



## Ausgabeformate HTML und CHM

<b>Ausgabeformat HTML.....</b>	<b>2</b>
Umlaute in Navigationsdatei.....	2
<b>Ausgabeformat HTML mit dem Plugin TOCJS.....</b>	<b>3</b>
Installation und Test im DITA-OT 1.4.....	3
Test und Anpassung im DITA-OT 1.5.....	3
Dateiübersicht.....	4
Build-Datei anpassen.....	5
Javascript-Navigation generieren.....	7
<b>Ausgabeformat CHM (Windows Help).....</b>	<b>8</b>
Symbolschaltflächen hinzufügen.....	8
Umlaute im Inhaltsverzeichnis.....	10
<b>Impressum.....</b>	<b>11</b>

# Ausgabeformat HTML

---

Tipps und Tools bei der HTML-Generierung.

## Umlaute in Navigationsdatei

---

Um deutsche Umlaute im Navigationsframe richtig darzustellen, ist eine kleine Anpassung im DITA-OT nötig.

In der Datei `\DITA-OT\xsl\map2htmltoc.xsl` ändern Sie im Template `generateCharset` folgenden Eintrag:

```
<meta http-equiv="Content-Type" content="text/html;  
charset=utf-8"/><xsl:value-of select="$newline"/>
```

um in:

```
<meta http-equiv="Content-Type" content="text/html;  
charset=ISO-8859-1"/><xsl:value-of select="$newline"/>
```

# Ausgabeformat HTML mit dem Plugin TOCJS

---

Tipps und Tools bei der HTML-Generierung mit einem Javascript-Navigationsframe.

## Installation und Test im DITA-OT 1.4

---

Anleitung: Original-Anleitung von Shawn McKenzie zur Installation und Integration des Plugins.

Once you have obtained the latest tocjs plugin, you need to install and integrate it into the DITA Open Toolkit

1. Obtain the latest version of the tocjs plugin. Ab DITA OT 1.5 ist dieses Plug bereits in der Auslieferung enthalten.
2. Unzip the plugin in your demo directory.  
This should give you a tocjs directory.
3. From your base DITA directory, type: `ant -f integrator.xml`.  
You should see output similar to the following message:

```
Buildfile: integrator.xml

BUILD SUCCESSFUL
Total time: 1 seconds
```

The tocjs transtype should be registered with the Open Toolkit.

4. Type: `ant -f demo/tocjs/buildsample.xml sample2tocjs`.  
The end of the message output should say:

```
dita2tocjs:

BUILD SUCCESSFUL
Total time: 6 seconds
```

5. Open `demo/tocjs/out/sample/frameset.html` in a browser.  
This documentation should be displayed in a frameset that includes a JavaScript TOC in the left pane.
6. Damit im linken Navigationsframe die deutschen Umlaute richtig dargestellt werden, editieren Sie die Datei `\demo\tocjs\xsl\tocjs.xml`. Ändern Sie den Wert für encoding von "utf-8" auf "ISO-8859-1".

```
<xsl:output method="text" encoding="ISO-8859-1"/>
```

## Test und Anpassung im DITA-OT 1.5

---

Anleitung: Da das Plugin bereits Bestandteil des OTs ist, bedarf es nur noch einer Anpassung.

1. Zum Testen des Plugins öffnen Sie per `startcmd.bat` eine Eingabeaufforderung des DITA-OTs und setzen Sie folgenden Befehl ab: `ant -f demo/tocjs/buildsample.xml sample2tocjs`.  
Zum Ende der Ergebnisausgabe sollte stehen:

```
dita2tocjs:

BUILD SUCCESSFUL
Total time: 6 seconds
```

- Öffnen Sie `demo/tocjs/out/sample/frameset.html` in einem Internetbrowser. Die Dokumentation zum Plugin TOCJS sollte sich mit einem Javascript-Inhaltsverzeichnis im linken Frame öffnen.
- Damit im linken Navigationsframe die deutschen Umlaute richtig dargestellt werden, editieren Sie die Datei `\demo\tocjs\xsl\tocjs.xml`. Ändern Sie den Wert für `encoding` von `"utf-8"` auf `"ISO-8859-1"`.

```
<xsl:output method="text" encoding="ISO-8859-1" />
```

## Dateiübersicht

Konzept: Die HTML-Frameset-Dateien und Javascript-Dateien im Outputverzeichnis des Plugins TOCJS.

