bx:recorddata

Das Tag `recorddata` gibt zusätzliche Daten innerhalb einer Containerliste (z.B. bx:containerloop oder bx:containerfilter) aus oder steuert die Ausgabe von Daten in verschiedenen Abhängigkeiten.

recorddata.id

```
<bx:recorddata.id [level="<n>" | parent | parent="<n>" | baseloop="<base>"] [{filter="<name>" | dynamicfilter="<name>"} [quiet]] />
```

Gibt die ID eines Datensatzes innerhalb einer Containerliste aus.

level

ID des DS in der angegebenen Ebene einer verschachtelten Containerliste wird zurückgegeben, wobei "0" die äußerste Containerliste ist und alle weiteren inneren Containerlisten jeweils um 1 erhöht werden (0,1,2,...).

parent / parent=...

hat die selbe Wirkung wie `level`, nur dass hier von der aktuellen Containerliste mit "0" ausgegangen wird und für jede weiter übergeordnete Containerliste 1 addiert wird (...,2,1,0). `parent` ist identisch mit `parent="1"`.

baseloop

übergeordneten Containerliste direkt mit Namen ansprechen, vorausgesetzt, es wurde auch ein Name vergeben (siehe Bx:containerloop und Bx:containerfilter).

filter / dynamicfilter

es kann ein JSP-Baustein als Filter angegeben werden

quiet

Fehlermeldungen des Filters wird unterdrückt

recorddata.creationdate

```
<bx:recorddata.creationdate  [pattern=""] [locale=""] />
```

Wandelt die ID in Datum um.

pattern

Ausgabe-Pattern, Standard: dd.MM.yyyy

locale

Sprachausgabe, Standard: de

recorddata.containerid

```
<bx:recorddata.containerid />
```

Gibt die Container-ID aus (ab Version 2.6.2).

recorddata.titel

```
<bx:recorddata.titel />
```

Gibt den Titel des aktuellen Datensatzes einer Containerliste aus. Der Titel ist der Selbe, der auch in der Containerdefinition als Anzeigeelement angegeben wurde.

recorddata.if

```
<bx:recorddata.if [first | last] [not] > Inhalt </bx:recorddata.if>
<bx:recorddata.if [request="{name}" | baseloop="{name}" | id="{ID}" | id="attribute:{Name}" |
id="clipboard:{Name}" ] [not]> Inhalt </bx:recorddata.if>
<bx:recorddata.if ... [true="{Ausdruck}"] [false="{Ausdruck}"] />
```

Mit diesem Tag kann auf einen Request-Parameter der Seite reagiert werden, der die ID eines Datensatzes der Containerliste enthält. Außerdem kann auf die DS-ID reagiert werden und die erste bzw. letzte Position ermittelt werden.

first | last

Abfrage, ob es der erste oder letzte DS der Datenliste ist

request

Name des Request-Parameter, der die gesuchte ID enthält. `{name}` kann auch einen Platzhalter in der Form `request="[platz]"` enthalten. In diesem Fall wird `[platz]` durch den Wert des Requestparameters `platz` ersetzt.

baseloop

Angabe

id

Angabe der gesuchten IDs (mit Komma oder Leerzeichen getrennt), entweder direkt oder mit einem Präfix. Abfrage, ob Dummy-Datensatz: `id=""`. Ab V2.5.9: Abfrage dummy mit `#null`

not

kehrt die Bedingung um

true | false

Im Tag ohne Inhalt wird der Wert von `true` bei Übereinstimmung ausgegeben, sonst der Wert von `false`.

