mmi consult gmbh

we make IT simple

Manfred Meise

IBM Certified Advanced Developer - Lotus Notes and Domino R3 – R8.5
IBM Certified Advanced Administrator - Lotus Notes and Domino R3 – R8, R8.5
IBM Certified Advanced Instructor – Lotus Notes and Domino R3- R8, R8.5

Certified for IBM Lotus software

Track 1 Session 4: 11:00 – 12:30 Raum ???

IBM Business Partner

zu meiner Person: Manfred Meise

- Studium Elektrotechnik (Dipl. Ing. (FH))
- Arbeit als Softwareingenieur seit mehr als 30 Jahren bei verschiedenen Computerherstellern und Softwarehäusern
- Gründer und Geschäftsführer der mmi consult gmbh
- Erfahrungen mit Lotus Notes/Domino seit 1992 - Markteinführung in Europa
  (als Leiter Strategische Projekte bei Lotus Development Deutschland)
- IBM Zertifizierungen als Anwendungsentwickler, Systemadministrator, Trainer für die Produktversionen R3 bis R8.5
- Tätigkeitsschwerpunkte im Entwicklungsbereich:
  CRM, Workflow, Objektorientierte Anwendungsarchitekturen, XPages
- Tätigkeitsschwerpunkte als Systemadministrator:
  Domänenzusammenführungen und –trennungen, Betriebshandbücher und Administrationsstandards, Versionswechsel, Infrastruktur-Audits, Client-Rollouts
- Erreichbar unter:
  - ***manfred.meise@mmi-consult.de***
  - ***http://www.mmi-consult.de***
  - ***http://www.mmi-consult.de/faq***

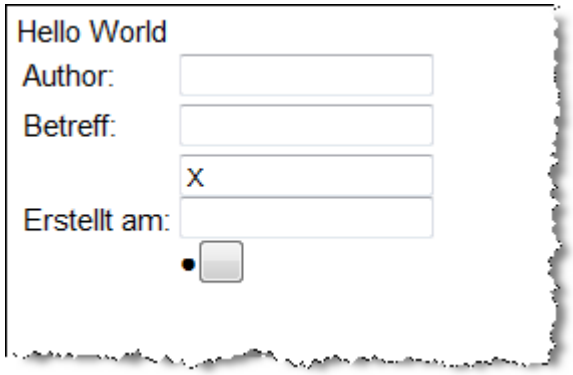
Meine Themen heute …

- XPages: Schnelle Programmierung – wo bleibt das UI?
- Einführung in XPages Styling
- Strukur des IBM oneUI Layout-Frameworks
- Verwendung von Motiven
- Anpassung einiger oneUI Elemente nach eigenen Vorstellungen
- Resumee und Ausblick

XPages:
Schnelle Programmierung – wo bleibt das UI?

XPage „Programmierung“

- Controls auf einer Seite zusammenstellen
- SSJS (und ggf. CSJS) berechnen Werte und steuern Funktionalität
- „Klick-Programming“
- Kann weiter verfeinert werden

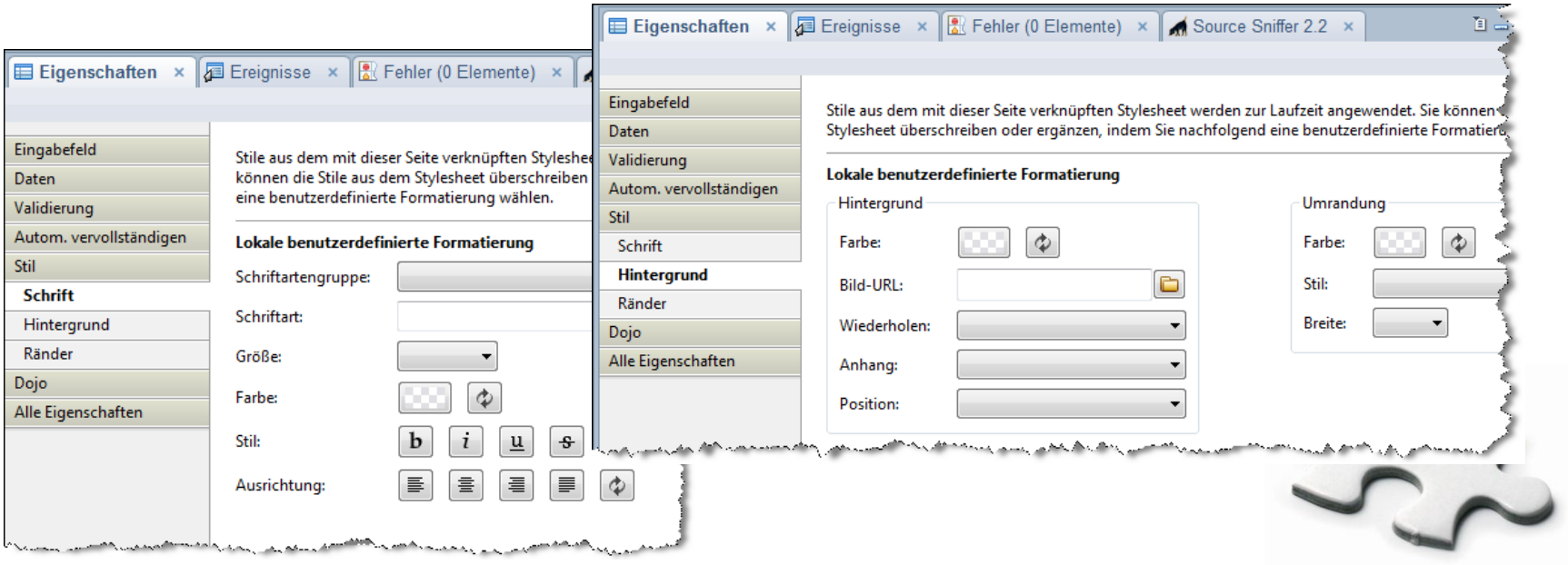


Ergebnis ist mager

- Nicht zu vergleichen, mit anderen Designs XPage basierter Seiten
- Beim Blick in Beispieldatenbanken (z.B. Diskussion) ist dort nicht einmal Styling angesetzt
- Mein Wunsch: ich will …
  - mich auf die Funktionalität und Datenstrukturen konzentrieren (und ansprechende Gestaltung möglichst erben)
  - bestehende Datenbanken mit geringem Aufwand in XPages Design umgestalten, um Browserfähigkeit zu integrieren
  - XPages Controls (z.B. von OpenNTF oder anderen) einfach in meine Anwendungen integrieren

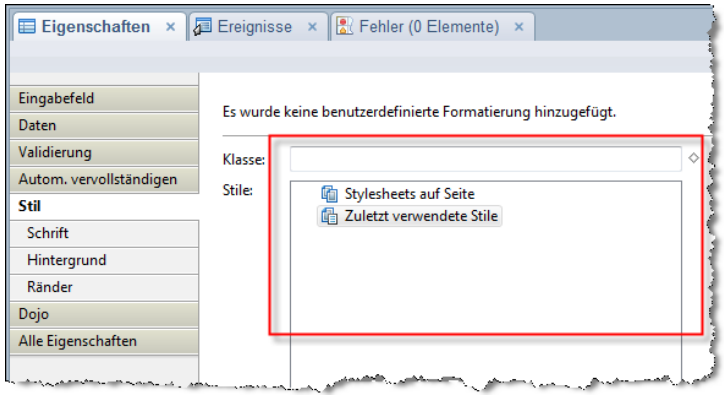
Object-Styling

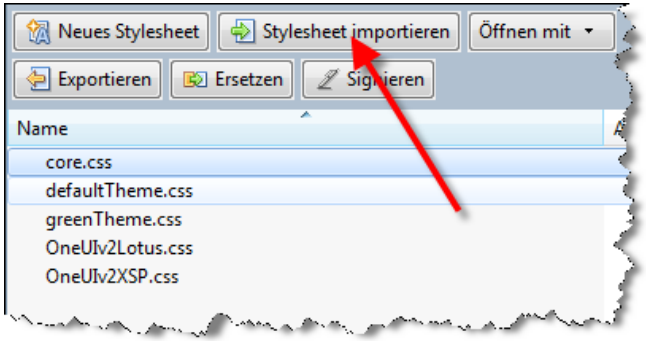
- Einzelne Objekte haben Stil-Attribute
  - … doch halt. Soll ich das für die ganze Anwendung konsequent durchziehen?
  - NEIN: Würde eine Menge Disziplin (besonders im Team), und Dokumentation erfordern. Wehe die Gestaltungswünsche ändern sich auf „halber Strecke“

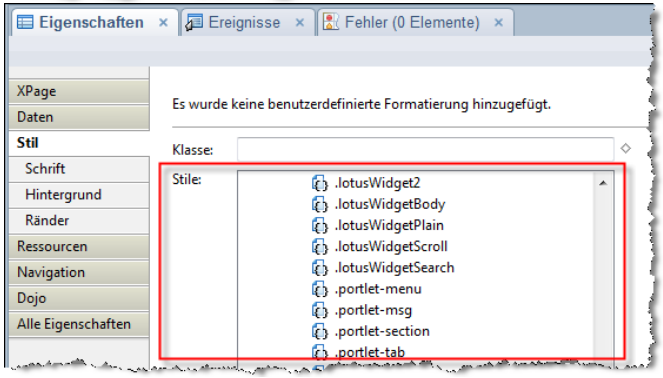


CSS StyleClasses einsetzen

- Da kann man doch auch Styles (CSS – Classes) einsetzen
- Warum kann ich nichts auswählen? Muss ich stets aufmerksam tippen, um keine Fehler zu machen?
- Würde ich StyleSheets (lediglich mit Class-Definitionen) einsetzen, wäre das schon einfacher
- Doch es ändert sich immer noch nichts







StyleClasses ausdefinieren

- Ach ja, man muss die StyleClasses ja auch ausformulieren

```
/*table-based form - use the div one moving forward*/
.lotusForm{margin:0;padding:0;zoom:1;background-color:#fafafa}
.lotusFormPlain, .lotusFormPlain .lotusTable, .lotusFormPlain .lotusFormTable td{background-color
page and not inline*/
.lotusForm h2 {margin:25px 15px 15px 25px;font-size:1.3em;}
.lotusForm h2 .lotusMeta{font-size:.8em;font-weight:normal;display:block}
.lotusForm tr.lotusFormFieldRow td, .lotusForm td.lotusFormFieldRow{padding-bottom:10px}
.lotusTable .lotusForm td, .lotusFormTable td {border-width:0 !important;padding-right:3px;vertica
.lotusForm td.lotusFormLabel {vertical-align:top;text-align:right;padding-right:10px;font-weight
.lotusForm td label{margin:0;padding:0;font-weight:bold;line-height:inherit;color:#222}
.lotusForm fieldset{margin:0;padding:0;border-width:0}/*this allows us to use fieldsets for non-s
.lotusForm .lotusFieldset{margin-left:0;padding:5px;border:1px solid #aaa}/*assign this to a fi
```

- Das Ergebnis sieht schon eher nach meinen Vorstellungen aus

- Doch darf ich nicht vergessen, diese StyleSheets und StyleClasses auf jeder XPage einzusetzen!

So kann es nicht weitergehen!

- Der anfängliche Spaß schneller XPage Gestaltung ist verflogen!
- Alle wollen mir klar machen, wie einfach das ist. Doch ich kann das nicht nachvollziehen: Frust!
- Wer stellt mir die ganzen StyleSheets und Grafiken für eine ansprechende Gestaltung zur Verfügung? Ich bin Programmierer und kein Designer!
- Irgendwie muss ich doch auch ein Design hinbekommen, wir es Beispiele von IBM oder OpenNTF zeigen – da ist ja auch nicht auf jeder Seite Styling Information enthalten.

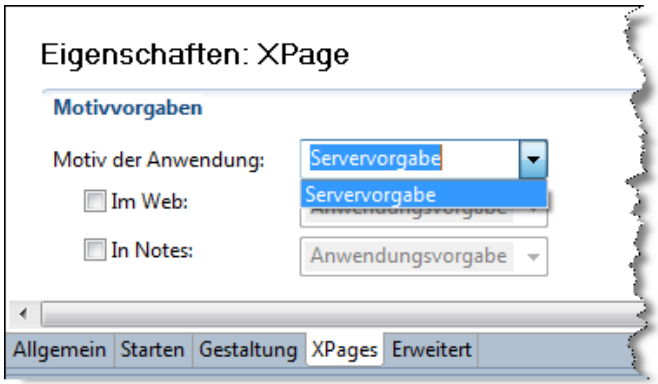
- AHHHH Layout Frameworks!!!!

Was bringt mir ein CSS Framework?

