

Begleitdokumentation Schulungsvorlagen

auf XML/XSLT-Basis für neue Mitarbeiter zur Verwendung in einem Handelsmarketingsystem

BRANDAD

SYSTEMS

BRANDAD Systems AG

Gebhardtstr. 5

90762 Fürth

Inhaltsverzeichnis

1	Einleitung.....	3
2	FO Elemente	3
2.1	fo:block.....	3
2.2	fo:inline.....	3
2.3	fo:inline mit Zeilenumbruch	4
2.4	fo:block-container.....	4
2.5	fo:table	5
2.6	Boxmodell	6
3	UG Elemente	7
3.1	ug:textarea	7
3.2	ug:buttongroup.....	8
3.3	ug:checkbox.....	8
3.4	ug:combobox	9
3.5	ug:title	9
3.6	ug:image	10
3.7	ug:invisible	10
4	UH Elemente	10
4.1	uh:if	10
4.2	uh:choose	11
4.3	uh:attribute	12
4.4	uh:variable	12
5	UP Elemente	13
5.1	up:page	13
6	Errorhandling.....	13
6.1	Vorgegebene Fehlerprüfungen	13
6.2	Postprocessing Fehlerprüfungen	14
6.3	eigens erstellte Fehlerprüfungen.....	14
7	Snippets.....	15

1 Einleitung

In dieser Begleitdokumentation wird zum besseren Verständnis und zur Steigerung des Lernerfolgs der Quelltext der Elemente, die in einer Vorlage verwendet werden, unabhängig zur Baumstruktur im Template Manager 3, erläutert. **Die Schulungsvorlagen sind für die weitere Bearbeitung notwendig.** Technische Zusammenhänge sollen durch die parallele Bearbeitung der Schulungsvorlagen und dieser Dokumentation klarer werden.

Zusammen mit dem Quelltext des Elements wird die gelayoutete Ausgabe im Template Manager 3, sofern möglich, veranschaulicht. Die Darstellung des Quelltextes gestaltet sich wie folgt: Text im grauen Kasten. Elementname in Schriftfarbe Schwarz und in den Erläuterungen *kursiv*. Dazugehörige Attribute werden **Dunkelrot** dargestellt. Verknüpfungen zweier Elemente über eine Funktion und dem Attribut **uniqueId** werden **Dunkelgrün** dargestellt. Entscheidungswerte sind **Dunkelblau**.

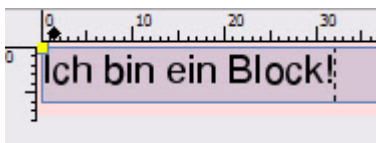
2 FO Elemente

Layoutelemente

2.1 fo:block

Ein fo:block wird meist benutzt um Absätze, Titel oder Überschriften zu formatieren.

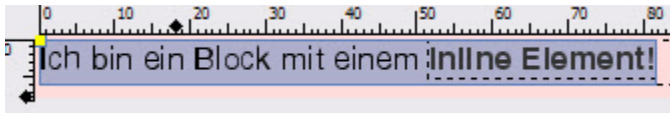
```
<fo:block>Ich bin ein Block!</fo:block>
```



2.2 fo:inline

Ein fo:inline wird genutzt um einen Teil eines Textes innerhalb eines fo:block anders zu formatieren oder hervorzuheben.

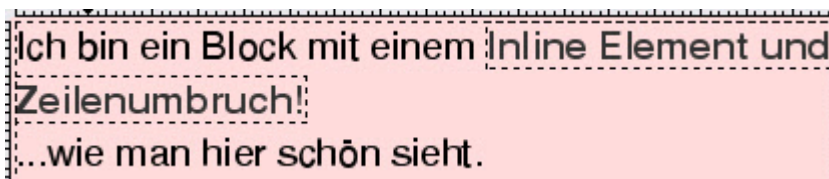
```
<fo:block>
  Ich bin ein Block mit einem
  <fo:inline color="g(80)" font-family="Helvetica" font-style="normal" font-
weight="bold">
    Inline Element!
  </fo:inline>
</fo:block>
```



2.3 fo:inline mit Zeilenumbruch

Mit dem Attribut `force-line-break-after="true"` kann man einen Zeilenumbruch am Ende des `fo:inline` Elements erzwingen.