Beispiele

```
<bx:containerfilter.Mitglieder pool="Mitglieder" force="list">
  <div style="border-top:1px solid black;<bx:recorddata.if last true="border-bottom:1px solid black;" false="margin-bottom:20px;"/>">
    <bx:recordfield.Titel/>
  </div>
</bx:containerfilter.Mitglieder>
```

Beim letzten DS der Liste soll die Anzeigeliste mit einem Strich abgeschlossen werden. Ansonsten wird ein Abstand zwischen den einzelnen Divs angezeigt.

```
<bx:containerfilter.Mitglieder pool="Mitglieder" force="list">
  <bx:recorddata.if request="[platz]">......<bx:recorddata.if>
</bx:containerfilter.Mitglieder>
```

Browseraufruf:

http://www.musterseiten.de/menuepunkt1/?platz=id2&id1=xxxxxxx&id2=yyyyyyyy&id3=zzzzzzzzz Es wird die id2 zur Überprüfung herangezogen.

Infos über Verhalten / Buginfo

v2.5.8 Inhalt wird ausgeführt, wenn die ID des Datensatzes der ID im Parameter entspricht:

```
<bx:recorddata.if id="{Batix-ID}" [not]> ... </bx:recorddata.if>
<bx:recorddata.if id="{Batix-ID}" [true="{angezeigter Text}"] [false="{angezeigter Text}"]/>
```

Inhalt wird ausgeführt, wenn man sich gerade in einem Dummy-Datensatz befindet:

```
v2.6.2 neu <bx:recorddata.if id="" [not]> ... </bx:recorddata.if>
v2.6.3 neu <bx:recorddata.if id="attribute:undefiniert" [not]> ... </bx:recorddata.if>
v2.6.3-v2.6.4.1** Bug** bei <bx:recorddata.if id="" [not]>
v2.7.1 neu <bx:recorddata.if id="#null" [not]> ... </bx:recorddata.if>
```

Inhalt wird ausgeführt, wenn die ID des Datensatzes einer der IDs im Parameter entspricht (oder #null):

```
v2.6.3 <bx:recorddata.if id="{Batix-ID1},{Batix-ID2}" [not]> ... </bx:recorddata.if>
v2.7.1 <bx:recorddata.if id="{Batix-ID1},{Batix-ID2},#null" [not]> ... </bx:recorddata.if>
```

Bug v2.6.3-v2.6.4.1: `<bx:containerloop force=single dummy><bx:recorddata.if id="request:recordid">`

'test.htm' [ok, ist dummy] 'test.htm?recordid=' [Dummy wird nicht mehr erkannt] 'test.htm?recordid=1234567ABCD' [ok, kein dummy] in v2.6.2 und v>2.6.4.1 geht es wieder

recorddata.currentindex

```
<bx:recorddata.currentindex [add="{n}"] [baseloop="{base}"] />
```

Gibt die Nummer des aktuellen Datensatzes einer Containerliste aus beginnend mit "1" für den Ersten. Auch wenn die Ausgabe über Blätterfunktionen eingeschränkt ist, wird immer die Nummer über die gesamte Liste ausgegeben, so als wären keine Beschränkungen da.

add

Bewirkt, dass immer die Ausgabe der aktuellen Nummer plus den Wert von `add` ausgegeben wird.

Standard ist 1, d.h., weil bei 0 angefangen wird, wird bei Nicht-Angabe des Add-Parameters bei 1 angefangen. Soll bei 0 angefangen werden, muß add="0" sein (heißt: zum Startwert 0 wird 0 hinzugezählt anstatt (Standard)1).

baseloop

Index einer übergeordneten Liste abfragen

recorddata.pageindex

```
<bx:recorddata.pageindex [add="<n>"] />
```

funktioniert im Prinzip genau wie `recorddata.currentindex`, nur, dass hier die Nummerierung Seitenweise erfolgt. Das heißt, auch bei einer Blätterfunktion werden nur die aktuell sichtbaren

Datensätze durchnummeriert, beginnend bei "1", unabhängig von der gesamten Anzahl an Datensätzen in der Containerliste.

recorddata.ifcontains

Diese Funktion ist entweder noch in Entwicklung oder sehr speziell. Bitte ggf. nachfragen!

recorddata.ifmatch

(zum Löschen vorgesehen)