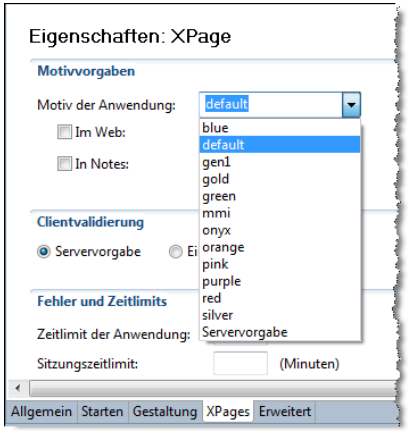
- Die Vorteile liegen ganze klar auf der Hand:
  - Es spart einem viel Zeit und vor allem Nerven, da man sich für die meisten Probleme keine Lösungen und nervige IE-Workarounds mehr suchen muss. Das Framework übernimmt diesen Job. Man muss nur noch wissen damit umzugehen

Motive, was ist das?

- Habe das was von Motiven gesehen, doch kann man hier keine auswählen?!



- Wie bekommt man dort etwas hinein? Ein Blick in Beispielanwendungen bringt neue Ideen.



- Baue ich ´mal ein. Mal sehen was passiert…

Einführung in XPages Styling

CSS Basics

- Cascading StyleSheets
  - Trennen Inhalt von Gestaltung einer Seite
  - Eigenständige Auszeichnungssprache, um die Darstellung eines Elementes zu bestimmen
  - Oft verwendet in HTML und XML Seiten

Regelwerk:

Eine CSS-Regel hat folgendes Aussehen:

```
Selektor [, Selektor2, …]
   {
      Eigenschaft-A: Wert-A;
      Eigenschaft-B: Wert-B
   }
/* Kommentar */
```

Beispiel:

CSS:

```
p.note {
  position: relative;
  left: 15%;
  width: 80%;
  padding: 30px;
  padding-bottom: 45px;
  border: 1px solid black;
  line-height: 1.5em;
  color: black;
  font-weight: bold;
  text-align: justify;
  background-color: #eeeeee;
}
```

HTML:

```
<p class="note">
  Dies ist ein kleiner Testabsatz. Dies ist ein kleiner Testabsatz...
</p>
```

CSS Selektoren

- CSS ID‘s
  - Identifizieren eindeutig ein Element auf der Seite
  - Selektor in CSS mit **#** gekennzeichnet

```
#myName {
   text-align: center;
   color:yellow;
   }
```

XPages CSS Basics – CSS IDs

- **ACHTUNG:**
  - Benutzerdefinierte Steuerelemente können mehrfach auf einer XPage verwendet werden
  - IDs müssen eindeutig berechnet werden, bevor sie an den Browser gesendet werden

- **Vorher (Definition):**

- **Nachher (Seiteninhalt für den Browser):**

```
<span id="view:_id1:myName">Label</span>
```

CSS Selektoren

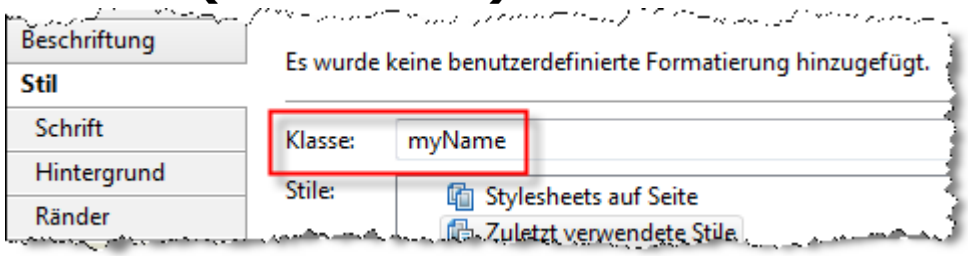
- CSS Klassen
  - Können beliebig oft auf einer Seite verwendet werden
  - Werden verwendet, um alle Elemente eines Typs einheitlich zu gestalten
  - Im Selektor mit einem „.“ gekennzeichnet

```
.myName {
   text-align: center;
   color:yellow;
   }
```

XPages CSS Basics – CSS Klassen

- **HINWEIS:**
  - Benutzerdefinierte Steuerelemente können mehrfach auf einer XPage verwendet werden
  - Gleichartige Stilklassen sorgen für eine einheitliche Darstellung

- **Vorher (Definition):**



- **Nachher (Seiteninhalt für den Browser):**

```
<span id="view:_id1:myName" class="myName">Label</span>
```

Strukur des IBM oneUI Layout-Frameworks

CSS Frameworks – Warum?

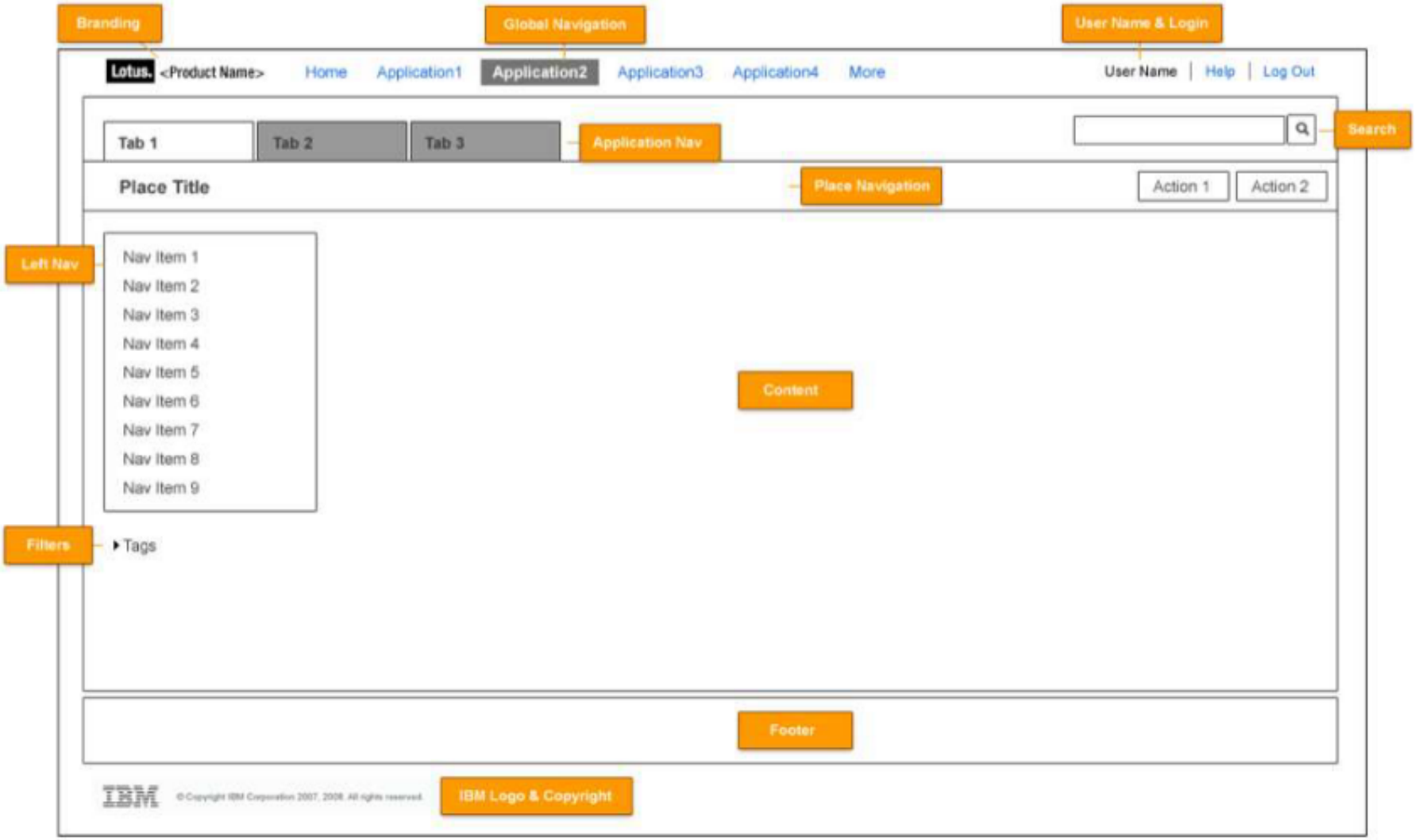
- Steigerung der Entwicklungsgeschwindigkeit
- Cross-Browser Support
  - Einschl. einiger Hacks für ältere Browser oder nicht unterstütze Features
- Stellt einheitliches „Look and Feel“ in mehreren Anwendungen sicher
- Einige (nicht alle) Frameworks verfügen über eine (gute) Dokumentation
- Einige Frameworks bieten zusätzliche Tools, um eigene Layouts zu definieren
- **Beispiele:**

Blueprint CSS	960 Grid System	jQuery UI CSS
YUI Grids CSS (Yahoo)	Elements CSS	Elastic Columns
IBM oneUI	YAML CSS Framework	Und viele weiter …

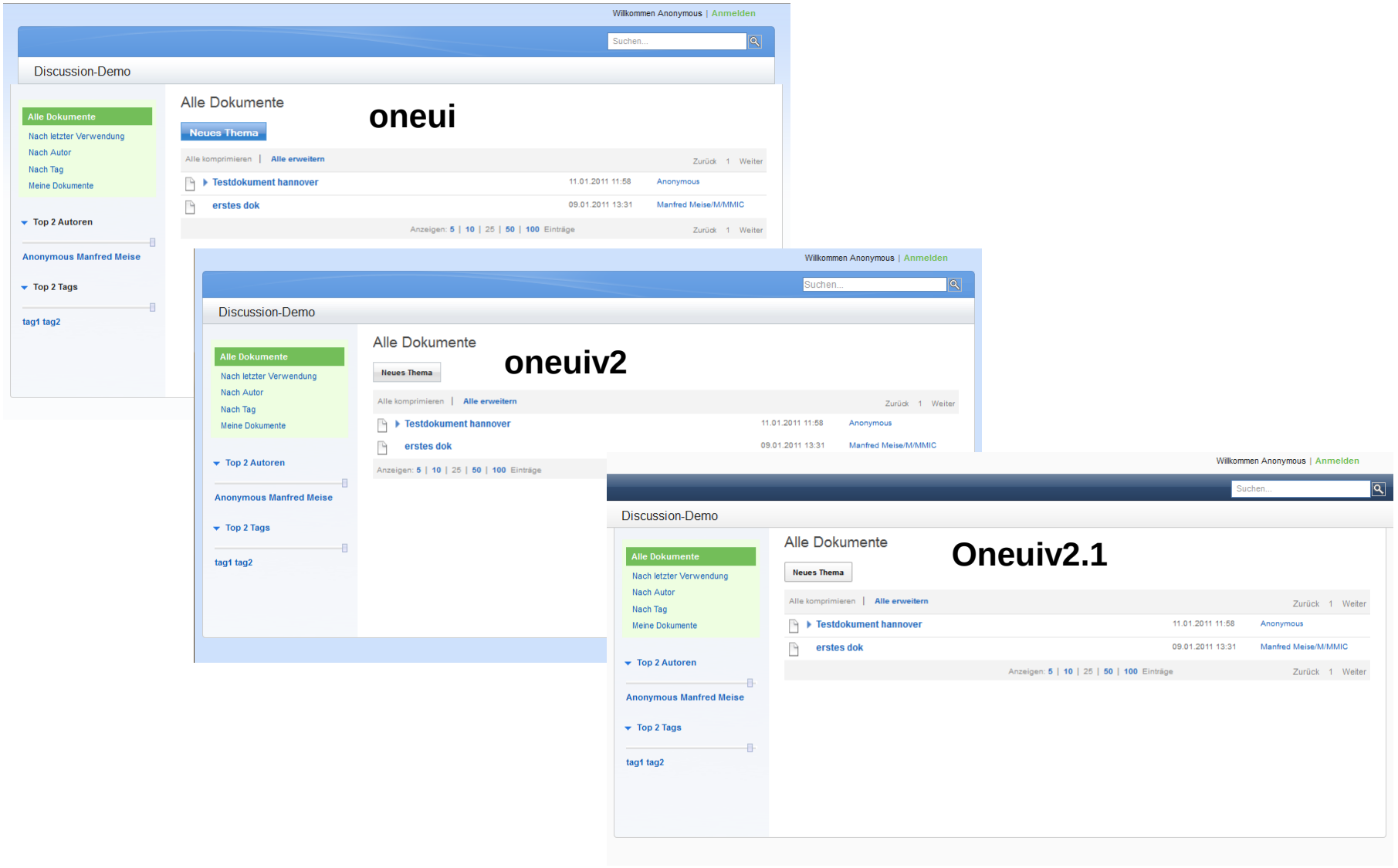
Das IBM oneUI Framework

- Layout und Presentation Framework
- Wird von IBM für alle modernen Webanwendungen eingesetzt
- Dokumentation:
  http://www-12.lotus.com/ldd/doc/oneuidoc/docpublic/index.htm
- Vorteile:
  - Integrierte Layout Funktionen
  - Standardisiertes Look and Feel für alle Gestaltungselemente
  - Zahlreiche Beispielanwendungen auf OpenNTF, welche dieses Framework verwenden
- Nachteile:
  - Basiert nicht auf Raster Layout
  - Umfangreich – erfordert intensive Einarbeitung
- Mittlerweile verfügbare Versionen
  - oneUI Version 1 (enthalten in 8.5.0)
  - oneUI Version 2 (enthalten in 8.5.2)
  - oneUI Version 2.1 (enthalten in 8.5.3)