### Output-Dateien

Nach einem erfolgreichem Build befinden sich folgende Dateien im Output-Verzeichnis:

Name	Größe	Typ
css		Dateiordner
images		Dateiordner
img		Dateiordner
topics		Dateiordner
delta.gif	1 KB	GIF-Bild
deltaend.gif	1 KB	GIF-Bild
event.js	38 KB	JScript Script File
frameset.html	1 KB	HTML Document
index.html	28 KB	HTML Document
tocnav.html	2 KB	HTML Document
toctree.js	52 KB	JScript Script File
treeview.js	48 KB	JScript Script File
yahoo.js	3 KB	JScript Script File

Abb. 1: Dateiübersicht des Outputs

### toctree.js


Diese Datei ist das eigentliche Hauptprodukt des Plugins. Ist diese nicht generiert worden, ist der build fehlgeschlagen. Sie verwandelt die bisher starre HTML-Navigation in eine ausklappbare Javascript-Verzeichnisbaum-Struktur. Sie wird im Output in die `tocnav.html` eingebunden und bildet mit dieser dann den linken Navigationsframe in Form eines Verzeichnisbaums.

### Basefiles

Die Ordner `css` und `img` sowie die Dateien `event.js`, `treeview.js`, `yahoo.js` und `tocnav.html` sind die sogenannten Basefiles des Plugins. Sie werden durch die build-Datei vom Plugin-Verzeichnis `\demo\tocjs\basefiles` in das HTML-Outputverzeichnis kopiert.

Die Datei `tocnav.html` im Ordner `\demo\tocjs\basefiles` könnten Sie eindeutschen, so dass nicht mehr **Expand all** und **Collapse all** im Navigationsbaum steht. Eine nicht sehr elegante Übersetzung wäre **Ausklappen** und **Einklappen**. Diese Datei stellt durch die Einbindung der individualisierten `toctree.js` den linken Navigationsframe.

### frameset.html

 **Hinweis:** Datei muss angepaßt werden!

Diese Datei ist im Output die Startdatei. Doppelklickt man diese, öffnet sich der HTML-Output korrekt mit der `tocnav.html` als Navigationsdatei und der entsprechend festgelegten Startdatei des Contentframes rechts. Sie wird innerhalb der XML-Quelldateien abgelegt und in das HTML-Outputverzeichnis kopiert (siehe dazu den Kopierbefehl in der Build-Datei). Folgende grün unterstrichenen Anpassungen bezüglich der Frames wurden vorgenommen:

```

C:\DITA-SRC\out\html-admin\frameset.html
1 <!-- This file is part of the DITA Open Toolkit project hosted on
2 Sourceforge.net. See the accompanying license.txt file for
3 applicable licenses.-->
4 <!-- (c) Copyright IBM Corp. 2004, 2005 All Rights Reserved. -->
5
6 <html>
7 <html xml:lang="de" lang="de">
8 <head>
9 <link rel="shortcut icon" href="favicon.ico" type="image/x-icon">
10 <link rel="stylesheet" type="text/css" href="commonltr.css"/>
11 <title>Dokupedia 04/09 (Auso/Civiso)</title>
12 </head>
13 <frameset cols="21%,*" bordercolor="#FFFFFF">
14 <frame name="tocwin" marginwidth="0" src="tocnav.html" scrolling="auto"/>
15 <frame name="contentwin" src="topics/common/a0_start_NAV_startseite.html"
16 </frameset>
17 <p>Dieses Projekt verwendet Frames. Bei Ihnen werden keine Frames ange
18 </noframes>
19 </frameset>
20 </html>

```

Abb. 2: Frames in der frameset.html

## index.html

Diese Datei ist das konventionelle Ergebnis der XHTML-Output-Generierung - und ein Zwischenschritt auf dem Weg zum Javascript-Verzeichnisbaum. Sie ist die alte starre HTML-Navigationsdatei, wie sie ohne das Plugin *tocjs* generiert wurde.