Vergleicht den aktuellen Datensatz in der Eltern-Containerschleife mit dem Datensatz in der Großeltern-Containerschleife und führt den Taginhalt nur aus, wenn beide aktuellen Datensätze den selben Containerdatensatz darstellen.

recorddata.cols

```
<bx:recorddata.cols num="{n}" equals="{n}" [not] > ... </bx:recorddata.cols>
<bx:recorddata.cols num="{n}" equals="{n}" true="{Ausdruck}" false="{Ausdruck}" />
<bx:recorddata.cols num="{n}" [col]{n}="<Ausdruck>" />
```

Dieses Kommando zeigt Daten abhängig von der Spaltennummer der Containerschleife an.

num

Anzahl an Spalten in der Ausgabe

equals

aktuelle Position in der Spalte

not

kehrt Bedingung um

true | false

bei Zutreffen bzw. Nichtzutreffen der Bedingung wird der jeweils angegebene Text ausgegeben

col[n]

es kann jede einzelne Spalte angesprochen werden (die Spalten können hier entweder 1, 2, 3 oder col1, col2, col3 heißen)

Beispiele

```
<div class="liste">
  <div class="item"<bx:recorddata.cols num="5" equals="5"> style="margin-
right:0;"</bx:recorddata.cols>>
   <a href="detail.htm?id=<bx:recorddata.id/>"><bx:recordfield.Titel/></a>
  </div>
  <bx:recorddata.cols num="5" equals="5"><div class="clear"></div></bx:recorddata.cols>
</div>
```

Der letzte DS in der Reihe, die aus 5 Elementen besteht, soll keinen rechten Abstand haben. Danach soll ein Clear-Div stehen.

```
<div class="liste">
  <div class="item" style="<bx:recorddata.cols num="5" equals="5" not true="margin-right:10px"
false="margin-right:0"/>;">
   <a href="detail.htm?id=<bx:recorddata.id/>"><bx:recordfield.Titel/></a>
  </div>
  <bx:recorddata.cols num="5" equals="5" true="<div class="clear"></div>"/>
</div>
```

Gleiches Beispiel wie oben, allerdings mit true|false

```
<div class="liste">
  <div class="item" style="background-color:<bx:recorddata.cols num="7" col1="red"
col2="orange" col3="yellow" col4="green" col5="lightskyblue" col6="blue"
col7="darkviolett"/>;">
   <a href="detail.htm?id=<bx:recorddata.id/>"><bx:recordfield.Titel/></a>
  </div>
  <bx:recorddata.cols num="7" equals="7" true="<div class="clear"></div>"/>
```

```
</div>
```

Die Spalten sollen Regenbogenfarben als Hintergrund bekommen (issn dämliches Beispiel, ich weiß - Vorschläge willkommen)

recorddata.active

```
<bx:recorddata.active [not]> ... </bx:recorddata.active>
<bx:recorddata.active [true="{Ausdruck}"] [false="{Ausdruck}"] />
```

führt den Ausdruck abhängig vom Aktivstatus des Containerdatensatzes aus (Häkchenfeld im DS im Container ganz oben. Man kann also auch inaktive DS anzeigen lassen. Auch dieses Tag kann entweder mit Inhalt (und optional mit `not`) versehen oder mit den Parametern `true` und/oder `false` verwendet werden.

Blätterfunktionen

Beispiel für eine Blätterfunktion

recorddata.nav

```
<bx:recorddata.nav [object="{bez}"] [labelname="{bez}"] [param="{bez}"] [indexparam="{bez}"]
[filename="{bez}"]> ... </bx:recorddata.nav>
```

Dieses Tag erzeugt keine Ausgabe, sondern fasst Parameter zusammen, die zur Navigation in Listen verwendet werden. Diese Parameter werden bei Tags, die .nav unterstützen nicht noch einmal aufgeführt! Beim Seitenblättern muss ein **Anzeigelimit** (Parameter `max`) bei der entsprechenden Liste definiert sein!

object / labelname

Name bzw. den Labelname des Containerloops

param

Name des Request-Parameters mit der Datensatz-ID (bei Blättern Einzeldatensatz). Standard: `param`

indexparam

Name des Request-Parameter, der die Index-Position enthält (bei Listenblättern). Standard: `index`

filename

(nur für alte Container) bezeichnet die Datei, auf die verlinkt wird (z.B. bei `.previouslist`).

