bitmotion

Web-Projekte, die begeistern.

Websites erstellen mit TYPO3

Jessica Jagodzinski

Agenda

- TYPO3 Einführung
- Extensions
- Templates
- TypoScript
- Umsetzung einer HTML-Vorlage
- Vertiefung

Agenda

- TYPO3 Einführung
- Extensions
- Templates
- TypoScript
- Umsetzung einer HTML-Vorlage
- Vertiefung

Agenda

- TYPO3 Einführung
- Extensions
- Templates
- TypoScript
- Umsetzung einer HTML-Vorlage
- Vertiefung

Templates

- CMS: Design und Content getrennt
- Statischer und dynamischer Content
- Template: bestimmt Erscheinungsbild, konfiguriert Extensions

Demonstration

- Der Template-Record

Templates

Aufbau eines Templates

- Hierarchie
- Template-Erweiterungen
- Standard-Templates (z.B. „css_styled_content“)

Tools für das Template

- Der Constant Editor
- Info/Modify
- Object Browser
- Template Analyzer

Templates

Demonstration

- Erstellen einer neuen Website mit Standard-Templates

Übung

- Erstellen einer neuen Seite mit neuem Template

Templates

Übung

- Verändern Sie mit Hilfe des Constant Editors das Aussehen der Seite
- Beobachten Sie die Veränderung des „Template Records“

Templates

- Root und +EXT Template (rootlevel, clear constants, setup)
- Static Templates sind Templates, die das System mitliefert und können nicht geändert werden.
- Static Templates bieten z.T. "ready-to-run" Setups, teilweise nur rudimentäre, aber wichtige Setups.
- Constants und Setup: Constants sind ähnlich Konstanten in der Programmierung. Es können Werte definiert werden, die später im Setup angesprochen werden.

Templates

- "include basis-templates" bietet eine Möglicheit Typoscript-"Libraries" zu bilden und diese einzubinden.
- "template on next level" bindet ein Template auf allen Unterseiten ein, nützlich vor allem für Startseite/Unterseiten.
- Templates können auch von Extensions kommen.

Agenda

- TYPO3 Einführung
- Extensions
- Templates
- TypoScript
- Umsetzung einer HTML-Vorlage
- Vertiefung

TypoScript

- TypoScript ist eine Syntax, um Informationen in einer hierarchischen Struktur zu definieren
- TypoScript verwendet "Objektbäume". Der Aufbau eines "Objektbaumes" bestimmt, welche Elemente wie ausgegeben werden.
- TypoScript macht selbst nichts (keine Programmiersprache)
- TypoScript-Informationen werden in ein PHP-Array transformiert

TypoScript

Beispiele für die Syntax

- meineSeite = rot
- „Objekt“ = „Wert“

„Objektpfad“

- meineSeite.hintergrund = blau
- meineSeite.linkziel = _blank
- „Objektpfad“ = „Wert“

Vereinfachte Schreibweise

```
meineSeite {
    hintergrund = blau
    linkziel = _blank
}
```

TypoScript

- meineSeite {
  hintergrund = blau
  hintergrund.transparenz = 95%
  linkziel = _blank
  }

- „hintergrund“ und „linkziel“ = Eigenschaft von „meineSeite“

- „transparenz“ = Eigenschaft von „hintergrund“

TypoScript

Beispiel: Bestimmung der Länge und Breite eines Zimmers auf einer Etage:

- meinHaus.2.5.laenge = 5 Meter
- meinHaus.2.5.breite = 3 Meter

TypoScript

Das Zimmer muss ich aber zunächt erzeugen und in den Objektbaum platzieren

```
meinHaus    = HAUS
meinHaus.0 = KELLER
meinHaus.1 = ETAGE
meinHaus.2 = ETAGE
meinHaus.3 = DACHBODEN
...
meinHaus.2.1   = ZIMMER (1.Zimmer auf der 2. Etage)
...
meinHaus2.5    = ZIMMER
...
meinHaus.2.5.laenge = 2 Meter
meinHaus.2.5.breite   = 3 Meter
```

TypoScript

Objektklassen

- HAUS
- KELLER
- ETAGE
- DACHBODEN
- ZIMMER

- „meinHaus“ ist ein selbstgewählter Name für das Wurzelobjekt
- „laenge“ und „breite“ sind konkrete Eigenschaften (Properties) eines Objektes vom Typ ZIMMER, die ich mit Werten belegen kann.

