

<date>2004-11-05T14:00:00</date>

</head>
</filesystem>XSLT Listing:<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml"/>
  <xsl:template match="/">
 <xsl:copy-of select="."/>
 </xsl:template>
</xsl:stylesheet>

xsl-if

XML Listing:<?xml version="1.0" encoding="UTF-8"?>
<test>
 <ergebnis>
  <name>Crystol Freak</name>
  <punkte>453</punkte>
 </ergebnis>
  <ergebnis>
 <name>Demian</name>
 <punkte>199</punkte>
  </ergebnis>
 <ergebnis>
  <name>Ueberflieger</name>
  <punkte>347</punkte>
  </ergebnis>
 <ergebnis>
  <name>CaptainX</name>
 <punkte>106</punkte>
 </ergebnis>
</test>XSLT Listing:<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
 <html><head></head><body>
 <table border="1"><xsl:apply-templates /></table>
 </body></html>
</xsl:template>
<xsl:template match="ergebnis">
  <xsl:if test="punkte >= 200">
  <tr><xsl:apply-templates /></tr>
 </xsl:if>
</xsl:template>
<xsl:template match="name">
 <td><xsl:value-of select="." /></td>
</xsl:template>
<xsl:template match="punkte">
 <td><xsl:value-of select="." /></td>
</xsl:template>
</xsl:stylesheet>

xsl-choose

XML Listing:<?xml version="1.0" encoding="UTF-8"?>
<test>
 <number>69</number>
 <number>12345</number>
 <number>743</number>
 <number>915743</number>
</test>XSLT Listing:<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
 <html><head></head><body>
 <h3>Die Zahlen lauten:</h3>
 <xsl:apply-templates />
 </body></html>
</xsl:template>
<xsl:template match="number">
  <div>
  <xsl:value-of select="." />
  <xsl:variable name="value" select="." />
  <xsl:choose>
  <xsl:when test="$value < 10000">
  <xsl:text>(eine kleine

Zahl)</xsl:text>
 </xsl:when>

<xsl:otherwise>
 <xsl:text> (eine grosse
Zahl)</xsl:text>
 </xsl:otherwise>

</xsl:choose>

</div>
</xsl:template>
</xsl:stylesheet>

xsl-for-each

XML Listing:<?xml version="1.0"
encoding="UTF-8"?>
<test>

<ergebnis>
 <name>Crystol
Freak</name>
 <punkte>453</punkte>

</ergebnis>
 <ergebnis>

<name>Demian</name>

<punkte>199</punkte>
 </ergebnis>

<ergebnis>
 <name>Ueberflieger</name>

<punkte>347</punkte>
 </ergebnis>

<ergebnis>
 <name>CaptainX</name>

<punkte>106</punkte>

</ergebnis>
</test>XSLT Listing:<?xml
version="1.0" encoding="UTF-8"?>
<xsl:stylesheet
version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template
match="/">

<html><head></head><body>

<table border="1"><xsl:apply-templates
/></table>

</body></html>
</xsl:template>
<xsl:t
emplate match="ergebnis">
 <xsl:if test="punkte >=
200">
 <tr><xsl:apply-templates
/></tr>

</xsl:if>
</xsl:template>
<xsl:template
match="name">
 <td><xsl:value-of select="."
/></td>
</xsl:template>
<xsl:template
match="punkte">
 <td><xsl:value-of
select="."
/></td>
</xsl:template>
</xsl:stylesheet&amp
;gt;

xsl-sort

XML Listing:<?xml version="1.0"
encoding="UTF-8"?>
<game>
 <player
name="Nina" points="19"/>
 <player name="Lea"
points="10"/>
 <player name="Oliver"
points="27"/>
</game>XSLT Listing:<?xml
version="1.0" encoding="UTF-8"?>
<xsl:stylesheet
version="2.0"

xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template
match="/">

<html><head></head><body>

<table border="1">
 <xsl:for-each
select="game/player">
 <xsl:sort select="@points"
order="descending"
 data-type="number"/>