Beispiele

Listenblättern

```
<bx:containerfilter.Voegel pool="Tiere" force="list" max="10">
  <a href="detail.htm?vid=<bx:recorddata.id/>"><bx:recordfield.Titel/></a><br>
</bx:containerfilter.Voegel>
...
<bx:recorddata.nav object="Voegel" indexparam="pos">
  <bx:recorddata.previouslist/> | <bx:recorddata.nextlist/>
</bx:recorddata.nav>
```

Blätterfunktion in einer Liste. Im Requestparameter `pos` steht dann beim Blättern die entsprechende Index-Position innerhalb der Liste. `nextlist` generiert einen Link, der ca. so aussieht: url/navpunkt/liste.htm?pos=11

DS-Blättern

```
<bx:containerfilter.Voegel pool="Tiere" idfield="vid" force="single">
  <a href="detail.htm?vid=<bx:recorddata.id/>"><bx:recordfield.Titel/></a><br>
</bx:containerfilter.Voegel>
...
<bx:recorddata.nav object="Voegel" param="vid">
  <bx:recorddata.previouselement/> | <bx:recorddata.nextelement/>
</bx:recorddata.nav>
```

DS-Blättern auf der Detailseite. Im Requestparameter `vid` steht die DS-ID. `previouselement` generiert einen Link, der ca. so aussieht: url/navpunkt/detail.htm?vid=1111111111

Labelverwendung

```
<bx:containerfilter label="Voegel" pool="Tiere" force="list" max="10" name="Klasse" type="8"
value="33333333333">
  <a href="detail.htm?vid=<bx:recorddata.id/>"><bx:recordfield.Titel/></a><br>
</bx:containerfilter>
...
<bx:recorddata.nav labelname="Voegel" indexparam="pos">
  <bx:recorddata.previouslist/> | <bx:recorddata.nextlist/>
</bx:recorddata.nav>
```

Wenn der Container keine Bezeichnung hat (z.B. bei Bedingung im Tag), dann kann auch ein Label zur Zuordnung verwendet werden.

Seitenblättern

recorddata.previouslist

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

```
<bx:recorddata.previouslist [type="link | url | index | if"] [plain] [show] [encode="plain |
html"] [not] /> <!-- Link wird erzeugt, Standardtext: zurück -->
<bx:recorddata.previouslist Parameter siehe oben> Linktext </bx:recorddata.previouslist>  <!--
eigener Linktext kann ausgegeben werden -->
```

Gibt einen Link zur vorherigen Seite der Listenansicht aus, falls es eine vorherige Seite gibt. Nicht alle Parameter sind gleichzeitig verwendbar (Standard ist "link").

type="link"

gibt einen Link aus (`<a href="...">zurück</a>`) verwendbare Parameter: show

type="url"

gibt nur den relativen Pfad aus (datei.htm?...) verwendbare Parameter: encode

type="index"

gibt nur den neuen Index aus

type="if"

gibt den Inhalt des Tags aus, wenn die vorherige Seite existiert; verwendbare Parameter: not

plain

setzt `type` auf `"url"`

show

zeigt den Text auch dann an, wenn keine Vorgängerseite existiert

encode

kodiert jedes Zeichen des Links bei Angabe von `html` als &#nnn; (ist standardmäßig gesetzt, um dies abzustellen `encode="plain"` angeben)

