Actions

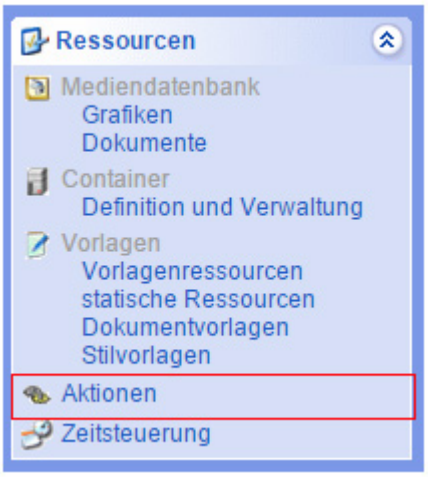
Alle Action-Bausteine (39 Seiten).

- Actions – Übersicht
- Batix Quelltext ausführen
- Benutzergruppe anlegen
- Bild hochladen
- Captcha Abfrage
- CMS-Seite versenden
- Container-Filteraction
- Container kopieren
- Containerdaten aus Email importieren
- Containerdaten speichern
- Containerdatensatz löschen
- Containerinhalt validieren
- Dokument hochladen
- Excel-Action
- Formular als Mail versenden
- Formulardaten validieren
- Intranet Login
- Intranet Logout
- Intranet-User anlegen/ändern
- JSP-Baustein ausführen
- Newsletter – Abo abmelden
- Newsletter – Abo ändern
- Newsletter – Abonnentenlogin
- Newsletter aktivieren
- Newsletter – Newsletter abonnieren
- Newsletter – Passwort vergessen (1)
- Newsletter – Passwort vergessen (2)

- Passwort des angemeldeten Users ändern
- Shop – Bestellung versenden
- Shop – Hinzufügen zu Warenkorb
- Shop – Löschen im Warenkorb
- Shop – Zusatzgebühr zum Warenkorb
- Sicheres Formular überprüfen (SecureformAction)
- Suche im Web
- Validierungsergebnis ausgeben
- Vorhandene Bedingung abfragen
- Weiterleitung
- Wert in Session speichern
- XHTML 2 PDF

Actions – Übersicht

Aktionen realisieren im Zusammenwirken mit Dokumentvorlagen an bestimmten Menüpunkten erforderliche Funktionalitäten - z.B. das Speichern von Formulardaten, das Versenden von Emails, das Anmelden bei einem Newsletter u.v.m.



Übersicht

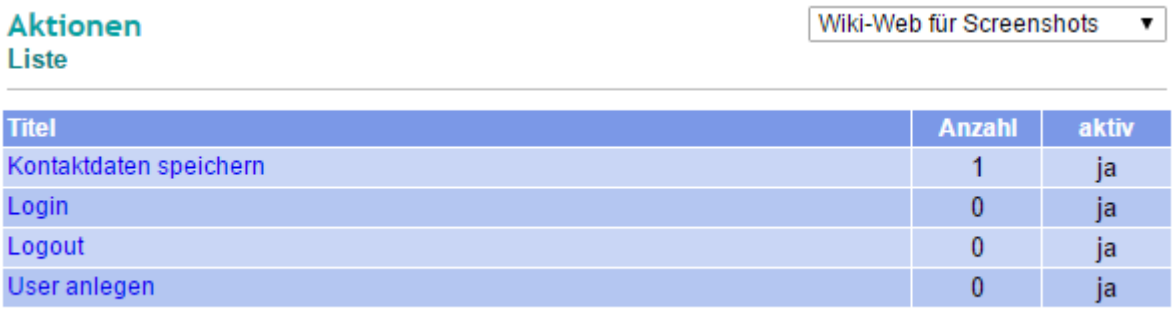
Kategorie	Actions
Speicherung	Bild hochladen Containerdaten aus Email importieren Dokument hochladen Formulardaten validieren
Container	Container kopieren Container-Filteraction Containerdaten speichern Containerdatensatz löschen Containerinhalt validieren Validierungsergebnis ausgeben
Benutzer und Zugang	Benutzergruppe anlegen Intranet Login Intranet Logout Intranet-User anlegen/ändern Passwort des angemeldeten Users ändern

Kategorie	Actions
Versenden von E-Mails	CMS-Seite versenden Formular als Mail versenden
Ausführung von Code	Batix Quelltext ausführen JSP-Baustein ausführen
Sonstiges	Captcha Abfrage Excel-Action Sicheres Formular überprüfen (SecureformAction) Suche im Web Vorhandene Bedingung abfragen Weiterleitung Wert in Session speichern XHTML 2 PDF
Veraltet	Newsletter – Abo abmelden Newsletter – Abo ändern Newsletter – Abonnentenlogin Newsletter aktivieren Newsletter – Newsletter abonnieren Newsletter – Passwort vergessen (1) Newsletter – Passwort vergessen (2) Shop – Bestellung versenden Shop – Hinzufügen zu Warenkorb Shop – Löschen im Warenkorb Shop – Zusatzgebühr zum Warenkorb

Abgesehen von oben angeführten Actionbausteinen können spezielle individuelle Aktionen unter Voraussetzung entsprechender Kenntnisse problemlos angelegt werden.

Aktionsliste

Wenn Sie den Eintrag Ressourcen->Aktionen anklicken, erhalten Sie etwa folgende Seite im Arbeitsbereich:

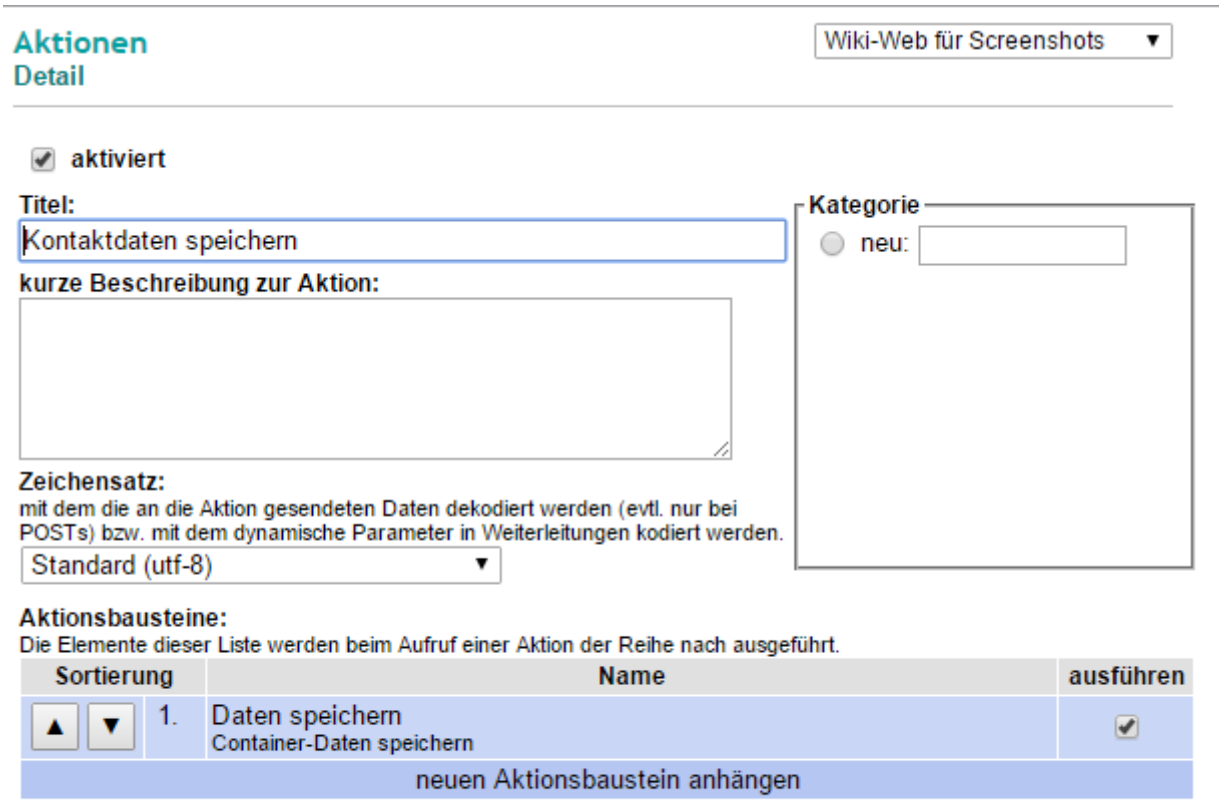


Es werden Ihnen alle momentanen Aktionen mit folgenden Informationen angezeigt:

- der Titel der Aktion
- die Anzahl der in der Aktion vorhandenen Aktionsbausteine
- der Status der Aktion (Nur aktive Aktionen und Aktionsbausteine können angewendet werden. Deaktivierte Aktionen und Aktionsbausteine verursachen bei Verwendung einen Fehler.)

Details der Aktion

Klicken Sie auf den Titel einer Aktion, um deren Bausteine einzusehen und allgemeine Einstellungen zur Aktion vorzunehmen.