TypoScript

- 0 bis x sind durchnummerierte Platzhalter (Array), in denen ich die Unterobjekte erzeuge.
- Der Programmierer der Objektklasse bestimmt, ob er solche Platzhalter für Unterobjekte zur Verfügung stellt oder nur Properties für Eigenschaften.
- Welche Properties es genau gibt steht jeweils in der Referenzdokumentation.

TypoScript

Was passiert in folgendem Beispiel? (TSref)

```
#Default PAGE obejct:
page = PAGE
page.typeNum = 0
page.10 = TEXT
page.10.value = HELLO WORLD!
```

TypoScript

Demonstration

- Erstellung einer Website ohne Standard-Template
- „Hello World“

Übung

- Erstellen eines neuen Objekts page.20
- Verändern Sie die Reihenfolge der Objekte page.10 und page.20
- Was passiert?

TypoScript

Content einbinden

- Mit dem CONTENT können Objekte die dynamischen Seiteninhalte gebunden werden
- Die Konfiguration der einzelnen Inhaltselemente ist sehr komplex, dafür wird die i.a.R. ein Standard-Template eingebunden
- styles.content.get aus css_styled_content

TypoScript

Demonstration

- css_styled_content
- Content einbinden

Übung

- Binden Sie Ihren Content ein
- Lassen Sie sich den Content aus einer anderen Spalte (z.B. „Left“) ausgeben

TypoScript

Menüs erstellen

- HMENU ist das „Oberobjekt“ (TLO) für alle hierarchischen Menüs
- Die Eigenschaften 1, 2, ... des HMENU Objektes stehen für die einzelnen Ebenen (keine beliebige Aufzählung)
- Die Ebene 1 muss vorhanden sein!
- Auf jeder Ebene 1, 2, ... wird eines der Objekte TMENU, GMENU, JSMENU, ... Eingesetzt, welches die eigentliche Ausgabe bestimmt
- Der „Normal-State“ NO = 1 muss eingesetzt sein!

TypoScript

Demonstration

- Erstellen eines einfaches Menüs für die erste und zweite Ebene

Übung

- Erstellen Sie ein einfaches Menü

TypoScript

Übung

- Lassen Sie alle Unterseiten immer anzeigen (Tipp: TSref TMENU)
- Umschließen Sie das TMENU mit einem UL-Tag
- Umschließen Sie jedes Element mit einem LI-TAG
- Kennzeichnen Sie das gerade aktive Menuelement mit der Klasse current
- Kennzeichnen Sie die übergeordneten Elemente des Aktive mit der Klasse active
- Fügen Sie als A-Tagtitle die Beschreibung der Seite hinzu. Falls die Beschreibung fehlt soll der Navigationstitel genommen werden. Falls dieser nicht existiert der Seitentitel.

TypoScript

Lösung

```
tmp.menu = HMENU
tmp.menu.1 = TMENU
tmp.menu.1.wrap = <ul> | </ul>
tmp.menu.1.expAll = 1
tmp.menu.1.NO = 1
tmp.menu.1.NO.ATagTitle.field = description // nav_title //
title
   tmp.menu.1.NO.wrapItemAndSub = <li>|</li>
   tmp.menu.1.ACT.wrapItemAndSub = <li
style="color:red">|</li>
   tmp.menu.1.CUR < page.5.1.NO
   tmp.menu.1.CUR.wrapItemAndSub = <li
style="color:green">|</li>
   tmp.menu.2 < page.5.1
```

TypoScript

Objekte kopieren

- In der Praxis “zerteilt” man TypoScript oft in kleine Happen und führt diese wieder zusammen
- Dies hat den Vorteil, dass ein besserer Überblick über die einzelnen Objekte gegeben ist

```
lib.menu = HMENU
lib.menu.1 = TMENU
lib.menu.1.NO = 1

page = PAGE
page.10 < lib.menu
```

Agenda

- TYPO3 Einführung
- Extensions
- Templates
- TypoScript

- Umsetzung einer HTML-Vorlage
- Vertiefung

HTML-Vorlagen einbinden

Umsetzung eines Designs mit einer HTML-Vorlage ist “Best Practice” Idee:

- Design wird zugeliefert
- Einbinden in TYPO3
- Festlegen
  - Was ist dynamisch und was statisch?
  - Wo sind welche Elemente zu platzieren?