## Verzeichnisse des HTML-Contents

Die Ordner *topics* und *images* sind hier im Beispiel die normalen Ergebnis-Dateien des XHTML-Outputs, wie er weiterhin im rechten Content-Frame zur Anzeige kommt.

## Build-Datei anpassen

Konzept: Beispiel einer Build-Datei für das Plugin TOCJS.

### DITA-Verzeichnisstruktur

Zum Verstehen der Build-Datei ist es nützlich, die DITA-Verzeichnisstruktur, wie sie im Beispiel genutzt wurde, zu kennen.

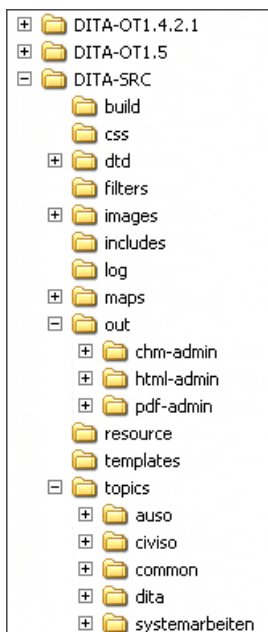


Abb. 3: Quelldateien parallel zu DITA-OT

Es wurden (08/2009) noch beide DITA-OT-Versionen genutzt, da das 1.5er noch keine Stichwörter (Index) im PDF2-Output generierte. Die Quelldateien wurden in einem Extraverzeichnis ausgelagert, damit sie schneller gesichert werden können. Diese Struktur ist nicht zwingend, soll aber das Beispiel der build-Datei illustrieren.

## Die build-Datei

Für den tocjs-Output wurde die bereits vorhandene XHTML-Build-Datei angepaßt. Das alte HTML-Output-Verzeichnis blieb dasselbe. Die grauen Zeilen kennzeichnen die neu hinzugekommenen TOCJS-Anpassungen. Der Übersicht wegen wurde auf die Notation `${file.separator}` verzichtet.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<project basedir="..\.." default="all" name="Administrator Dokumentation">
<property name="dita.dir" value="C:\DITA-OT1.5"/>
<import file="\${dita.dir}\integrator.xml"/>

<import file="\${dita.dir}\build.xml" />

<target name="all" depends="integrate" >
<ant antfile="\${dita.dir}\build.xml" target="init">
<property name="args.logdir" value="C:\DITA-SRC\log" />
<property name="args.input" value="C:\DITA-SRC\ditalog.ditamap" />
<property name="output.dir" value="C:\DITA-SRC\out\html-admin" />
<property name="transtype" value="xhtml" />
<property name="args.css" value="C:\DITA-SRC\css\ditalog.css" />
<property name="args.csspath" value="css" />
<property name="args.dita.locale" value="de-de" />
<property name="args.copycss" value="yes" />
<property name="args.hdr" value="C:\DITA-SRC\includes\header.xml" />
<property name="args.ftr" value="C:\DITA-SRC\includes\footer.xml" />
<property name="args.xsl" value="\${dita.dir}\xsl\dita2xhtml.xsl" />
<property name="dita.input.valfile" value="C:\DITA-SRC\filters\ditalog.ditaval" />
<property name="validate" value="true" />
</ant>
</target>
```

Hier wird die angepaßte frameset.html, die im Quellordner \includes liegt, in das HTML-Output-Verzeichnis kopiert:

```
<copy todir="C:\DITA-SRC\out\html-admin">
<fileset dir="C:\DITA-SRC\includes">
<include name="frameset.html" />
</fileset>
</copy>
```

Hier werden sämtliche Bilddateien in das HTML-Output-Verzeichnis kopiert:

```
<copy todir="C:\DITA-SRC\out\html-admin\images">
<fileset dir="C:\DITA-SRC\images">
<include name="**/*.gif" />
</fileset>
</copy>

</target>
```

Es folgt die eigentliche Anpassung für das Javascript-Menü. Die Datei toctree.js wird nach der Generierung ins HTML-Output-Hauptverzeichnis abgelegt. Innerhalb des Copy-Befehls werden sämtliche Basefiles des Plugins aus dem Ordner \demo\tocjs\basefiles in das HTML-Output-Hauptverzeichnis kopiert.