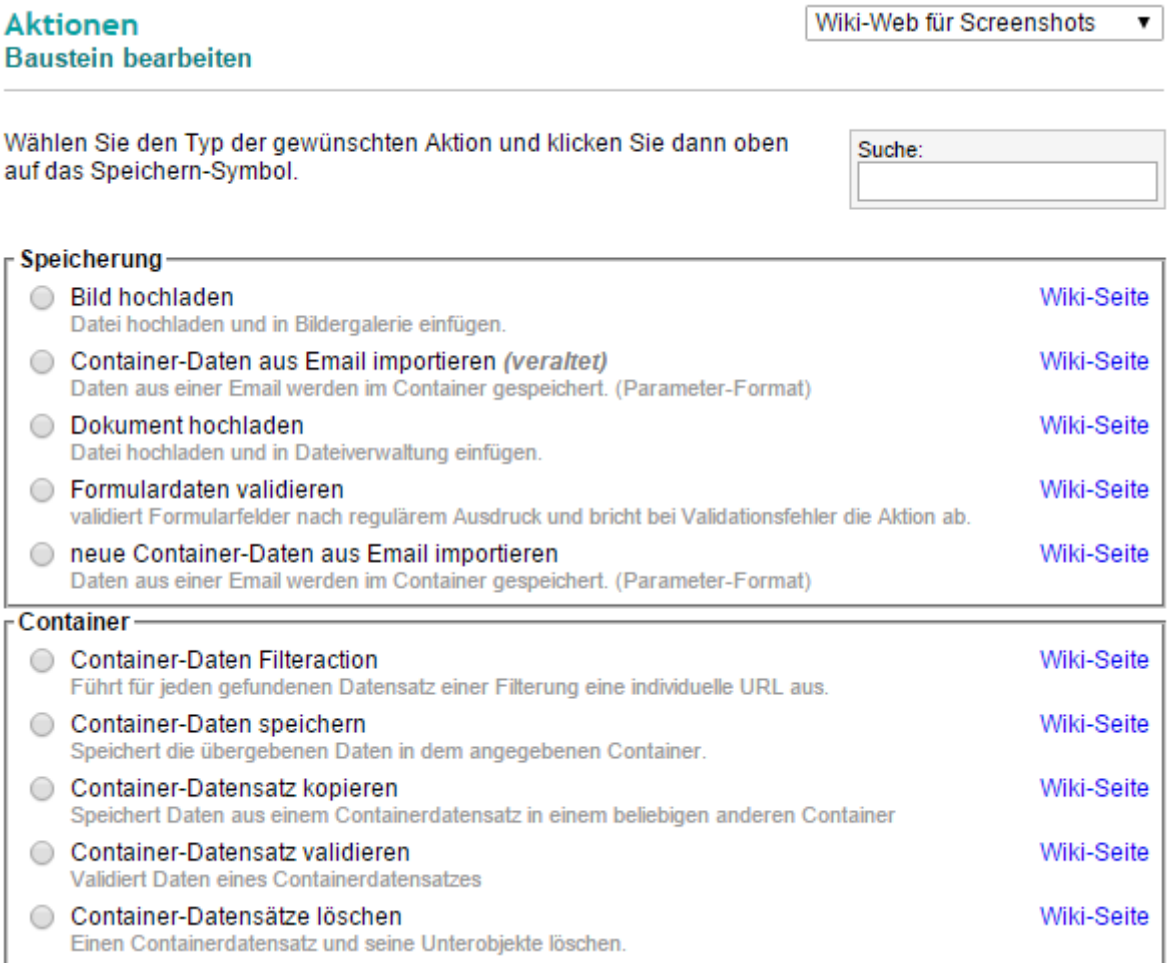


Hier können Sie:

- die Aktion aktivieren/deaktivieren
- einen Titel sowie eine kurze Beschreibung der Aktion eingeben (wird auf der Übersichtsseite und anderen Bestandteilen des CMS als Erläuterung ausgegeben)
- die Aktionsbausteine ändern:
  - Mittels "neuen Aktionsbaustein anhängen" generieren Sie assistengesteuert einen neuen Baustein (s.u.)
  - Mit einem Klick auf die Zeile wechseln Sie zur Detailansicht des gewählten Aktionsbausteines (s.u.).
  - Aktivieren/Deaktivieren Sie das Kontrollkästchen "ausführen", um die Nutzung des Bausteines festzulegen.
- In der Toolbox finden Sie die durch das CMS verwendete interne ID der Aktion, sowie eine Auflistung der Verwendungsarten und -orte.

Neuen Aktionsbaustein anhängen

Um eine neue Aktion anzulegen, klicken Sie - wenn Sie sich in der Übersichtsseite aller Aktionen befinden - auf den Button  /  in der Symbolleiste. Hier müssen Sie noch die Felder ausfüllen und die Aktion abspeichern ( / ). Anschließend werden Sie aufgefordert, einen neuen Aktionsbaustein hinzuzufügen. Klicken Sie auf diese Zeile und Sie sehen folgenden Assistenten:



Klicken Sie auf den Auswahlbutton neben der gewünschten Aktion und anschließend zum Speichern auf / .

Details des Aktionsbausteins

Hier können Sie:

- den Baustein aktivieren/deaktivieren
- einen Titel sowie eine Kurzbeschreibung festlegen
- die Felder für die Inhalte des Aktionsbausteins bearbeiten (diese unterscheiden sich je nach Baustein erheblich)

Batix Quelltext ausführen

Führt Batix Quelltext aus

Werte können mit `bx:pagedata.setscriptattribute` an den nächsten Action-Baustein übergeben werden.

```
<bx:pagedata.setscriptattribute name="<name>"> Inhalt </bx:pagedata.setscriptattribute>
```

Ausgabe im Log: <bx:tools.log>...</bx:tools.log>

Benutzergruppe anlegen

Dieses Action erzeugt eine neue Benutzergruppe.

Parameter

groupname	Feld mit Gruppennamen Parametername des Feldes, in dem der Benutzergruppenname steht (nur angeben, wenn Anlegen einer Gruppe)
field-id	Parametername des Gruppen-ID-Feldes Feldname mit dem die Benutzergruppen-ID übergeben wird (nur angeben, wenn Änderung einer Gruppe)
activate	Gruppe aktiv schalten, Standard: ja
attribute-groupid	Name für neue Gruppen-ID Attributname, in dem die ID der neuen Gruppe gespeichert werden soll

Bild hochladen

Diese Aktion verschiebt eine hochgeladene Datei in die Dateiverwaltung. Dabei ist es wichtig, dass im Upload-Formular `enctype="multipart/form-data"` gesetzt ist.

Parameter

fileparam	gibt den Request-Parameter an, der die hochgeladene Datei enthält
folderid	falls das Bild in einen bestimmten Ordner der Bildergallerie gespeichert werden soll, kann hier die ID des Ordners angegeben werden
folderparam	eine Alternativmöglichkeit um das Bild in einen bestimmten Ordner zu speichern, die ID wird aus dem hier angegebenen Request-Parameter ausgelesen (Standard-Parametername ist "folderid")
folderpath	die dritte Möglichkeit zur Angabe des Ordners, hier wird der Name des Ordners angegeben
targetweb	um das Bild in ein anderes Projekt/Web zu speichern, hier die ID des Webs angeben
titel	mit diesem Parameter kann der Titel des Bildes festgelegt werden
titelparam	um den Titel aus einem Request-Parameter auszulesen, hier den Parametername angeben
autor	hiermit kann der Autor des Bildes direkt festgelegt werden
autorparam	der Autor kann alternativ auch über einen Parameter angegeben werden

Bemerkungen

Bei Werten, die durch verschiedene Parameter festgelegt werden können, gilt folgende Reihenfolge der Überprüfung (es wird dabei abgebrochen, sobald ein Wert gefunden wurde):

- **Ordner**: `folderid` -> `folderparam` (wenn nicht angegeben, dann Standardwert "folderid") -> `folderpath`
- **Titel**: `titel` -> `titelparam`

- **Autor**: `autor` -> `autorparam`

In den Metadaten des Bildes werden unter "uploader" und "author" der eingeloggte Benutzer (falls er eingeloggt ist) sowie unter "uploaderIP" die IP des Uploaders eingetragen.

Multi-Upload (ab V 2.6.3)

Mehrere Bilder werden gleichzeitig mit HTML5 über ein Multiupload hochgeladen und in einem verknüpften Container gespeichert. **Achtung**! Da die Bilder im Hintergrund gespeichert werden, muß es schon einen Eltern-Datensatz geben, d.h., der Upload-Actionbaustein kommt nach dem Speicher-Actionbaustein.

url	weiterleitende URL (z. B. zu einem Save-Action)

Beispiel

Detailseite

```
<form action="send.act" method="post" enctype="multipart/form-data">
  ...
  <input name="Bilder" type="file" multiple>
  <input type="submit">
</form>
```

Formular bauen, file-input-Feld muß den Parameter `multiple` haben

Action bauen: der Upload-Actionbaustein kommt nach dem Speicher-Actionbaustein

Angaben beim Upload-Actionbaustein:

- bei `fileparam` muß "Bilder" stehen (Name im Formular)
- bei `Scriptausführung je Bild: (url)` steht die Weiterleitung zu einem Speicheraction, das die einzelnen Bilder im Untercontainer speichert, z.B. `/www/beispielweb/menupunkt/savePic.act?Bild\=[[Bilder]]&Datum\=[[Datum]]&Bildergalerie\=[[galerieID]]` Im anschließenden savePic.act wird in einem Bildercontainer das Bild mit Bildergalerie-Verknüpfung(Elterndatensatz-Verknüpfung) und Datum gespeichert

Captcha Abfrage

Diese Aktion prüft, ob der zum CAPTCHA-Bild eingegebene Text korrekt ist. Dazu muß auf der Seite ein Bild mit dem Namen SimpleCaptcha.jpg, das die visuelle Ausgabe enthält, und ein Input-Feld mit `name="captcha"` eingebunden sein.

Parameter

requestField	der Requestparameter, der die Antwort enthält (standardmäßig "j_captcha_response")
successpage	die Seite, auf die weitergeleitet wird, wenn die Antwort richtig ist
failpage	die URL, die bei falscher Antwort aufgerufen wird
appendRequest	gibt an, ob der aktuelle Request an die Folgeseite weitergegeben wird

Ein verfremdetes Bild mit einer Zeichenfolge wird ausgegeben und muß in ein Eingabefeld geschrieben werden.

Quelltext dazu:

```
<div>
  <img src="/SimpleCaptcha.jpg<bx:pagedata.sessionurl/>" alt="Captcha-Bild">
  <bx:pagedata.request name="fehler" value="1">Die eingegebene Zeichenfolge war nicht
korrekt.</bx:pagedata.request>
  <input type="text" name="captcha" value="">
</div>
```

Die Felder werden anlog des erstellten HTML-Designs ausgefüllt.

> Wenn das Captcha in einem IFrame verwendet wird, muß beim Bild und bei der Form `<bx:pagedata.sessionurl/>` angehängt werden.

Beispiel: `<form action="save.htm<bx:pagedata.sessenurl/>">...</form>`

CMS-Seite versenden

Eine normale Seite im Projekt wird gelesen und dann als HTML-Email versendet.

Da diese Aktion auf Formular als Mail versenden aufbaut, sind alle dort angegebenen Parameter auch hier verwendbar, mit den unten aufgeführten Änderungen bzw. Neuerungen.