Das IBM oneUI Framework – Layout Elemente



Gegenüberstellung der Darstellungen



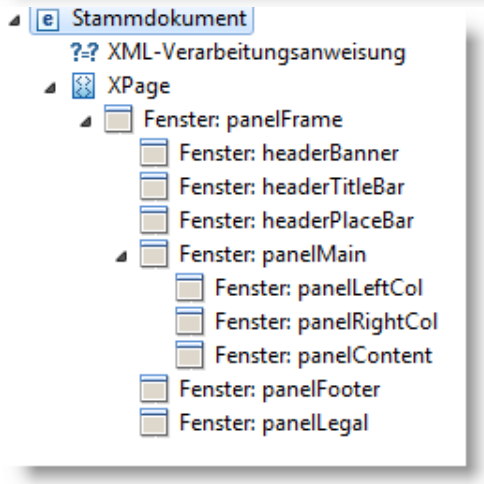
Das IBM oneUI Framework - Layout

- Fenster Steuerelemente mit Stilklassendefinitionen legen Strukturen fest

```
<xp:panel styleClass="lotusFrame" id="panelFrame">
    <xp:panel styleClass="lotusBanner" id="headerBanner">Banner info goes here...</xp:panel>
    <xp:panel styleClass="lotusTitleBar" id="headerTitl                                    anel>
    <xp:panel styleClass="lotusPlaceBar" id="headerPlac                                    anel>

    <xp:panel styleClass="lotusMain" id="panelMain">
        <xp:panel styleClass="lotusColLeft" id="panelLeftCol">Left side nav info goes here...</xp:panel>
        <xp:panel styleClass="lotusColRight" id="panelRightCol">Right sidebar info goes here...</xp:panel>
        <xp:panel styleClass="lotusContent" id="panelContent">Main content goes here...</xp:panel>
    </xp:panel>
    <xp:panel styleClass="lotusFooter" id="panelFooter">Footer info goes here...</xp:panel>
    <xp:panel styleClass="lotusLegal" id="panelLegal">Legal info goes here...</xp:panel>
</xp:panel>
```

Reihenfolge beachten



Mit optionalen weiteren Unterstukturen

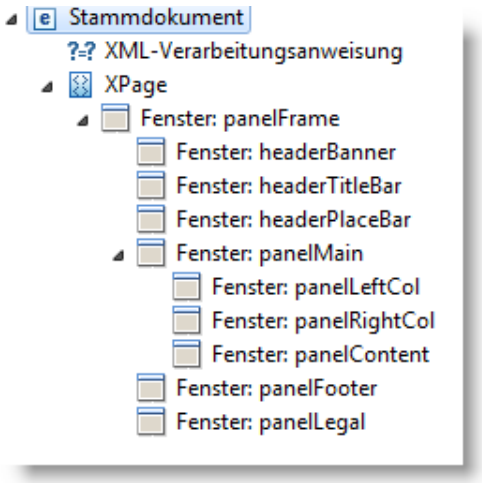
Empfohlenes Design-Pattern

- Die einzelnen Layout-Bereiche in separate benutzerdefinierte Steuerelemente auslagen
  - Erhöht Übersichtlichkeit
  - Erhöht Änderungsfreundlichkeit
  - Erlaubt es, abgewandelte Designs mit gleichen Kernmodulen zu erstellen

```
<xp:panel styleClass="lotusFrame">
    <xc:layout_banner></xc:layout_banner>
    <xc:layout_title></xc:layout_title>
    <xc:layout_place></xc:layout_place>
    <xp:panel styleClass="lotusMain">
        <xc:layout_left></xc:layout_left>
        <xp:panel styleClass="lotusContent">
            <xp:callback facetName="facet1" id="callback1"></xp:callback>
        </xp:panel>
    </xp:panel>
    <xc:layout_footer></xc:layout_footer>
</xp:panel>
```

Anleitung: oneUI Layouts anwenden

1. Erstellen Sie die Grundstruktur eines oneUILayouts und weisen den verwendeten Fenstern die oneUI Stilklassen zu



Tip: Verwenden Sie eine StyleSheet Datei mit Namensplatzhaltern (um keine Schreibfehler zu begehen)

2. Testen Sie im Browser

Das Ergebnis ist unbefriedigend?

Banner info goes here...
TitleBar info goes here...
PlaceBar info goes here...
Left side nav info goes here...
Right sidebar info goes here...
Main content goes here...
Footer info goes here...
Legal info goes here...