HTML-Vorlagen einbinden

Prinzip

- HMTL-Template und CSS-File auf Server ablegen
- TypoScript-Template „mitteilen“ HTML-Template zu benutzen

Wichtige TypoScript-Objekte

- TEMPLATE mit den Eigenschaften
  - „workOnSubparts“ und
  - „subparts“

HTML-Template muss mit „Markern“ vorbereitet werden

HTML-Vorlagen einbinden

```
 1 <html>
 2 <head>
 3 <!-- ###DOCUMENT_HEADER### begin-->
 4 <link rel="stylesheet" type="text/css" href="fileadmin/template/formate.css">
 5 <style type="text/css">
 6 <!--
 7 -->
 8 </style>
 9 <!-- ###DOCUMENT_HEADER### end-->
10 </head>
11
12
13 <body>
14 <!-- ###DOCUMENT_BODY### start-->
15 <div id="boxueberall">
16   <div id="banner">Banner</div>
17   <div id="boxmenuoben"><!-- ###menu_3### --> <!-- ###menu_3### --></div>
18   <div id="boxmenulinks"><!-- ###menu_1### --> <!-- ###menu_1### --></div>
19   <div id="boxcontent"><!-- ###contentzelle### --> <!-- ###contentzelle### --></div>
20   <div id="newsticker"></div>
21 </div>
22 <!-- ###DOCUMENT_BODY### stop-->
23 </body>
24
25 </html>
```

HTML-Vorlagen einbinden

Vorgehen

- Im HTML-Quelltext identifizieren, wo die dynamischen Stellen sind
- Diese mit Markern <!-- ###MARKE ### --> umfassen
- Weiterhin muss TYPO3 den HEADER und BODY in der HTML-Vorlage erkennen können
- Daher muss dieser auch markiert werden
  - <!-- ###DOCUMENT_HEADER### -->
  - <!-- ###DOCUMENT_BODY### -->

HTML-Vorlagen einbinden

```
157 ##########
158 # BODY
159 ##########
160 temp.mainTemplate = TEMPLATE
161 temp.mainTemplate {
162   template = FILE
163   template.file = fileadmin/template/mytemplate.html
164   workOnSubpart = DOCUMENT_BODY
165   subparts.menu_1 < temp.menu_1
166   subparts.menu_3 < temp.menu_3
167   subparts.contentzelle < styles.content.get
168 }
```

HTML-Vorlagen einbinden

- Im TypoScript kann man nur die HTML-Vorlage laden

```
#########
# BODY
#########
temp.mainTemplate = TEMPLATE
temp.mainTemplate {
  template = FILE
  template.file = fileadmin/schulung/index.html
```

- Und TYPO3 mitteilen, ob der HEADER oder der BODY verändert werden soll

```
workOnSubpart = DOCUMENT_BODY
```

HTML-Vorlagen einbinden

- Einzelne Marken in der HTML-Datei werden dann wie folgt angesprochen

```
subparts.HAUPTMENU < lib.hauptmenu
subparts.MENULINKS < lib.menulinks
subparts.CONTENTLINKS < styles.content.getLeft
subparts.CONTENTLINKS.slide = -1
subparts.CONTENTMAIN < styles.content.get
```

HTML-Vorlagen einbinden

Kochbuch Dynamisierung 1/3

- In der HTML-Vorlage und in der CSS-Datei müssen eventuell Pfade zu Grafiken angepasst werden
- In der HTML-Datei nach dem <HEAD> und vor dem </HEAD> jeweils eine Marke <!-- ###DOCUMENT_HEADER### --> setzen. Weiterhin nach dem <BODY> und vor dem </BODY> eine Marke <!-- ###DOCUMENT_BODY### --> setzen
- In der HTML-Datei an den Stellen, die von TypoScript erzeugt werden, jeweils eine umschließende Marke setzen (der Teil zwischen den Marken wird durch das generierte HTML von TYPO3 ersetzt)
- Die Dateien auf dem Server unter fileadmin/templates ablegen

HTML-Vorlagen einbinden

Kochbuch Dynamisierung 2/3

- TypoScript Template auf der obersten Seite anlegen
- Das TypoScript für das Auslesen des HTML-Headers eingeben

```
#########
# HEAD
#########
temp.headTemplate = TEMPLATE
temp.headTemplate {
  template = FILE
  template.file = fileadmin/schulung/index.html
  workOnSubpart = DOCUMENT_HEADER
}
```

HTML-Vorlagen einbinden

Kochbuch Dynamisierung 3/3

- Das TypoScript für das Auslesen des HTML-Bodys eingeben

```
#########
# BODY
#########
temp.mainTemplate = TEMPLATE
temp.mainTemplate {
  template = FILE
  template.file = fileadmin/schulung/index.html
  workOnSubpart = DOCUMENT_BODY
  subparts.HAUPTMENU < lib.hauptmenu
  subparts.CONTENTMAIN < styles.content.get
}
```