```
<target name="sample2tocjs" description="Generate tocjs javascript file"
depends="all">
<property name="content.frame" value="contentwin"/>
<antcall target="dita2tocjs">
<param name="transtype" value="tocjs"/>
<param name="args.input" value="C:\DITA-SRC\ditalog.ditamap"/>
<param name="output.file" value="C:\DITA-SRC\out\html-admin\toctree.js"/>
</antcall>
<copy todir="C:\DITA-SRC\out\html-admin">
<fileset dir="\${dita.dir}\demo\tocjs\basefiles"><include
```

```
name="**/*" /></fileset>
</target>

</project>
```

## Javascript-Navigation generieren

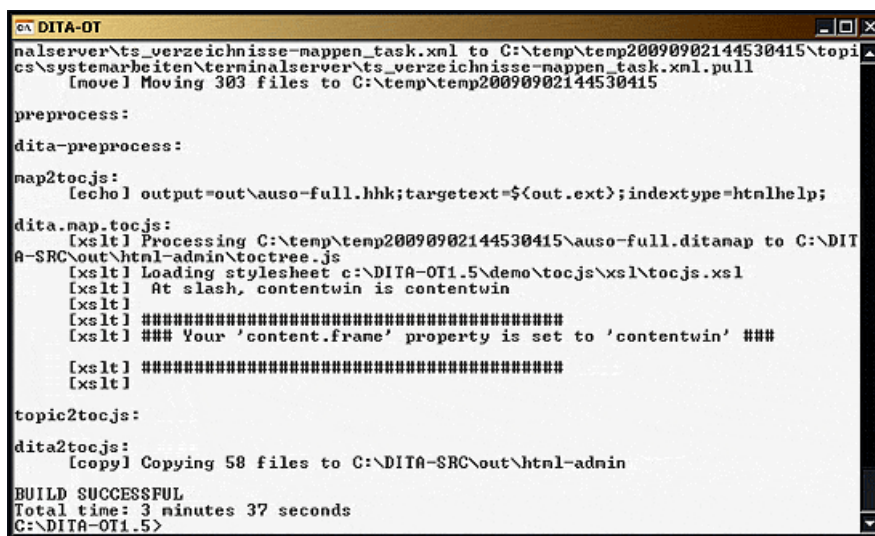
Anleitung: So generieren Sie HTML-Output mit Javascript-Navigation.

Das Plugin wurde ins DITA-OT integriert, die frameset.html und die build-Datei wurden angepaßt!

1. Doppelklicken Sie im Toolkit-Verzeichnis auf die Datei startcmd.bat.
2. Im Beispiel heißt die build-Datei admin-html.xml. Der Ant-Befehl mit dem tocjs-Target lautet dementsprechend:

```
ant -f C:\DITA-SRC\build\admin-html.xml sample2tocjs
```

3. Die Meldung über einen erfolgreichen Build kommt mit vielen Rauten daher. Nur diese Meldung gibt Ihnen Gewißheit, dass der Build geklappt hat.