not

kehrt den Effekt bei `type="if"` um

recorddata.nextlist

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

(identisch mit .previouslist, nur dass hier die nächste Seite verlinkt wird und der Linktext standardmäßig "weiter" lautet)

```
<bx:recorddata.nextlist [type="link | url | index | if"] [plain] [show] [encode="plain |
html"] [not] />        <!-- Link wird erzeugt, Standardtext: weiter-->
<bx:recorddata.nextlist Parameter siehe oben> Linktext </bx:recorddata.nextlist
>                                  <!-- eigener Linktext kann ausgegeben werden -->
```

recorddata.firstlist

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

(identisch mit .previouslist, nur dass hier die erste Seite verlinkt wird und der Linktext standardmäßig "erste Seite" lautet)

```
<bx:recorddata.firstlist[type="link | url | index | if"] [plain] [show] [encode="plain |
html"] [not] /> <!-- Link wird erzeugt, Standardtext: erste Seite-->
```

```
<bx:recorddata.firstlist Parameter siehe oben> Linktext </bx:recorddata.firstlist>  <!--
eigener Linktext kann ausgegeben werden -->
```

recorddata.lastlist

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

(identisch mit .previouslist, nur dass hier die letzte Seite verlinkt wird und der Linktext standardmäßig "letzte Seite" lautet)

```
<bx:recorddata.lastlist[type="link | url | index | if"] [plain] [show] [encode="plain | html"]
[not] /> <!-- Link wird erzeugt, Standardtext: letzte Seite-->
<bx:recorddata.lastlist Parameter siehe oben> Linktext </bx:recorddata.lastlist>  <!-- eigener
Linktext kann ausgegeben werden -->
```

Datensatzweise blättern

recorddata.previouselement

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

```
<bx:recorddata.previouselement [type="link | url | if | id | title"] [encode="html | plain"]
[not] [show] [loop]/>      <!-- Link wird erzeugt, Standardtext: zurück-->
<bx:recorddata.previouselement Parameter siehe oben> Linktext </bx:recorddata.previouselement
>                          <!-- eigener Linktext kann ausgegeben werden -->
<bx:recorddata.previouselement
type="object"><bx:recordfield.Titel/></bx:recorddata.previouselement >
```

Diese Funktion generiert einen Link zum vorherigenElement der Liste (nur bei Detailansicht).

type="link"

(Standard) produziert einen klickbaren Link wenn das entsprechende Element existiert (immer anzeigen mittels show)

type="url"

gibt die relative URL aus

type="if"

gibt den Inhalt des Tags nur aus, wenn das Element existiert (kann mit not umgekehrt werden)

type="id"

zeigt nur die ID des Datensatzes an

type="title"

zeigt nur den Name des Datensatzes an (so wie er in der Verwaltung angezeigt wird)

type="object"

ab V 2.6.2: Tag wird als offenes Tag benutzt und ermöglicht, direkt auf das erste Element (Datensatz) zuzugreifen, um z.B. den Titel anzuzeigen. Sonst wird nur die ID übergeben und man hat keine Möglichkeit, die DS-Felder auszulesen.

encode

`encode="html"` (nur bei `type="url"` unterstützt) bewirkt, dass jedes Zeichen in &#nnn; umgewandelt wird (ist standardmäßig gesetzt, um dies abzustellen `encode="plain"` angeben).

loop

signalisiert, dass es sich um eine Endlosliste handelt, d.h. Vorgänger vom ersten Element ist das letzte Element

recorddata.nextelement

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen. (identisch mit .previouselement, nur dass hier das nächste Element verlinkt wird und der Linktext standardmäßig "weiter" lautet, ebenfalls kann mit dem Parameter `loop` signalisiert werden, dass es sich um eine Endlosliste handelt, d.h. Nachfolger vom letzten Element ist das erste Element)

```
<bx:recorddata.nextelement [type="link | url | if | id | title"] [encode="html | plain"] [not]
[show] [loop]/>      <!-- Link wird erzeugt, Standardtext: weiter-->
<bx:recorddata.nextelement Parameter siehe oben> Linktext </bx:recorddata.nextelement
>                            <!-- eigener Linktext kann ausgegeben werden -->
<bx:recorddata.nextelement type="object"><bx:recordfield.Titel/></bx:recorddata.nextelement >
```