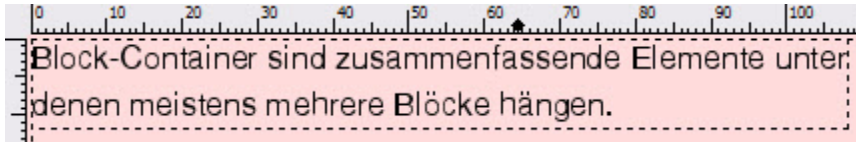
```
<fo:block>
  Ich bin ein Block mit einem
  <fo:inline color="g(80)" font-family="Helvetica" font-style="normal"
font-weight="bold" force-line-break-after="true">
    Inline Element und Zeilenumbruch!
  </fo:inline>
  ...wie man hier schön sieht.
</fo:block>
```



2.4 fo:block-container

Ein `fo:block-container` wird genutzt um auf einer block-Ebene mehrere `fo:block` in einem Bereich zusammenzufassen. Der Zweck darin ist eine logische Strukturierung der Elemente.

```
<fo:block-container>
  <fo:block>
    Block-Container sind zusammenfassende Elemente unter denen meistens
    mehrere Blöcke hängen.
  </fo:block>
</fo:block-container>
```



2.5 fo:table

Wer eine Tabelle in HTML kennt wird sich hier schnell zurechtfinden da sie ähnlich aufgebaut ist. Am Anfang der Tabelle müssen die Tabellenspalten (`fo:table-column`) definiert werden. Im Tabellenbody (`fo:table-body`) werden die Tabellenzeilen (`fo:table-row`) mit ihren Zellen (`fo:table-cell`) definiert. Es müssen so viele Zellen existieren wie es Spalten gibt. Eine Zelle lässt sich per Attribut aber über mehrere Zellen in einer Zeile spannen (`number-columns-spanned`).

```
<fo:table>
  <fo:table-column column-width="50%" />
  <fo:table-column column-width="50%" />
  <fo:table-body>
    <fo:table-row>
      <fo:table-cell>
        <fo:block>Tabellen sind relativ simpel aufgebaut.</fo:block>
      </fo:table-cell>
      <fo:table-cell background-color="green" />
    </fo:table-row>
    <fo:table-row background-color="yellow">
      <fo:table-cell>
        <fo:block>1</fo:block>
      </fo:table-cell>
      <fo:table-cell>
        <fo:block>default</fo:block>
      </fo:table-cell>
    </fo:table-row>
  </fo:table-body>
</fo:table>
```

```

</fo:table-cell>
</fo:table-row>
<fo:table-row>
  <fo:table-cell number-columns-spanned="2">
    <fo:block>
      ebenso kann man mit dem Attribut columns-span eine Anzahl an
      Tabellenzellen zu einer verbinden.
    </fo:block>
  </fo:table-cell>
</fo:table-row>
</fo:table-body>
</fo:table>

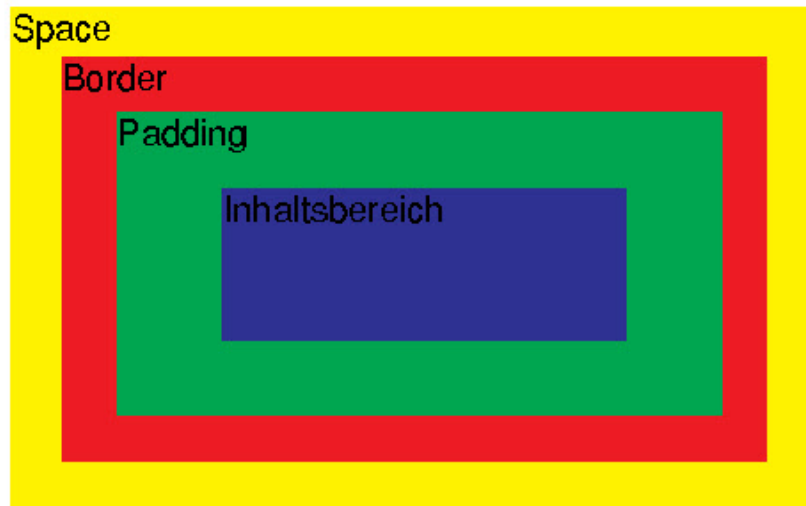
```

Tabellen sind relativ; simpel aufgebaut.	ebenso kann man mit dem Attribut columns-span: eine anzahl an Tabellenzellen zu einer verbinden.
1:	default

2.6 Boxmodell

Das XSL Formatting Objects Boxmodell hat im Prinzip die gleiche Funktionsweise wie das Boxmodell in CSS (Cascading Style Sheets). Für jedes Element in der Vorlage wird nach den FO-Regeln ein rechteckiger Bereich reserviert, der in dem sog. Boxmodell beschrieben ist. Dieser Bereich besteht aus dem eigentlichen Inhalt (Inhaltsbereich), einem Innenabstand zu dem Rahmen des Elements (Padding), dem Rahmen (Border) und dem Abstand zu anderen Elementen, die auf einer Seite zu finden sind (Space). Attribute wie z.B. font-size, font-family usw. werden auf die untergeordneten Elemente, sofern sie diese auch besitzen, vererbt (weitergegeben).