```
DITA-OT
nalserver\ts_verzeichnisse-mappen_task.xml to C:\temp\temp20090902144530415\topi
cs\systemarbeiten\terminalserver\ts_verzeichnisse-mappen_task.xml.pull
[movel Moving 303 files to C:\temp\temp20090902144530415

preprocess:
dita-preprocess:
map2tocjs:
[echo] output=out\auso-full.hhk;targettext=${out.ext};indextype=htmlhelp;
dita.map.tocjs:
[xslt] Processing C:\temp\temp20090902144530415\auso-full.ditamap to C:\DIT
A-SRC\out\html-admin\toctree.js
[xslt] Loading stylesheet c:\DITA-OT1.5\demo\tocjs\xsl\tocjs.xsl
[xslt] At slash, contentwin is contentwin
[xslt] #####
[xslt] ### Your 'content.frame' property is set to 'contentwin' ###
[xslt] #####
[xslt]

topic2tocjs:
dita2tocjs:
[copy] Copying 58 files to C:\DITA-SRC\out\html-admin

BUILD SUCCESSFUL
Total time: 3 minutes 37 seconds
C:\DITA-OT1.5>
```

Abb. 4: Build succesful

4. Diese Dateien wurden im HTML-Output-Verzeichnis generiert. Öffnen Sie die frameset.html im Browser.

Name	Größe	Typ
css		Dateiordner
images		Dateiordner
img		Dateiordner
topics		Dateiordner
delta.gif	1 KB	GIF-Bild
deltaend.gif	1 KB	GIF-Bild
event.js	38 KB	JScript Script File
frameset.html	1 KB	HTML Document
index.html	28 KB	HTML Document
toctree.js	52 KB	JScript Script File
treeview.js	48 KB	JScript Script File
yahoo.js	3 KB	JScript Script File

Abb. 5: toctree.js innerhalb der toctree.html

5. Sollten im Navigationsbaum die deutschen Umlaute falsch dargestellt werden, muß die Datei C:\DITA-OT1.5\demo\tocjs\xsl\tocjs.xsl angepaßt werden. Bei Output Method muss stehen: <xsl:output method="text" encoding="ISO-8859-1" />.

# Ausgabeformat CHM (Windows Help)

Tipps und Tools bei der CHM-Generierung.

## Symbolschaltflächen hinzufügen

Es sollen neue Schaltflächen in der CHM-Hilfe hinzukommen.

### Standard-Schaltflächen

Eine Standard-Schaltflächenleiste im CHM-Standard-Output sieht folgendermaßen aus:



Abb. 6: Nur 4 Schaltflächen im CHM-Output

### Neue Schaltflächen hinzufügen

Es sollen einige Schaltflächen hinzukommen:



Abb. 7: Weitere Schaltflächen hinzufügen

Besonders wichtig erscheinen mir hier die Navigationsschaltflächen.

- **Vorherige** - Vorheriges Topic im Inhaltsverzeichnis. Im Beispiel also nach oben auf *Nachrichtentelegramm*.
- **Nächstes** - Nächstes Topic nach unten im Inhaltsverzeichnis.
- **Zurück** - Zum zuvor geöffneten Topic in der Historie.
- **Vorwärts** - Zum danach geöffneten Topic in der Historie.

Die beiden Schaltflächen, die das Blättern durch das Inhaltsverzeichnis ermöglichen, benötigen als Voraussetzung den Eintrag `Binary TOC=Yes`. Dieser Eintrag steht in der HTML-Workshop-Projekt-Datei `Dateiname.hhp`. Entweder Sie editieren diesen Eintrag direkt im Editor oder nutzen dafür die Oberfläche des HTML-Workshops.

Einen kleinen Nachteil hat jedoch der Einsatz der Inhaltsnavigation. Es sind nun keine individuellen oder erweiterten Icons für das Inhaltsverzeichnis mehr möglich. Es können nur noch die Standard-Icons **Buch** und **Seite** eingesetzt werden.

HTML-Workshop erlaubt den Einsatz von 2 individuellen Schaltflächen, die zur einer Inhaltsseite Ihrer Wahl verlinkt werden kann. Im Beispiel verlinkt die Schaltfläche **Was ist neu** zu einer Seite, die die Neuerungen in der Software als auch in der Online-Hilfe aufzeigt.

### Die Projektdatei .hhp

Es mussten einige Einträge in die Projektdatei geschrieben werden. Im Bereich [ OPTIONS ] kamen hinzu:

```
Binary TOC=Yes
Default Window=main
```

Zwischen OPTIONS und FILES wurde ein weiterer Bereich eingefügt: [WINDOWS]. Dem folgt eine lange Zeile. Wie diese Zeile zustande kommt, ist in vielen HTML-Workshop-Internetseiten gut dokumentiert. Ein guter Weg wäre es auch, die Schaltflächen in der Oberfläche des HTML-Workshops hinzuzufügen und das Ergebnis in der Projektdatei Dateiname.hhp anschließend zu betrachten und als Grundlage für das DITA-xsl-Stylesheet zu nehmen. Hier der WINDOWS-Bereich zum obigen Screenshot.

```
[ WINDOWS ]
main=, "Dokupedia.hhc", "Dokupedia.hhk", "topics\common\default.html",
"topics\common\home.html",
"topics\common\erste-individuelle-schaltflaeche.html",
"Beschriftung Individuelle Schaltfläche",,,,0x3520,,0x64204e,,,,,,0
```

Man beachte jedoch, dass nach dem Wort main alles in einer Zeile geschrieben sein muss!

### Das Stylesheet map2hhpImpl.xsl

Damit die Individualisierung der CHM-Datei automatisch bei jedem build aus dem DITA-OT generiert wird, müssen die Neuerungen der Projektdatei in das für die Projektdatei verantwortliche XSL-Stylesheet geschrieben werden. Dies ist die Datei DITA-OT\xsl\map2htmlhelp\map2hhpImpl.xsl im XSL-Ordner der Toolkit-Installation. An 2 Stellen erfolgt die Anpassung im Stylesheet. Diese sind jeweils grün gefärbt.

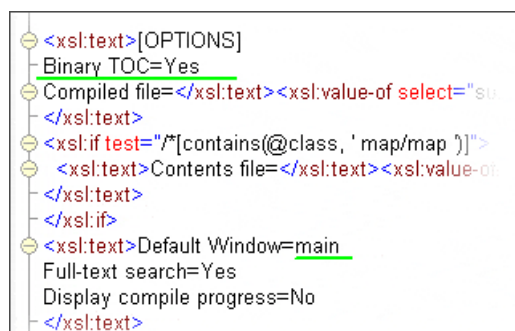


Abb. 8: Options-Einträge

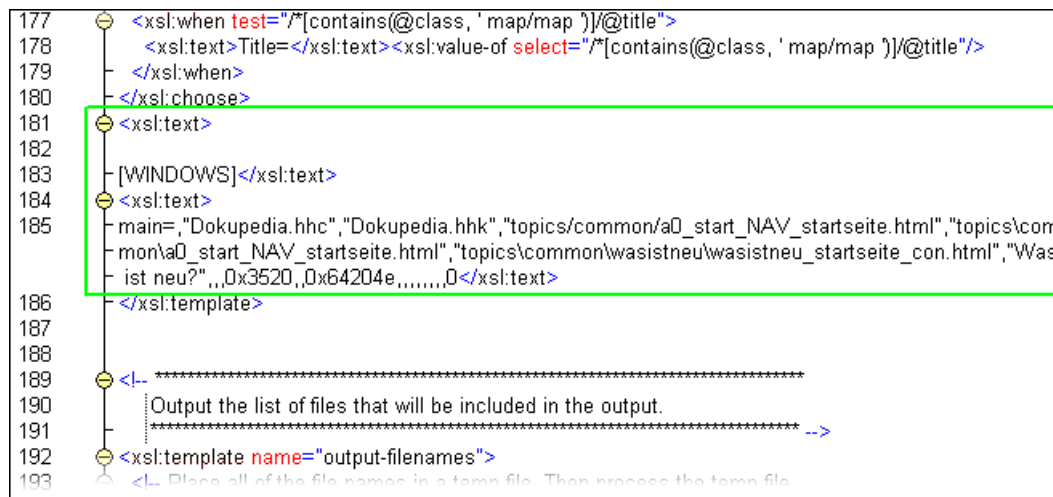


Abb. 9: Windows-Einträge

Im Beispiel der WINDOWS-Einträge ist die default-Datei eine Datei namens start\_NAV\_startseite.html. Diese Datei ist zugleich der Symbolschaltfläche **Home** zugeordnet. Es wurde eine individuelle Schaltfläche **Was ist neu?** hinzugefügt, die auf die Seite wasistneu\_startseite\_con.html verlinkt.

### Nachteil der Hartkodierung

Sobald man obige Windows-Einträge in die DITA-XSL-Datei hartkodiert, provoziert man bei anderen Projekten anderen Namens unweigerlich eine Fehlermeldung. Vielleicht wäre eine Plugin-Lösung besser?

## Links zum Artikel

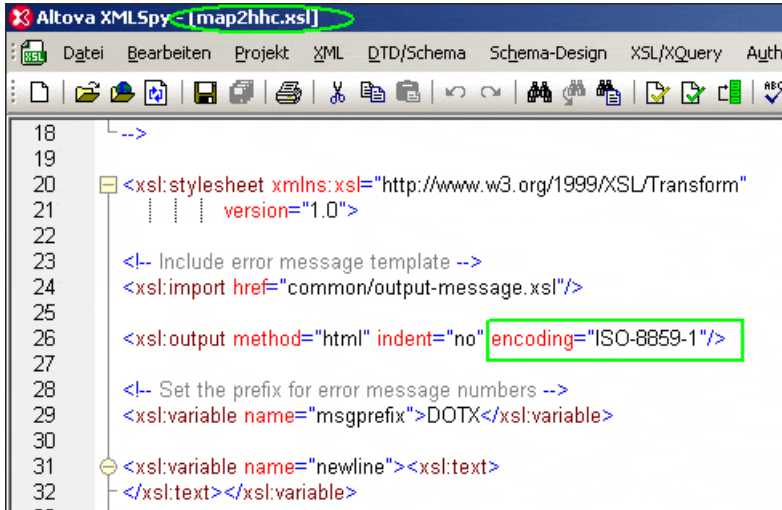
- [Inoffizielle HTML-Help-Spezifikation](#)

## Umlaute im Inhaltsverzeichnis

Anleitung: So stellen Sie deutsche Umlaute im Inhaltsverzeichnis der CHM-Datei richtig dar.

 **Hinweis:** Dies tritt insbesondere beim DITA-OT 1.5 auf!

1. Editieren Sie die Datei `C:\DITA-OT1.5\xsl\map2hhc.xsl` folgendermaßen:



```

18  <!--
19
20  <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
21  |         |         |         |
22  |         |         |         |         |
23  |         |         |         |         |         |
24  <xsl:import href="common/output-message.xsl"/>
25
26  <xsl:output method="html" indent="no" encoding="ISO-8859-1"/>
27
28  <!-- Set the prefix for error message numbers -->
29  <xsl:variable name="msgprefix">DOTX</xsl:variable>
30
31  <xsl:variable name="newline"><xsl:text>
32  </xsl:text></xsl:variable>

```

Abb. 10: encoding-Attribut hinzufügen

2. Fügen Sie wie im Bild angeben beim `xsl:output` das Attribut `encoding="ISO-8859-1"` hinzu.

# Impressum

---

Autor:	Andreas Petersell petersell@ditalog.com www.ditalog.com
Ausgabe:	Ausgabeformate HTML und CHM
Redaktionsschluß:	18.04.2011

# Index

## B

Binary TOC 8

## C

CHM-Datei 10

Inhaltsverzeichnis 10

CHM-Datei konfigurieren 8

## H

hhp-Datei 8

## M

map2hhc.xsl 10

map2hhp.xsl 9

## P

Plugin tocjs 4

Projektdatei .hhp 8

## S

Schaltflächen in CHM-Datei 8

## T

tocjs 4, 5

basefiles 4

Builddatei anpassen 5

frameset.html 4

tocnav.html 4

toctree.js 4

## U

Umlaute 10

Umlaute in HTML-Navigation 2

## V

Verzeichnisbaum als Navigation 4