de·la·Haye
KOMMUNIKATIONSDESIGN

Modulentwicklung.

TYPOlight

Usertreffen 2009

Was Euch in den nächsten zwei Stunden erwartet:

Theorie	Planung eines TYPOlight-Moduls
	Das Ordner- und Dateisystem
	Der Modulgenerator
	- - Fragen, Ergänzungen & Kommentare hierzu - -
Praxis	Ein Beispielmodul
	Das Modul über den Generator anlegen
	Erstellung und Konfiguration des Moduls
Ergänzungen	Hooks
	Kleine Änderungen
	Ressourcen für Entwickler
Fazit	Offene Fragerunde & Abschlussdiskussion

Konfigurieren statt Programmieren

- TYPOlight setzt auf eine festgelegte Ordnerstruktur
- Module klinken sich in das System ein
- Core-Funktionalitäten können verändert oder erweitert werden, ohne dass der Originalcode verändert wird
- Module nutzen vorhandene Eigenschaften, Libraries, Variablen, Objekte etc.

Grundsätzliche Überlegungen bei der Modulentwicklung

- Geht es (teilweise) auch mit „Bordmitteln“?
  - ➔ Listenausgaben sind z.B. oft mit dem Modul *Auflistung* machbar.
- Kann eine vorhandene Funktionalität erweitert werden?
  - ➔ Bsp.: Erweiterung der Mitgliederdaten statt einem neuen Modul

Planung einer Erweiterung

- Ohne Planung kein Modul
- Umfang der Planung hängt von der Komplexität ab

Große Erweiterungen - umfangreiche Funktionen:

- Gesamtkonzept
- (UML-)Diagramme für Aktivitäten, Klassen, Objekte etc.
- Genaue Datenbank-Entwürfe
- Analyse ggf. bereits vorhandener Klassen und Funktionalitäten
- Prüfung: Sind ggf. separate Teil-Extensions sinnvoll?

Mittlere, „normale“ Erweiterungen:

- Grundkonzept
- Planung der Abläufe und Aktivitäten
- Entwurf der benötigten Tabellen

Planung einer Erweiterung

Und für kleinere Erweiterungen:

Benennungen von Erweiterungen und Variablen

- Namenskonflike vermeiden!
- Modulnamen können nur 1x vergeben werden
- Systemweit genutzte Variablen ebenfalls
- Präfixe sorgen für korrekte, eindeutige Zuordnungen
  - ➔ bei Modul- und Verzeichnisnamen
  - ➔ bei Variablen-Erweiterungen (z.B. in der Tabelle *tl_modules*)

	Schlecht	Besser
Modulname	MeineFilmsammlung	fm_MeineFilmsammlung
Variable in tl_modules	abteilung_template	fm_abteilung_template
Variable in Session	schuhgroesse	fm_schuhgroesse

Ausnahme: Erweiterungen, die möglichst als Letztes geladen werden sollen, könnten mit „*z_*“, „*zz_*“, „*zzz_*“ o.ä. benannt werden.

Das Grundgerüst: Ordner und Dateien

Verzeichnisstruktur in/TL_ROOT/system/modules/

mein_modul	
mein_modul.php	vom Modul benötigter PHP-Code
config	
config.php	allgemeine Konfiguation, Registrierung von BE- und FE-Modulen, einzubindendes Javascript, Hooks etc
database.sql	SQL-Definitionen aller benötigten Tabellen
dca	
tl_mein_modul.php	DCA + BE-Klassen für eigene Modultabelle
z.B. tl_module.php etc.	DCA-Ergänzungen für bestehende DCAs
languages	
de	
default.php	Sprachsnippets und -variablen für ...das Modul selbst, allgemeine Angaben etc
modules.php	...die Modulanzeige im BE
tl_mein_modul.php	...für FE-Module
z.B. tl_member.php	...ggf. Ergänzungen zu TL-eigenen Tabellen
templates	
mein_template.tpl	Ausgabetemplate(s)

Die Struktur kann um eigene Verzeichnisse erweitert werden, z.B. */html* für Icons.

DCA = Data Container Array

- Zentrales Element des TL-Backends
- Backend generiert sich aus den DCA der einzelnen Module
- Ermöglicht Pflege aller mit *tl_* beginnenden Datentabellen
- DCA-Erstellung und Manipulation:
  - → über Konfigurationsdateien im Modul
  - → on the Fly im Modul (Beispiel: Die Katalog-Extension)
  - → über die *dcaconfig.php* (siehe Ergänzungen)

Mögliche Abschnitte eines DCA:

- *config*
  - → Tabellenfestlegung, allgemeine Konfiguration
- *list*
  - → Auflistungsmöglichkeiten
  - → globale Operationen
  - → Datensatz-Operationen
- *palettes*
  - → Festlegung der pflegbaren Felder (Paletten)
- *subpalettes*
  - → Unter-Paletten
- *fields*
  - → Definitionen der Feld-Eingabemasken.

