Schnelleinstieg für Entwickler

Letzte Aktualisierung: 15.03.2017 um 09:09 Uhr von Dan, Günther

Wenn Sie die Frontend Administration in Kombination mit Layouts und Blöcken verwenden möchten, muss Ihr Design bestimmte Voraussetzungen erfüllen. Welche das sind um schnell in den Genuss der neuen Features zu kommen, nennen wir hier.

Das EGOCMS 16 erlaubt es nun **.html** oder **.tpl** Templates zu verwenden. Wenn Sie lieber mit **.tpl** Dateien entwickeln, weil so z.B. Ihr bevorzugter Editor ein besseres *Smarty Highlighting* ermöglicht, können Sie das nun endlich machen. Im folgenden werden **.tpl** Dateien für die Beispiele genannt. Diese funktionieren aber auch mit **.html** Dateien.

In diesem Beispiel ermöglichen wir dem Design ein Layout zu verwenden, welches die volle Breite annimmt und erstellen einen Block, der *Bild* und *Text* Element nebeneinander beinhaltet.

Kopfbereich vorbereiten

In Ihrem **index.tpl** Template binden Sie die neue Smarty Funktionein **include_head_tags** ein. Diese kümmert sich darum, dass Sie immer alle notwendigen CSS, Javascript Dateien und auch Meta Tags einbinden. Zusammen mit **include_module_files** sieht Ihr *HEAD* Element bereits so aus:

```
<head>
   {include_head_tags}
   {include_module_files}
</head>
```

Hauptbereich vorbereiten

Damit das EGOCMS weiß, welcher Bereich im Design das Layout und somit alle Blöcke beinhaltet, müssen Sie einen Container mit dem Attribut **data-edit-template="layout"** definieren.

```
<div data-edit-template="layout">
  {include file=$typeTemplate}
</div>
```

Layout definieren

Definieren Sie eine **layout.tpl** Datei, welches Ihr Standard Layout beinhaltet. Standardmäßig kann dies einfach ein Container für die zentralen Blöcke sein. Ein Container folgt einer Orientierung, welche standardmäßig **center** ist.

```
<div data-edit-template="center">
   {$page->getBlocks("center")}
</div>
```

Wenn Sie z.B. ein Layout möchten, dass aus zwei Spalten besteht, könnte dies so aussehen.

```
<div data-edit-template="center" style="float: left;">
   {$page->getBlocks("center")}
</div>
<div data-edit-template="right" style="float: right;">
   {$page->getBlocks("right")}
</div>
```

Ihre **layout.tpl** liegt im selben Verzeichnis wie Ihre **index.tpl**.

Im Verzeichnis **layouts** können Sie noch weitere Layouts definieren. Alle Layouts die hier definiert sind, sind für alle Seitentypen verfügbar. Diese tragen als Namen eine beliebige Bezeichnung, z.B. **image_text.tpl**.

Sie können auch ein ganz anderes *Layout* als Standard Layout definieren.

```
{
   "default_layout": "other_layout"
}
```

Blöcke definieren

Blöcke definieren Sie im Verzeichnis **blocks**. Alle Blöcke die hier definiert sind, sind für alle Seitentypen verfügbar. Das HTML im Block kann beliebig aussehen. Wenn Sie Elemente wie *Bild, Text,* usw. einbinden möchten, verwenden Sie die Smarty Funktion **value**.

```
<div style="float: left;">
```

```
    {value var="image1" type="image" title="Bild"}
</div>
<div style="float: right;">
    {value var="content1" type="content" title="Inhalt"}
</div>
<br style="clear: both;"/>
```

var ist die Bezeichnung des Wertes für das Extrafeld. Es wird empfohlen einheitliche Bezeichnungen zu verwenden, wie z.B. image1, image2, image3, content1, content2, content3, usw. Damit lassen sich Blöcke besser ersetzen, da die Werte des alten Blockes bereits für den neuen Block passen. Auch bei einem Designwechsel stimmen dann bereits alle Inhalte.

type ist der Typ des Elements.

- **image:** erzeugt ein Bild Element.
- **content:** minimalistischer WYSIWYG Editor.
- **editor:** komplexer WYSIWYG Editor.
- **text:** Editor, in welchem nur Text eingegeben werden darf (kein HTML).

title gibt den Text an, der bei der **Visualisierung** für dieses Element angezeigt wird.

Erweiterte Möglichkeiten

Bis hierhin können Sie bereits **Layouts und Blöcke** verwenden und mit der **Frontend Administration** durchstarten. Es gibt aber noch weitaus mehr Möglichkeiten.

In **site/<Mandant>/admin/conf.json** können Sie als JSON formattiert weitere Einstellungen vornehmen. z.B. können Sie dem erstellen Layout und Block weitere Eigenschaften zusagen.

```
{
    "layouts": {
        "default": {
            "title": "Mein Standard Layout"
        }
    },
    "blocks": {
        "image_text": {
            "title": "Bild und Text",
            "description": "Bild und Text in zwei Spalten aufgeteilt.",
            "pattern": "[33.33:image][66.66:text]"
        }
    }
}
```

Mit **title**, bzw. **description** können Sie die angezeigten Texte zur besseren Identifizierung bei der Auswahl bestimmen.