- Das TypoScript für das Menü erzeugen
- PAGE-Objekt anlegen

HTML-Vorlagen einbinden

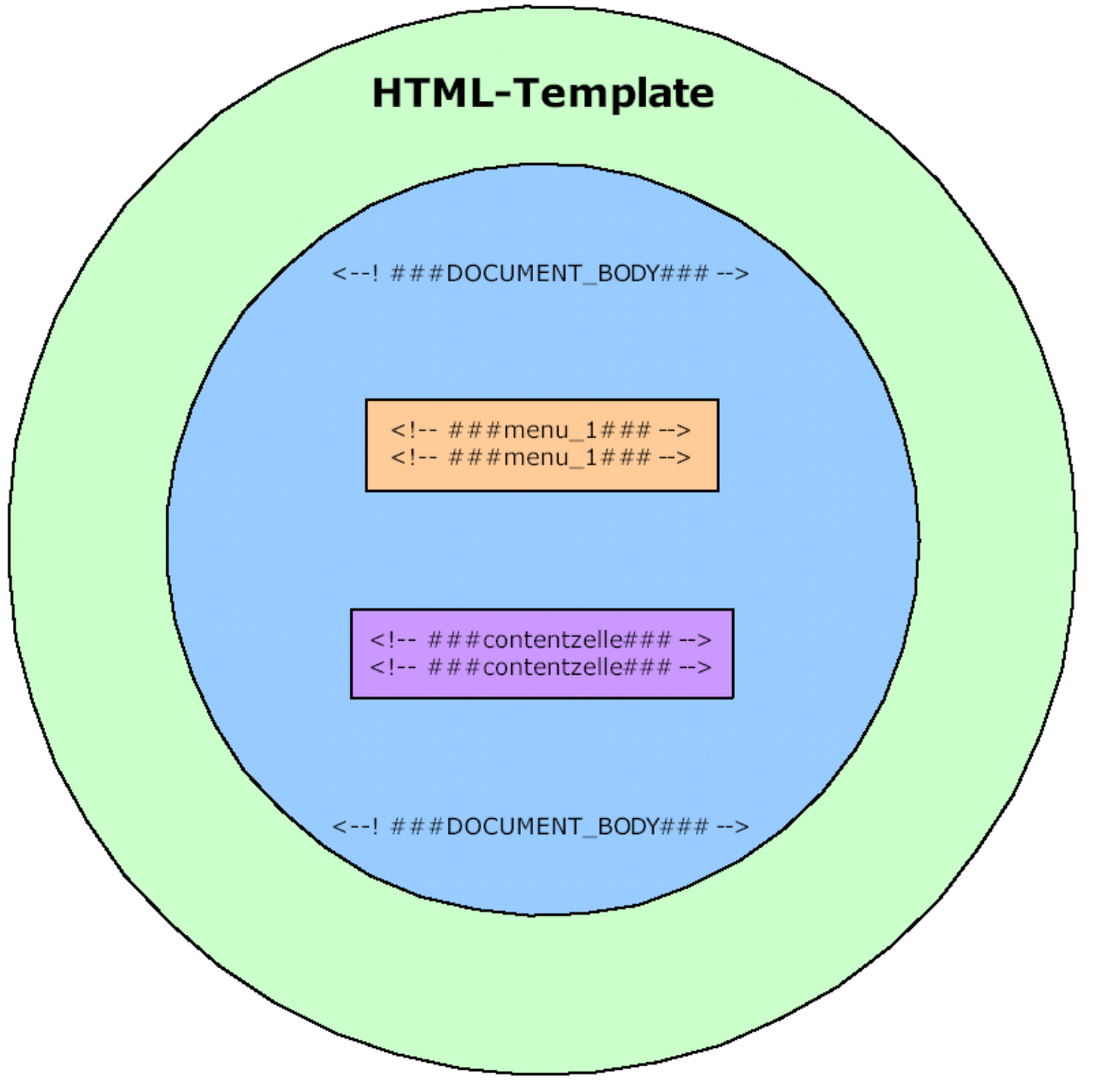
Demonstration

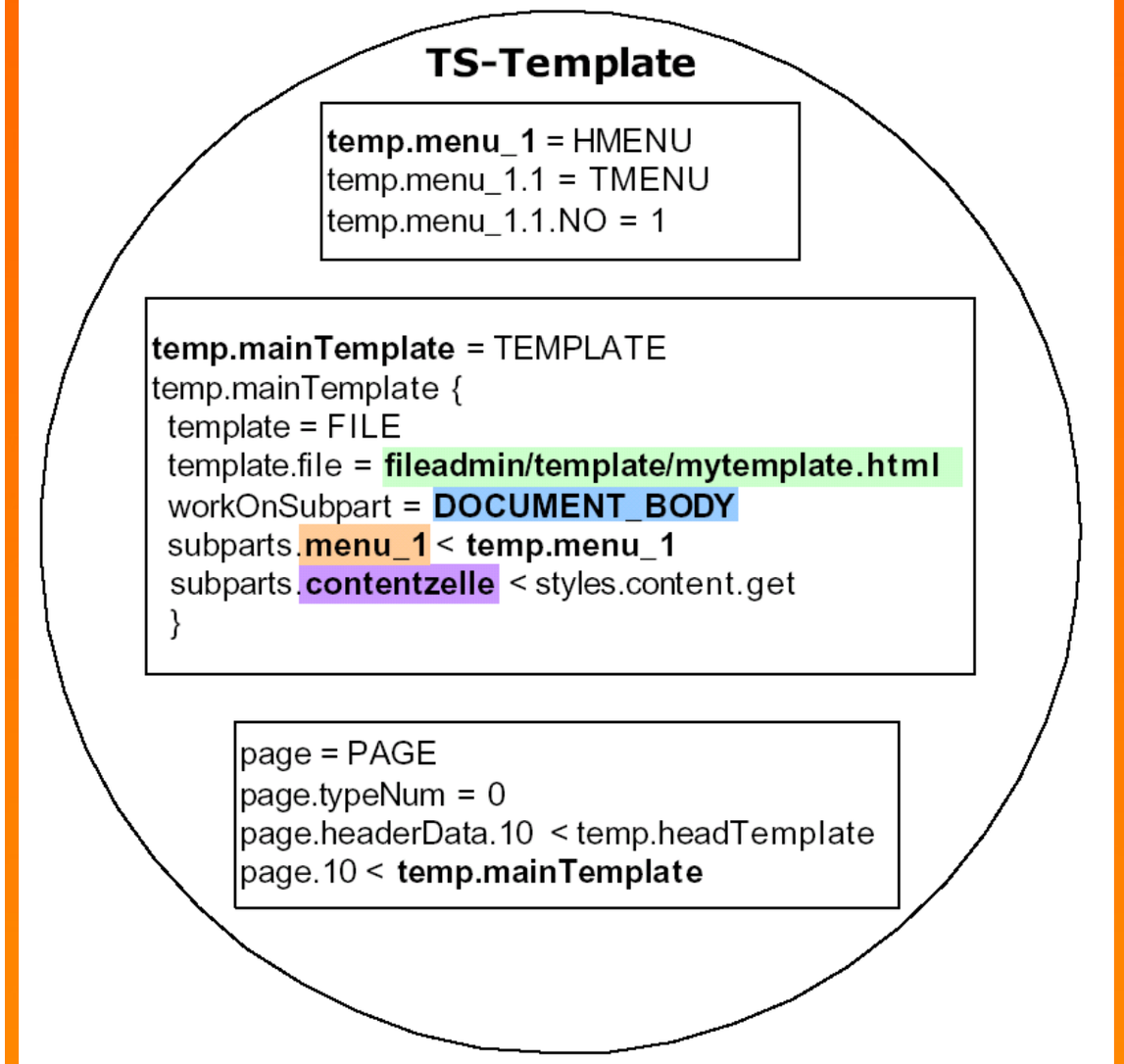
- HTML-Vorlage mit Markern versehen und Menü so wie Content durch TypoScript generieren lassen

Übung

- Binden Sie das HTML-Template ein

HTML-Vorlagen einbinden





Agenda

- TYPO3 Einführung
- Extensions
- Templates
- TypoScript
- Umsetzung einer HTML-Vorlage
- Vertiefung

Weiterführende Themen

Template und Template-Erweiterungen

Demonstration

- Template modularisieren
- Achtung: Reihenfolge der Abarbeitung beachten!