Das FO Boxmodell



3 UG Elemente

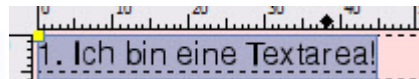
Benutzerinteraktionselemente

3.1 ug:textarea

Eine `ug:textarea` ist ein Element auf der Userpage, vergleichbar wie ein `input` Textfeld in HTML. Der Text der hier eingegeben wird erscheint zugleich auf der Ausgabe im PDF.

```
<fo:block>
  <fo:inline force-line-break-after="true">
    <ug:textarea label="1." uniqueId="2">
      1. Ich bin eine Textarea!
    </ug:textarea>
  </fo:inline>
```

1.

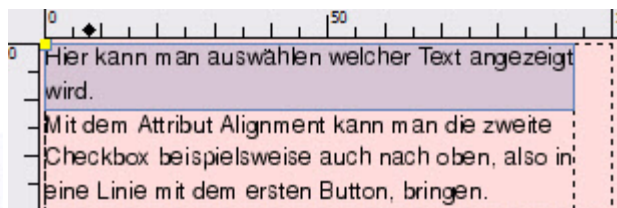


3.2 ug:buttongroup

Eine ug:buttongroup erscheint auf der Userpage als sog. Radiobuttons (ug:radiobutton). Der Benutzer hat die Wahl von <Anzahl Radiobuttons> aber kann sich nur für eine Option entscheiden.

```
<ug:buttongroup auto-reload="true" label="Auswahl Text "
uniqueId="btngrp1">
  <ug:radiobutton description="1. Text">
    <fo:block>
      Hier kann man auswählen welcher Text angezeigt wird.
    </fo:block>
  </ug:radiobutton>
  <ug:radiobutton description="2. Text">
    <fo:block>
      Kann ebenfalls mit anderen Elementen verknüpft werden.
    </fo:block>
  </ug:radiobutton>
</ug:buttongroup>
```

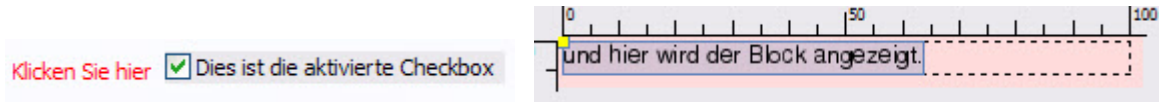
Auswahl Text 1. Text
 2. Text



3.3 ug:checkbox

Eine ug:checkbox ist eine Auswahloption für den Benutzer. Wenn der Haken in der Box gesetzt ist wird der fo:block der darunter „hängt“ gelayoutet.

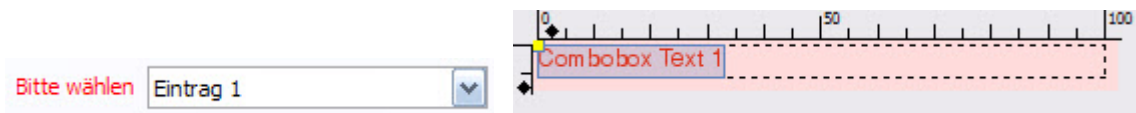
```
<ug:checkbox auto-reload="true" description="Dies ist die aktivierte
Checkbox" label="Klicken Sie hier" uniqueId="1">
  <fo:block>und hier wird der Block angezeigt.</fo:block>
</ug:checkbox>
```



3.4 ug:combobox

Eine `ug:combobox` bietet dem Benutzer eine Auswahl aus einer Liste von Einträgen/Items (`ug:item`). Je nachdem welches Item ausgewählt ist wird der entsprechende `fo:block` der darunter „hängt“ gelayoutet.

```
<ug:combobox auto-reload="true" label="Bitte wählen" uniqueId="1">
  <ug:item description="Eintrag 1">
    <fo:block color="red">Combobox Text 1</fo:block>
  </ug:item>
  <ug:item description="Eintrag 2">
    <fo:block color="blue">Combobox Text 2</fo:block>
  </ug:item>
</ug:combobox>
```



3.5 ug:title

Ein `ug:title` wird nur auf der Userpage als Text angezeigt und hat keine Auswirkung auf das Layout. Das Attribut `label` ist eine Art Beschreibung, der Inhalt ist der tatsächlich dargestellte Text.

```
<ug:title keep-together="always" label="Die Darstellung eines Titels:">
  dies ist ein Text auf der Editseite.
</ug:title>
```

Die Darstel...nes Titels: dies ist ein Text auf der Editseite.