DCA: Gliederung


```
$GLOBALS['TL_DCA']['TABELLENNAME'] = array (
    'config' => array (
        ... ALLGEMEINE KONFIGURATION ...
    ),
    'list' => array (
        'sorting' => array (
            ... SORTIERUNGEN ...
        ),
        'label' => array (
            ... ANGABEN IN DER LISTENANSICHT ...
        ),
        'global_operations' => array (
            ... GLOBALE OPERATIONEN ...
        ),
        'operations' => array (
            ... OPERATIONEN PRO DATENSATZ ...
        )
    ),
    'palettes' => array (
        ... PALETTEN: WELCHE FELDER WERDEN GEPFLEGT? ...
    ),
    'subpalettes' => array (
        ... UNTERPALETTEN ...
    ),
    'fields' => array (
        ... FELDDEFINITIONEN ...
    )
);
```

- Genaue Angaben zu allen DCA-Optionen: http://dev.typolight.org

Der Modul-Generator

- Kann als Extension „developement“ installiert werden und steht dann als „Extension Creator“ im BE zur Verfügung
- Ist ein Hilfsmittel, um grundlegende Moduleigenschaften festzulegen
- Ist <u>keine</u> *klick-mich-ich-mach-ein-fertiges-Modul*-Erweiterung
- Erzeugt eine passende Verzeichnisstruktur mit rudimentären Dateien

→

/mein_modul.php
- Modul-Klasse mit Zuweisung der Template-Variable und der Methode compile() für den PHP-Code

/config
config.php
- Hilfskommentare

database.sql
- Basis-SQL für Tabellen

/dca
tl_mein_modul.php
- Rudimentäres DCA

/languages
/de
default.php
- Leere Bsp.-Variablen

modules.php
- Sprachen-Variablen

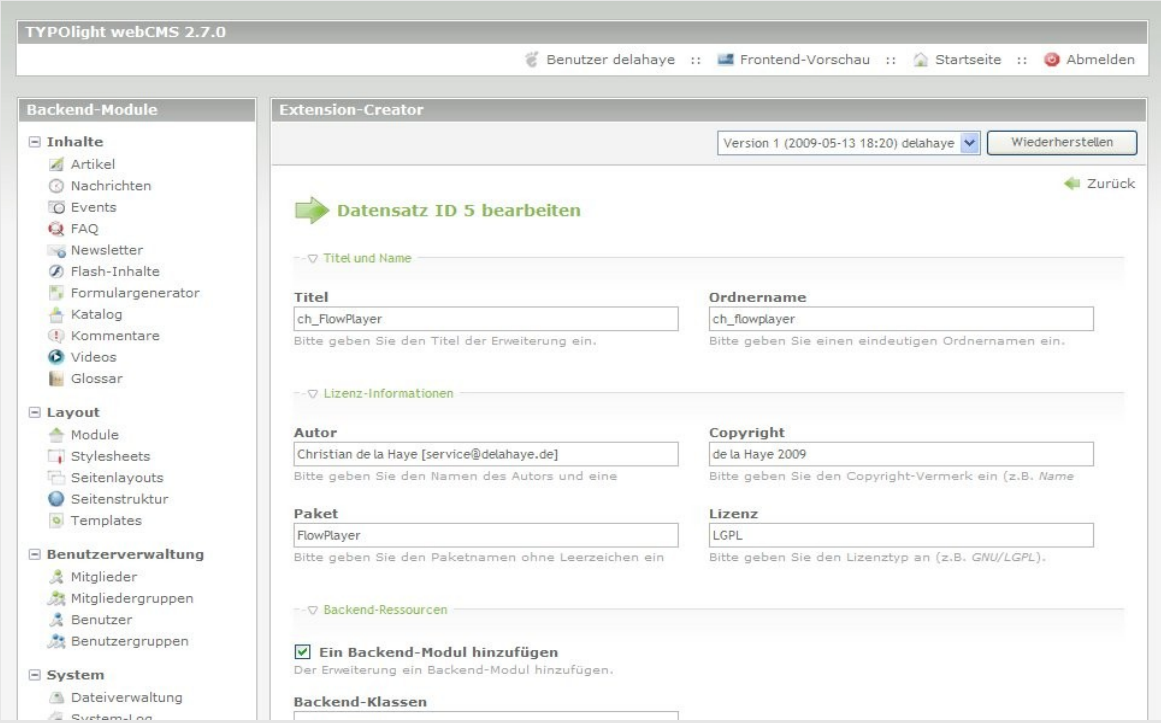
tl_mein_modul.php
- Sprachen-Variablen

/templates
mein_modul.php
- Leeres Template

Der Modul-Generator

Die erzeugten Dateien enthalten:

- die allgemeine Struktur
- alle angegebenen Klassen
- viele Kommentare mit Hilfetexten
- Basis-SQL für alle Tabellen
- Beispiele für die Variablennotation



TYPOlight

Usertreffen 2009:
Modulentwicklung

Der Modul-Generator

Was passiert bei nachträglichen Änderungen?

- Das Tool baut nur das Grundgerüst auf
- Neuerstellung = Änderungen werden überschrieben!

- Empfehlung, wenn Änderungen nötig sind:
  - ➔ Sicherung des bisherigen Standes inkl. der Anpassungen
  - ➔ Kopie im Modul-Generator in eine neue Erweiterung mit neuem Namen und anderem Ordner
  - ➔ Manuelle Übernahme bestehender Anpassungen in die neu erzeugten Dateien
  - ➔ Entfernen des alten Ordners in der TL-Installation

Fragen, Ergänzungen & Kommentare

Hier ist jetzt Platz und Zeit für

- Fragen
- Kommentare
- Anregungen
- Ergänzungen
- Hinweise aus der Entwicklergemeinde

... zur Theorie. Nach dem Praxisteil ist eine ausführliche Diskussion geplant.