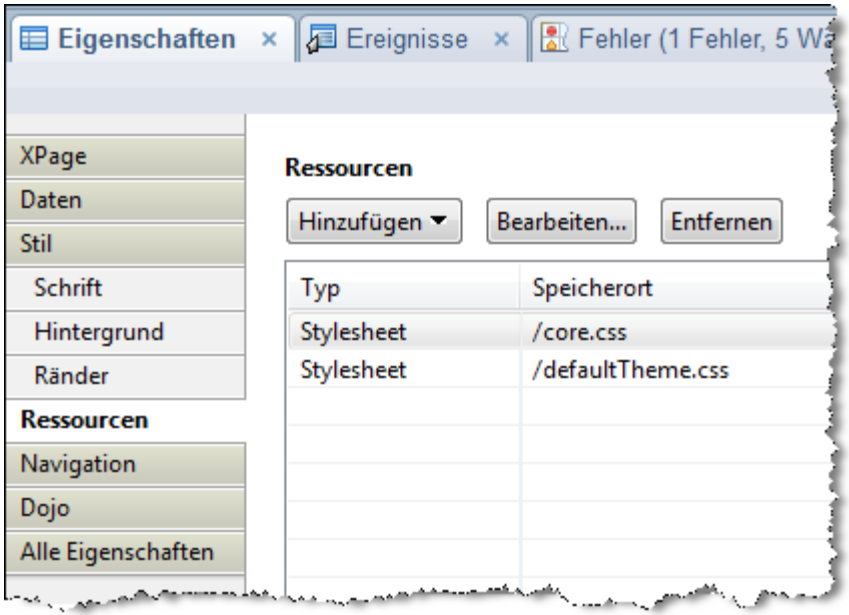
- Klar: wir haben CCS Stilklassen referenziert, die Ressourcen jedoch nicht eingebunden (bzw. nur unseren Platzhalter)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html lang="de">
<head>
<title></title>
<link rel="stylesheet" type="text/css" href="/xsp/.ibmxspres/.mini/css/@Da&amp;@Ib&amp;2Tfxsp.css&amp;2TfxspLTR.css&amp;2TfxspFF.css.css">
<script type="text/javascript" src="/xsp/.ibmxspres/dojoroot-1.6.1/dojo/dojo.js" djConfig="locale: 'de-de'"></script>
<script type="text/javascript" src="/xsp/.ibmxspres/.mini/dojo/.de-de/@Iq.js"></script>
<link rel="stylesheet" type="text/css" href="/EC12/BasicStyling.nsf/OneUIv2Lotus.css">
</head>
<body class="xspView tundra">
<form id="view:_id1" method="post" action="/EC12/BasicStyling.nsf/basicStyling.xsp" class="xspForm" enctype="multipart/form-data">
<div id="view:_id1:panelFrame" class="lotusFrame">
<div id="view:_id1:headerBanner" class="lotusBanner">
```

Anleitung: Optik herstellen

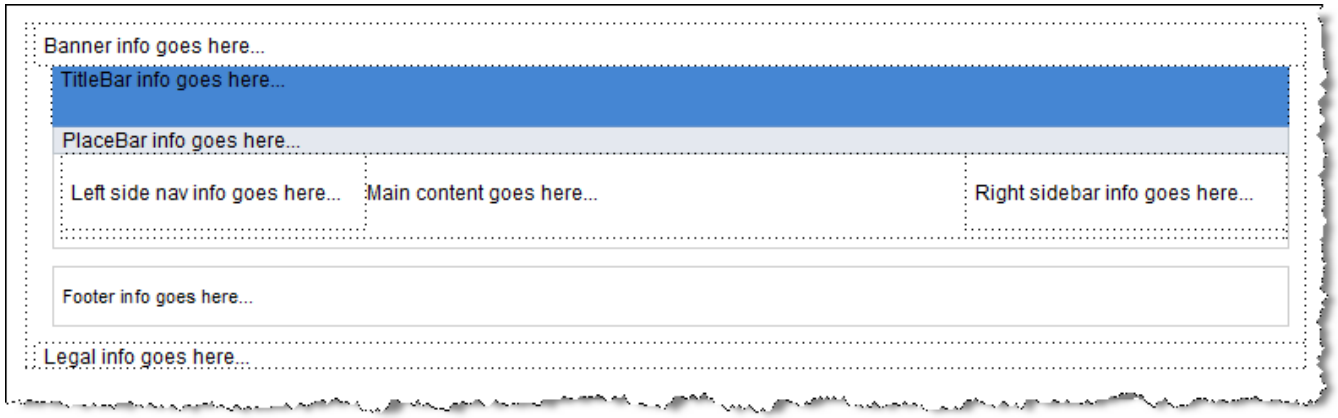
1. Importieren Sie folgende CSS StyleSheets aus dem Verzeichnis „`<NotesData>\domino\html\oneuiv2`“:
   - `base\core.css`
   - `defaultTheme\defaultTheme.css`

2. Fügen Sie diese Ressourcen zur XPage hinzu

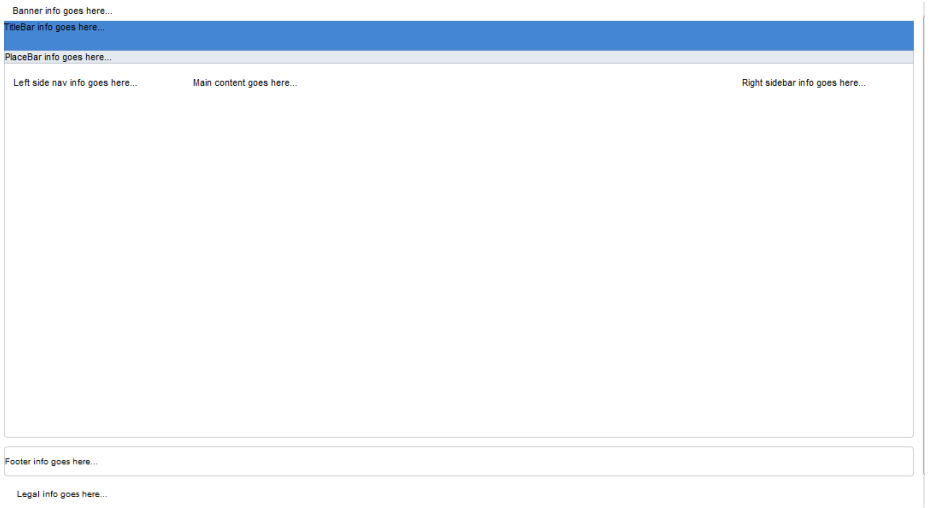


Sieht ja schon besser aus …

- Sowohl im XPage-Editor …

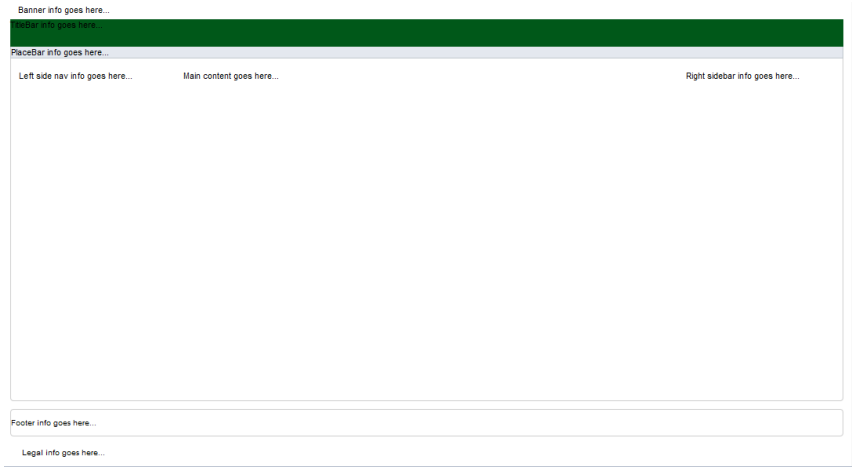


- als auch im Browser …

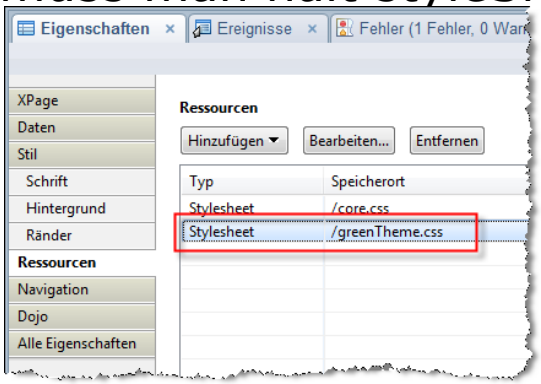


Wir können auch die Farbe wechseln …

- …wenn jemand grün schöner findet

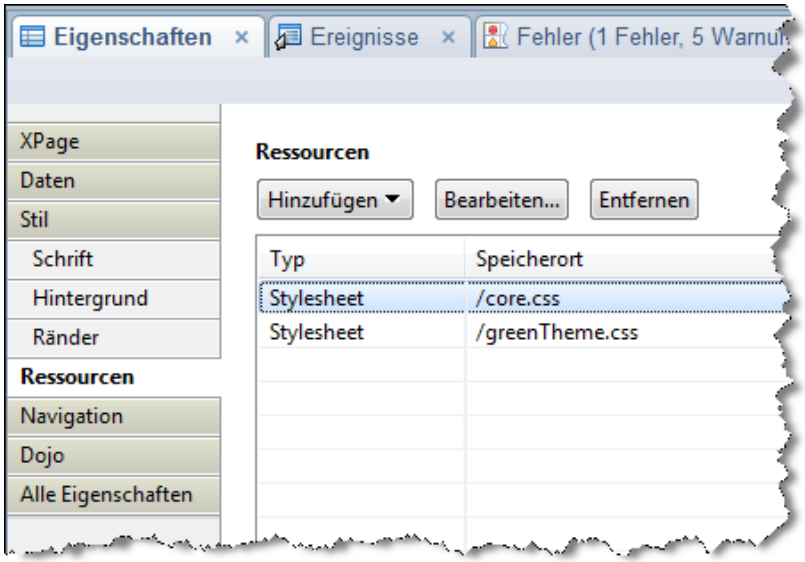


- muss man halt styleSheets austauschen



Anleitung: Farbmuster wählen

1. Importieren Sie die entsprechenden CSS StyleSheets aus dem gewünschten Unterverzeichnis der MotivDateien, z.B.: „<NotesData>\domino\Html\oneuiv2“:
   `greenTheme\greenTheme.css`
   oder
   `redTheme\redTheme.css`
2. Und fügen diese statt der „`defaultTheme.css`“ zur XPage hinzu



Verwendung von Motiven

Was ist ein Motiv (Theme)?

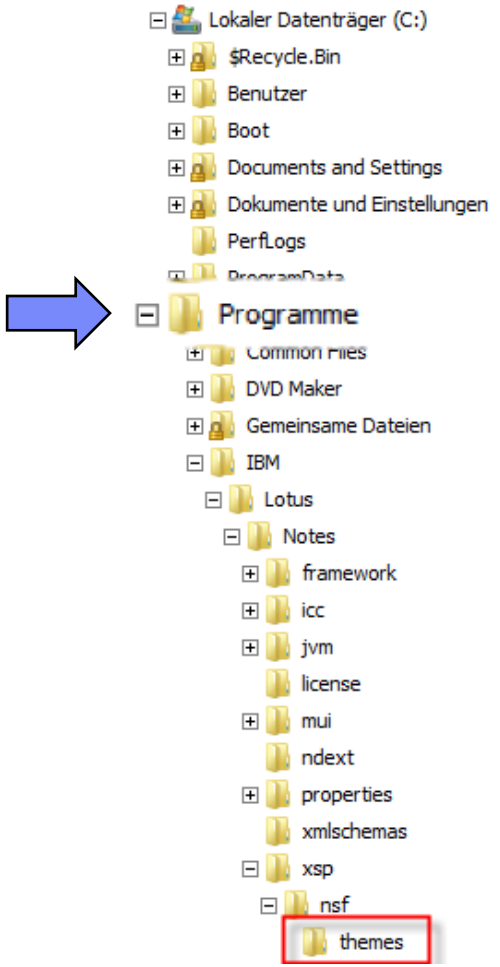
- XML Datei mit dem root element “theme”
- JSF Gestaltungselement (Ressourcen\ Motive) seit Domino 8.5
- Kann serverweit oder anwendungsbezogen definitiert werden
  - Server enthalten bereits einige Vorgabemotive
- Steuert Seitenmodifikationen zur Laufzeit
  - Resourceneinschluss (JavaScript / CSS)
  - Manipulation der Attribute von Steuerelementen

XPage - Motive

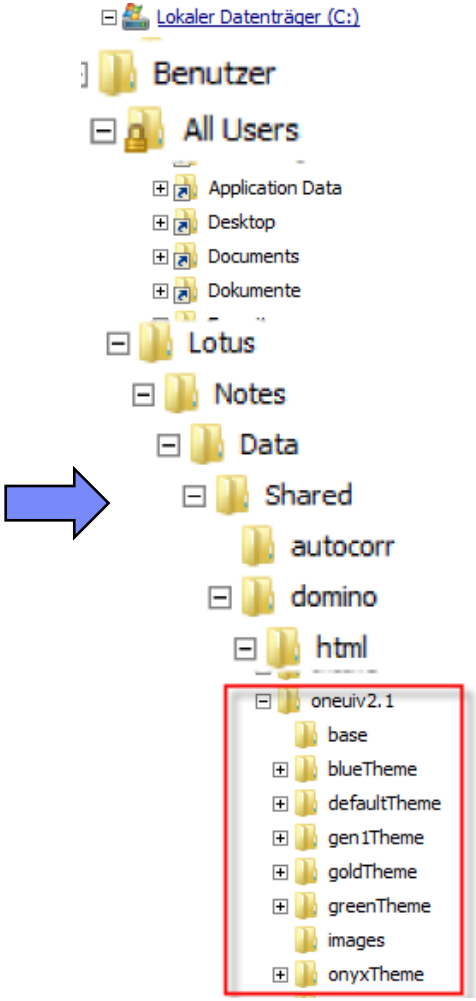
- Abbildung des entsprechenden JSF Elementes
- Steuern Aussehen und Verhalten der generierten Web-Page zur Laufzeit
- Kann serverweit oder anwendungsbezogen definitiert werden
  - Server enthalten bereits einige vorinstallierte Motive
  - Anwendungsbezogen
    - zentrales Gestaltungselement (seit Notes/Domino 8.5)
    - kann mehrfach vorhanden sein
- Ein Motiv kann bereits bestehende Motive erweitern
- Erlaubt es Standardressourcen zu laden, ohne diese explizit auf den einzelnen XPages laden zu müssen
- Erlaubt es, die Attribute eines jeden Elementes einer XPage zur Laufzeit zu beeinflussen
  - statisch
  - konditional
- XML basiert und einfach zu erlernen
  - Muster wird bei Erstellung eines neuen Motives erstellt
  - Bearbeitbar mit Texteditor oder Eclipse XML-Edior

IBM OneUI im Filesystem

- Theme-Definitionen



- Theme-Ressources



Vorinstallierte Motive

- Auf Server (und Client) gespeichert in:
  `[domino root]/xsp/nsf/themes/`
  ACHTUNG: nicht im Datenverzeichnis
- Hinzufügen eigener serverweiter Motive durch Speicherung im gleichen Verzeichnis
- Hierdurch können alle Anwendungen eines dieser Motive verwenden
- können weiter verfeinert werden (serverweit oder anwendungspezifisch
- sollten nicht verändert werden (werden bei Server/Client-Updates überschrieben)

Vorinstallierte Motive

- webstandard (default theme)
- notes
- oneui
- oneuiv2
- oneuiv2.1

- oneuiv2_gold
- oneuiv2_green
- oneuiv2_metal
- oneuiv2_red

- oneuiv2.1_blue
- oneuiv2.1_gen1
- oneuiv2.1_gold
- oneuiv2.1_green
- oneuiv2.1_onyx
- oneuiv2.1_orange
- oneuiv2.1_pink
- oneuiv2.1_purple
- oneuiv2.1_red
- oneuiv2.1_silver

Motivauswahl (oneUI V2) der Diskussions-Datenbank

blue

gold

green

red

Das einfachste Beispiel eines Motives

- Dieses Motiv hat keine Funktion, ist jedoch gültig

Home × useless.theme ×

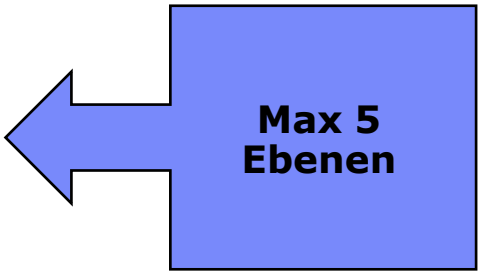
```
1 <theme />
```

- Motive können andere Motive erweitern:

Home × useless.theme ×

```
1 <theme extends="webstandard" />
```

  - webstandard
  - oneui
  - oneuiv2
  - oneuiv2.1
  - notes
  - …

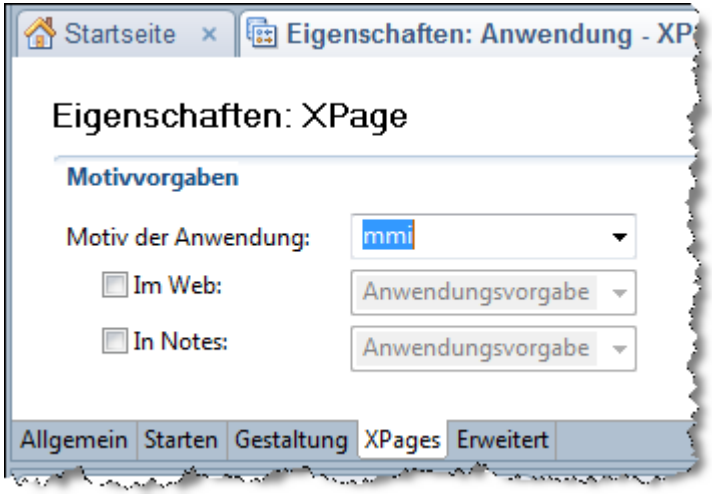


- Damit Motive Wirkungen zeigen, müssen:
  - Ressourcen zur generierten Seite hinzugefügt werden oder
  - Attribute für Komponenten gesetzt werden

Auswahl / Aktivierung eines Motives

- Servervorgabe
  - Datei `xsp.properties`
- Anwendungsvorgabe
  - Eigenschaft `XPages`

```
40
41  # ######################################
42  # THEME
43  # ######################################
44
45  # Name of the XSP theme to use
46  #xsp.theme=webstandard
47
48  # Name of the XSP theme to use when running on the web
49  # If this property is not defined, the xsp.theme is used
50  #xsp.theme.web=
51
52  # Name of the XSP theme to use when running on the notes client
53  # If this property is not defined, the xsp.theme is used
54  #xsp.theme.notes=
```



Aufbau einer Motiv Datei

- Jede Motiv Datei folgt diesem Aufbau

```
<theme>
    <resource>
        <content-type>text/css</content-type>
        <href>mystylesheet.css</href>
    </resource>

    <control>
        <name>[Control Name]</name>
        <property>
            <name>[property Name]</name>
            <value>[property Value]</value>
        </property>
    </control>

</theme>
```

- ..wobei die beiden Haupt-Elemente

  `<resource>`

  `<control>`

- … gar nicht oder auch mehrfach vorkommen können

Das Motiv-Element: <resource>

- Kann recht einfach sein:
  - Referenz einer internen Anwendungs-Ressource

```
<resource>
    <content-type>text/css</content-type>
    <href>blue.css</href>
</resource>
```

  - Referenz einer vorinstallierten Server-Ressource

```
<resource dojoTheme="true">
    <content-type>text/css</content-type>
    <href>/.ibmxspres/dojoroot/dijit/themes/tundra/tundra.css</href>
</resource>
<resource dojoTheme="true">
    <content-type>text/css</content-type>
    <href>/.ibmxspres/dojoroot/ibm/domino/widget/layout/css/domino-default.css</href>
</resource>
```

… Das Motiv-Element: <resource>

- oder auch ziemlich komplex
  - Referenz mehrerer CSS Dateien
    - intern / extern
  - Definition von Client-Side Javascript Dateien
  - Definition von „Rendert“ Eigenschaft für Ressourcen
    - Auswahl verschiedener CSS Dateien in Abhängigkeit des Browsers zur Laufzeit

- Resource nur verwenden, wenn DoJo Steuerelemente verwendet werden

