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.