Übung

- Modularisieren Sie Ihr Template

Weiterführende Themen

Type-Konzept

- TSRef
  typeNum
  This decides the typeId of the page
  NOTE: This value MUST be set and be unique!
  - typeNum wird hauptsächlich für verschiedene Ansichten verwendet (Druckansicht, RSS-Feed, PDF-Aufgabe)
  - Aufruf mit „&type=1“ an der URL

Übung

- Erstellen Sie ein neues PAGE-Objekt „alt_page“ mit typeNum=1 und sprechen Sie dieses an

Weiterführende Themen

Condition

- Mit Conditions kann man das Erscheinungsbild in Abhängigkeit von „Zuständen“ definieren

Beispiel

```
...
page.25 = TEXT
...
[dayofweek = 0]
page.25.value = Sonntag
[else] ...
page.25.value = Anderer Tag
```

Weiterführende Themen

Operatoren innerhalb einer Condition

- =
- <
- >

Conditions verknüpfen

- AND: [browser = msie] && [system = win]
- OR: [browser = opera] II [browser = netscape]
- Höhere Priorität hat „AND“

Demonstration

- Conditions: Browser ausgeben

Weiterführende Themen

Funktion: stdWrap

- „Schweizer Messer“
- Kann überall da eingesetzt werden, wo Objekteigenschaften vom Datentyp „stdWrap“ vorhanden sind
- **Beispiel**:
  - page.10 = TEXT
  - page.10.field = title

- Teilt sich in drei große Bereiche
  - Get Data: Auslesen von Daten
  - Override/Conditions: Überschreiben und Vergleichen von Werten
  - Parse Data: Eigenschaften zur Verarbeitung von Daten

Weiterführende Themen

Get Data

- Ließt den Wert eines Feldes aus

```
1 page.10 = TEXT
2 page.10.filelist = fileadmin/template
```

Override

- Überschreibt Werte

```
1 page.10 = TEXT
2 page.10.filelist = fileadmin/template
3 page.10.listNum = 1
```

Parse Data

- Verändert Werte

```
1 page.10 = TEXT
2 page.10.filelist = fileadmin/template
3 page.10.listNum = 1
4 page.10.case = upper
```

- Vorhandene Felder anzeigen mit .debugData =1!
- Übung: Lassen Sie den Seitentitel in dem Layer links unten anzeigen

Weiterführende Themen

Option Split

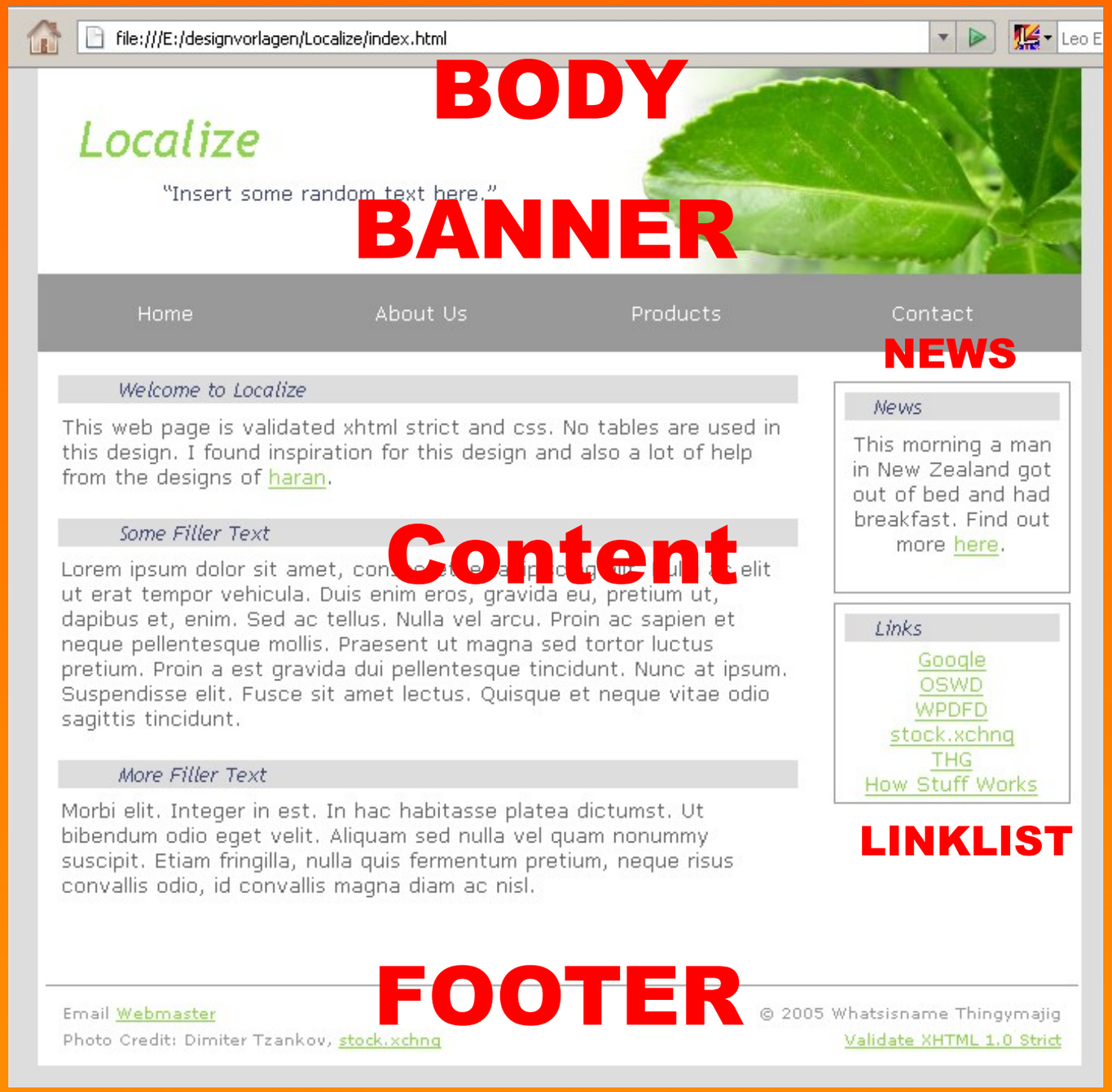
- Steht in TS-Eigenschaften zur Verfügung die mit „optionSplit“ Datentypen gekennzeichnet sind
- Häufig verwendet, um Menüs zu gestalten
- Erlaubt es, mehreren Elementen Eigenschaften zuzuweisen
- **Grundregel**
  - I*I Unterteilt Werte in Erster, Mittlerer und Letzter
  - II Unterteilt Erster, Mittlerer, Letzter in Unterbereiche
- **Beispiel**
  - temp.menu_1.1.NO.before = <||>|*|-|*|<||>
  - Weitere Beispiele: s. TSRef