```
<resource dojoTheme=”true”>
```

- Browser/Client Erkennung

```
<resource rendered="#{javascript:context.getUserAgent().isIE(0,6) == true}">
<resource rendered="#{javascript:context.getUserAgent().isFirefox()}">
<resource rendered="#{javascript:context.getUserAgent().isSafari()}">
<resource rendered="#{javascript:context.isRunningContext('Notes')}">
```

- Erkennung Schreibrichtung

```
<resource rendered="#{javascript:context.isDirectionRTL()}">
<resource rendered="#{javascript:context.isDirectionLTR()}">
```

- Referenz interner/externer Dateien

```
<href>/.ibmxspres/global/theme/oneui/iehacks.css</href>
<href>http://www.someserver.com/resources/application.css</href>
```

Pfade auf Server-Ressourcen: HTML Verzeichnis

Alias	Physikalisch	HTTP
/.ibmxspres/domino	<NotesData>domino/html	http://<server>/

- Motiv-Definition

theme	(description?, resource*, property*, control*)
extends	webstandard
xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance
xsi:noNamespaceSchemaLocation	platform:/plugin/com.ibm.designer.domino.stylekits/schema/stylekit.xsd
resource	(description?, content-type, href)
content-type	text/css
href	/.ibmxspres/domino/oneuiv2/base/core.css

- Browser-HTML

```
<head>

<link rel="stylesheet" type="text/css" href="/oneuiv2/base/base.css">

</head>
```

Pfade auf Server-Ressourcen: XPages Global Verzeichnis

Alias	Physikalisch	HTTP
/.ibmxspres/global	<NotesData>domino/java/xsp	http://<server>/domjava/xsp

- Motiv-Definition

theme	(description?, resource*, property*, control*)
resGlobaextends	webstandard
xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance
xsi:noNamespaceSchemaLocation	platform:/plugin/com.ibm.designer.domino.stylekits/schema/stylekit.xsd
resource	(description?, content-type, href)
content-type	text/css
href	/.ibmxspres/global/theme/webstandard/xsp.css

- Browser-HTML

```
<head>

<link rel="stylesheet" type="text/css" href="/xsp/.ibmxspres/.mini/css/@Da&amp;@Ib&amp;2Tfxsp.css&amp;2TfxspLTR.css&amp;2TfxspFF.css.css">

</head>
```

Pfade auf Server-Ressourcen: DoJo Verzeichnis

Alias	Physikalisch	HTTP
/.ibmxspres/dojoroot	<NotesData>domino/js/dojo-1.5.1	http://<server>/domjs/dojo-1.5.1

- Motiv-Definition

theme	(description?, resource*, property*, control*)
extends	webstandard
xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance
xsi:noNamespaceSchemaLocation	platform:/plugin/com.ibm.designer.domino.stylekits/schema/stylekit.xsd
resource	(description?, content-type, href)
dojoTheme	true
content-type	application/x-javascript
href	/.ibmxspres/dojoroot/ibm/xsp/widget/layout/xspClientDojo.js

- Browser-HTML

```
<head>

<script type="text/javascript" src="/xsp/.ibmxspres/dojoroot-1.6.1/dojo/dojo.js" djConfig="locale: 'de-de'"></script>

</head>
```

Beispiel eines abgeleiteten Motives

Home | oneuiv2_gold.theme

```
1  <theme extends="oneuiv2">
2
3      <resource rendered="#{javascript:context.isDirectionLTR()}">
4          <content-type>text/css</content-type>
5          <href>/.ibmxspres/domino/oneuiv2/goldTheme/goldTheme.css</href>
6      </resource>
7      <resource rendered="#{javascript:context.isDirectionRTL()}">
8          <content-type>text/css</content-type>
9          <href>/.ibmxspres/domino/oneuiv2/goldTheme/goldThemeRTL.css</href>
10     </resource>
11
12 </theme>
```

- Schließt weitere Ressoucen ein

… Das Motiv-Element: <control>

- Attribute des „`Control`“ Elementes definieren die Auswirkung auf gesteuerte Attribute

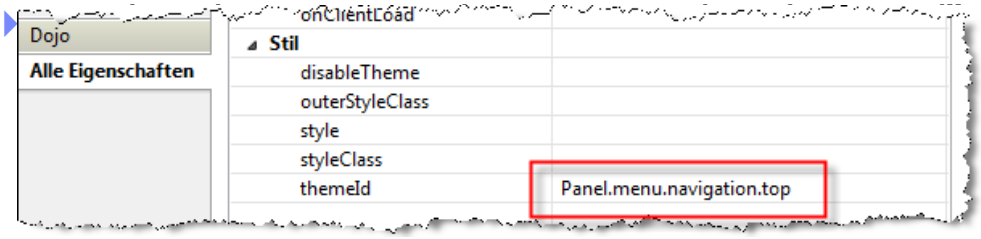
  `mode=”concat”`

  - **erweitert** Eigenschaft (Vorgabe, wenn nicht angegeben)

  `mode=”override”`

  - **ersetzt** Eigenschaft durch hier definierte Eigenschaften
- Auswahl des zu beeinflussenden Steuerelementes
  - Das `<control>` Element selektiert xsp-Elemente nach Default

```
<control>
    <name>Panel.TitleBar</name>
    <property>
        <name>styleClass</name>
        <value>lotusTitleBar</value>
    </property>
</control>
```



zugewiesene Namen

oneUIv2.0: Default Theme-IDs

```
<!-- OneUI v2.0.1 Control ThemeID List:

        lotusui.Frame

        lotusui.Banner
                    Banner.RightCorner
                    Banner.Inner
                    Banner.Logo
                    Banner.Utility
                    Banner.Applications

        lotusui.TitleBar
                    TitleBar.RightCorner
                    TitleBar.Inner
                    TitleBar.Tabs
                    TitleBar.Search

        lotusui.PlaceBar
                    PlaceBar.RightCorner
                    PlaceBar.Inner
                    PlaceBar.Buttons

        lotusui.Main
                    Main.ColLeft
                    Main.Content
                    Main.ColRight

        lotusui.Footer

        lotusui.Menu
                    Menu.RightCorner
                    Menu.Inner

        lotusui.ActionButton
                    ActionButton.Disabled
                    ActionButton.Special
-->
```

ThemeIDs ermitteln

- Um allen Objekten eines identischen Typs die gleiche StyleClass zuzuweisen, können wir die notwendigen Definitionen im Theme hinterlegen
- Dieses erfordert, das die Theme-ID des Objektes bekannt ist
- einige ThemeIDs können intuitiv errraten werden, andere nicht unbedingt
- diese kann man programmatisch durch das StyleKitFamily-Attribut der Komponente ermitteln
- Hinterlegen in einer Bereichsvariablen
- Anzeigen auf einer Testmaske mit berechtetem Feld, das an die Bereichsvariable gebunden ist

```
var comp = getComponent('buttonNewTopic');
sessionScope.buttontheme = comp.getStyleKitFamily();
```

Eigenschaften von Standardkomponenten setzen

Seitentitel und Schriftarten setzen

```
<control override="false">
    <name>ViewRoot</name>
    <property>
        <name>pageTitle</name>
        <value>#{database.title}</value>
    </property>
    <property>
        <name>pageIcon</name>
        <value>/icons/crystal/32x32/apps/fileshare.png
        </value>
    </property>
    <property>
        <name>dojoParseOnLoad</name>
        <value>#{true}</value>
    </property>
    <property>
        <name>dojoTheme</name>
        <value>#{true}</value>
    </property>
    <property mode="concat">
        <name>styleClass</name>
        <value>soria</value>
    </property>
</control>
```

Eigenschaften von Standardkomponenten setzen

Einheitliche Gestaltung von Seitensteuerelementen

```
<control>
    <name>Pager</name>
    <property mode="override">
        <name>layout</name>
        <value>Previous Separator Group Separator Next</value>
    </property>
    <property mode="override">
        <name>partialRefresh</name>
        <value>#{true}</value>
    </property>
</control>
```

Auszug aus „oneuiv2.1_blue.theme “

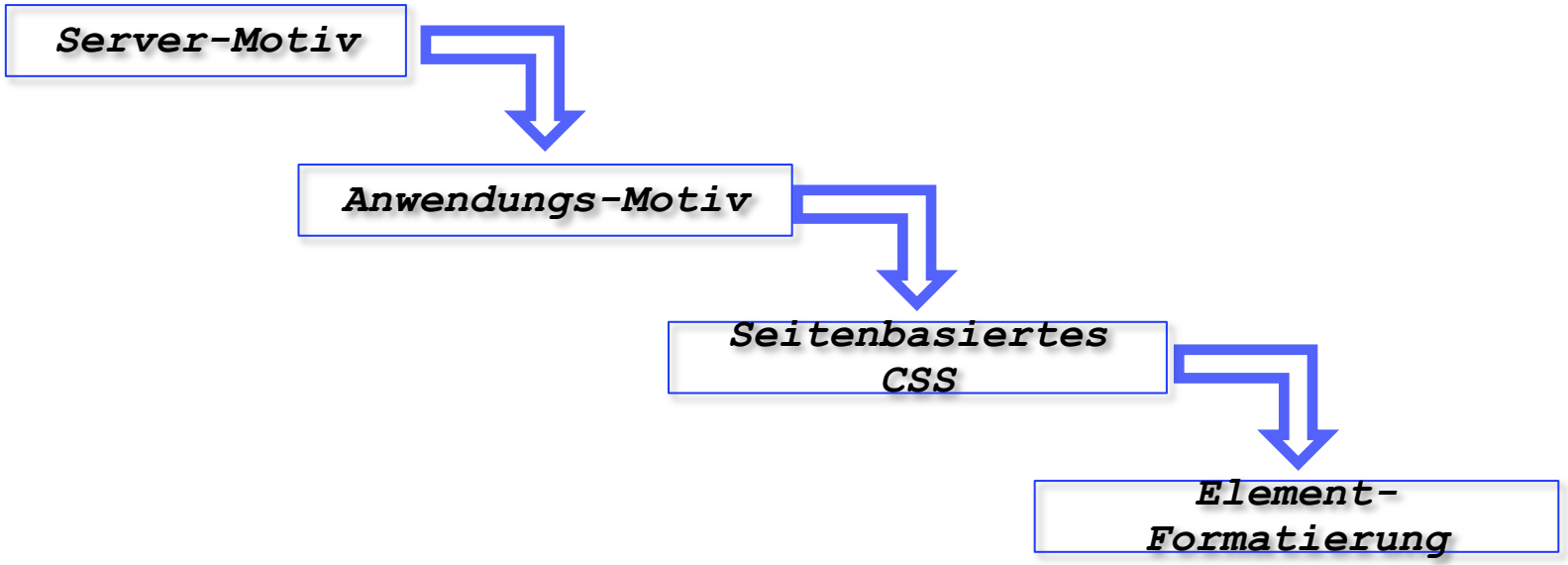
Steuerung von Schaltflächen

```
<!-- Command Button -->
<control>
    <name>Button.Command</name>
    <property>
        <name>styleClass</name>
        <value>lotusBtn</value>
    </property>
</control>

<!-- Submit Button -->
<control>
    <name>Button.Submit</name>
    <property>
        <name>styleClass</name>
        <value>lotusBtn</value>
    </property>
</control>

<!-- Cancel Button -->
<control>
    <name>Button.Cancel</name>
    <property>
        <name>styleClass</name>
        <value>lotusBtn</value>
    </property>
</control>
```

Erweitertes Styling mit Motiven: Was gilt?!