<tr>
 <td><xsl:value-of select="@name"
/></td>
 <td><xsl:value-of
select="@points" /></td>
 </tr>

</xsl:for-each>

</table></body></html>
</xsl:template&am
p;gt;
</xsl:stylesheet>

xsl-element

XML Listing:<?xml version="1.0" encoding="UTF-8"?>
<shopping>
 <products>
 <product name="Schaukelstuhl" price="99" currency="EUR" />
 <product name="Kaffeemaschine" price="49,95" currency="EUR" />
 <product name="Schraubenzieher" price="4,95" currency="EUR" />
 <product name="Sofakissen" price="29,95" currency="EUR" />
 </products>
</shopping>XSLT Listing:<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:attribute-set name="green">
 <xsl:attribute name="bgcolor">#00FF00</xsl:attribute>
 </xsl:attribute-set>
 <xsl:attribute-set name="yellow">
 <xsl:attribute name="bgcolor">#FFFF00</xsl:attribute>
 </xsl:attribute-set>
 <xsl:template match="/">
 <html>
 <head>
 <title>xsl:element sample</title>
 </head>
 <body>
 <table border="1">
 <xsl:apply-templates
 select="shopping/products/product" />
 </table>
 </body>
 </html>
 </xsl:template>
 <xsl:template match="product">
 <xsl:choose>
 <xsl:when test="position() mod 2 = 0">
 <xsl:element name="tr" use-attribute-sets="green">
 <xsl:call-template name="product" />
 </xsl:element>
 </xsl:when>
 <xsl:otherwise>
 <xsl:element name="tr" use-attribute-sets="yellow">
 <xsl:call-template name="product" />
 </xsl:element>
 </xsl:otherwise>
 </xsl:choose>
 </xsl:template>
 <xsl:template name="product">
 <td>
 <xsl:value-of select="@name" />
 </td>
 <td>
 <xsl:value-of select="@price" />
 </td>
 </xsl:template>
</xsl:stylesheet>

xsl-attribute

XML Listing:<?xml version="1.0" encoding="UTF-8"?>
<doc>
 <countries>
 <country code="de">Germany</country>
 <country code="fr">France</country>
 <country code="it">Italy</country>
 <country code="ch" default="true">
 Switzerland
 </country>
 </countries>
</doc>XSLT Listing:<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
 <html>
 <head><title>xsl:attribute sample</title></head>
 <body>
 <form>
 country:
 <select name="country">
 <xsl:for-each select="doc/countries/country">
 <option value="{@code}">
 <xsl:if test="@default = 'true'
 or @default = 'yes'">
 <xsl:attribute name="selected">
 true

</xsl:attribute>
 </xsl:if>
 <xsl:value-of select="."/>
 </option>
 </xsl:for-each>
 </select>
 </form>
 </body>
 </html>
</xsl:template>
</xsl:stylesheet>

xsl-text

XML Listing:<?xml version="1.0" encoding="UTF-8"?>
<doc>
 <text>Die Welt ist ein Dorf</text>
</doc>XSLT Listing:<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
 <html><head></head><body>
 <p><xsl:value-of select="/doc/text" /></p>
 <script type="text/javascript">
 <xsl:text>
 <!--
 document.write("<p>" +
 document.lastModified +
 "</p>");
 //-->
 </xsl:text>
 </script>
 </body></html>
</xsl:template>
</xsl:stylesheet>

xsl-key

XML Listing:<?xml version="1.0" encoding="UTF-8"?>
<doc>
 <countries>
 <country code="de">Germany</country>
 <country code="fr">France</country>
 <country code="it">Italy</country>
 <country code="ch"
 default="true">
 Switzerland
 </country>
 </countries>
</doc>XSLT Listing:<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:key name="country" match="/doc/countries/country" use="@code"/>
 <xsl:template match="/">
 <html>
 <body>
 ch : <xsl:value-of select="key('country','ch')"/>
 de : <xsl:value-of select="key('country','de')"/>
 fr : <xsl:value-of select="key('country','fr')"/>
 it : <xsl:value-of select="key('country','it')"/>
 </body>
 </html>
</xsl:template>
</xsl:stylesheet>

xsl functions (compareCI)