3.6 ug:image

Mit einem `ug:image` kann man Bilder nur auf der Userpage einbinden und hat keine Auswirkung auf das Layout. Das Attribut `label` ist die Beschreibung des Bildes, `src` ist die Quelle.

```
<ug:image label="Bild 1" src="a5cabrio_front_4c.jpg" />
```



3.7 ug:invisible

Ein `ug:invisible` kann als unsichtbarer Anker auf der Userpage genutzt werden um Userpage Elemente neu zu organisieren bzw. positionieren.

```
<ug:invisible comment="Meine Checkbox (Anker)" uniqueId="anker1" />
```

4 UH Elemente

Flusskontrollelemente

4.1 uh:if

Das Element `uh:if` stellt im Prinzip einen praktischen Schalter dar, der Elemente in der Vorlage nach einer oder mehreren bestimmten Bedingung/-en an- und abschalten kann (sichtbar oder nicht sichtbar). In diesem Beispiel wird eine `ug:buttongroup` mit zwei `ug:radiobutton` Elementen eingesetzt. Der erste Radiobutton hat den Wert `on`, der zweite den Wert `off`. Das `uh:if` prüft nun im Attribut `test` durch die Funktion `input()` das Element mit dem Attribut `uniqueId=„optionen“` auf den Wert `„on“`. Wenn ja wird der darunter „hängende“ `fo:block` gelayoutet, wenn nein dann wird diese Ausgabe ignoriert. Die Bedingungen können mit dem booleanschen Algebra (`and`, `or`, ...) verknüpft werden.

```
<ug:buttongroup auto-reload="true" uniqueId="optionen">
  <ug:radiobutton description="Option 1" value="on" />
  <ug:radiobutton description="Option 2" value="off" />
</ug:buttongroup>

<uh:if test="input('optionen') = 'on'">
  <fo:block>Beispieltest</fo:block>
</uh:if>
```

4.2 uh:choose

Ein uh:choose Block ist im Prinzip das gleiche wie ein uh:if, nur dass man die einzelnen Entscheidungen durch ein uh:when prüft. Sollte keines davon zutreffen wird der Standardfall uh:otherwise verwendet.

```
<fo:block>
  <ug:textarea input-verification="required(integer)" uniqueId="text1">
    0
  </ug:textarea>
  -
  <ug:textarea input-verification="required(integer)" uniqueId="text2">
    20
  </ug:textarea>
</fo:block>

<uh:choose>
  <uh:when test="input('text1') = input('text2')">
    <fo:block>Die Textfelder sind gleich groß</fo:block>
  </uh:when>
  <uh:otherwise>
    <fo:block>Die Textfelder sind unterschiedlich groß.</fo:block>
  </uh:otherwise>
</uh:choose>
```

4.3 uh:attribute

Mit dem uh:attribute Element können Attribute des Elements, unter dem uh:attribute „hängt“, dynamisch verändert werden.

```
<ug:buttongroup auto-reload="true" uniqueId="btngrpcolor">
  <ug:radiobutton description="Text rot" value="rot" />
  <ug:radiobutton description="Text blau" value="blau" />
</ug:buttongroup>

<fo:block>
  <uh:attribute name="color">
    <uh:choose>
      <uh:when test="input('btngrpcolor') = 'rot'">red</uh:when>
      <uh:otherwise>blue</uh:otherwise>
    </uh:choose>
  </uh:attribute>
  Beispieltext
</fo:block>
```

4.4 uh:variable

Eine Variable speichert einen vorher definierten Wert. Dieser Wert lässt sich mit einem uh:copy-of Element auslesen, indem man im Attribut select den Namen der Variable, vorangestellt mit einem „\$“, angibt.

```
<uh:variable name="meinevariable">Beispieltext</uh:variable>

<fo:block><uh:copy-of select="$meinevariable" /></fo:block>
```

5 UP Elemente

Seitenelemente

5.1 up:page

Hiermit lässt sich eine PDF Seite und eine Userpage erstellen.

```
<up:page name="Seite 1" page-width="10cm">
  <fo:block-container><fo:block>text</fo:block></fo:block-container>
</up:page>
<up:page name="Seite 2" page-width="10cm">
  <fo:block-container><fo:block>text</fo:block></fo:block-container>
</up:page>
```