```
<theme extends="oneuiv2">
   ...
</theme>
```

Wie werden Motive oft eingesetzt?

- Garantierte Einbindung von Stylesheets und Scriptbibliotheken
  - Verwendung von Standardframeworks (z.B. Dojo, Blueprint CSS)
  - Firmenweite Gestaltungsrichtlinien (myCompanyStyles.css)
  - Anwendungsspezifische Ressourcen (myAppStyles.css)
- Erweiterung/Ersetzung von Komponentenattributen erzwingen
  - style (z.B. font-family: arial; font-size: 9pt;)
  - styleClass (e.g. xspTableCell)

Beispiel für eine (bedingte) Ressoucenkonfiguration: Schreibrichtung

```
<resource>
    <content-type>text/css</content-type>
    <href>/.ibmxspres/global/theme/webstandard/xsp.css</href>
</resource>
<resource rendered="#{javascript:context.isDirectionLTR()}">
    <content-type>text/css</content-type>
    <href>/.ibmxspres/global/theme/webstandard/xspLTR.css</href>
</resource>
<resource rendered="#{javascript:context.isDirectionRTL()}">
    <content-type>text/css</content-type>
    <href>/.ibmxspres/global/theme/webstandard/xspRTL.css</href>
</resource>
```

Beispiel für eine (bedingte) Ressoucenkonfiguration: Browserspezifisch

```
<!-- FireFox Specific -->
<resource rendered="#{javascript:context.getUserAgent().isFirefox()}">
    <content-type>text/css</content-type>
    <href>/.ibmxspres/global/theme/webstandard/xspFF.css</href>
</resource>

<!-- Safari Specific -->
<resource rendered="#{javascript:context.getUserAgent().isSafari()}">
    <content-type>text/css</content-type>
    <href>/.ibmxspres/global/theme/webstandard/xspSF.css</href>
</resource>
```

Beispiel für eine (bedingte) Ressoucenkonfiguration: Browserspezifisch

```
<!-- iehacks == if IE6 -->
<resource rendered="#{javascript:context.getUserAgent().isIE(6,6)}">
    <content-type>application/x-javascript</content-type>
    <href>/.ibmxspres/global/theme/oneuiv2/js/ie6.js</href>
</resource>
<!-- iehacks == if IE7 -->
<resource rendered="#{javascript:context.getUserAgent().isIE(7,7)}">
    <content-type>application/x-javascript</content-type>
    <href>/.ibmxspres/global/theme/oneuiv2/js/ie7.js</href>
</resource>
```

Beispiel einer Komponentenmodifikation

Zuweisung von styleSheets

```
<!-- View Root (Page Body) -->
<control>
    <name>ViewRoot</name>
    <property mode="concat">
        <name>styleClass</name>
        <value>xspView tundra</value>
    </property>
</control>
```

```
<control>
    <name>lotusui.Banner</name>
    <property>
        <name>styleClass</name>
        <value>lotusBanner</value>
    </property>
</control>
```

Eigenschaften von Standardkomponenten setzen

```
<control override="false">
    <name>ViewRoot</name>
    <property>
        <name>pageTitle</name>
        <value>#{database.title}</value>
    </property>
    <property>
        <name>pageIcon</name>
        <value>/icons/crystal/32x32/apps/fileshare.png
        </value>
    </property>
    <property>
        <name>dojoParseOnLoad</name>
        <value>#{true}</value>
    </property>
    <property>
        <name>dojoTheme</name>
        <value>#{true}</value>
    </property>
    <property mode="concat">
        <name>styleClass</name>
        <value>soria</value>
    </property>
</control>
```

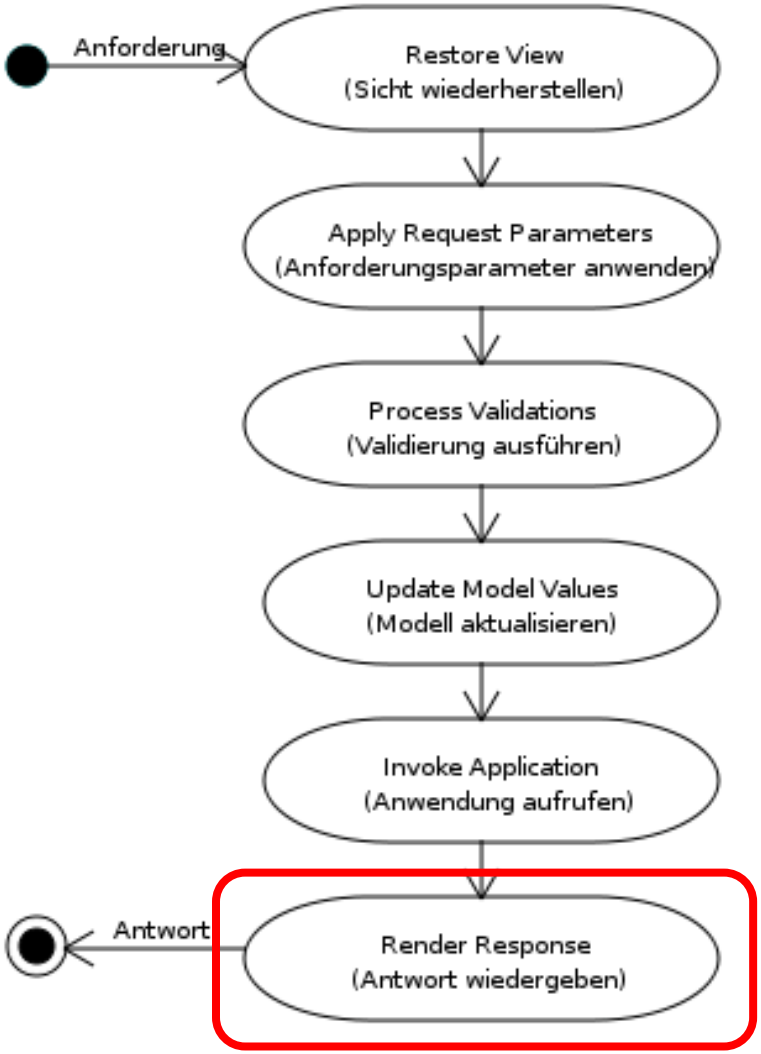
Funktioniert für jedes Attribut...

- pageTitle
- var
- rows
- partialRefresh
- ...Werte

Nahezu jedes Attribut (sofern sie nicht read-only sind) können mit Motiven gesteuert werden

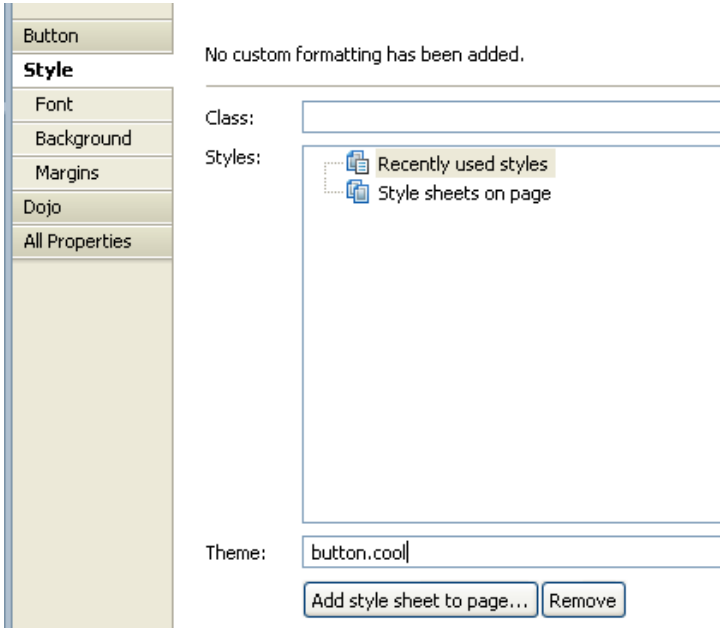
Wie arbeiten Motive?

- Motive sind wie “server-side CSS”
- Werden in der JSF Rendering-Phase angewendet
- Ressourcen werden in den Komponentenbaum integriert
- Komponenten werden mit einer Vorgabe- oder vergebener ID identifiziert (ThemeID)
- Jede Instanz einer XPage Komponente ist ein Java-Bean
  - Jedes Attribut hat eine “getter” Methode
  - Jedes Attribut hat eine “setter” Methode



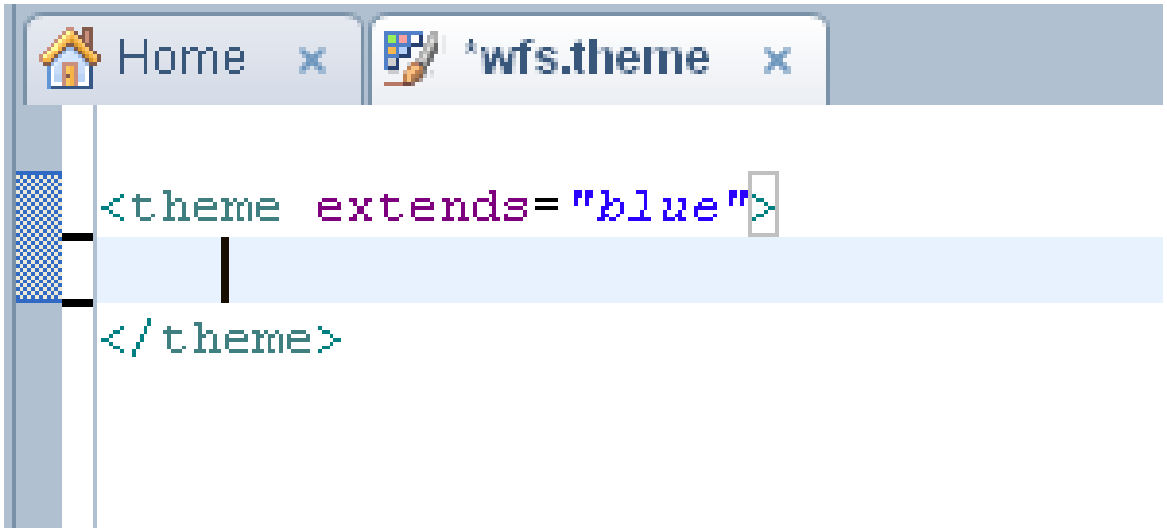
Anleitung: Komponenten steuern

1. Motive können auch Vorgabestile für Komponenten festlegen
2. Wenn z.B. alle Schaltflächen eine neue Gestalt annehmen sollen, können wir:
   - die verwendete CSS StyleClass setzen
   - eine eigene StyleClass implementieren
3. Hierzu muss das benutzerdefinierte Steuerelement “actionsBar” bearbeitet werden
4. Selektieren Sie die Schaltfläche “New Topic”
5. Auf den “Style”-Reiter geben Sie “button.cool” als Theme-ID ein
6. Speichern Sie die XPage



Erstellen eigener Motive

- Statt bestehende Motive um eigene Definition anzupassen erscheint es sinnvoller eigene Modifikationen in einem eigenen Motiv zu definieren
- Dieses Motriv kann einen beliebigen Namen tragen
- Und muss z.B. das Motiv “blue” erweitern



Erstellen eigener Motive: bestehende erweitern

- Damit das eigene Motiv nicht alle notwendigen Definitionen aufnehmen muss, kann es bestehende erweitern und anpassen

```
<theme extends="blue">
    <resource>
        <content-type>text/css</content-type>
        <href>custom.css</href>
    </resource>

    <control>
        <name>button.cool</name>
        <property mode="override">
            <name>styleClass</name>
            <value>coolbutton</value>
        </property>
    </control>
</theme>
```

Anleitung: Hinzufügen einer Control-Definition

1. Erstellen Sie ein neues Motiv in Erweiterung des Motives “blue” mit einem beliebigen Namen
2. Kopieren Sie einen Control-Abschnitt und fügen ihn vor </theme> ein.
3. Setzen Sie den Namen “`button.cool`”, wie zuvor in XPages referenziert
4. Ändern Sie das Attribut für “`styleClass`” zu “`coolButton`”

```
    <control>
        <name>button.cool</name>
        <property mode="override">
            <name>styleClass</name>
            <value>coolButton</value>
        </property>
    </control>
</theme>
```

5. Speichern Sie das Motiv
6. Aktivieren dieses Motiv in den Anwendungseigenschaften

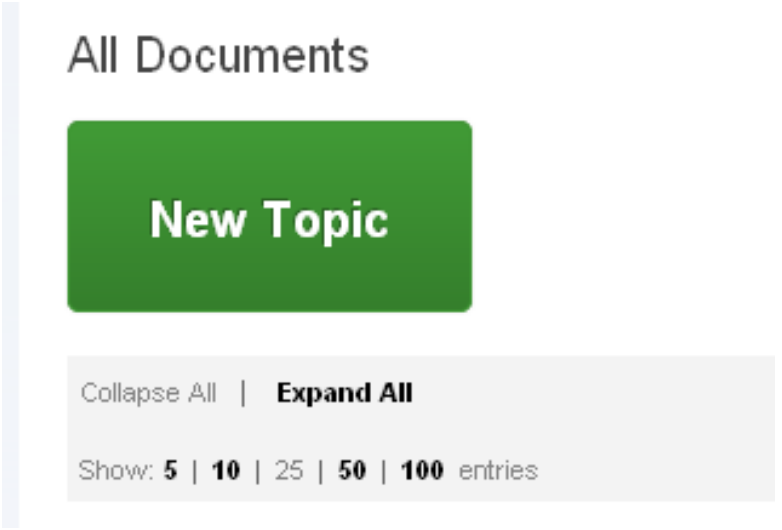
Anleitung: Erweitern des “custom.css”

1. Im Stylesheet “custom.css” muss eine Klassendefinition für “coolButton” eingefügt werden

2. Zu Testzwecken probieren wir einmal eine richtig große grüne Schaltfläche aus

3. Oben sehen wir die Klassendefinition und unten das Ergebnis im Browser