Parameter

mailFrom	Die Email-Adresse, unter der die Mail versandt wird. Falls beim Versenden ein Fehler auftritt oder auf diese Mail geantwortet wird, geht bei dieser Adresse eine Email ein. Bei Kontakt könnte hier auch email angegeben werden, dann kann auf diese Mail direkt geantwortet werden (nicht mehr verwenden). Besser ist jedoch Reply-To (Beschreibung ganz unten) zu verwenden.
mailFromName	Der im Email-Programm angezeigte Absendername (auch möglich z.B. [[Name]] [[Vorname]])
mailTo	Der Empfänger der Email
subject	Der Betreff der Mail
file	Der virtuelle Dateiname des zu versendenden HTML-Designs
appendRequest	gibt an, ob der Request, welcher der Aktion übergeben wurde, an die URL der CMS-Seite angehängt werden soll (wenn diese z.B. eine ID enthält)

Datei anhängen

Um Email-Anhänge zu erzeugen kann an einer beliebigen Stellen des Email-Templates der Platzhalter für den Anhang hinzugefügt werden.

Die Syntax ist folgende:

```
<![ATTACHMENT[Dokument-ID]]>
```

oder

```
<![ATTACHMENT[Anhang-Dateiname|URL für Anhangsdaten]]>
```

Um die Anhänge zu parsen muß im Action bei weitere Eigenschaften eine Einstellung eingetragen werden.

```
parse-attachments=true
```

Beispiel

```
<![ATTACHMENT[<bx:pagedata.request name="pdf" />]]>
```

wenn die Dokument-ID mit dem Request-Parameter "pdf" übergeben wird

```
<![ATTACHMENT[export_<bx:tools.datum/>.css|exporttemplate.txt]]>
```

wenn die Ausgabe des Templates namens exporttemplate.txt aus dem selben Navigationspunkt als Anhang mit aktuellen Datum im Namen sesendet werden soll.

```
<bx:containerfilter.Sammlung pool="..." max="10">
 <![ATTACHMENT[<bx:recordfield.Dokument type="id"/>]]>
</bx:containerfilter.Sammlung>
```

Hängt alls gefilterten Dokumente des Containers an die Mail an.

Reply-To

Ein Reply-To Header kann im Freieingabe-Feld gesetzt werden (ganz unten: Freigabe weiterer Eigenschaften):

```
reply-to=[[Kontakt_Mail]]
```

Container-Filteraction

Mit diesem Action kann man einen Container nach bestimmten Filter-Kriterien durchsuchen und darauf reagieren.

Parameter

list	die ID der Datenliste (Container), die gefiltert werden soll (Pflichtfeld)
xml	die Definition des Filters (Pflichtfeld)
url	die für jeden gefundenen Datensatz aufgerufene URL. Das kann ein weiteres Action sein, z.B. ein Speicheraction. (Pflichtfeld) Kann auch frei gelassen werden, wenn der Filter nur auf Erfolg oder Mißerfolg prüfen soll, z.B. ob eine bestimmte E-Mail schon vorhanden ist oder nicht.
Test-Modus (für Entwicklung gedacht)	Im Testmodus werden keine URLs ausgeführt, die Actionausführung wird nach dem Action abgebrochen und stattdessen eine Infoseite angezeigt, auf der aufgelistet wird, welche Urls ausgeführt würden.
emptyresulturl	eine URL die nur dann ausgeführt wird, wenn keine URL im Filter ausgeführt wurde. Die Ausführung erfolgt im Hintergrund, z.B. wenn eine bestimmte Email noch nicht angelegt ist, dann kann sie damit angelegt werden und die Action-Abfolge wird nicht unterbrochen.
successurl	wenn ausgefüllt und mindestens eine URL ausgeführt wurde, wird die Ausführung nach diesen Action abgebrochen und an die angegebene Seite weitergeleitet.
failurl	wenn ausgefüllt und keine URLs ausgeführt wurden, wird die Ausführung nach diesen Action abgebrochen und an die angegebene Seite weitergeleitet.
request-method	falls der Request mittels POST geschickt werden soll, muss dieser Parameter den Wert **POST** enthalten

spezielle Platzhalter bei successurl und failurl

(ab v2.6) bei "*weiteren Eigenschaften*" können Parameter definiert werden, die statistische Werte in Actionattribute mit den dort definierten Namen schreiben:

record-counter-param=filterdurchläufe	Anzahl der DS, die durch die Filterung gekommen sind
processed-counter-param=versuche	Anzahl der URL-Aufrufe, es fallen evt. DS weg z.B. durch bx:if
success-counter-param=erfolge	Anzahl der erfolgreichen URL-Aufrufe
error-counter-param=fehler	Anzahl der nicht erfolgreichen URL-Aufrufe, aufgerufene Seite/Action bringt Fehler

In anschließenden Actions kann auf die Werte über den angegebenen Name in eckigen Klammer zugegriffen werden (z.B. auswertung.htm?anzahl=[[anzAufruf]]&fehler=[[anzFehler]])

diese Werte kann man dann z. B. im Success-URL-Feld benutzen:

```
erfolg.htm?report=[[versuche]]+Datensaetze+verarbeitet,+davon+[[fehler]]+mit+Fehlern
```

oder z. B. auch in einem folgenden Weiterleitungsaction oder Saveaction

Bemerkungen

In der URL können Batix-Tags benutzt werden, z.B. bx:pagedata um der Aktion übergebene Parameter weiterzuleiten oder bx:recordfield bzw. bx:recorddata um Informationen zum aktuellen Datensatz zu übergeben.

Die Parameter der URL müssen selbst URL-Encoded werden. Dazu kann man `<bx:tools.urlencode>Text</bx:tools.urlencode>` benutzen.

Im URL-Feld können beliebig Zeilenumbrüche und Leerzeichen gesetzt werden, um den Code entwicklerfreundlich zu formatieren. In der ausgeführten URL werden diese Zeichen vorher entfernt. Leerzeichen, die in Parametern übergeben werden sollen, müssen vorher kodiert werden.

Container kopieren

Diese Aktion kopiert Daten aus einem Container in einen anderen. Felder mit gleichen Namen und Typ werden automatisch übernommen. Für weitere Felder kann über eine Konfiguration eine Zuordnung geschaffen werden. Über solch eine Konfigurationseinstellung können auch mehrere Felder in ein Feld hineinkopiert werden.

Parameter

`source-listid`	gibt die ID der Quell-Liste an
`target-listid`	dies ist die ID der Ziel-Liste
`idfield`	der Name des Request-Parameters, der die ID des zu kopierenden Datensatzes enthält
`config`	die Kopierzuordnung in XML

Auf die ID des erzeugen Kopie-DS kann man in einem weiteren Actionbaustein über das Action-Attribute `copyrecord` zugreifen.

Weitere Eigenschaften: `include-child-lists=true` (es werden die Untercontainer mit kopiert - wie in der Verwaltung - ab V2.6.6) `attribute-recordid=copyrecord` (es kann statt `copyrecord` für die ID des neuen DS ein anderer Name definiert werden)

XML-Syntax

Die Syntax der Kopierzuordnung folgt diesem Muster:

```
<rules>
  <field name="Titel" source="Name"/>
  <field name="Datum"><![CDATA[<bx:recordfield.Datum_Beginn pattern="dd.MM.yyyy"/><bx:if> -
<bx:recordfield.Datum_Ende pattern="dd.MM.yyyy"/></bx:if>]]></field>
  <field name="Alter"><![CDATA[<bx:pagedata.request name="neuesAlter"/>]]></field>
  <field name="Kategorie/Titel" source="Titel"/>
</rules>
```


name	Feldname des Zielfelds
source	Feldname des Quellfelds
<![CDATA[...]]>	zu speichernder Inhalt als Tag-Inhalt es können auch Tags wie `bx:recordfield` verwendet werden
Feld/Unterfeld	Unterelemente einer Einzelverknüpfung ansprechen

Beispiele

Die XML Datei hat folgendes Format:

Beispiel Titel

```
<rules>
  <field name="Datum" source="Datum_Beginn"/>
  <field name="Flat_Ausgabe"><![CDATA[
Beschreibung:
<bx:recordfield.Beschreibung type="plain"/>
Bemerkungen:
<bx:recordfield.Bemerkungen type="plain"/>
]]></field>
  <field name="Kategorie" source="Kategorie/Titel"/>
  <field name="Zeitstempel" source=""/>
</rules>
```

Das Haupt-Tag heißt `<rules>`. Dazwischen dürfen beliebig viele `<field>`-Einträge stehen, die das Pflicht-Attribut `name` haben müssen, das den Feldnamen des Zielcontainers enthält.

Bei geschlossenen `<field/>`-Tag muß in einem Parameter `source` der Name des Feldes im Quellcontainer angegeben werden. Mit "/" kann auch ein Feld hinter einer Einzelverknüpfung aufgerufen werden.

Wenn ein Feld aus dem Quellcontainer nicht übernommen werden soll, kann bei `name` der Feldname und `source=""` angegeben werden. (Hier im Beispiel würde nicht der Inhalt des Timestamp-Feldes des Quellcontainers kopiert werden und so ein aktueller Timestamp im Zielcontainer erstellt werden)

Wenn `<field>` geöffnet wird, können Batix-Tags benutzt werden, um den Inhalt von Textfeldern im Zieldatensatz aus mehreren Feldern des Quelldatensatzes zusammenzusetzen. An <bx>-Tags können nur einige verwendet werden, z. B. bx:pagedata um der Aktion übergebene Parameter weiterzuleiten oder bx:recordfield bzw. bx:recorddata um Informationen zum aktuellen Datensatz

zu übergeben. Der Inhalt sollte in einem CDATA-Block eingeschlossen sein, wenn er XML-Sonderzeichen (z. B. "<", "&") enthalten könnte.