recorddata.firstelement

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen. (fast identisch mit .previouselement, nur dass hier das erste Element verlinkt wird und der Linktext standardmäßig "erster" lautet)

```
<bx:recorddata.firstelement [type="link | url | if | id | title"] [encode="html | plain"]
[not] [show] />      <!-- Link wird erzeugt, Standardtext: erster-->
<bx:recorddata.nextelement Parameter siehe oben> Linktext </bx:recorddata.nextelement
>                        <!-- eigener Linktext kann ausgegeben werden -->
<bx:recorddata.nextelement type="object"><bx:recordfield.Titel/></bx:recorddata.nextelement >
```

recorddata.lastelement

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen. (identisch mit .previouselement, nur dass hier das letzte Element verlinkt wird und der Linktext standardmäßig "letzter" lautet)

```
<bx:recorddata.firstelement [type="link | url | if | id | title"] [encode="html | plain"]
[not] [show] />      <!-- Link wird erzeugt, Standardtext: letzter-->
<bx:recorddata.nextelement Parameter siehe oben> Linktext </bx:recorddata.nextelement
>                        <!-- eigener Linktext kann ausgegeben werden -->
<bx:recorddata.nextelement type="object"><bx:recordfield.Titel/></bx:recorddata.nextelement >
```

Ausgabe von aktuellen Parametern und Positionen (oder wie könnte man das nennen???)

recorddata.total

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

```
<bx:recorddata.total/>
```

Gibt die gesamte Anzahl an Datensätzen aus.

recorddata.startindex

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

```
<bx:recorddata.startindex/>
```

gibt die Nummer des ersten Datensatzes aus, der auf der aktuellen Seite angezeigt wird bzw. die Nummer des aktuellen Detaildatensatzes innerhalb der angegebenen Filterung und Sortierung.

recorddata.endindex

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

```
<bx:recorddata.endindex/>
```

gibt die Nummer des letzten Datensatzes auf der aktuellen Seite aus.

recorddata.listindex

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

```
<bx:recorddata.listindex max="<n>" />
```

Mit diesem Kommando erhält man die Indexnummer, auf die man bei einem Link zurück zur Übersicht verlinken muss. `max` ist Pflicht.

Wenn z. B. Datensatz 14 angezeigt wird und auf der Listseite 6 Datensätze angezeigt werden, gibt dieses Tag 12 aus.

recorddata.currentpage

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

```
<bx:recorddata.currentpage/>
```

Dieses Tag gibt die Nummer der aktuellen Seite (beginnend bei 1) aus.

recorddata.totalpages

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

```
<bx:recorddata.totalpages/>
```

Dieses Tag gibt die Anzahl der Anzeigeseiten aus.

recorddata.navlist / recorddata.navdetail

```
<bx:recorddata.navlist [max="<n>"] [boundary="<text>"] />
<bx:recorddata.navlist [max="<n>"] [boundary="<text>"]> ... </bx:recorddata.navlist>
<bx:recorddata.navdetail [max="<n>"] [boundary="<text>"] />
<bx:recorddata.navdetail [max="<n>"] [boundary="<text>"]> ... </bx:recorddata.navdetail>
```

Mittels `.navlist` wird eine verlinkte Liste an benachbarten Seiten (`.navlist`) bzw. Detailseiten (`.navdetail`) ausgegeben.

max

Wenn `max` nicht angegeben wird oder "0" ist, werden alle Seiten angezeigt

boundary

legt das Trennzeichen zwischen den Zahlen fest (Standard: " | ")

Beispiel: Eine Liste besteht aus 23 Datensätzen und füllt wegen `max="3"` (Parameter der Containerliste, nicht von `.navlist`) 8 Seiten. Wir befinden uns auf Seite 5 und `max` von `.navlist` ist auf `"2"` gesetzt. Folgendes wird ausgegeben (mit entsprechender Verlinkung): 3 | 4 | 5 | 6 | 7