Erweiterungs-Entwicklung in der Praxis

Was wir machen werden:

- Erstellung mit dem ***Modul-Generator***
- Typ ***Modul*** mit Parent-/Child-Tabellen erstellen
- Einen ***Hook*** nutzen (durchsuchbare Einzelseiten)
- Typ ***Content-Element*** erstellen
- ***DCA*** sehen & verstehen
- ***SQL-Tabellen*** neu erstellen & modifizieren
- ***Javascript*** im Seiten-Header platzieren
- ***Vorhandene Felder*** nutzen

Was wir nicht machen können:

- Alle DCA-Möglichkeiten durchgehen
- Den Code erst vor Ort entwickeln und Testen

Ein Beispielmodul

Beispielmodul: Der **FlowPlayer**

- Anzeige von FLV-, MP4- und weiteren Filmformaten
- Flexibel über Parameter erweiterbarer Player

Download: www.flowplayer.org

Das Modul steht ab sofort auch im Repository zur Verfügung.

Features

- als Modul:
  - Bündelung von FLV-Filmen in Listen mit Voransicht
  - Verzweigung in Filmansicht
  - Autostart ja/nein
  - Player mit vorgeschaltetem JPG-Screen ja/nein
  - erweiterte Parameter
- als Content-Element:
  - Autostart ja/nein
  - Player mit vorgeschaltetem JPG-Screen ja/nein
  - erweiterte Parameter

2 Module, 1 Content-Element

- Modul 1: ***FlowPlayer Liste***
  - → Liste mit Filmen in bestimmter Reihenfolge
  - → Verlinkung auf Detailseite
  - → Vorschau-Bild
  - → Titel, Kurzbeschreibung

- Modul 2: ***FlowPlayer Reader***
  - → Player mit Film aus Liste
  - → Vorschau-Bild kann in Player erscheinen (ohne Autostart)

- Content-Element: ***FlowPlayer***
  - → Player mit Einzelfilm
  - → Vorschau-Bild kann in Player erscheinen (ohne Autostart)

Tabellen

tl_module

- vorhandene TL-Tabelle
- Daten der Module
  - ➔ Detailseite
    - ➔ Array der Playlisten
    - ➔ Automatischer Filmstart
    - ➔ Größe des Players

tl_content

- vorhandene TL-Tabelle
- Daten eines (Einzel-)films
  - ➔ Titel des Films
  - ➔ Beschreibung
  - ➔ Filmdatei
  - ➔ Daten für (Vorschau-)Bild
  - ➔ Erweiterte Parameter für Film