Containerdaten aus Email importieren

Diese Aktion speichert Container- und Untercontainerdatensätze. Die nötigen Daten werden von einem Mail-Server abgerufen.

singlemode	wenn gewählt wird mit jedem Aufruf der Aktion nur eine Mail gelesen, ansonsten alle
pop3server	der Server, von dem die Mails geholt werden
username	der Benutzername zur Anmeldung am Server
password	das Passwort zum Benutzername
delmail	sollen gelesene Mails gelöscht werden?
listid	ID des Containers, in den die Daten gespeichert werden sollen
recordidfield	Name des Tags, der die ID eines vorhandenen Datensatzes enthält (Standard ist "recordid")
dokumentContainerId	ID des Containers, in den die der Mail angehängten Dateien gespeichert werden sollen
dokumentFolder	ein Request-Parameter, der angibt, in welches Verzeichnis der Dateiverwaltung, die in der Mail angehängten Dateien gespeichert werden sollen, kann auch als Eigenschaft der Aktion festgelegt werden mittels documentFolder
documentField	Name des Feldes im Dokument-Container, dem die Datei zugewiesen werden soll
nextpage	auf diese Seite wird nach Fertigstellung weitergeleitet, diese URL kann auch im Request-Parameter backfile stehen
active	gespeicherten Datensatz aktivieren?
timestamp	gibt ein Feld an, in dem der Bearbeitungszeitpunkt des Imports im Container festgehalten werden kann

weitere Eigenschaften:

appendrecordid	falls gesetzt, gibt diese Eigenschaft den Parameter an, über den der Folgeseite die ID des gespeicherten Datensatzes mitgeteilt wird

targetweb	um die Daten in ein anderes Web zu speichern, hier die ID des Webs angeben

Containerdaten speichern

iese Aktion speichert einen Containerdatensatz, dessen Felder im Request übergeben wurden.

Parameter

nextpage	auf diese Seite wird nach Fertigstellung weitergeleitet, diese URL kann auch im Request-Parameter **backfile** stehen
active	soll der erzeugte Datensatz aktiviert werden?
appendrecordid	der Name eines zusätzlichen Parameters, welcher der Folgeseite übergeben wird und die ID des erzeugten Datensatzes enthält
listid	ID der Liste, in die der neue Datensatz gespeichert wird
recordidfield	der Name des Parameters, der die ID des zu überschreibenden Datensatzes enthält
timestamp	der Name eines Datum-Feldes, in dem der Zeitpunkt der Erstellung festgehalten wird
createnew	gibt an, ob ein neuer Datensatz erstellt werden darf (nur wenn keine recordid übergeben wurde)

weitere Eigenschaften:

`update-active=j`
 	
`update-active=n`
	Datensatz aktivieren oder deaktivieren
	es ist auch möglich den Wert via Platzhalter zu übergeben: `update-active=[[paramName]]`
	ein leerer oder nicht vorhandener Parameter führt zur Deaktivierung

`log-meta=false`	speichert keine MetaDaten (erstellt bzw. letzte Änderung)

Bemerkungen

Durch bestimmte Name und Verwendung der Request-Parameter kann der Ablauf dieser Aktion beeinflusst und angepasst werden.

Möchte man einen Datensatz aktivieren bzw. deaktivieren, so ist der Request-Parameter "activ" mit dem Wert "j" bzw."n" zu übergeben.

einfaches Feld speichern

Hier entspricht der Name des Parameters dem Feldname. Veraltet: Bei Verknüpfungen werden "Name.LINKLIST" und "Name.LINKRECORD" verwendet.

Mehrfachverknüpfung

Um bei einer Mehrfachverknüpfung einzelne DS hinzufügen oder löschen zu können, wir Name.MODE angegeben (set, add oder delete).

Beispiel

```
Aus Hobby-Liste entfernen:
  <form action="save.act" method="post">
    <input type="hidden" name="Hobbies.MODE" value="delete">
    <bx:recordfield.Hobbies>
      <input type="checkbox" name="Hobbies" value="<bx:recorddata.id/>">
<bx:recordfield.Titel/>
    <bx:recordfield.Hobbies>
  </form>

Weitere Hobbies hinzufügen:
  <form action="save.act" method="post">
    <input type="hidden" name="Hobbies.MODE" value="add">
    <bx:containerfilter.Hobbies pool="Hobbies" force="list" orderby="Titel">
      <input type="checkbox" name="Hobbies" value="<bx:recorddata.id/>">
<bx:recordfield.Titel/>
    </bx:containerfilter.Hobbies>
  </form>
```

Checkbox: Wert "unchecked" speichern

Um den Zustand einer Checkbox in einem Wahrheitswert zu speichern, erstellt man ein hidden-Field für den nein-Wert, das den Name der Checkbox.CHECKBOX hat. Beispiel: `<input type="checkbox" name="agb" value="j"><input type="hidden" name="agb.CHECKBOX" value="n">`

Radiobox: Wert "null" in einem Wahrheits-Feld speichern

Um den Zustand null - also nicht ja und nicht nein - zu speichern, gibt man als Wert "#null" an. Beispiel: `<input type="checkbox" name="feldname" value="#null">`

in Untercontainer speichern

Es kann "Untercontainerfeld/Feld" verwendet werden, um Felder in einem Untercontainer anzusprechen. Ein bestimmter Untercontainer kann über seine ID angesprochen werden: "Untercontainerfeld.UNTERCONTAINERID/Feld". Es können auch mehrere DS auf einen Rutsch in Untercontaienr gespeichert werden: "Untercontainerfeld.new-n/Feld"

Beispiele

```
<bx:containerfilter.Firmen pool="Firmen">
  <bx:recordfield.Telefonnummern>
    <input type="text" name="Telefonnummern.<bx:recorddata.id/>/Nummer"
value="<bx:recordfield.Nummer/>">
  </bx:recordfield.Telefonnummern>
</bx:containerfilter.Firmen>
```

Untercontainer "Telefonnummern" im Container "Firmen", speichern von Änderungen in einem Rutsch

mehrere Datensätze speichern

Um mehrere Datensätze zu ändern kann "DATENSATZID/Feld" benutzt werden.

mehrere Datensätze erstellen

Weiterhin können mit einer Aktion mehrere Datensätze angelegt werden, die Parameter müssen dann wie folgt benannt werden: "new-n/Feld", wobei "n" durch eine Zahl ersetzt wird (also z.B. "new-1/Titel" und "new-2/Titel").

Verknüpftes Bilder in einem Bild-Field wieder löschen

Beispiele

```
<form>
  <input type="file" name="bild" value="">
  <input type="checkbox" name="bild.archivid" value="">
</form>
```

Feld wird mit dem Checkbox-Inhalt überschrieben.

Containerdatensatz löschen

Es wird ein bestimmter Datensatz mitsamt allen Feldern und Untercontainern gelöscht.

Parameter

`table`	die ID der Liste, aus der ein Datensatz gelöscht werden soll
`nextpage`	auf diese Seite wird nach Fertigstellung weitergeleitet, diese URL kann auch im Request-Parameter backfile stehen
`ID-Field`	der Request-Parameter, der die ID des zu löschenden Datensatzes enthält (Standard ist "recordid")

Containerinhalt validieren

Diese Aktion liest einen Datensatz aus einem Container und validiert ihn anhand von Kriterien.

Parameter

idfield	der Request-Parameter, der die ID des Datensatzes enthält
list	die Liste, aus welcher der Datensatz entnommen wird (Container-ID oder Containername)
config	die Validierungsregeln alternativ auch Dateiname, wo die Regeln drin stehen

XML-Syntax der Validierungsregeln

```
<model>
<bind nodeset="Name" index="alias" required="true()" match="..." constraint="..."
type="xsd:anyURI"/>
<bind nodeset="Feld1" required="Feld2 = 'Wert'"/>
<bind nodeset="Location" .../>
</model>
```

Es kann auch ein Pfad angegeben werden, wo dann die Regeln hinterlegt sind (für mehr Flexibilität, da kann man die Regeln im Container speichern oder so)

Attribute

nodeset

gibt das Feld im Container an, welches validiert werden soll

index

ist ein Bezeichner für diesen Test (beliebiger Text)

required

`required="true()"` - Feld muß gefüllt sein

`required="Feld2 = 'Wert'"` - Feld muß in Abhängigkeit eines anderen Feldwertes gefüllt sein. Wichtig sind die Leerzeichen vor und nach dem `=`. Allerdings keine Leerzeichen vor und nach Feldname bzw. Wertangabe.

Beispiele

Feld muß gefüllt sein

```
...
<bind nodeset="Zaehlernummer" required="true()"/>
...
```

Die Zählernummer (z.B. eines Stromanschlusses) muß immer angegeben werden (im Container muß im entsprechenden Textfeld etwas stehen).

Feld muß in Abhängigkeit eines anderen Feldwertes gefüllt sein

```
...
<bind nodeset="Zaehlernummer" required="Neuanschluß = false"/>
...
```

Die Zählernummer muß angegeben werden, wenn im Formular das Häkchen bei "Neuanschluß" nicht gesetzt wurde (im Container angelegtes Häckchenfeld darf nicht angehakt sein).

Feld muß in Abhängigkeit eines anderen Feldwertes gefüllt sein

```
...
<bind nodeset="Zaehlernummer" required="Neuanschluß = 'nein'"/>
...
```

Gleiches Beispiel wie oben, nur daß das Feld "Neuanschluß" ein Textfeld ist und das Wort "nein" drin steht.

match

kann einen regulären Ausdruck enthalten (für einzeilige und mehrzeilige Textfelder)

Beispiel: E-Mail-Validierung

```
...
<bind nodeset="Email" index="2" match="^[_a-zA-Z0-9-](\.{0,1}[_a-zA-Z0-9-])*@([_a-zA-Z0-9-
]{2,}\.){0,}[_a-zA-Z0-9-]{3,}(\.[_a-zA-Z]{2,4}){1,2}$"/>
...
```

Der im Feld "Email" gespeicherte Wert muß dem angegebenen Regex ensprechen (dieser Regex ist nur einer von vielen).

constraint

kann eine Bedingung angegeben werden, die der Feldinhalt erfüllen muß (z.B. `. > 18:00`, zur Zeit `< >` für Datumsangaben, `< > =` für Zahlen und `=` für einzeiligen Text). Details siehe Abschnitt **constraint** unten.

Beispiel: Datum prüfen

```
...
<bind nodeset="Geburtsdatum" index="2" constraint="Geburtsdatum < #today"/>
...
```

Das Geburtsdatum muß vor heute liegen.

type

gibt den xsd-Typnamen an (zur Zeit noch nicht unterstützt)

constraint

Mit `constraint` werden Gültigkeitsbedingungen des Feldinhalts angegeben. Diese sind je nach Typ des Feldes unterschiedlich. Der Wert im Parameter `constraint` muß folgende Form haben:

```
<bind nodeset="..." constraint="{Containerfeld} {Operator} {Vergleichswert}" />
```

Wichtig sind die Leerzeichen, die die 3 Werte trennen.