Um die Verlinkung individueller zu gestalten verwenden Sie die zweite Form dieses Tags. Der Inhalt wird dann für jede anzuzeigende Seite ausgegeben. Sie können eine Reihe von Unterelementen einbinden:

recorddata.navlistpage / recorddata.navdetailpage

```
<bx:recorddata.navlistpage/>
<bx:recorddata.navdetailpage/>
```

gibt die Nummer der Seite bzw. Detailseite aus

recorddata.navlistindex

```
<bx:recorddata.navlistindex/>
```

gibt die Indexnummer zurück, die zum Aufruf der im Schleifendurchlauf repräsentierten Seite benötigt wird

recorddata.navdetailid

```
<bx:recorddata.navdetailid/>
```

gibt die Datensatz-ID zurück, die zum Aufruf der im Schleifendurchlauf repräsentierten Detailseite benötigt wird

recorddata.navlistpath / recorddata.navdetailpath

```
<bx:recorddata.navlistpath [encoding="html | plain"] />
<bx:recorddata.navdetailpath [encoding="html | plain"] />
```

gibt den Pfad zurück, über den die im Schleifendurchlauf repräsentierte Seite bzw. Detailseite aufgerufen werden kan Mit `encoding` wird das Ausgabeformat gesteuert. Standard ist `"html"` (Zeichen werden durch &#nnn; ersetzt), wenn Sie normalen Text ausgeben wollen setzen sie `encoding` auf `"plain"`.

recorddata.navlistcurrent / recorddata.navdetailcurrent

```
<bx:recorddata.navlistcurrent [not]> Inhalt </bx:recorddata.navlistcurrent>
<bx:recorddata.navdetailcurrent [not]> Inhalt </bx:recorddata.navdetailcurrent>
```

führt den Inhalt nur aus, wenn die anzuzeigende Zahl die aktuelle Seite bzw. Detailseite ist. `not` kehrt das Verhalten um.

recorddata.navlistfirst / recorddata.navdetailfirst

```
<bx:recorddata.navlistfirst [not]> Inhalt </bx:recorddata.navlistfirst>
<bx:recorddata.navdetailfirst [not]> Inhalt </bx:recorddata.navdetailfirst>
```

führt den Inhalt nur aus, wenn die anzuzeigende Zahl die erste Seite bzw. Detailseite ist. `not` kehrt das Verhalten um.

recorddata.navlistlast / recorddata.navdetaillast

```
<bx:recorddata.navlistlast [not]> Inhalt </bx:recorddata.navlistlast>
<bx:recorddata.navdetaillast [not]> Inhalt </bx:recorddata.navdetaillast>
```

führt den Inhalt nur aus, wenn die anzuzeigende Zahl die letzte Seite bzw Detailseite ist. `not` kehrt das Verhalten um.

recorddata.navlink

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

```
<bx:recorddata.navlink move="<n>" />
<bx:recorddata.navlink move="<n>"> Linktext </bx:recorddata.navlink>
```

Dieses Tag schreibt einen Link zu einer vorhergehenden oder nachfolgenden Seite (falls mehrere Seiten vorhanden sind).

<n>

ganze Zahl

`move` gibt dabei die Schrittweite an (kann auch negativ sein). Falls die erste Form verwendet wird, ist der Linktext standardmäßig "...".

recorddata.detaillink

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

```
<bx:recorddata.detaillink move="{n}" />□□□□□□□<!-- Linktext standardmäßig "..." -->
<bx:recorddata.detaillink move="{n}"> Linktext </bx:recorddata.detaillink>
```

Dieses Tag schreibt einen Link zu einem vorhergehenden oder nachfolgenden Datensatz (falls mehrere Dantesätze vorhanden sind).