Tabellen

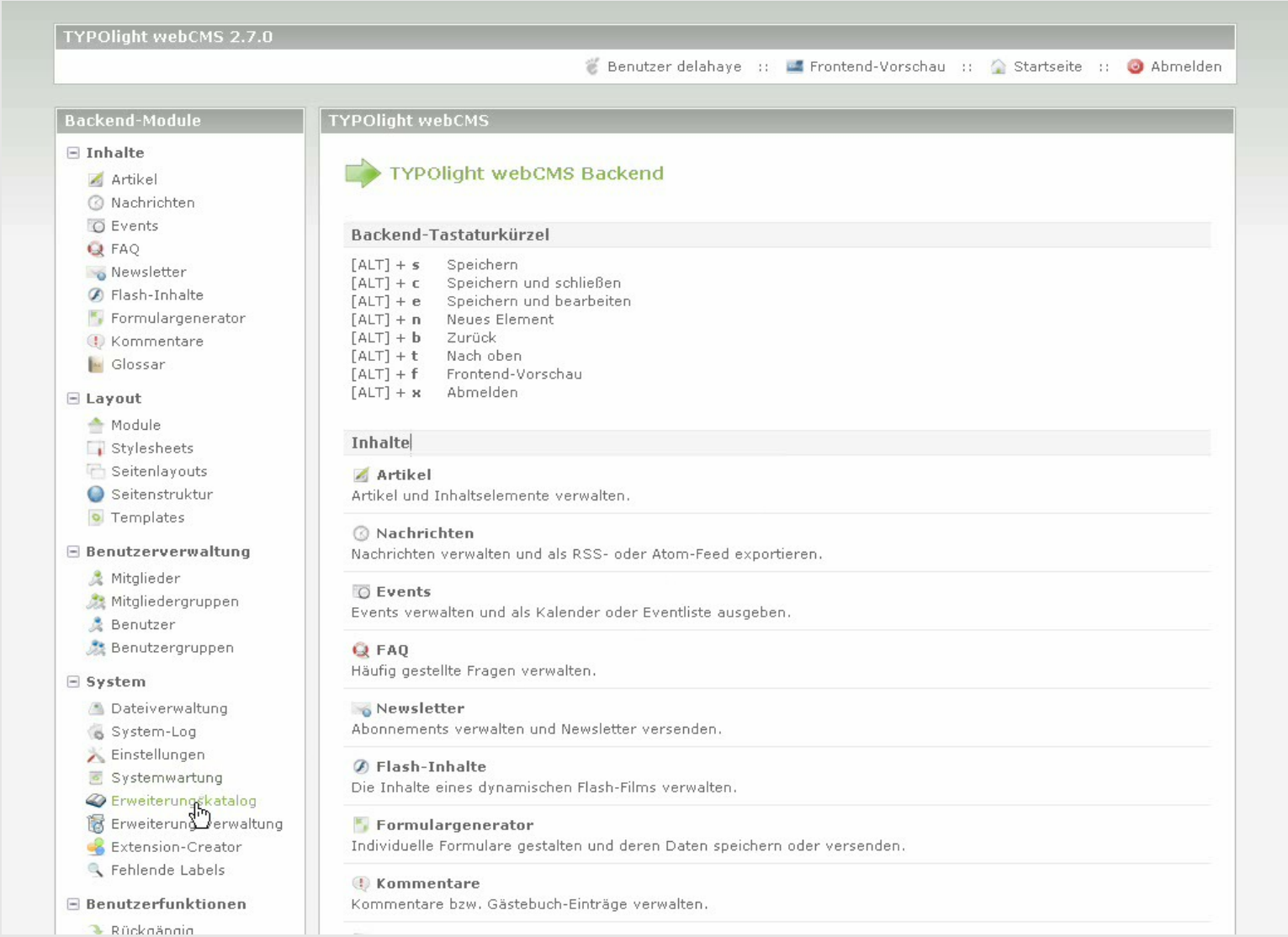
tl_ch_flowplayer

- Daten der Playlisten
  - → nur Titel erforderlich

tl_ch_flowplayer_playlist

- Daten der einzelnen Filme in der Playlist
  - → Bezug zu *tl_ch_flowplayer*
  - → Freie Sortierung
  - → Veröffentlichung
  - → Titel des Films
  - → Alias für Links
  - → Beschreibung
  - → Filmdatei
  - → Daten für (Vorschau-)Bild
  - → Erweiterte Parameter für Film

Das Modul über den Generator anlegen



TYPOlight

Usertreffen 2009:
Modulentwicklung

Anpassung: config.php

- In der Datei *config/config.php* registrieren wir 1 BE-Modul ...

```
array_insert($GLOBALS['BE_MOD']['content'], 15, array (
    'ch_flowplayer' => array (
        'tables' => array('tl_ch_flowplayer',
                'tl_ch_flowplayer_playlist'),
        'icon' => 'system/modules/ch_flowplayer/html/icon.gif')));
```

- ... 2 FE-Module ..

```
array_insert($GLOBALS['FE_MOD']['miscellaneous'], 5, array (
    'ch_flowplayer_reader' => 'Module_chFlowPlayerReader',
    'ch_flowplayer_list' => 'Module_chFlowPlayerList'));
```

- ... 1 Content-Element ...

```
array_insert($GLOBALS['TL_CTE']['images'], 3, array (
    'ch_flowplayer' => 'Content_chFlowPlayer'));
```

- ... fügen 1 Javascript in den Seitenheader ein ...

```
$GLOBALS['TL_JAVASCRIPT'][] = 'system/modules/ch_flowplayer/
html/flowplayer/flowplayer-3.1.0.min.js';
```

- ... und machen die Detailseiten durchsuchbar.

```
$GLOBALS['TL_HOOKS']['getSearchablePages'][] = array(
'Module_chFlowPlayer', 'getSearchablePages');
```

Anpassung: database.sql

- *tl_module* & *tl_content*
  - ➔ Alle Felder entfernen (bereits vorhanden) + neue einfügen:

```
CREATE TABLE `tl_module` (
  `ch_playlists` blob NULL,
  `ch_autoplay` char(1) NOT NULL default '1',
  `ch_playersize` varchar(64) NOT NULL default ''
) ENGINE=MyISAM DEFAULT CHARSET=utf8;

CREATE TABLE `tl_content` (
  `ch_movie` varchar(255) NOT NULL default '',
  `ch_description` text NULL,
  `ch_autoplay` char(1) NOT NULL default '1',
  `ch_preview` varchar(255) NOT NULL default '',
  `ch_playersize` varchar(255) NOT NULL default '',
  `ch_params` text NULL
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

- *tl_ch_flowplayer*
  - ➔ pid, sorting, key pid entfernen (nicht benötigt) + einfügen:

```
CREATE TABLE `tl_ch_flowplayer` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `tstamp` int(10) unsigned NOT NULL default '0',
  `title` varchar(64) NOT NULL default ''
  PRIMARY KEY  (`id`),
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Anpassung: database.sql