Hier ist die Liste der unterstützten Bedingungen, geordnet nach Feldtyp:

Textfelder (einzeilig, mehrzeilig)

{Vergleichswert}: Gültig sind:

- Werte in Apostroph für Vergleich mit festen Wert
- Name eines anderen Feldes

- Spezialwerte
  - #null" (Feldinhalt ist null)
  - "#empty" (Feldinhalt ist Leerstring)
  - "#nodata" (Feldinhalt ist null oder Leerstring) (ab v2.5.8)

{Operator}:

- `=` prüft, ob Feld dem Testwert entspricht
- `!=` Prüft, ob Feld dem Testwert nicht entspricht (ab v2.5.8)
- `MATCH` Prüft, ob Feld dem regulären Ausdruck in Testwert entspricht (ab v2.5.8)

Datumsfelder

{Vergleichswert}

- Als Vergleichswert ist bisher (v2.5.8) nur ein fester Wert möglich. Er sollte wegen Kompatibilität zu zukünftigen Erweiterungen in Apostrophs eingeschlossen werden.
- Zeitfelder müssen in der Form "{Stunden}:{Minuten}[:{Sekunden}]" angegeben sein
- Datumsfelder müssen in der Form "{Tag}.{Monat}.{Jahr}" angegeben sein
- Datum-/Zeitfelder in der Form "{Tag}.{Monat}.{Jahr} {Stunden}:{Minuten}[:{Sekunden}]"
- Erweiterung mit ab v2.5.9: Schlüsselwort: "#today" (constraint="datum is #today+2")
- Erweiterung mit ab v2.5.9: Vergleich mit anderen Feld (constraint="vondatum before bisdatum" Bei zwei zu vergleichenden Feldern muß das Containerfeld als erstes angegeben werden.)

{Operator}

- `<` (`<`) - Prüft, ob Feld zeitlich vor dem Testwert liegt
- `>` (`>`) - Prüft, ob Feld zeitlich nach dem Testwert liegt
- "before" - Prüft, ob Feld zeitlich vor dem Testwert liegt (ab v2.5.9)
- "after" - Prüft, ob Feld zeitlich nach dem Testwert liegt (ab v2.5.9)
- "=" oder "is" - Prüft, ob Feld gleich dem Testwert ist (gleicher Tag) (ab v2.5.9)
- "lt" - Prüft, ob Feld zeitlich vor dem Testwert liegt (ab v2.5.9)
- "gt" - Prüft, ob Feld zeitlich nach dem Testwert liegt (ab v2.5.9)
- "beforeOrEqual" - Prüft, ob Feld zeitlich vor oder genau auf dem Testwert liegt (ab v2.5.9)
- "afterOrEqual" - Prüft, ob Feld zeitlich nach oder genau auf dem Testwert liegt (ab v2.5.9)
- "lte" - Prüft, ob Feld zeitlich vor oder genau auf dem Testwert liegt (ab v2.5.9)
- "gte" - Prüft, ob Feld zeitlich nach oder genau auf dem Testwert liegt (ab v2.5.9)

Zahlenfelder

{Vergleichswert}

- Als Vergleichswert ist bisher (v2.5.8) nur ein fester Wert möglich. Als Dezimaltrenner muß das Komma verwendet werden.

{Operator}

- `<` prüft, ob Feld kleiner als der Testwert ist
- `>` Prüft, ob Feld größer als der Testwert ist
- `=` Prüft, ob Feld gleich dem Testwert ist

Häkchenfelder (Wahrheitswert)

{Vergleichswert}

- 'j', 'ja', 'true', 'y', 'yes' für Test auf wahr
- '#null' für Test auf nicht gewählt (ab v2.5.8)
- alles andere für Test auf falsch

{Operator}

- `=` prüft, ob Feld dem Testwert entspricht
- `!=` prüft, ob Feld nicht dem Testwert entspricht (ab v2.5.8)

Verknüpfungen, Untercontainer

werden in Version 2.5.8 noch nicht unterstützt

Dokument hochladen

Diese Aktion verschiebt ein hochgeladenes Dokument in die Dateiverwaltung. Dabei ist es wichtig, dass im Upload-Formular enctype="multipart/form-data" gesetzt ist.

fileparam	gibt den Request-Parameter an, der die hochgeladene Datei enthält
folderid	falls das Dokument in einen bestimmten Ordner der Dokumentenverwaltung gespeichert werden soll, kann hier die ID des Ordners angegeben werden
folderparam	eine Alternativmöglichkeit um das Dokument in einen bestimmten Ordner zu speichern, die ID wird aus dem hier angegebenen Request-Parameter ausgelesen (Standard-Parametername ist "folderid")
folderpath	die dritte Möglichkeit zur Angabe des Ordners, hier wird der Name des Ordners angegeben
targetweb	um das Dokument in ein anderes Projekt/Web zu speichern, hier die ID des Webs angeben
info	mit diesem Parameter kann die Beschreibung des Dokumentes festgelegt werden
infoparam	um die Beschreibung aus einem Request-Parameter auszulesen, hier den Parametername angeben

Bemerkungen

Bei Werten, die durch verschiedene Parameter festgelegt werden können, gilt folgende Reihenfolge der Überprüfung (es wird dabei abgebrochen, sobald ein Wert gefunden wurde):

- **Ordner**: `folderid` -> `folderparam` (wenn nicht angegeben, dann Standardwert "folderid") -> `folderpath`
- **Titel**: `titel` -> `titelparam`
- **Autor**: `autor` -> `autorparam`

In den Metadaten des Dokumentes werden unter "uploader" und "author" der eingeloggte Benutzer (falls er eingeloggt ist) sowie unter "uploaderIP" die IP des Uploaders eingetragen.

Multiupload

funktioniert wie beim Bilderupload.

Excel-Action

Erzeugt eine Excel-Datei, die gleich geöffnet oder als Download angeboten werden kann.

Parameter

createnew	Sheets anlegen, falls diese noch nicht existieren? (Standard = nein) Wird keine Vorlage verwendet, muß `"ja"` gewählt sein.
downloadname	Dateiname der erzeugten Datei (nicht ausfüllen, um Datei nicht zum Download anzubieten)
dokumentname	Dateiname für die Dokumentenverwaltung (nicht ausfüllen um die erzeugte Datei nicht abzuspeichern) ACHTUNG! `"ordnerid"` muß mit angegebenw werden.
ordnerid	ID des Ordners in der Dokumentenverwaltung, in der die fertige Datei abgelegt werden soll Muß mit angegeben werden, wenn in der Dokumentenverwaltung gespeichert werden soll.
xml	XML Daten direkt eingeben (wenn leer wird xmlfile genommen) Parameter mit [[[...]]] angeben
xmlfile	Dateiname der Seite, welche die XML Daten liefert
templateid	Server-Dateiname der Excel-Datei in der Dokumentenverwaltung, die als Vorlage dienen soll (wegen Styling oder Formeln). Leer lassen um neue Datei zu erzeugen, die dann allerdings nicht gestylt werden kann.
overwrite	Soll eine evtl. vorhande Datei überschrieben werden (Standard: nein) - wird nicht mehr genutzt
beschreibung	Beschreibung für die gespeicherte Datei in der Dokumentenverwaltung (nur für neu erstellte Dateien)
forcehttp	Soll in jedem Fall http anstatt https zum template laden benutzt werden? (Standard: nein)

Aufbau der XML-Datei

```
<?xml version="1.0" encoding="utf-8" ?>
<data>
  <style name="fettkursiv" bold="true" italic="true"/>
```

```
  <style name="comic" font="Comic Sans MS" size="14" color="red"/>

  <sheet name="Sheet-Name" [widthA="20" widthB="10" widthC="40"]>

    <insert row="0" col="0" style="fettkursiv">Zeile0 Spalte0</insert>
    <insert row="0" col="1">Zeile0 Spalte1</insert>
    <insert row="0" col="2">Zeile0 Spalte2</insert>

    <insert row="1" col="0">Zeile1 Spalte0</insert>
    <insert row="1" col="1">Zeile1 Spalte1</insert>
    <insert row="1" col="2">Zeile1 Spalte2</insert>

  </sheet>
</data>

ODER

<?xml version="1.0" encoding="utf-8" ?>
<data>
  <sheet idx="0">

    <insert adress="A1">Zeile0 Spalte0</insert>
    <insert adress="B1">Zeile0 Spalte1</insert>
    <insert adress="C1">Zeile0 Spalte2</insert>

    <insert adress="A2">Zeile1 Spalte0</insert>
    <insert adress="B2">Zeile1 Spalte1</insert>
    <insert adress="C2">Zeile1 Spalte2</insert>

  </sheet>
</data>
```

bold	true/false (oder weglassen)
italic	true/false (oder weglassen)

color	siehe hier, nur Farbnamen LIGHT_CORNFLOWER_BLUE r=204 g=204 b=255 ROYAL_BLUE r=0 g=102 b=204 CORAL r=255 g=128 b=128 ORCHID r=102 g=0 b=102 MAROON r=127 g=0 b=0 LEMON_CHIFFON r=255 g=255 b=204 CORNFLOWER_BLUE r=153 g=153 b=255 WHITE r=255 g=255 b=255 LAVENDER r=204 g=153 b=255 PALE_BLUE r=153 g=204 b=255 LIGHT_TURQUOISE r=204 g=255 b=255 LIGHT_GREEN r=204 g=255 b=204 LIGHT_YELLOW r=255 g=255 b=153 TAN r=255 g=204 b=153 ROSE r=255 g=153 b=204 GREY_25_PERCENT r=192 g=192 b=192 PLUM r=153 g=51 b=102 SKY_BLUE r=0 g=204 b=255 TURQUOISE r=0 g=255 b=255 BRIGHT_GREEN r=0 g=255 b=0 YELLOW r=255 g=255 b=0 GOLD r=255 g=204 b=0 PINK r=255 g=0 b=255 GREY_40_PERCENT r=150 g=150 b=150 VIOLET r=128 g=0 b=128 LIGHT_BLUE r=51 g=102 b=255 AQUA r=51 g=204 b=204 SEA_GREEN r=51 g=153 b=102 LIME r=153 g=204 b=0 LIGHT_ORANGE r=255 g=153 b=0 RED r=255 g=0 b=0 GREY_50_PERCENT r=128 g=128 b=128 BLUE_GREY r=102 g=102 b=153 BLUE r=0 g=0 b=255 TEAL r=0 g=128 b=128 GREEN r=0 g=128 b=0 DARK_YELLOW r=128 g=128 b=0 ORANGE r=255 g=102 b=0 DARK_RED r=128 g=0 b=0 GREY_80_PERCENT r=51 g=51 b=51 INDIGO r=51 g=51 b=153 DARK_BLUE r=0 g=0 b=128 DARK_TEAL r=0 g=51 b=102 DARK_GREEN r=0 g=51 b=0 OLIVE_GREEN r=51 g=51 b=0 BROWN r=153 g=51 b=0 BLACK r=0 g=0 b=0
widthA, widthB ...	Spaltenbreite nach Spaltenname

Formular als Mail versenden

Die Daten aus dem Request werden in ein Email-Design mit Platzhaltern eingefügt. Dann wird es als Email versendet.