Demonstration

- optionSplit

Fragen und Antworten

Bereiche Festlegen



Bereiche durch Marker kennzeichnen

Marker in der HTML-Vorlage setzten

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>…Localize</title>
<link rel="stylesheet" type="text/css" href="screenstyle.css" />
</head>
<body>
<!-- ###BODY### subpart begin -->
<div class="container">
<div class="titleblock"><h1>Localize</h1><p>&ldquo;Insert some random text here.&rdquo;</p></div>
        <div>
        <!-- ###MENU_TOP### begin -->
          <ul class="navbar">
             <li><a href="#" class="nav">Home</a></li>
             <li><a href="#" class="nav">About Us</a></li>
             <li><a href="#" class="nav">Products</a></li>
             <li><a href="#" class="nav">Contact</a></li>
          </ul>
        <!-- ###MENU_TOP### end -->
        </div>
<div class="rightcontainer">
<div class="rightbox"><h2>News</h2><p>This morning a man … <a href="#">here</a>.</p></div>
  <div class="rightbox linkbox"><h2>Links</h2>
               <a href="http://www.google.com/" title="Google">Google</a>
<!-- ###BODY### subpart end -->
</body>…
</html>
```

Bereiche durch Marker kennzeichnen

Marker in der HTML-Vorlage setzten

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
…
<body>
<!-- ###BODY### subpart begin -->
<div class="titleblock"><h1>Localize</h1><p>&ldquo;Insert some random text
here.&rdquo;</p></div>
        <div>
        <!-- ###MENU_TOP### begin -->
          <ul class="navbar"> …</ul>
        <!-- ###MENU_TOP### end -->
        </div>
<div class="rightcontainer"> …
<div class="content">
<!-- ###CONTENT### begin -->
                <h2>Welcome to Localize</h2>
                <p>This web page is validated xhtml …
                <h2>Some Filler Text</h2>
                <p>Lorem ipsum dolor sit amet, consectetuer…
                <h2>More Filler Text</h2>
                <p>Morbi elit. Integer in…</p>
<!-- ###CONTENT### end -->
</div>
<div class="footer">
<!-- ###DOCUMENT_BODY### subpart end -->
</body>…
</html>
```