- *tl_ch_flowplayer_playlist*
  - ➔ Felder einfügen

```
CREATE TABLE `tl_ch_flowplayer_playlist` (
  `id` int(10) unsigned NOT NULL auto_increment,
  `pid` int(10) unsigned NOT NULL default '0',
  `sorting` int(10) unsigned NOT NULL default '0',
  `tstamp` int(10) unsigned NOT NULL default '0',
  `published` char(1) NOT NULL default '',
  `title` varchar(64) NOT NULL default '',
  `alias` varchar(128) NOT NULL default '',
  `description` text NULL,
  `movie` varchar(32) NOT NULL default '',
  `addImage` char(1) NOT NULL default '',
  `singleSRC` varchar(255) NOT NULL default '',
  `size` varchar(64) NOT NULL default '',
  `alt` varchar(255) NOT NULL default '',
  `caption` varchar(255) NOT NULL default '',
  `floating` varchar(32) NOT NULL default '',
  `imagemargin` varchar(255) NOT NULL default '',
  `fullsize` char(1) NOT NULL default '',
  `params` text NULL,
  PRIMARY KEY  (`id`),
  KEY `pid` (`pid`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Anpassung: DCA *tl_module.php & tl_content.php*

- Für *dca/tl_module.php* und *dca/tl_content.php* müssen nur Paletten und Eingabemasken für noch nicht in den TL-Tabellen vorhandene Felder definiert werden. Z.B.:

```
$GLOBALS['TL_DCA']['tl_module']['palettes']['ch_flowplayer_reader'] = '
    {title_legend},name,headline,type;
    {config_legend},ch_playlists,ch_autoplay,ch_playersize;
    {protected_legend:hide},protected;
    {expert_legend:hide},guests,cssID,space';

$GLOBALS['TL_DCA']['tl_module']['palettes']['ch_flowplayer_list'] = '
    {title_legend},name,headline,type,jumpTo;
    {config_legend},ch_playlists;
    {protected_legend:hide},protected;
    {expert_legend:hide},guests,cssID,space';



$GLOBALS['TL_DCA']['tl_module']['fields']['ch_playlists'] = array (
    'label'      => &$GLOBALS['TL_LANG']['tl_module']['ch_playlists'],
    'exclude'    => true,
    'inputType'  => 'checkboxWizard',
    'foreignKey' => 'tl_ch_flowplayer.title',
    'eval'       => array('multiple'=>true, 'mandatory'=>true)
);
```

Anpassung: DCA *tl_ch_flowplayer.php* & *tl_ch_flowplayer_playlist.php*

- Für *dca/tl_ch_flowplayer.php* und *dca/tl_ch_flowplayer_playlist.php* müssen vollständige DCA definiert werden. Z.B.:

```
$GLOBALS['TL_DCA']['tl_ch_flowplayer'] = array
(
    'config' => array
    (
        'dataContainer'    => 'Table',
        'ctable'           => array('tl_ch_flowplayer_playlist'),
        'switchToEdit'     => true, // 'Speichern und bearbeiten'
        'enableVersioning'=> true // Mit Versionierung
    ),
    'list' => array
    (
        'sorting' => array
        (
            'mode'        => 1, // Standard-Sortierung
            'fields'      => array('title'),
            'flag'        => 1, // Nach Anfangsbuchstaben aufsteigend
            'panelLayout' => 'filter;search,limit',
                // Einschänkungen u. Felder im Kopf der Liste
        ),
        'label' => array // In Liste angezeigte Daten
        (
            'fields' => array('title'),
            'format' => '%s' // Ersetzte Anzeigedaten
        ),

                                    ...
```

Anpassung: DCA *tl_ch_flowplayer.php & tl_ch_flowplayer_playlist.php*


```
                                        ...

    'global_operations' => array
    (
        'all' => array // Alle bearbeiten
        (
            'label'       => &$GLOBALS['TL_LANG']['MSC']['all'],
            'href'        => 'act=select',
            'class'       => 'header_edit_all',
            'attributes'  => 'onclick="Backend.getScrollOffset();"'
        )
    ),
    'operations' => array
    (
        'edit' => array // Element bearbeiten
        (
            'label'   => &$GLOBALS['TL_LANG']
                                ['tl_ch_flowplayer']['edit'],
            'href'    => 'table=tl_ch_flowplayer_playlist',
                                // In Detail-Tabelle wechseln
            'icon'    => 'edit.gif'
        ),
        'copy' => array // Element kopieren
        (
            'label'   => &$GLOBALS['TL_LANG']
                                ['tl_ch_flowplayer']['copy'],
            'href'    => 'act=copy',
            'icon'    => 'copy.gif'
        ),

                                        ...
```

Anpassung: DCA *tl_ch_flowplayer.php & tl_ch_flowplayer_playlist.php*


```
                                        ...

            'delete' => array // Element löschen
            (
                'label'       => &$GLOBALS['TL_LANG']
                                   ['tl_ch_flowplayer']['delete'],
                'href'        => 'act=delete',
                'icon'        => 'delete.gif',
                'attributes'  => 'onclick="if (!confirm(
                    \'' . $GLOBALS['TL_LANG']['MSC']
                    ['deleteConfirm'] . '\')) return false;
                    Backend.getScrollOffset();"'
                    // Sicherheitsabfrage
            ),
            'show' => array // Element zeigen
            (
                'label'  => &$GLOBALS['TL_LANG']
                                ['tl_ch_flowplayer']['show'],
                'href'   => 'act=show',
                'icon'   => 'show.gif'
            )
        )
    ),
    'palettes' => array // Pflegbare Felder
    (
        'default' => '{title_legend},title'
            // Legende des (klappbaren) Bereichs
            // Feldnamen mit Unterteilungen (, und ; für Linie)
    ),

                                        ...
```

```
                                        ...