```
.coolButton {
    -moz-border-radius:5px 5px 5px 5px;
    -moz-box-shadow:0 0 1px #DDDDDD;
    background:-moz-linear-gradient(-90deg, #41
    border:1px solid #3E9533;
    color:#FFFFFF;
    display:block;
    font-size:20px;
    font-weight:bold;
    line-height:15px;
    margin:0 0 8px;
    padding:26px 30px 24px;
    text-decoration:none;
    text-shadow:-1px -1px 0 rgba(0, 0, 0, 0.3);
}
```



Anpassung einiger oneUI Elemente nach eigenen Vorstellungen

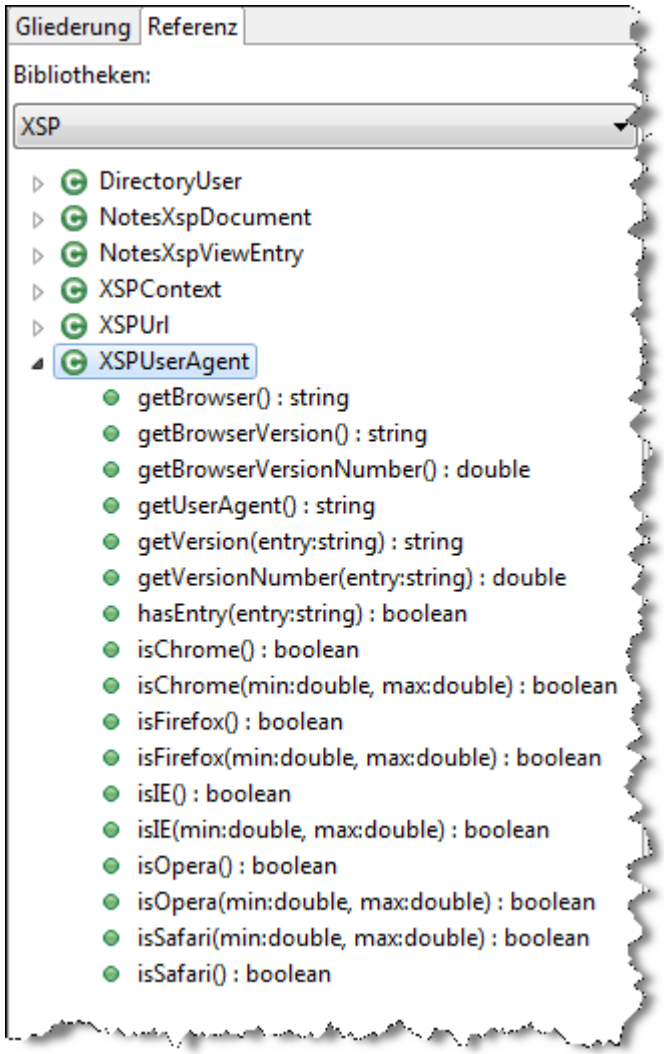
Benötigte Werkzeuge und Kenntnisse

- Spezialitäten der zu unterstützenden Browser
- Verwendung von CSS2 oder CSS3
- JSF Motive
- IBM oneUI Layout-Struktur (Hierarchie der Panel und CSS Klassen-IDs)
- Kenntnis der Theme-IDs von Standardelementen
- Entwickler-Werkzeuge im Browser (zB. Firebug, ColorZilla, …)

Motive: Häufig eingesetzte Funktionen

- Um bedingte Anweisungen in Motiven einzusetzen, lohnt ein Blick auf Methoden von

  „com.ibm.xsp.designer.context.XSPUserAgent“



Laufzeitumgebung ermitteln

- context.isRunningContext(String)

- z.B.: context.isRunningContext(„Domino“)
- z.B.: context.isRunningContext(„Notes“)

- ```
  context.getProperty("mxpd.theme.info")
  ```


  ```

Properties setzen

- <theme extends="webstandard">
- <property>
- <name>mxpd.theme.info</name>
- <value>example</value>
- </property>
- <resource
- rendered="#{javascript:context.getUserAgent().isIE()}:\">
- <content-type>text/css</content-type>
- <href>example.css</href>
- </resource>
- </theme>

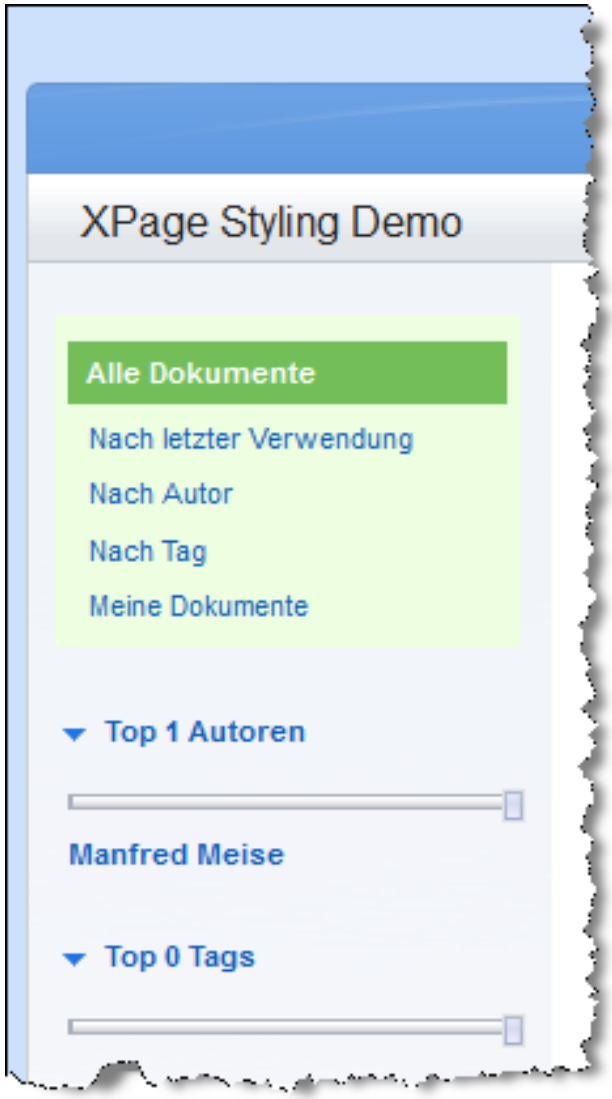
```
context.getProperty("mxpd.theme.info")
```

Anleitung: Unabhängigkeit der Seitengestaltung herstellen

1. Erstellung eines eigenen Motivs (in Ableitung eines vorgegebenen Motivs)
   ACHTUNG: max. 5 Ebene Vererbung unterstützt
2. Einbeziehung eigener CSS StyleClasses (durch Separierung in eigenen StyleSheet Ressourcen und Definition/Berechnung in Motiven)
3. Entfernen jeglicher (oder nahezu jeglicher) Style-Definitionen und Theme-ID Definitionen von den bestehenden Seiten

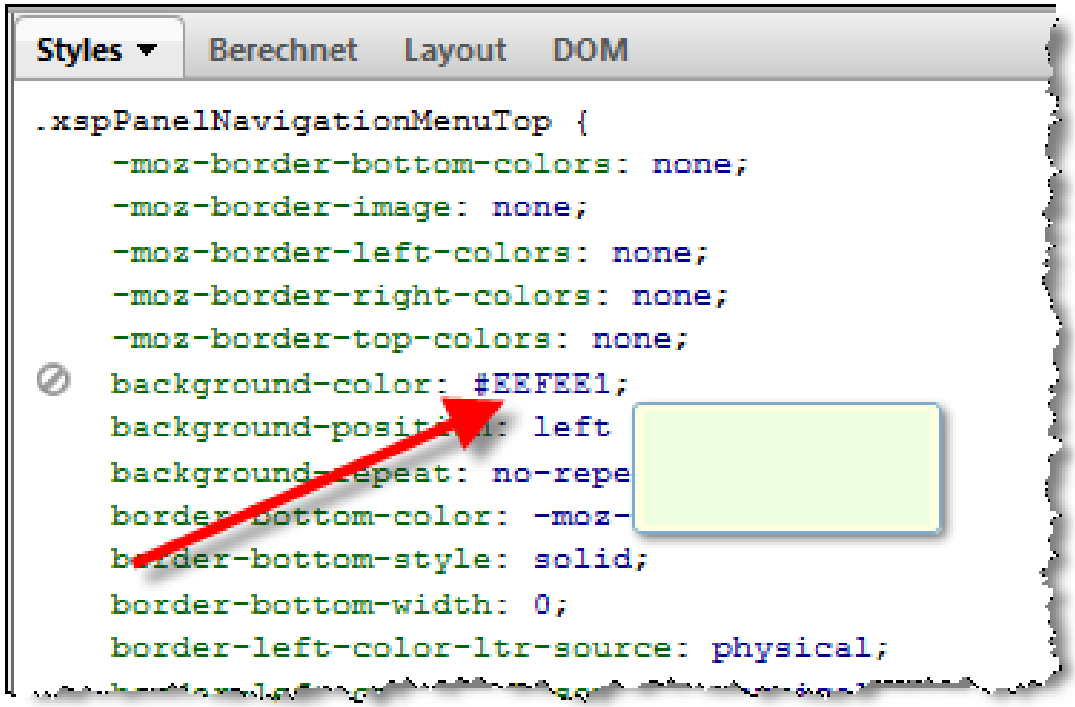
Beispiel 1: Menünavigation umgestalten

- Obwohl des “blue” Motiv recht ansprechend ist, mögen viele Anwender den grünen Hintergrund nicht
- Mit Hilfe von z.B. Firebug kann man herausfinden, welches Stylsheet dafür verantwortlich ist:
  - .xspPanelNavigationMenuTop



Beispiel 1: Menünavigation umgestalten …

- Firefox ermöglicht die einfache Analyse der Seitengestaltung und temporäre Anpassung von CSS Definitionen



Beispiel 1: Menünavigation umgestalten …

- Farbe(n) nach eigenen Wünschen ändern
  - .xspPanelNavigationMenuTop
    - background-color: #CEE1FC
  - .xspPanelNavigationMenuItemSelected a, .xspPanelNavigationMenuItemSelected a:visited
    - background-color: red
    - border-top-color: red
    - border-bottom-color: red
    - border-left-color: red
    - border-right-color: red
- Hierbei liefern Tools wie ColorZilla gute Dienste
- … gut: über Geschmack kann man streiten

Anleitung: Neue Ressourcen dem Motiv hinzufügen

1. Die bisherige Ressourcendefinition kopieren
2. Den Dateinamen des neu erstellten (2. Eintrages) in “custom.css” ändern
3. Wichtig: Nach der Ursprungsangabe einstellen! Warum ?
   - Weil wir einige Style-Class Definitionen **überschreiben** wollen
   - Das erfolgt nur, wenn diese Reihenfolge eingehalten wird

```
<resource>
    <content-type>text/css</content-type>
    <href>blue.css</href>
</resource>

<resource>
    <content-type>text/css</content-type>
    <href>custom.css</href>
</resource>
```

Anleitung: Erstellen des StyleSheets “Custom.css”

1. Erstellen Sie in Ressources → Stylesheets ein neues Stylesheet
2. Benenen Sie es “custom.css”, wie in der Resssource Definition des Motivs zuvor
3. Für alle in Firebug identifizierten Klassen die notwendigen **Änderungen** eintrage
4. z.B. für die Navigationshintegrund .xspPanelNavigationMenuTop
   - Wir müssen lediglich das/die zu änderde Attribut(e) neu einstellen. Alle anderen bleiben erhalten

```
1  /** Author: Manfred Meise, mmi consult GmbH **/
2
3  /* main menu body */
4  .xspPanelNavigationMenuTop {
5      background-color:#CEE1FC;
6      }
7
8  /* main menu links */
9  .xspPanelNavigationMenuItemSelected a, .xspPanelNavigationMenuItemSelected {
10     background-color:red;
11     border-top-color: red;
12     border-bottom-color: red;
13     border-left-color: red;
14     border-right-color: red;
15     }
16
```

