bx:secureform

Das Tag `bx:secureform` wird benutzt, um Formulare gegen Manipulationen abzusichern. Es ist verfügbar ab *Version 2.6.5.*

> [Unter diesem Link](/books/cms-handbuch-entwickler/page/manipulationsgeschuetzte-formulare-mit-secureform) gibt es eine ausführliche Anleitung, wie dieses Tag mit dem [zugehörigen Action](/books/cms-handbuch-entwickler/page/sicheres-formular-ueberpruefen) zu benutzen ist.

secureform

```
<bx:secureform> ... </bx:secureform>
```

Das Tag wird um eine `<form>` geschachtelt und definiert somit einen Gültigkeitsbereich für die inneren Tags (s.u.).

Beispiele

```
<bx:secureform>
  <form action="save.act">
    ...
  </form>
</bx:secureform>
```

secureform.field

```
<bx:secureform.field name="{parametername}" [cut]> ... </bx:secureform.field>
<bx:secureform.field name="{parametername}" hidden-when-empty [cut]> ...
</bx:secureform.field>
```

Dieses innere Tag wird um die einzelnen Werte von `<input>`s oder Ähnlichem geschachtelt. Es merkt sich den Name und Wert des Parameters, um später über alle gesammelten Parameter den Hash zu bilden.

Der Wert wird auch wieder vom Tag ausgegeben, es funktioniert also wie `[bx:clipboard.copy](/books/cms-handbuch-entwickler/page/bx-clipboard)`. Falls der Wert nicht ausgegeben werden soll, kann `cut` angegeben werden (funktioniert dann wie `[bx:clipboard.cut](/books/cms-handbuch-entwickler/page/bx-clipboard)`).

> Falls im Inhalt andere Tags benutzt werden, muss darauf geachtet werden, dass diese **kein** HTML- oder sonstiges Encoding vornehmen (d.h. es muss ggf. `type="plain"` oder `encode="plain"` angegeben werden)! Das `secureform.field` Tag macht automatisch ein HTML-Encoding, falls es nicht von `tools.htmlencode` umschlossen ist.

Der Parameter `hidden-when-empty` muss angegeben werden, falls leere Werte dieses Parameters vom Browser nicht mitgesendet werden. Das würde sonst die Hashbildung verfälschen, denn das Tag beachtet den Parameter und merkt sich den leeren Wert, allerdings bekommt das Action den Parameter nicht übergeben und zieht ihn entsprechend *nicht* zur Hashbildung heran. Ist `hidden-when-empty` also angegeben und der Taginhalt leer, dann fließt dieser Parameter nicht mit in die Hashberechnung ein.

> Als Wert (Tag-Inhalt) darf kein User-Input verwendet werden, die dieser ja nicht vom Server festgelegt wird.

Im Normalfall darf also z.B. kein `bx:pagedata.request` innerhalb von `bx:secureform.field` stehen.

parametername	**entspricht dem Name des Parameters, der vom Browser später mitgeschickt wird ist meistens gleich dem `name`-Attribut eines `<input>`s**

Beispiele

```
<!-- statischer Wert -->
<input type="hidden" name="gruppe" value="<bx:secureform.field
name="gruppe">12345ABC</bx:secureform.field>">

<!-- dynamischer Wert -->
<input type="hidden" name="kid" value="<bx:secureform.field name="kid"><bx:recordfield.Kunde
type="id" /></bx:secureform.field>">

<!-- Wert, bei dem das HTML-Encoding abgeschaltet werden muss -->
<input type="hidden" name="fester_titel" value="<bx:secureform.field
name="fester_titel"><bx:text.Titel_aus_Verwaltung type="plain" /></bx:secureform.field>">
```

```
<!-- Feld, dass z.B. bei <bx:containerfilter dummy> wegfallen kann und somit hidden-when-empty benutzt werden muss -->
<bx:if><input type="hidden" name="editid" value="<bx:secureform.field name="editid" hidden-when-empty><bx:recorddata.id /></bx:secureform.field>"></bx:if>
```

secureform.values

```
<bx:secureform.values name="{parametername}" [cut]>...</bx:secureform.values>
```