Parameter

nextpage	auf diese Seite wird nach Fertigstellung weitergeleitet
mailFrom	die Email-Adresse, unter der die Mail versandt wird
mailFromName	der im Email-Programm angezeigte Absendername
mailTo	der Empfänger der Email
subject	der Betreff der Mail
sendHTMLFormat	wenn gewählt, wird die Email als HTML-Mail versandt
cc	Email-Adressen für CC (durch Semikolon trennen)
bcc	Email-Adressen für BCC (durch Semikolon trennen)

Bemerkungen

Platzhalter der Form "[[param]]" können in Adressen und Betreff verwendet werden. Diese werden dann durch die entsprechenden Request-Parameter oder Aktions-Attribute ersetzt. Ebenfalls kann im Betreff "{format}" verwendet werden, das aktuelle Datum wird dann gemäß "format" formatiert (bei fehlerhafter Angabe wird "dd.MM.yyyy" verwendet). .

Formulardaten validieren

Mit dieser Aktion ist es möglich, Formulardaten auf korrekte Syntax zu überpüfen.

failpage	die seite, auf die weitergeleitet wird, wenn die Validierung fehlschlägt
validate:param	für jeden Parameter, der validiert werden soll, muss es einen solchen Eintrag geben (param ist dabei durch den Name des zu überprüfenden Parameters zu ersetzen) der Wert stellt den regulären Ausdruck dar, mit dessen Hilfe der Parameter überprüft wird

Bemerkungen

Um Email-Adressen zu validieren kann als Parameter-Wert statt dem regulären Ausdruck einfach "email" geschrieben werden, es wird dann ein Ausdruck benutzt, der auf die meisten gängigen Adressen passt. Ein erweiterter regulärer Ausdruck für Email-Adressen ist z.B. dieser (Quelle: http://www.regular-expressions.info/email.html): `[A-Za-z0-9!#$%&'*+/=?^_`{|}~-]+(?:.[A-Za-z0-9!#$%&'*+/=?^_`{|}~-]+)*@(?:[A-Za-z0-9](?:[A-Za-z0-9-]*[A-Za-z0-9])?\.)+[A-Za-z0-9](?:[A-Za-z0-9-]*[A-Za-z0-9])? Um also wirklich sicher zu gehen, empfiehlt sich die Verwendung dieses Ausdrucks beim Überprüfen von Email-Adressen.

Innerhalb des regulären Ausdrucks können Platzhalter der Form [:param:] stehen, diese werden dann durch den Wert des Action-Parameters "param" ersetzt.

Beispiel für ein simples Formular mit den dazugehörigen Erweiterungen für die Validation

```
<form name="" method="" action="">
  <input type="hidden" name="validate:Vorname" value=".+">
  <input type="text" name="Vorname" value="">
  <input type="hidden" name="validate:Name" value=".+">
  <input type="text" name="Name" value="">
  <input type="hidden" name="validate:Strasse" value=".+">
  <input type="text" name="Strasse" value="">
  <input type="submit" value="senden">
</form>
```

Die Werte in den Value-Parametern der Hiddenfields sind reguläre Ausdrücke

Intranet Login

Um einen User im Intranetbereich zu authentifizieren wird diese Aktion benutzt.

Parameter

successpage	gibt die URL an, auf die bei Erfolg weitergeleitet wird, kann auch im Request übergeben werden (hat dann Vorrang)
failpage	auf diese URL wird bei Misserfolg weitergeleitet, kann auch als Request-Parameter übergeben werden (hat dann Vorrang)
block-counter	wenn ausgefüllt: Anzahl der erlaubten Fehlversuche bis zur Sperrung des Benutzers
block-timeout	Zeit in Minuten nach der Fehlversuche neu gezählt werden (Standard=60)
minimale Sessionlänge	(min-session-timeout) Angabe in Minuten. Die evtl. beim Benutzer eingestellte Sessionlänge wird für diese Session überschrieben
maximale Sessionlänge	Angabe in Minuten. Die evtl. beim Benutzer eingestellte Sessionlänge wird für diese Session überschrieben

Freigabe weitere Eigenschaften

username	wird im Request übergeben, alternative Namen sind: j_username, user, name, login
passwort	wird im Request übergeben, alternative Namen sind: password, j_password, pass

Intranet Logout

Diese Aktion invalidiert die Session und loggt damit den Benutzer aus.

Parameter

nextpage	auf diese Seite wird anschließend weitergeleitet

Intranet-User anlegen/ändern

Mit dem Action kann man einen Intranet-User anlegen und Angaben ändern (z.B. bei Paßwort vergessen). Meist in Verbindung mit einem Anmeldeformular kann man die im Formular eingegebenen und validierten Daten zum Anlegen des Users nutzen. field-user wäre z.B. Email und field-password wäre eines der beiden Paßwörter. Das Action gibt die ID des Users zurück (attribute-userid), die dann anschließend im Container gespeichert werden muß. Zusätzlich kann man noch die ID einer Benutzergruppe angeben, der der User zugeordnet sein soll. Mit dologin legt man schließlich fest, ob der User nach Abschluß der Anmeldung eingeloggt sein soll oder nicht.

Parameter

field-user	Parametername des Benutzerfeldes Feldname mit dem der Benutzername übergeben wird (nur wenn neuer Benutzer)
field-id	Parametername des Benutzer-ID-Feldes Feldname mit dem die Benutzer-ID übergeben wird (nur wenn Änderung eines Benutzers)
field-password	Parametername des Passwortfeldes Feldname mit dem das Passwort übergeben wird
activate	Benutzer aktiv schalten, Standard: ja
attribute-userid	Name für neue Benutzer-ID Attributname, in dem die ID des neuen Benutzers gespeichert werden soll
groups	zuzuordnende Gruppen, Liste von Gruppen-IDs
dologin	Benutzer sofort anmelden, (Standard nein) Ansonsten wird der aktuelle Benutzer ausgeloggt

Freigabe weiterer Eigenschaften:

anrede=...

name=...

vorname=...

email=...

anschrift=...

telefon=...

fax=...

bemerkung=...

Löschen

Mit dem Action-Parameter (unter Freieingabe) `delete` (auf "true" setzen) kann der Benutzer auch gelöscht werden.

JSP-Baustein ausführen

Ein JSP-Baustein wird ausgeführt.

Parameter

jsp-id	der auszuführende Baustein (ist aus einer Liste auswählbar)
quiet	wenn "true" werden keine keine Fehler angezeigt

Newsletter – Abo abmelden

Kurzbeschreibung

Funktionen:

Parameter

`entfernt`	`URL zur Folgeseite, wenn der Newsletter erfolgreich abgemeldet wurde`
`nichtvorhanden`	`die Folgeseite, falls für die angegebene Email-Adresse kein Newsletter vorhanden ist`
`falsch`	`Weiterleitung, falls das Passwort falsch ist`

Request-Parameter

`email`	`die Email-Adresse`
`aboid`	`die ID des Abos`
`passwort`	`das Passwort zur Email-Adresse`

Bemerkungen

Es werden entweder die Request-Parameter `email` und `aboid` **oder** `email` und `passwort` benötigt.

Newsletter – Abo ändern

Einstellungen des Newsletterabo werden geändert.

Parameter

nextpage	die Folgeseite

Request-Parameter

id access rubrik email htmlmail passwort1 passwort2

Newsletter – Abonnentenlogin

Authentifizierung um Newsletterabo zu ändern.

Parameter

login	Folgeseite, wenn Login OK war
error	Folgeseite, wenn der Login fehlgeschlagen ist

Request-Parameter

email	die Email-Adresse
password	das Passwort

Newsletter aktivieren

Ein abonnierter Newsletter wird freigeschalten.

Parameter

successpage	die Seite, die nach erfolgreicher Freischaltung angezeigt wird
failpage	die angezeigte Seite, falls das Abo nicht aktiviert werden konnte
requestparam	der Request-Parameter der die Aktivierungs-ID enthält (Standard: "id"), alternativ kann im Request auch email und aboid übergeben werden

Es kann eine Mail verschickt werden mit dem Optin-Link, mit dem der NL aktiviert werden kann.

Beispiele

Sie haben sich bei unserem Newsletter angemeldet. Bitte bestätigen Sie Ihre Email-Adresse, indem Sie auf unten stehenden Link klicken.:

https://www.testseite.de/optin.act?nlid=[[activateid]]

Newsletter – Newsletter abonnieren

Jemand wird zur Newsletter-Kundenliste hinzugefügt und erhält eine Email mit seinen Zugangsdaten.

Parameter

abonniert	URL der Folgeseite bei erfolgreicher Anmeldung
schonvorhanden	URL der Fehlerseite, wenn der Newsletter bereits abonniert wird
setactive	Abonnent ohne Prüfung sofort aktivieren?
subject	Betreff der Bestätigungsmail
mailFrom	die Email-Adresse des Absenders (wird bei Fehlermeldungen und beim Antworten auf Bestätigungsemail benutzt)
mailFromName	der in der Mail angezeigte Absendername

Request-Parameter

email	die Email-Adresse
rubrik	(optional) mehrmals angeben, um die Rubriken zu spezifizieren (Standard: alle)
passwort1 passwort2	(optional) Passwort und -Bestätigung (Standard: zufall)
htmlmail	(optional) Newsletter im HTML-Format senden ('j' oder 'n', Standard: 'n')
firma, ansprech, strasse, ort	(optional) Abonenntendaten
frage, antwort	(optional) geheime Frage und Antwort

Newsletter – Passwort vergessen (1)

Diese Aktion verlangt eine Eingabe der Email-Adresse um die geheime Antwort eingeben zu können.