XSLT Listing:<xsl:stylesheet version="2.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns:std="https://www.simtech-ag.ch">
 <xsl:output omit-xml-declaration="yes" />
 <xsl:function name="std:compareCI">
 <xsl:param name="left" />
 <xsl:param name="right" />
 <xsl:value-of
 select="compare(upper-case($left),upper-case($right))" />
 </xsl:function>
 <xsl:template match="/">
 compareCI red,blue:
 <xsl:value-of select="std:compareCI('red','blue')" />
 compareCI red,red:
 <xsl:value-of

select="std:compareCI('red','red')" />
 compareCI red,Red:
 <xsl:value-of select="std:compareCI('red','Red')" />
 compareCI red,Yellow:
 <xsl:value-of select="std:compareCI('red','Yellow')" />
 </xsl:template>
</xsl:stylesheet>

xsl functions (reverse)

XSLT Listing:<xsl:stylesheet version="2.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:std="https://www.simtech-ag.ch">
 <xsl:output omit-xml-declaration="yes" />
 <xsl:function name="std:reverse" as="xs:string">
 <xsl:param name="sentence" as="xs:string" />
 <xsl:sequence
 select="
if (contains($sentence, ' '))
 then concat(std:reverse(
 substring-after(
 $sentence, ' ')),
 ' ',
 substring-before($sentence, ' '))
 else $sentence" />
 </xsl:function>
 <xsl:template match="/">
 <xsl:value-of select="std:reverse('All over the world')"/>
 </xsl:template>
</xsl:stylesheet>

multiple document

XML Listing:<?xml version="1.0" encoding="UTF-8"?>
<address>
 <firstname>Hans</firstname>
 <lastname>Muster</lastname>
 <street>Musterstrasse 6</street>
 <postalcode>3000</postalcode>
 <city>Musterstadt</city>
</address>XML Listing (i18n):<?xml version="1.0" encoding="UTF-8"?>
<i18n>
 <lang code="de" default="true">
 <item name="title">Titel</item>
 <item name="lastname">Nachname</item>
 <item name="firstname">Vorname</item>
 <item name="street">Strasse</item>
 <item name="postalcode">PLZ</item>
 <item name="city">Ort</item>
 </lang>
 <lang code="en">
 <item name="title">Title</item>
 <item name="lastname">Lastname</item>
 <item name="firstname">Firstname</item>
 <item name="street">Street</item>
 <item name="postalcode">ZIP Code</item>
 <item name="city">City</item>
 </lang>
 <lang code="fr">
 <item name="title">Titre</item>
 <item name="lastname">Nom</item>
 <item name="firstname">Prenom</item>
 <item name="street">Route</item>
 <item name="postalcode">Code Postal</item>
 <item name="city">Lieu</item>
</lang>
</i18n>XSLT Listing:<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:variable name="i18n" select="document('i18n.xml')/i18n"/>

<xsl:variable name="lang" select="$i18n/lang[@default = 'true']"/>
<xsl:template match="/">
 <html>
 <head>
 <title><xsl:value-of select="$lang/item[@name = 'title']"/></title>
 </head>
 <body>...</body>
 </html>
</xsl:template>
</xsl:stylesheet>