Beispiel 2: Motivauswahl zur Laufzeit

- Umschaltung von Motiven zur Laufzeit
  - Links auf Seite
  - Kombox
  - Benutzerpräferenzen und Scoped Variables

```
var t = context.getSubmittedValue();

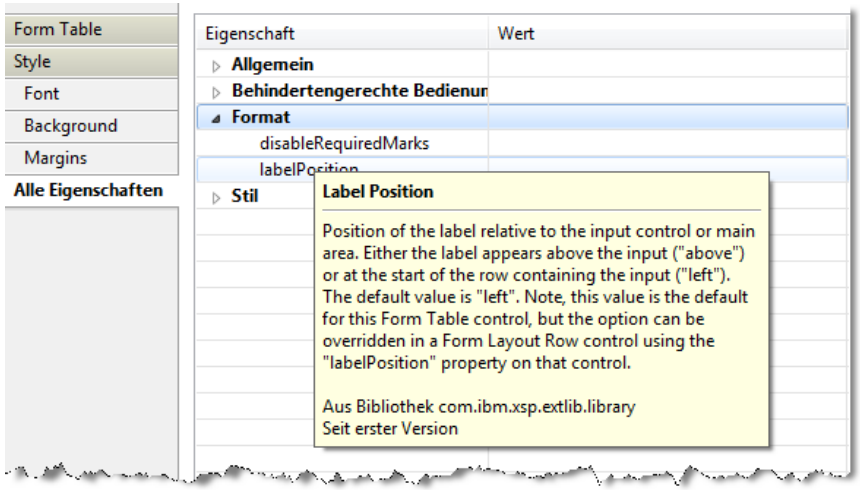
if (t=="blue" || t=="gen1" || t=="gold" || t=="green" || t=="pink" || t=="purple" || t=="red" ||t=="orange" || t=="onyx" || t=="silver") {
        context.setSessionProperty("xsp.theme",t);
    } else {
        context.setSessionProperty("xsp.theme","default");
    }
context.redirectToPage("themes");
```

- Im Kern:
  - `Context.setSessionProperty(„xsp.theme“, „blue“)`
  - `Context.redirectToPage („nameOfPage“)`

Beispiel: oneUIV2.1 Demo von Mark Leusink

Beispiel 3: Position von Feldbeschriftungen auf Eingabemasken

- Merkmal des „Form Table“ Controls



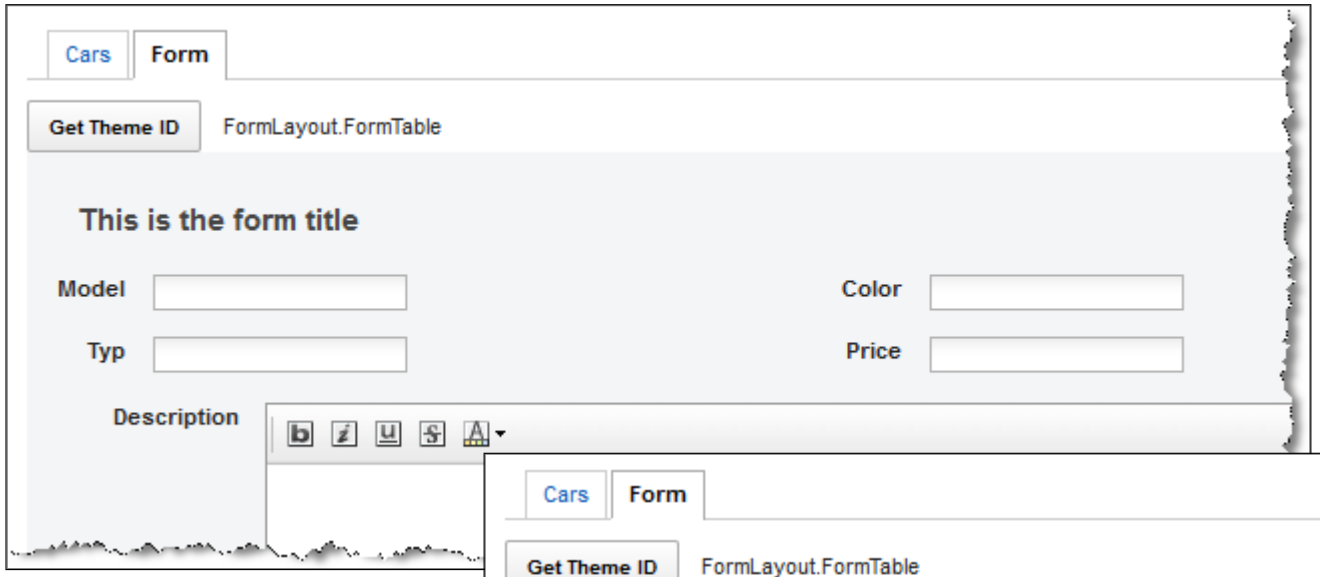
- Motivsteuerung in Motiv „mmi“

```
<theme extends="gold" xmlns:xsi="http://www.w3.or

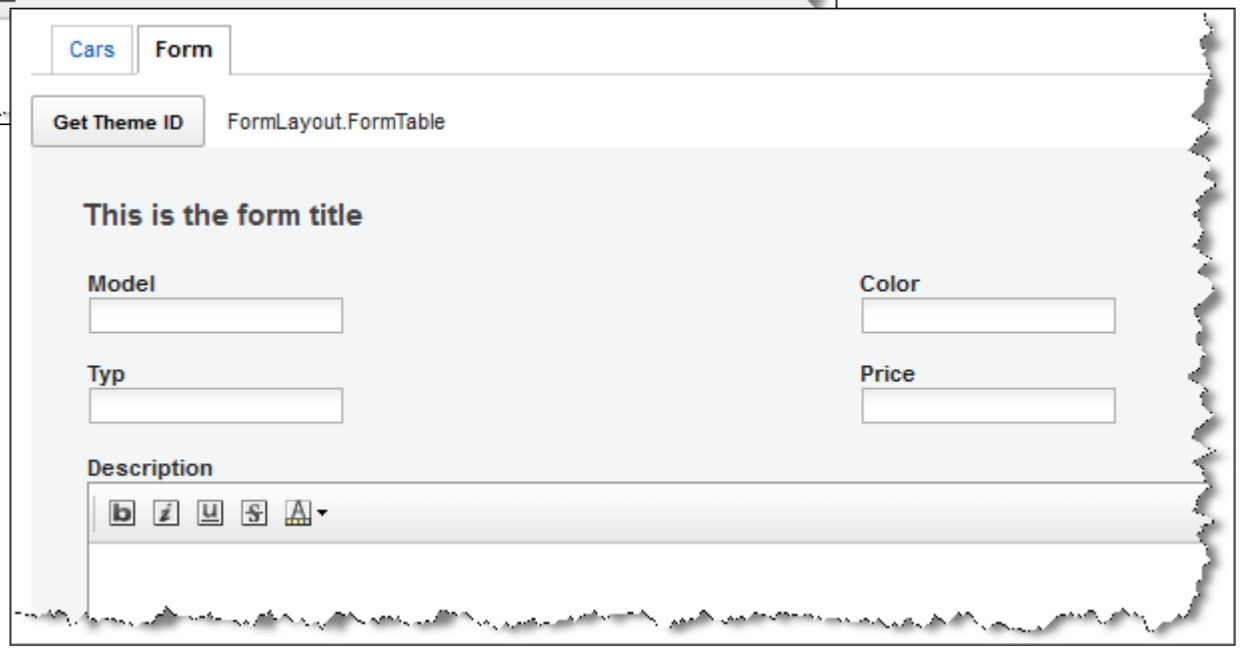
    <control>
        <name>FormLayout.FormTable</name>
        <property>
            <name>labelPosition</name>
            <value>above</value>
        </property>
    </control>

</theme>
```

Darstellungen vorher/nachher



Ohne Motivsteuerung



Mit Motivsteuerung

Anleitung: Eingabetexte vor oder über den Feldern

1. Öffnen Sie die Beispieldatenbank „Demo\oneUIV21.nsf“
2. Erstellen Sie ein neues Motiv „mmi“, welches ein bestehendes z.B. „gold“ erweitert
3. Fügen Sie eine Control Definition für „ FormLayout.FormTable“ hinzu
4. Setzen Sie das Attribut „labelPosition“ auf entweder „above“ oder left
5. Speichern Sie dieses Motiv
6. Zu Laufzeit wählen Sie das Motiv „mmi“ aus der Motivauswahl der Startseite
7. Auf dem 2 Tab der tabbed Table erscheint eine Eingabemaske gemäß der im Motiv hinterlegten Regeln für die Labelposition

Resumee und Ausblick

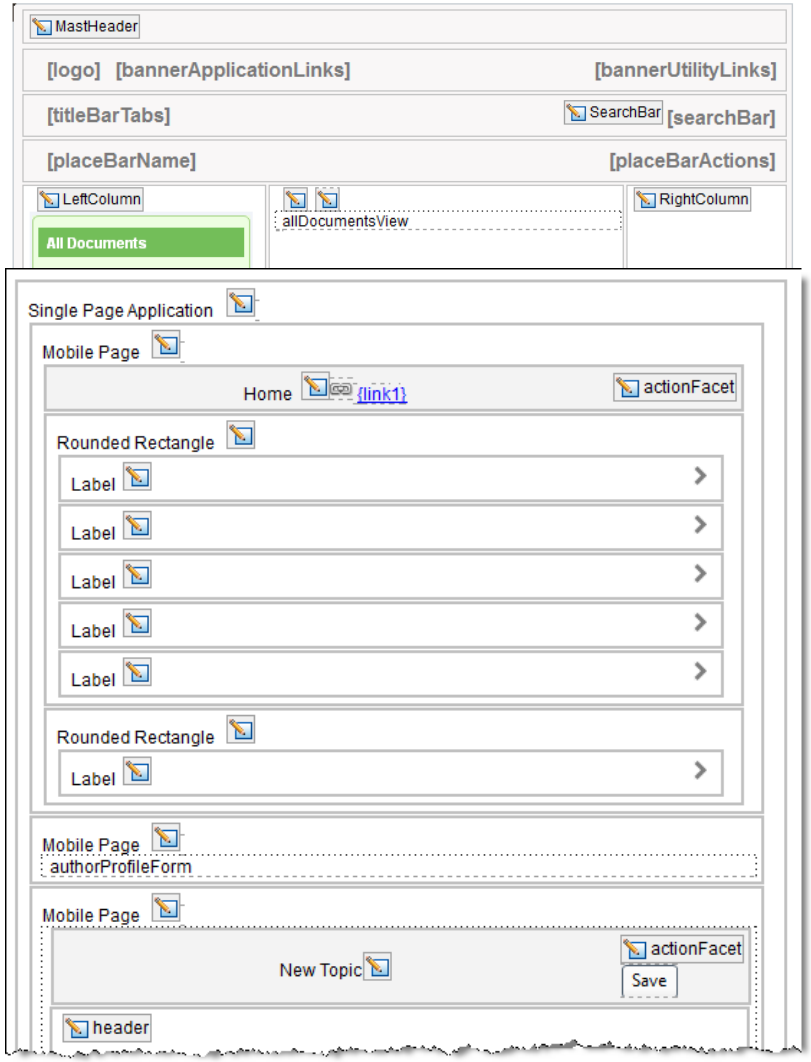
Streamline team development

- Durchgängige Verwendung von Motiven unterschützt optimal den Einsatz einer 3-Schichten Architektur der Anwendung
  - Erfahrene Entwickler erstellen und pflegen Skriptbibliotheken und Managed Beans
  - Anwendungsentwickler erstellen/pflegen Motive und definieren in Form von Komponenten die Nutzung von Bibliotheken und APIs
  - Neulinge setzen Komponenten zusammen “click-programming” und wählen vorgegebene Stile aus

Vereinfachtes Layout durch 8.5.3 UP1

- Application Layout Control
  - Keine Notwendigkeit mehr, separate Layout Steuerelemente zu definieren
  - Benutzerdefinierte Steuerelemente nehmen unmittelbar
    - Navigation
    - Anwendungselemente

  auf
- Mobile Page
  - SinglePage Gestaltung mit Ankern
  - Für iPhone und Android geeignet
- Dokumentation und Tutorials auf Developerworks



Steigerung der Entwicklungsgeschwindigkeit

- Keine zeitaufwändigen Gestaltungen bei Einhaltung der Struktur erforderlich
- Einheitliches Design aller Anwendungen
- Motive und Stile wachsen mit den Anforderungen
- Konzentration auf Logik und nicht auf Gestaltung

Noch Fragen offen geblieben?

http://www.mmi-consult.de
http://www.mmi-consult.de/faq
mailto:manfred.meise@mmi-consult.de

Quellennachweise

Quellenhinweise

Quellen:

- http://www-10.lotus.com/ldd/ddwiki.nsf/dx/Create_a_website_layout
- http://www.eview.com/eview/VOLR6.nsf/0/2FB0E91250A0BBAF852577F3006C34C7?openDocument
- http://www.xpagesblog.com/XPagesHome.nsf/SearchResults.xsp?search=themes
- http://xmage.gbs.com/blog.nsf/d6plinks/TTRY-8FHVLW
- http://www.slideshare.net/WorkFlowStudios/1-one-u-iv2-theme

Werkzeuge:

- https://addons.mozilla.org/de/firefox/addon/firebug/
- https://addons.mozilla.org/de/firefox/addon/colorzilla/