    'fields' => array // Einzelne Felddefinition
    (
        'title' => array
        (
            'label'       => &$GLOBALS['TL_LANG']
                                    ['tl_ch_flowplayer']['title'],
                                    // Feldüber- und unterschrift
            'search'      => true, // nach Feld kann gesucht werden
            'exclude'     => true, // aber nur Admins sehen es!
            'inputType'   => 'text', // Was für ein Feld?
            'eval'        => array('mandatory'=>true, 'maxlength'=>255)
            // Diverse Angaben zum Feld. Pflichtfeld? Wie viele Zeichen?
        )
    )
);
```

- Viele weitere DCA-Möglichkeiten vorhanden
- **Wichtig: Callback-Funktionen**
  - Funktionen, die eigene Überprüfungen, DCA-Änderungen 'on the fly' oder Zwischenschritte ermöglichen - ähnlich Hooks. Z.B:
    - Füllen einer Auswahlliste mit Werten aus einer eigenen Tabelle, spezielle Zugriffsrechte implementieren, Dateien nach (User-)Aktionen erstellen ...
  - An vielen Stellen im DCA nutzbar (siehe http://dev.typolight.org)

DCA: Klassen und Methoden ergänzen

- In *dca/tl_ch_flowplayer_playlist.php* fehlen noch eine Methode zur Aliaserstellung und eine zur Elementdarstellung im BE:

```
class tl_ch_flowplayer_playlist extends Backend
{
    public function generateAlias($varValue, DataContainer $dc)
    {
        ... GENERIERE ALIAS AUS DEM TITEL ...
    }

    public function listMovies($arrRow)
    {
        ... ZEIGT TYP DES ELEMENTS IM BE ...
    }
}
```

- Ohne Aliaserstellung ist eine FE-Verlinkung zur Detailseite nur numerisch möglich (z.B. *www.irgendwas.de/filme/film/2215.html*)
- Hier sind auch weitere BE-Klassenerweiterungen möglich

Anpassung: Sprachendateien

- *languages/de/default.php*
  - → Bezeichnungen für CE und Modul ch_flowplayer setzen

```
$GLOBALS['TL_LANG']['MOD']['ch_flowplayer'] = array(
'Videos','FLV-Filme mit dem FlowPlayer verwalten.');

$GLOBALS['TL_LANG']['CTE']['ch_flowplayer'] = array(
'Video','FLV-Film mit dem FlowPlayer wiedergeben.');
```

- *languages/de/modules.php*
  - → Bezeichnungen für FE-Module setzen

```
$GLOBALS['TL_LANG']['FMD']['ch_flowplayer_reader'] =  array(
'FlowPlayer Reader',
'Mit diesem Modul binden Sie den
    FlowPlayer zur Wiedergabe einer Playlist ein.');

$GLOBALS['TL_LANG']['FMD']['ch_flowplayer_list'] = array(
'FlowPlayer Liste',
'Mit diesem Modul binden Sie eine
    Playliste für den FlowPlayer ein.');
```

Anpassung: Sprachendateien

- *languages/de/tl_ch_flowplayer.php*
  - ➔ Felder, Buttons und Legenden

```
$GLOBALS['TL_LANG']['tl_ch_flowplayer']['title'] = array(
    'Titel', 'Titel der Playlist');

$GLOBALS['TL_LANG']['tl_ch_flowplayer']['tstamp'] = array(
    'Änderungsdatum', 'Datum und Uhrzeit der letzten Änderung');


$GLOBALS['TL_LANG']['tl_ch_flowplayer']['new'] = array(
    'Neue Playlist', 'Eine neue Playlist anlegen');

$GLOBALS['TL_LANG']['tl_ch_flowplayer']['edit'] = array(
    'Playlist bearbeiten', 'Playlist ID %s bearbeiten');

$GLOBALS['TL_LANG']['tl_ch_flowplayer']['copy'] = array(
    'Playlist duplizieren', 'Playlist ID %s duplizieren');

$GLOBALS['TL_LANG']['tl_ch_flowplayer']['delete'] = array(
    'Playlist löschen', 'Playlist ID %s löschen');

$GLOBALS['TL_LANG']['tl_ch_flowplayer']['show'] = array(
    'Playlistdetails', 'Details der Playlist ID %s anzeigen');


$GLOBALS['TL_LANG']['tl_ch_flowplayer']['title_legend'] = 'Titel';
```

Anpassung: Sprachendateien

- *languages/de/tl_module.php*
  - ➔ Felder

```
$GLOBALS['TL_LANG']['tl_module']['ch_playlists'] = array(
'Playlisten','Wählen Sie die Playlisten mit den
    gewünschten Filmen aus.');

$GLOBALS['TL_LANG']['tl_module']['ch_autoplay'] = array(
'Autoplay', 'Sollen die Filme automatisch starten?');

$GLOBALS['TL_LANG']['tl_module']['ch_playersize'] = array(
'Breite und Höhe des Players', 'Geben Sie Breite
    und Höhe des FlowPlayers ein.');
```

- Der Aufbau der Sprachendateien ist ansonsten weitgehend gleich
- Weitere Sprachen werden durch exakt analoge Dateien in entsprechend benannten Verzeichnissen erstellt.

Anpassung: Logik Content-Element

- *Content_ch_FlowPlayer.php*
  - → Stellt ein Content-Element FlowPlayer im FE zur Verfügung
  - → Template-Variable setzen
  - → Code für das Template einfügen