Bei **layouts** steht **default** für das Standard Layout. Wenn Sie weitere *Layouts* anbieten, steht an dieser Stelle die Bezeichnung des Dateinamens.

pattern ermöglicht es Ihnen für Blöcke ein Muster zu zeichnen, welches optisch erahnen lassen soll, wie der Block aufgebaut ist. **[]** stellen einzelne Elemente dar, welche auch *kommasepariert für mehrzeilige* Darstellungen möglich sind. Der erste Wert ist die prozentuale Breite des Elements. Der zweite Wert nach dem Doppelpunkt kann ein beliebiger Schriftzug sein. Wird als Schriftzug *image* verwendet, wird ein Platzhalter Bild angezeigt. Sie können anstatt **pattern** auch **image** und somit den Pfad zu einem Bild verwenden. Mit **max** können Sie sogar angeben, wie oft dieser Block gleichzeitig verwendet werden darf.

Die Einstellungen aus **conf.json**, sowie die **layouts** und **blocks** Verzeichnisse, führen sich rekursiv zusammen. Bedeutet: Sie können einzelnen Seitentypen weitere *Layouts* oder *Blöcke* definieren oder einzelne Einstellungen aus **conf.json** überschreiben (ohne dabei die gesamte **conf.json** kopieren zu müssen).

In **layouts** können Sie auch definieren, welche Blöcke standardmäßig verwendet werden oder welche erlaubt, bzw. nicht erlaubt sind. Mehrere Blöcke werden kommasepariert angegeben. Mit einem @ können Sie die Einstellungen einer anderen *Orientierung* verwenden und erweitern.

```
{
    "layouts": {
        "default": {
            "blocks": {
                "center": {
                    "default": "image_text",
                    "allow": "block1,block2",
                    "disallow": "block3,block4"
                },
                "right": {
                    "default": "@center,block5"
                }
            }
        }
```

```
    }
}
```

Listenpunkte erweitern

Im WYSIWYG Editor können Sie nummerierte Listen und Aufzählungen verwenden. Diese können Sie ergänzen, um Listen mit eigener CSS Klasse auswählen zu können. Hierzu wird die **conf.json** erweitert.

```
{
    "editor": {
        "lists": {
            "bullist": [{
                "text": "Offener Pfeil",
                "cls": "open-arrow"
            }],
            "numlist": [{
                "text": "Plus",
                "cls": "plus-arrow"
            }]
        }
    }
}
```

Einstellungen erweitern

Über die **conf.json** können Sie auch die Mandanten Einstellungen um weitere Tabs erweitern.

```
{
    "navigation": {
        "design": {
            "name": "Design",
            "title": "Design",
            "template": "site/admin/conf_design.html"
        },
        "social": {
            "name": "Soziale Netzwerke",
            "title": "Soziale Netzwerke",
            "template": "site/admin/conf_social.html"
        },
        "footer": {
            "name": "Fußbereich",
            "title": "Fußbereich",
            "template": "site/admin/conf_footer.html"
        }
    }
}
```

site_admin_misc

Die Einstellungen hier werden im **Site** Objekt gespeichert, so dass Sie diese immer parat haben. Alle Einstellungen aus **$site->site**, **$site->admin** und **conf.json** existieren in **$site->conf**. Das selbe gilt auch für **$page->conf**, wo einige Einstellungen zu finden sind. Das Template eines solchen Reiters sieht z.B. so aus:

```
<form name="extra" action="{$action_url}" method="post">
    <div align="center">
        <table class="table">
            <tr>
                <td colspan="2" class="cell">
                    {input type="combo" name="footer_links" title="Nützliche Links" full=true no_users=true no_rights=true no_text=true sitename=$conf_site->name}
                </td>
            </tr>
        </table>
    </div>
</form>
<script type="text/javascript">
{literal}
function do_load_extra() {
    set_input_value('footer_links', parent.get_conf('footer', 'links'));
}
function do_unload_extra() {
    parent.set_conf('footer', 'links', get_input_value('footer_links'));
}
{/literal}
</script>
```

Damit entfällt es einer Seite, z.B. der Startseite, Einstellmöglichkeiten zu geben, die man für alle Seiten benötigt und man somit im Code immer die Startseite ermitteln muss.

Weitere Methoden

Verwenden Sie folgende Methode um den ersten Wert aller Blöcke zu ermitteln.

```
{$page->getFirstValue("image1")}
```

Verwenden Sie folgende Methode um einen bestimmten Wert eines Blockes einer Orientierung zu ermitteln.

```
{$page->getValue("image1", $orient = "center", $index = 0)}
```

Verwenden Sie folgende Methode um Bilder **Retina ready** einzubinden und so auch höhere Pixeldichten optimal zu bedienen.

```
{picture src=$page->getFirstValue("image1") attr.alt=$page->field.name}
```