move

Schrittweite (kann auch negativ sein)

recorddata.navdetailobject

Hinweis: Dieses Tag kann innerhalb von `recorddata.nav` benutzt werden und von dort alle relvanten Parameter übernehmen.

```
<bx:recorddata.navdetailobject> ... </bx:recorddata.navdetailobject>
```

repräsentiert den jeweiligen Containerdatensatz. Funktioniert wie `recorddata.nextelement type="object"` (oder die anderen artverwanden Tags)

Beispiele

Beispiel 1

```
<bx:recorddata.navdetail max="3">
  <bx:recorddata.navdetailobject><bx:recordfield.Titel/></bx:recorddata.navdetailobject>
</bx:recorddata.navdetail>
```

Die Wirbeltierklassen (Säugetiere, Vögel, Amphibien, Reptilien, Fische) werden ausgegeben und in Klammern die Anzahl der Tiere aus dem Container "Tiere", die mit der jeweiligen Klasse verbunden sind.

recorddata.meta

```
<bx:recorddata.meta (key="{metakey}" | name="{metakey}") [date [pattern="{format}"]] />
```

Gibt den Inhalt eines Metawertes aus.

key | name

Name einer Metaeigenschaft

date

Datum soll angezeigt werden

pattern

Datumsformatierung (Standard ist "EEEEEE, d. MMMMMM yyyy")

recorddata.katcount

```
<bx:recorddata.katcount pool="{liste}" field="<feld>" />
<bx:recorddata.katcount pool="{liste}" field="<feld>"> Text </bx:recorddata>
```

Dieses Tag gibt die Anzahl der Datensätze aus einer Recordliste zurück, welche dort über ein Feld mit dem Datensatz im aktuellen Schleifendurchlauf verknüpft sind. Ebenso kann auch ein Text ausgegeben werden, falls die Anzahl größer als Null ist.

pool

ID oder Alias des Containers

field

Name des Feldes (muss vom Typ Einzelverknüpfung sein)

Beispiele

Beispiel 1

```
<bx:containerfilter.Wirbeltierklassen pool="Klassen">
  <bx:recordfield.Titel/> ( <bx:recorddata.katcount pool="Tiere" field="Klasse"/> )<br>
</bx:containerfilter.Wirbeltierklassen >
```

Die Wirbeltierklassen (Säugetiere, Vögel, Amphibien, Reptilien, Fische) werden ausgegeben und in Klammern die Anzahl der Tiere aus dem Container "Tiere", die mit der jeweiligen Klasse verbunden sind.

recorddata.empty

```
<bx:recorddata.empty [object="{name}"] [not] > Inhalt </bx:recorddata.empty>
```

Der Inhalt wird nur ausgeführt, wenn in der Schleife keine Datensätze zum Anzeigen vorhanden sind. Wenn das Tag nicht innerhalb von `<bx:recorddata.nav>` verwendet wird müssen der Parameter `object` mit dem Namen der zugehörigen Containerschleife und evtl. auch der Parameter `label` mit angegeben werden. *ab Version 2.5.6*

object

Name der Containerschleife

not

kehr die Funktion um

Beispiele

Beispiel 1

```
<bx:recorddata.empty object="Newsliste">z. Z. keine Newsbeiträge
vorhanden</bx:recorddata.empty>
<bx:recorddata.empty object="Newsliste" not><bx:recorddata.total/> News
vorhanden:</bx:recorddata.empty>

oder

<bx:recorddata.nav object="Newsliste">
  <bx:recorddata.empty>z. Z. keine Newsbeiträge vorhanden</bx:recorddata.empty>
  <bx:recorddata.empty not><bx:recorddata.total/> News vorhanden:</bx:recorddata.empty>
<bx:recorddata.nav>
```

Beispiel 2

```
<bx:containerfilter.Suche_Tier pool="Tiere">
  ...
</bx:containerfilter.Suche_Tier>

<bx:recorddata.empty object="Suche_Tier">
  Leider wurde kein Tier mit dem gesuchten Begriff gefunden.
</bx:recorddata.empty>
```