Parameter

question-page	Seite zur geheimen Frage (Standard: "./frage.htm")
unknown-page	Seite, wenn die Email-Adresse nicht in der Datenbank ist (Standard: "./")
unsupported-page	Seite, falls keine Frage festgelegt wurde (Standard: "./")
email	(Request-Parameter) die Email-Adresse

Newsletter – Passwort vergessen (2)

Nach Eingabe der geheimen Antwort wird eine Email mit dem Passwort versandt.

Parameter

ok-page	Ziel URL, wenn die Antwort richtig war
fail-page	Ziel URL wenn Antwort auf die Frage falsch war
mailFrom	Email-Adresse, unter der die Mail versandt wird
mailFromName	im Email-Programm angezeigter Absendername
subject	Betreff
sendHTMLFormat	Email im HTML-Format senden?

Request-Parameter

aboid	die Abo-ID
antwort	die Antwort auf die geheime Frage

Passwort des angemeldeten Users ändern

Das Passwort des gerade angemeldeten Users kann mittels dieser Aktion geändert werden.

Parameter

oldpassword-field	Feld für altes Passwort Optional. Falls ausgefüllt, muss das alte Passwort in diesem Request-Paramater übergeben werden. Diese Eigenschaft muss ggf. über die weiteren Eigenschaften angegeben werden.
password1-field	Feld für Passwort Request-Feldname, der Passwortfeld enthält.
password2-field	Feld für Passwortwiederholung Request-Feldname, die Passwortwiederholung enthält.
failpage	Fehlerseite Fehlerseite falls Passwort leer, zu kurz oder ungleich der Wiederholung

Bemerkungen

Die beiden übergebenen Eingaben müssen übereinstimmen und das Passwort muss mindestens 3 Zeichen lang sein.

Shop – Bestellung versenden

Diese Aktion versendet den Warenkorb per Email.

Parameter

name	der Name des Warenkorbes (falls mehrere getrennte Warenkörbe verwaltet werden sollen)
mailFrom	die Email-Adresse des Absenders
mailFromName	der anzuzeigende Absender-Name im Email-Programm
subject	der Betreff
mailTo	Empfänger-Adresse
ccMailField	der Request-Parameter, der die Adresse für die Bestätigungsmail enthält
ccSubject	der Betreff für die Bestätigungsmail
art-titel	Name des Feldes, was die Artikelbezeichnung enthält
art-preis	Name des Feldes, was den Preis enthält
art-nummer	Name des Feldes, was die Artikelnummer enthält
backfile	URL zur Bestätigungsseite (standardmäßig "../")
currency	das Währungssymbol
clearbasket	Warenkorb leeren?
extraChargeField	Feldname für den Betrag eines zusätzlichen, produktabhängigen Postens
extraChargeName	der Name des zusätzlichen Postens (Standard ist der Wert von extraChargeField)

Bemerkungen

Weiterhin können Parameter der Form "addCharge1Field" und "addCharge1Name" verwendet werden ("1" kann dabei jeweils erhöht werden, also "addCharge2Field, addCharge2Name, addCharge3Field, addCharge3Name, ..."). Diese zusätzlichen Posten werden der Summe hinzugefügt.

Shop – Hinzufügen zu Warenkorb

Durch diese Aktion wird ein neues Element in den Warenkorb aufgenommen.

Parameter

name	Name des Warenkorbes (falls mehrere getrennte Warenkörbe verwaltet werden sollen), standardmäßig der Wert des Request-Parameters "basket"
basketlink	die URL der Folgeseite, hier kann "[[url]]" verwendet werden, die Folgeseite steht dann im Request-Parameter "url"
extraParameterPrefix	eine Markierung für zusätzlich Parameter, die dem Warenkorb übergeben werden (alle Parameter, die mit diesem Wert anfangen werden übergeben)

Request-Parameter

basket	der Name des Warenkorbes
artikelid	die Artikel-ID
type	nur für CM-Seiten-Shop-Modus
url	die Folgeseite
anzahl	Bestellmenge
index	nur für Updatemodus

Shop – Löschen im Warenkorb

Diese Aktion löscht einen Artikel aus dem Warenkorb oder leert den Warenkorb komplett.

Parameter

name	der Name des Warenkorbes
nextpage	die Folgeseite
clear	löscht den gesamten Warenkorb (zu verwenden nach Versendung der Bestellung)
remove	(Request-Parameter) enthält die zu löschenden Posten (falls als erster Posten "all" übergeben wird, werden alle Einträge gelöscht)

Shop – Zusatzgebühr zum Warenkorb

Eine Zusatzgebühr (z. B. Nachnahme, Lieferkosten, Versicherung) wird in den Warenkorb gelegt oder entfernt.

Parameter

basketname	der Name des Warenkorbes
group	der Name des Postens (kann auch im Request übergeben werden, als "name" oder "group")
value / element	(Request-Parameter) enthält den Wert des Postens, falls hier nichts steht wird der Posten entfernt

Sicheres Formular überprüfen (SecureformAction)

Das SecureForm-Action wird benutzt, um Formulare gegen Manipulationen abzusichern. Es ist verfügbar ab *Version 2.6.5.*

> Unter diesem Link gibt es eine ausführliche Anleitung, wie dieses Action mit dem zugehörigen Tag zu benutzen ist.

Parameter

securelist	Dies ist eine komma-getrennte Liste der Parameternamen, welche zur Hashbildung herangezogen werden. Tauchen hier Parameternamen auf, die nicht im Request sind, so werden diese Parameter nicht zur Hashbildung benutzt. Hier können auch Parameter stehen, die nie an das Action übergeben werden dürfen - da diese im Frontend nicht mitgehasht werden und höchstens durch Manipulation im Request landen, schlägt die Prüfung fehl und das Action bricht ab. Zusätzlich zu dieser manuellen Liste gibt es eine im System verankerte Liste, welche einige Standardparameter enthält, die immer geschützt sein müssen (z.B.: recordid, listid).
failurl	Falls die Prüfung fehlschlägt, wird zu dieser URL weitergeleitet. Ist hier nichts eingetragen und die Prüfung schlägt fehlt, wird das Action mit einer Fehlermeldung abgebrochen.

Alle Parameter in securelist müssen, falls vorhanden, abgesichert sein. Das heißt ihr Wert muss entweder direkt in den Hash eingeflossen sein, oder einen der erlaubten Werten entsprechen (und diese Liste muss dann in den Hash eingeflossen sein).

Erlaubte Werte

Falls ein spezieller Parameter für mehrere erlaubte Werte mitgeschickt wurde, wird zusätzlich geprüft, ob der originale Parameter einen dieser Werte entspricht.

Actionklasse: com.batix.action.SecureformAction

Suche im Web

Es wird eine Suche in vorher indizierten Seiten des Webs ausgeührt. Das Suchergebnis wird in der Benutzersitzung gespeichert und kann danach auf der Ergebnisseite wieder ausgelesen werden.

Diese Suchergebnisse können dann mittels bx:websearch in eine HTML-Seite eingefügt werden.

Parameter

searchparam	Request-Parameter der den Suchbegriff enthält (Standard ist "search")
resultpage	die Seite, auf die danach weitergeleitet wird
useAccessFilter	sollen Menüpunkte, auf die der Benutzer kein Zugriff hat herausgefiltert werden?
useNavFilter	aktiviert die Parameter includeNav, excludeNav sowie searchroot
includePattern	ein regulärer Ausdruck, der angibt, was in die Suche eingeschlossen wird
excludePattern	ein regulärer Ausdruck, der angibt, was aus der Suche ausgeschlossen wird
includeNav	Liste von Navigations-ID's von in die Suche eingeschlossenen Pfaden
excludeNav	Liste von Navigations-ID's von aus der Suche ausgeschlossenen Pfaden
searchroot	gesamtes Web durchsuchen? (wenn nicht speziell angegeben)
boost-date	Aufwertung neuerer Seiten im Suchergebnis, Standard 0,5. `0` schaltet sie ab *(ab Version 3.3.2)*
halflife-days	Halbwertszeit der Aufwertung in Tagen, Standard 365 *(ab Version 3.3.2)*
sortparam	Request-Parameter, mit dem der Besucher die Sortierung umschalten kann (Standard "sort") *(ab Version 3.3.2)*
similarity-classic	Trefferbewertung wie in Versionen vor 3.3.2 (TF-IDF statt BM25) *(ab Version 3.3.2)*

Bemerkung

Sobald `useAccessFilter` verwendet wird, werden alle anderen Parameter zur Einschränkung (wie z.B. `useNavFilter`, `include...` oder `exclude...`) nicht mehr beachtet.

Aktualität im Suchergebnis

(ab Version 3.3.2) Neuere Seiten werden im Suchergebnis bevorzugt. Die frischeste Seite wird um den in `boost-date` angegebenen Anteil besser bewertet, mit zunehmendem Alter läuft die Aufwertung aus: bei 0,5 bekommt eine ganz neue Seite 50 % Zuschlag, eine ein Jahr alte 25 %, eine fünf Jahre alte knapp 2 %. Über `halflife-days` wird eingestellt, nach wie vielen Tagen der Zuschlag auf die Hälfte gefallen ist – für Projekte, in denen es überwiegend um Nachrichten geht, sind 90 bis 180 Tage sinnvoller als die voreingestellten 365.

Voraussetzung ist, dass die Seite ihr Datum ausgibt: `<meta name="date" content="2026-08-13">` im Kopf der Seite. Seiten ohne Datum – typisch Übersichts- und Listenseiten – werden nicht aufgewertet, was so gewollt ist. Ausgeben lässt sich das Datum mit bx:websearch.date, dort steht auch, welche weiteren Schreibweisen gelesen werden.