```
class Content_chFlowPlayer extends Module
{

    /**
     * Template
     * @var string
     */
    protected $strTemplate = 'ce_ch_flowplayer';


    /**
     * Generate module
     */
    protected function compile()
    {

        PHP-LOGIK FÜR DAS TEMPLATE

    }
}
```

Anpassung: Logik Modul Liste

- *Module_ch_FlowPlayerList.php*
  - → Stellt ein Modul FlowPlayer Liste im FE zur Verfügung
  - → Template-Variable setzen
  - → Code für BE-Darstellung einfügen

```
class Module_chFlowPlayerList extends Module
{
    protected $strTemplate = 'mod_ch_flowplayer_list';

    public function generate()
    {
        if (TL_MODE == 'BE')
        {
            DARSTELLUNG DES MODULS IM BE
        }

        $this->ch_playlists = deserialize($this->ch_playlists, true);
        if (!is_array($this->ch_playlists)
            || count($this->ch_playlists) < 1)
        {
            NICHT ZEIGEN, WENN KEINE PLAYLISTS VORHANDEN
        }

        return parent::generate();
    }
                                    ...
```

Anpassung: Logik Modul Liste

- *Module_ch_FlowPlayerList.php*
  - ➔ Code für das Template einfügen
  - ➔ Variable für mögliche Zielseiten setzen
  - ➔ Zielseite für Detailansicht ermitteln

```
                                        ...

    protected function compile()
    {
        PHP-LOGIK FÜR DAS TEMPLATE
    }

    protected $arrTargets = array();

    protected function generateMovieLink(Database_Result $objMovies)
    {
        GENERIERUNG VON LINKS ZUR DETAILSEITE
    }

}
```

Anpassung: Logik Modul Reader

- *Module_ch_FlowPlayerReader.php*
  - Stellt ein Modul FlowPlayer Reader im FE zur Verfügung
  - Template-Variable setzen
  - Code für BE-Darstellung analog zu *Module_ch_FlowPlayerList.php* einfügen
  - Code für das Template einfügen

```
class Module_chFlowPlayerReader extends Module
{
    protected $strTemplate = 'mod_ch_flowplayer_reader';

    public function generate()
    {
        DARSTELLUNG DES MODULS IM BE
        ANALOG ZU Module_ch_FlowPlayerList.php
    }

    protected function compile()
    {
        PHP-LOGIK FÜR DAS TEMPLATE
    }
}
```

Anpassung: Logik Modul FlowPlayer

- *Module_ch_FlowPlayer.php*
  - Bindet die Detailseiten der Playlisten in die TL-Suche ein (analog zu de News, den FAQ etc)
  - Enthält die Methode, die in der config/config.php für den Hook registriert wurde
  - Änderung: Extends Frontend!
  - Änderungen: Variable $strTemplate u. Methode compile() raus
  - Code für die Ermittlung möglicher Film-Detailseiten einfügen