In dieser Form wird ein erlaubter Wert eines Parameters gesammelt (siehe Mehrere erlaubte Werte). Es verhält sich wie `bx:secureform.field` (`cut`, HTML-Encoding). Bitte die Hinweise weiter oben auch hier beachten!

parametername	entspricht dem Name des Parameters, der vom Browser später mitgeschickt wird

Die gesammelten, erlaubten Werte müssen dann für jedes Feld ausgegeben werden. Dies geschieht in folgender Form:

```
<bx:secureform.values name="{parametername}" />                        <!-- hidden Feld wird erzeugt -->
<bx:secureform.values name="{parametername}" param [encode="..."] />   <!-- nur Parametername ausgeben -->
<bx:secureform.values name="{parametername}" values [encode="..."] />  <!-- nur gesammelte Werte ausgeben -->
```

Die erste Variante (nur ein geschlossenes Tag mit `name` Parameter) erzeugt ein `input type="hidden"` mit passenden `name` und `value` Attributen.

Falls man selbst einen Link zusammenbaut, kann `param` bzw. `values` benutzt werden, um entweder den Name des speziellen Parameters bzw. dessen Wert auszugeben. Je nach Fall muss die Rückgabe entsprechend encoded werden, z.B. `encode="javascript"` oder `encode="url"`.

> Falls keine Werte für den Parameter gesammelt wurden, geben alle drei Varianten nichts aus.

secureform.hash

```
<bx:secureform.hash />        <!-- hidden Feld wird erzeugt -->
<bx:secureform.hash name />   <!-- nur Hash-Parametername ausgeben -->
<bx:secureform.hash value />  <!-- nur Hash-Wert ausgeben -->
```

Dieses innere Tag durchläuft alle gesammelten Parameter-Werte-Paare und bildet mit anderen, geheimen Faktoren einen Hashwert. Ohne Parameter wird ein `<input type="hidden">` mit passenden `name` und `value` Attributen in die Seite geschrieben.

Möchte man nur Name oder Wert haben, ist dies auch möglich (um diese z.B. in Scripts zu verwenden).

> Nach `` oder `` darf kein weiteres `` oder `` folgen. Alle Tags müssen außerdem innerhalb von `` stehen.

secureform.check

```
<bx:secureform.check securelist="{parameterliste}" [url="{url}"] [forward] />
```

Hiermit können in einem Template die abgesicherten Parameter geprüft werden. Die Parameter ähneln die des Actions.

parameterliste	**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 die Seite übergeben werden dürfen - da diese im Frontend nicht mitgehasht werden und höchstens durch Manipulation im Request landen, schlägt die Prüfung fehl und die Seite 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).**
url	Falls die Prüfung fehlschlägt, wird zu dieser URL weitergeleitet. Ist zusätzlich `forward` angegeben, wird statt dem Redirect ein Forward gemacht (URL bleibt im Browser stehen). Ist hier nichts eingetragen und die Prüfung schlägt fehlt, wird die Abarbeitung der Seite abgebrochen und der generierte Quelltext wird nicht an den Client ausgeliefert. Manche Tags führen schon Aktionen aus, bevor `<bx:secureform.check>` die Parameter prüft, diese sind aber meistens unkritisch.

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).

> Dieses Tag sollte in der ersten Zeile des Quelltextes stehen, um sicherzustellen, dass die Abarbeitung der restlichen Seite so früh wie möglich abgebrochen wird.

Erlaubte Werte

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

Verweise auf diese Seite:

- Sicheres Formular überprüfen (SecureformAction)
- Sicheres Formular überprüfen (SecureformAction)
- Manipulationsgeschützte Formulare mit Secureform
- Manipulationsgeschützte Formulare mit Secureform

(4 Verweise)