6 Errorhandling

6.1 Vorgegebene Fehlerprüfungen

Durch die input-verification lässt sich der eingegebene Text des Benutzers prüfen. Im folgenden Beispiel benötigt die ug:textarea einen Gleitkommawert. Die einzelnen Überprüfungsverfahren die im Attribut input-verification angegeben werden müssen findest du auch im Wiki.

```
<fo:block>
  <ug:textarea input-verification="type(float, *|2, points, 0)" label="Gleitkommazahl" uniqueId="1">sf</ug:textarea>
</fo:block>
```

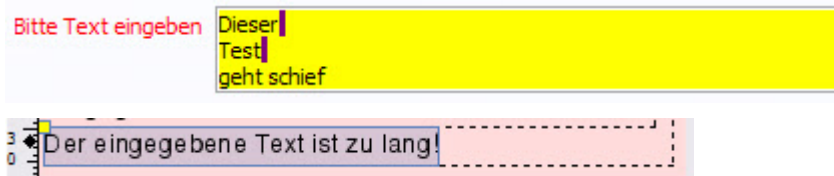
uf.error.type.float
Gleitkommazahl

6.2 Postprocessing Fehlerprüfungen

Mit der layout Überprüfung kann man nach dem Layout durch die Engine die Ausgabe auf dem PDF prüfen lassen. Nützlich ist dies z.B. wenn der eingegebene Text des Benutzers ein Bild nicht überdecken soll kann man den Absatz auf eine bestimmte Anzahl von Zeilen beschränken

(`maxlines(1)`). Das lässt sich natürlich auch mit der Höhe in Punkt machen (`maxheight(10pt)`).

```
<ly:choose>
  <ly:when verification="maxlines(1)">
    <fo:block>
      <ug:textarea label="Bitte Text eingeben" rows="3" uniqueId="1" />
    </fo:block>
  </ly:when>
  <ly:otherwise>
    <fo:block>Der eingegebene Text ist zu lang!</fo:block>
  </ly:otherwise>
</ly:choose>
```



6.3 eigens erstellte Fehlerprüfungen

Man kann Benutzereingaben aber auch gezielt mit komplexeren Mechanismen prüfen. Im Beispiel werden 2 Eingaben aus `ug:textarea` Elementen (geprüft auf Ganzzahl) miteinander summiert. Wenn das Ergebnis nicht 10 ist wird eine Fehlermeldung ausgegeben.

```
<fo:block>
  <ug:textarea cols="3" input-verification="required(integer)"
    uniqueId="text1" />
```

```
<ug:title alignment="right" keep-together="always">+</ug:title>+
<ug:textarea alignment="right" cols="3"
      input-verification="required(integer)" uniqueId="text2" />
<ug:title alignment="right" keep-together="always">=</ug:title>
<ug:title alignment="right" keep-together="always">10</ug:title>=
<uh:if test="(string-length(input('text1')) > 0) and
      (string-length(input('text2')) > 0)">
  <uh:copy-of select="number(input('text1')) +
      number(input('text2'))" />
</uh:if>
<uh:if test="number(input('text1')) +
      number(input('text2')) != 10">
  <ug:title keep-together="always" position="errorOnTop"
      style="attention">
    Das Ergebnis ist leider nicht 10.
  </ug:title>
</uh:if>
</fo:block>
```

7 Snippets

Ein Snippet ist eine Sammlung komplexerer Funktionen, Strukturen und Logik die dem Einpfleger abgenommen werden. Es repräsentiert immer die CI des Kunden. Durch Attribute lässt sich das Erscheinungsbild des Snippets ändern. Das sollte aber nur in abgesprochenen Ausnahmefällen geschehen.

```
<adb:adresskonfigurator columns="3" font-size="10pt" line-height="12pt"
  uniqueIdPart="1" />
```

Adressblock...nfigurator

Für demo001: Adrian Brand -...ht kein Logo zur Verfügung!

Anzahl Adressspalten: 2

Adressblock für demo001 Adrian Brand - Filiale 1

Adresse: Adrian Brand - Filiale 1
 Gebhardtstr. 5
 90762 Fürth
 Telefon 0911 / 756658-0
 Fax 0911 / 756658-88
 info@brand-ad.de
 www.brand-ad.de

Adressblock für demo002 Adrian Brand - Filiale 2

Adresse: Adrian Brand - Filiale 2
 Gebhardtstr. 5
 90762 Fürth
 Telefon 0911 / 756658-0
 Fax 0911 / 756658-88
 info@brand-ad.de
 www.brand-ad.de