Der Besucher kann pro Suche umschalten. Dafür genügt ein Radio-Button-Paar im Suchformular:

```
<label><input type="radio" name="sort" value="aktuell" checked> Relevanz und
Aktualität</label>
<label><input type="radio" name="sort" value="relevanz"> nur Relevanz</label>
```

Beim Wert `relevanz` wird nur nach Relevanz sortiert, bei jedem anderen Wert nach Relevanz und Aktualität. Heißt der Parameter im Formular anders, wird sein Name über `sortparam` angegeben.

Validierungsergebnis ausgeben

Diese Aktion wertet das Ergebnis einer Container-Validierung aus. Es kann ein Report in ein Textfeld des Datensatzes geschrieben werden, ein Häkchen im Containerdatensatz geschalten werden oder das Action abgebrochen und auf eine andere URL weitergeleitet werden.

Parameter

reportfield	das Feld, in dem das Validierungsergebnis gespeichert werden soll
format	das Format, in dem das Validierungsergebnis geschrieben werden soll ("xml" oder "properties")
resultfield	das Häkchenfeld in dem der Validierungsstatus gespeichert werden soll (angehakt bedeutet "Validation nicht bestanden")
fail-url	hierhin wird weitergeleitet, wenn die Validierung fehlgeschlagen ist
success-url	hierhin wird weitergeleitet, wenn die Validierung erfolgreich war
list	zu validierende Datenliste, falls mehrere Container gleichzeitig validiert werden

Bemerkung

Es muss vorher eine Validierung stattgefunden haben.

Sobald eine URL aufgerufen wird (`fail-url` oder `success-url`) wird die weitere Aktionsverarbeitung abgebrochen. In der URL-Angabe können auch Platzhalter der Form "[[param]]" verwendet werden, diese werden dann durch den jeweiligen Wert des Request-Parameters "param" ersetzt.

Vorhandene Bedingung abfragen

Dies ist eine Aktion, die einen Abbruch der Aktionsverarbeitung bewirkt, wenn ein Requestparameter ungleich einem bestimmten Wert ist.

Parameter

requestField	der Requestparameter, dessen Wert abgefragt werden soll
compareValue	der Wert, den der Requestparameter haben muss, um die Bedingung zu erfüllen es können auch Spezailwerte verwendet werden: #null - Feld ist nicht im Request enthalten #empty - Feld ist leer im Request #nodata - Feld ist nicht im Request oder ist leer Ferner ist es möglich, reguläre Ausdrücke der Form ^regex hier$ zu verwenden. Auch kann man zwei Request-Werte vergleichen, indem man bei compareValue den zweiten Parameter in zwei eckige Klammern stellt.
conditionTrue	zu dieser URL wird weitergeleitet, wenn die Bedingung erfüllt wird
conditionFalse	falls die Überprüfung fehlschlägt, wird hierhin weitergeleitet

Die URLs können Platzhalter der Form [[Parametername]] sowie die speziellen Platzhalter {querystring} und {request} enthalten. Ebenfalls kann den speziellen Platzhaltern "?" oder "&" vorangestellt werden (z.B. {&request}), diese Zeichen werden dann mit ausgegeben, falls der Platzhalter zu einem Wert aufgelöst werden kann.

Klassenname

`com.batix.action.ConditionAction`

Weiterleitung

Diese Aktion leitet an eine andere Seite weiter.

Funktionen:

Parameter

destination	die Ziel-URL
appendQueryString	gibt den Query-String des aktuellen Requests an die Ziel-URL weiter

Bemerkungen

In der URL können Platzhalter der Form "[[param]]" verwendet werden, diese werden dann durch die Werte der entsprechenden Request-Parameter ersetzt. Weiterhin gibt es zwei spezielle Platzhalter:

[[session-id]]	die Session-ID
[[newrecordid]]	ID eines Datensatzes, der durch eine vorherige Speicher-Aktion angelegt wurde

ACHTUNG!

Nach Update von einer V2.6.6 auf eine aktuelle Version wichtig:

Stilistischer Aufbau: /index.htm ->Liste /detail.htm ->Detailseite /save.act ->Speichern-Aktion

Bei Formular absenden an save.act haben Weiterleitungen die bisher mit "?query=foo" den Browser bisher wegen einem Tomcat "Bugfeature" auf "./?query=foo" weitergeleitet. Das ist eigentlich **falsch** und inzwischen gefixt. Nun wird aber der Browser bei *Verwendung" des Bugs als Endlosschleife immer wieder auf das Save-Action geleitet: "./save.act?query=foo". Daher muss in den aktuellen Versionen für eine Weiterleitung auf die Index-Seite des Menüpunktes "./?query=foo"

verwendet werden, da sonst korrekterweise auf die Adresse geleitet wird, die der Browser auch zuletzt angefragt hat. Fehlerbild wären also seltsame leere Actionaufrufe aus dem Frontend und Benutzer, die melden, dass ihr Browser nach Formular-Absenden keine sinnvolle Seite mehr darstellt und stattdessen was von "Zu viele Weiterleitungen" berichtet.

Wert in Session speichern

Alle Request-Parameter mit dem angegebenen Präfix werden in eine Hashmap in der Session geschrieben. Der Name des Session-Parameters ist "com.batix.sessionlist:**mapname**".

Parameter

mapname	Name der Hashmap (Standard: "default")
fieldname	Name des Parameters
fieldnamePrefix	alternativ: Präfix der zu speichernden Parameter
savename	(ggf. über Freieingabe setzen) Name des Session-Attributes, in dem der Wert aus dem Request gespeichert werden soll

Wird **savename** nicht angegeben, wird als Name des Session-Attributes der Wert aus **fieldname** verwendet.

XHTML 2 PDF

Liest URLs aus einem Container,einem Request-Parameter oder mit Vorgabe im Action und erstellt daraus PDF-Dateien. Vollständige und strikte Unterstützung von **CSS 2.1** sowie einigen Erweiterungen von **CSS 3.0**. (siehe [1])

Funktionen:

Parameter

Dokumentverwaltung	*in Dokumentverwaltung speichern (Standard = nein)* Wenn hier nein angegeben wird, dann wird das PDF-Dokument direkt zum Browser gestreamt
Verzeichnis-Id Eingabe bei weitere Eigenschaften: **folderparam=parametername** **folderpath=/pfad/** **targetweb=Projekt-ID**	*ID des Zielordners in des Dokumentenverwaltung* Wenn bei Dokumentverwaltung ein ja gewählt ist, kann man in Attribut Verzeichnis-Id die ID eines Ordners der Dokumentenverwaltung angeben oder alternativ diese ID in einem Request-Parameter namens 'Verzeichnis-Id' oder in einem Request-Parameter dessen Name in Attribut folderparam eingetragen ist. Der Ordner kann auch über dessen Namen angegeben werden in Attribut folderpath. Zusätzlich kann man im Attribut targetweb ein anderes Projekt des selben Mandanten angeben.
Dokumentname	*Dateiname des erzeugten PDF* Wenn bei Dokumentverwaltung der Standard nein gewählt ist, kann hier der vorgeschlagene Dateiname angegeben werden (Platzhalter sind erlaubt). bei einem ja wird dies als Originaldateiname verwendet
Dateifeldname	*Name des Feldes, in dem die Verknüpfung in nächsten Action gespeichert werden soll* nur wenn bei Dokumentverwaltung = ja gewählt ist: Hier kann ein Action-Parameter für Folgeactions definiert werden, z. B. um das Dokument zu einem Containerdatensatz zuzuweisen.
URL	*Pfad zur Seite* Im Hintergrund wird die URL aufgerufen und dann das zurückgelieferte HTML in eine PDF-Datei gerendert.

Request-Name	*Name des Requestparameters der eine URL enthält* Ein Parameter mit diesen Namen muß im Request, der dieses Action aufruft, mit übertragen werden. Er muß eine URL enthalten und diese URL wird aufgerufen und dann das zurückgelieferte HTML in eine PDF-Datei gerendert. (Wird ignoriert wenn bei URL etwas angegeben ist)
Container-Id	*Container in welchem die URLs gespeichert sind* Bei Angabe der ID eines Containers (pool) wird in diesen Container in allen Datensätzen bzw. diesen, die den Kriterien in Container-Filter entsprechen, nach dem Feld Container-Feld geschaut. Dieses Feld enthält die URL zum Quellcode für das PDF und von allen gefilterten Datensätzen werden diese Quellcodes zusammenkopiert und als ein großes PDF gerendert
Container-Feld	*Containerfeld welches die komplette URL enthält*
Containerfilter	*Filterkriterien des Containers im XML-Format* (z.B. Session-ID prüfen)
Eingabe bei weitere Eigenschaften: **info=irgendeine Info** **infoparam=parametername**	*Dateiname des erzeugten PDF* Hier kann eine Dokumentbeschreibung für den Eintrag in der Dokumentverwaltung eingetragen werden. Entweder ein fester Wert in Attribut info oder der Name eines anderen Request-Parameters im Attribut infoparam

Quelldatenmodi:

- Direkte URL angeben: Feld URL ausfüllen
- URL in Requestparameter: Request-Name ausfüllen
- URL in Containerfeld: "Container-"-Felder ausfüllen

Ausgabemodi:

- Direkt streamen: Dokumentverwaltung = nein auswählen
- Speichern in Dokverwaltung: Dokumentverwaltung = ja auswählen

Tips

Seitennummern rechts unten:

```
@page {
  @bottom-right {
    content: "Seite " counter(page);
    font-family: Arial, Helvetica, Sans-Serif;
    font-size: 12pt;
  }
```

```
}
```

Seitengröße:

814 x 1134 Pixel (Größe für Hintergrundbilder, die die gesamte Seite ausfüllen sollen)

PDF und Schriften

Beispiel für CSS-Style zur Einbettung von Schriften in ein PDF-Dokument:

```
@font-face{
font-family:Source Sans Pro Light;
src:url(/static/<bx:pagedata.webdir/>/assets/fonts/SourceSansPro-Light.ttf)
format("truetype");
font-style:normal;
font-weight:400;
-fs-pdf-font-embed: embed;
-fs-pdf-font-encoding: Identity-H;
}
```

Der Schriftname bei font-family muss identisch sein mit der Bezeichnung in der TrueType-Font-Datei. Als Windows-User findest du die korrekte Bezeichnung mit Doppelklick auf die ttf-Datei.