Marker in der HTML-Vorlage setzten

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>…Localize</title>
<link rel="stylesheet" type="text/css" href="screenstyle.css" />
</head>
<body>
<!-- ###DOCUMENT_BODY### subpart begin -->
<div class="container">
<div class="titleblock"><h1>Localize</h1><p>&ldquo;Insert some random text
quo;</p></div>
        <div>
        <!-- ###MENU_TOP### begin -->
          <ul class="navbar">
             <li><a href="#" class="nav">Home</a></li>
             <li><a href="#" class="nav">About Us</a></li>
             <li><a href="#" class="nav">Products</a></li>
             <li><a href="#" class="nav">Contact</a></li>
           </ul>
        <!-- ###MENU_TOP### end -->
        </div>
<div class="rightcontainer">
<div class="rightbox"><h2>News</h2><p>This morning a man …
"#">here</a>.</p></div>
  <div class="rightbox linkbox"><h2>Links</h2>
              <a href="http://www.google.com/" title="Google">Google</a>
<!-- ###DOCUMENT_BODY### subpart end -->
</body>…
</html>
```

```
temp.mainTemplate = TEMPLATE
temp.mainTemplate {
  template = FILE
template.file=fileadmin/template.html
workOnSubpart=DOCUMENT_BODY
```

Marker in der HTML-Vorlage setzten

```
<div class"container">
<div class="titleblock"><h1>Localize</h1><p>&ldquo;Insert some random text quo;</p></div>
        <div>
        <!-- ###MENU_TOP### begin -->
         <ul class="navbar">
            <li><a href="#" class="nav">Home</a></li>
            <li><a href="#" class="nav">About Us</a></li>
            <li><a href="#" class="nav">Products</a></li>
            <li><a href="#" class="nav">Contact</a></li>
         </ul>
        <!-- ###MENU_TOP### end -->
        </div>
<div class="rightcontainer">
<div class="rightbox"><h2>News</h2><p>This morning a man … "#">here</a>.</p></div>
  <div class="rightbox linkbox"><h2>Links</h2>
                <a href="http://www.google.com/" title="Google">Google</a>
```

```
temp.mainTemplate = TEMPLATE
temp.mainTemplate {
  template = FILE
template.file=fileadmin/template.html
workOnSubpart=DOCUMENT_BODY
subparts.MENU_TOP < temp.menu1
…
```

Marker in der HTML-Vorlage setzten

```
<!-- ###MENU_TOP### begin -->
  <ul class="navbar">
    <li><a href="#" class="nav">Home</a></li>
    <li><a href="#" class="nav">About Us</a></li>
    <li><a href="#" class="nav">Products</a></li>
    <li><a href="#" class="nav">Contact</a></li>
  </ul>
<!-- ###MENU_TOP### end -->

<div class"container">
<div class="titleblock"><h1>Localize</h1><p>&ldquo;Insert some random text quo;</p></div>
        <div>
          <ul class="navbar">
            <li><a href="#" class="nav">Willkommen</a></li>
            <li><a href="#" class="nav">Unternehmen</a></li>
            <li><a href="#" class="nav">Support</a></li>
          </ul>
        </div>
<div class="rightcontainer">
<div class="rightbox"><h2>News</h2><p>This morning a man … "#">here</a>.</p></div>
  <div class="rightbox linkbox"><h2>Links</h2>
            <a href="http://www.google.com/" title="Google">Google</a>
```

```
temp.mainTemplate = TEMPLATE
temp.mainTemplate {
  template = FILE
template.file=fileadmin/template.html
workOnSubpart=DOCUMENT_BODY
```

generiertes HTML

Ausgabe HTML zusammenstellen

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
      <link rel="stylesheet" type="text/css" href="typo3temp/stylesheet_e99aac0be0.css" />
      <title>Localize</title>
      <link rel="stylesheet" type="text/css" href="fileadmin/vorlagen/screenstyle.css" />
      <title>Willkommen</title>
      <meta name="generator" content="TYPO3 4.0 CMS" />
      <script type="text/javascript" src="typo3temp/javascript_757c080409.js"></script>
</head>
<body>
<div class="container">
<div class="titleblock"><h1>Localize</h1><p>“Insert some random text here.”</p></div>
<div>
   <ul class="navbar">
     <li><a href="index.php?id=4" onfocus="blurLink(this);" class="nav">Willkommen</a></li>
     <li><a href="index.php?id=3" onfocus="blurLink(this);" class="nav">Unternehmen</a></li>
     <li><a href="index.php?id=2" onfocus="blurLink(this);" class="nav">Support</a></li>
    </ul>
</div>
<div class="rightcontainer">
       <div class="rightbox"><h2>News</h2><p>This morning a man … <a href="#">here</a>.</p></div>
<div class="rightbox linkbox"><h2>Links</h2>

<div class="content"> …
INHALT VON styles.content.get wird analog generiert und eingefügt
…
</body>
</html>
```

Bisher genredertes Ausgab- HTML

Das neu generierte Menü wird hier eingefügt