multiple output documents

XML Listing (toc.html):<!DOCTYPE HTML><html>
 <head>
 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
 <title>Table of Contents</title>
 </head>
 <body>
 <h1>Table of Contents</h1>
 <p>Introduction</p>
 <p>Jump Start</p>
 <p>XPath</p>
 <p>Templates</p>
 <p>Processing Instructions</p>
 <p>Multiple Source Documents</p>
 <p>XSL Tiles</p>
 <p>XSL Formatting Objects</p>
 <p>XSL Applications</p>
 </body>
</html>XSLT Listing:<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output name="toc" method="html"/>
 <xsl:output name="section" method="html"/>
 <xsl:template match="/">
 <xsl:result-document href="toc.html" format="toc">
 <html><head><title>Table of Contents</title></head><body>
 <h1>Table of Contents</h1>
 <xsl:for-each select="/*/body/(*[1] | h1)">
 <p><xsl:value-of select="."/></p>
 </xsl:for-each>
 </body></html>
 </xsl:result-document>
 <xsl:for-each-group select="/*/body/*" group-starting-with="h1">
 <xsl:result-document href="section{position()}.html"
 format="section">
 <html><head><title><xsl:value-of select="."/></title></head>
 <body>
 <xsl:copy-of select="current-group()"/>
 </body></html>
 </xsl:result-document>
 </xsl:for-each-group>
 </xsl:template>
</xsl:stylesheet>

xsl-import

XSLT Listing (main.xsl):<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:import

href="layout.xsl" />
 <xsl:template match="/">

<xsl:call-template name="layout" />

</xsl:template>
 <xsl:template name="header">

my header
 </xsl:template>
 <xsl:template
name="menubar">
  my menubar
 </xsl:template>

<xsl:template name="menutree">
  my menutree

</xsl:template>
 <xsl:template name="footer">

my footer
 </xsl:template>
 <xsl:template
name="main">
  my main

</xsl:template>
</xsl:stylesheet>XSLT Listing
(layout.xsl):<?xml version="1.0"
encoding="UTF-8"?>
<xsl:stylesheet version="2.0"

xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:template
name="layout">
  <html>
  <head>

 <title>XML/XSL Tiles Sample</title>
    ...

</head>
  <body topmargin="0" leftmargin="0"
rightmargin="0"
    bottommargin="0">
    ...

<xsl:call-template name="header" />
    ...
    ...

<xsl:call-template name="menubar" />
    ...
    ...

<xsl:call-template name="menutree" />
    ...
    ...

<xsl:call-template name="main" />
    ...
    ...

<xsl:call-template name="footer" />
    ...

</body>
  </html>
 </xsl:template>

<xsl:template name="header">
  default header

</xsl:template>
 <xsl:template
name="menubar">
  default menubar

</xsl:template>
 <xsl:template
name="menutree">
  default menutree

</xsl:template>
 <xsl:template name="footer">

default footer
 </xsl:template>
 <xsl:template
name="main">
  default main

</xsl:template>
</xsl:stylesheet>

xsl-include

XSLT Listing (main.xsl):<?xml version="1.0"
encoding="UTF-8"?>
<xsl:stylesheet version="2.0"

xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:include
href="trace.xsl" />
 <xsl:template match="/">

<html>
  <head>
  <xsl:call-template
name="trace">
  <xsl:with-param
name="text">
  start header

</xsl:with-param>
  </xsl:call-template>

</head>
  <body>
  <xsl:call-template
name="trace">
  <xsl:with-param
name="text">
  start body
  </xsl:with-param>

 </xsl:call-template>
  </body>

</html>

</xsl:template>
</xsl:stylesheet>XSLT Listing
(trace.xsl):<?xml version="1.0"
encoding="UTF-8"?>
<xsl:stylesheet version="2.0"

xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:template
name="trace">
  <xsl:param name="text" />

<xsl:text
disable-output-escaping="yes"><--</xsl:text>

<xsl:value-of select="$text" />
  <xsl:text
disable-output-escaping="yes">--></xsl:text>

</xsl:template>
</xsl:stylesheet>

Simtech AG
Finkenweg 23
3110 Münsingen
Schweiz

Impressum

Das Copyright für sämtliche Inhalte dieser Website liegt bei Simtech AG, Schweiz.
Beachten Sie auch unsere Hinweise zum Urheberrecht, Datenschutz und Haftungsauschluss.
Jeder Hinweis auf Fehler nehmen wir gerne entgegen.

Copyright

2024 Simtech AG, All rights reserved, Powered by stack.ch written in Golang by Daniel Schmutz

https://www.simtech-ag.ch/k%C3%B6niz