```
class Module_chFlowPlayer extends Frontend
{

    public function getSearchablePages($arrPages, $intRoot=0)
    {
        PHP-LOGIK ZUR SEITENERMITTLUNG
    }
}
```

Anpassung: Template Content Element FlowPlayer

- *templates/ce_ch_flowplayer.tpl*
  - Grundaufbau analog zu fast allen TL-CE-Templates
  - Gliederung des Templates:
    - Block mit Template-Klasse, Id & Headline (in praktisch allen TL-CE-Elementen gleich)
    - Ausgabecode für den Player, hierbei:
      - Einbindung des Players
      - Javascript zur Player-Steuerung mit
        - entweder gewähltem Startbild oder Autoplay
        - oder freien Parametern
  - Kurzbeschreibung des Films

Anpassung: Template Modul FlowPlayer Liste

- *templates/mod_ch_flowplayer_list.tpl*
  - Grundaufbau analog zu fast allen TL-Modul-Templates
  - Auskommentierung der Ausgabe für die TL-Suche

  - Gliederung des Templates:
    - Block mit Template-Klasse, Id & Headline (in praktisch allen TL-Modulen gleich)
    - Schleife, die alle möglichen Playlists durchläuft mit:
    - Schleife, die alle Filme der aktuellen Playlist auflistet mit:
    - Vorschau-Bild
    - Kurzbeschreibung
    - Link

Anpassung: Template Modul FlowPlayer Liste

Fertiges Template *templates/mod_ch_flowplayer_list.tpl*:

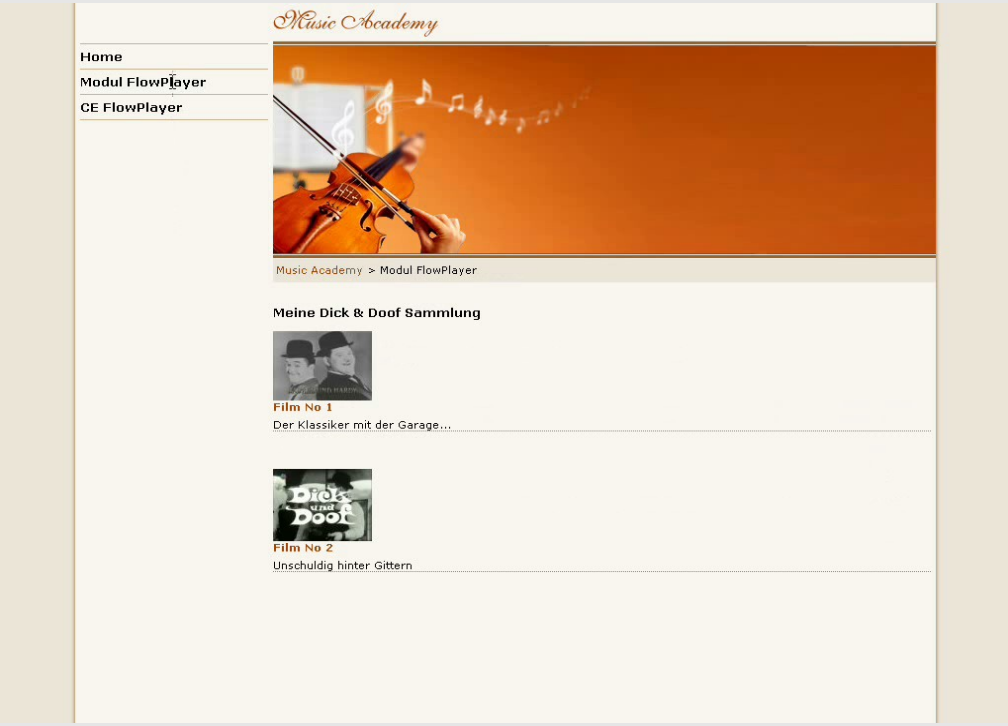
```
<!-- indexer::stop -->
<div class="<?php echo $this->class; ?> block"<?php echo $this->cssID; ?><?php if ($this-
>style): ?> style="<?php echo $this->style; ?>"<?php endif; ?>>
<?php if ($this->headline): ?>
  <<?php echo $this->hl; ?>><?php echo $this->headline; ?></<?php echo $this->hl; ?>>
<?php endif; ?>
<?php foreach($this->playlists as $playlist): ?>
  <div class="playlist <?php echo $playlist['class']; ?>">
  <?php foreach($playlist['items'] as $entry): ?>
    <div class="movie <?php echo $entry['class']; ?>">
    <?php if ($entry['addImage']): ?>
      <div class="image_container"<?php if ($entry['margin'] || $entry['float']): ?>
                  style="<?php echo $entry['margin'] . $entry['float']; ?>"<?php endif; ?>>
      <?php if ($entry['fullsize']): ?>
        <a href="<?php echo $entry['href']; ?>" title="<?php echo $entry['alt']; ?>"
          rel="lightbox">
      <?php endif; ?>
      <img src="<?php echo $entry['src']; ?>"<?php echo $entry['imgSize']; ?>
         alt="<?php echo $entry['alt']; ?>" />
      <?php if ($entry['fullsize']): ?></a><?php endif; ?>
      <?php if ($entry['caption']): ?>
        <div class="caption"><?php echo $entry['caption']; ?></div>
      <?php endif; ?>
      </div>
    <?php endif; ?>
    <h2><a href="<?php echo $entry['link']; ?>" title="<?php echo $entry['title']; ?>">
        <?php echo $entry['title']; ?></a></h2>
    <?php if($entry['description']) : ?>
      <div class="description"><?php echo $entry['description']; ?></div>
    <?php endif; ?>
    </div>
  <?php endforeach; ?>
  </div>
<?php endforeach; ?>
</div>
<!-- indexer::continue -->
```

Anpassung: Template Modul FlowPlayer Reader

- *templates/mod_ch_flowplayer_reader.tpl*
  - Aufbau praktisch 1:1 wie für das Content-Element
  - Grundaufbau analog zu fast allen TL-Detail-Templates
  - Gliederung des Templates:
    - Block mit Template-Klasse, Id & Headline (in praktisch allen TL-Modulen gleich)
    - Ausgabecode für den Player, hierbei:
      - Einbindung des Players
      - Javascript zur Player-Steuerung mit
        - entweder gewähltem Startbild oder Autoplay
        - oder freien Parametern
  - Kurzbeschreibung des Films
  - Backlink zur letzten besuchten Seite (i.d.R. die Liste)

Das Ergebnis

- Nach Fertigstellung und Installation des Moduls kann es beliebig in TYPOlight genutzt werden:



- Die Erweiterung kann als „ch_FlowPlayer“ ab sofort im Repository installiert oder heruntergeladen werden.

Hooks nutzen

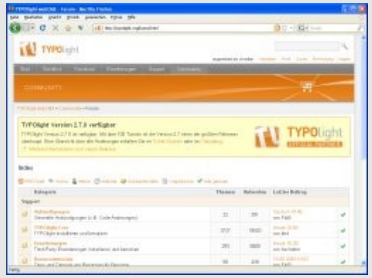
- Hooks = Stellen, an denen eigene Logik in bestehende Abläufe eingefügt werden kann
  - → Vor oder nach bestimmter Aktion wird verzweigt
    - → externe Logik aus externen Dateien/Klassen abgearbeitet
  - → und wieder in die restliche Verarbeitung zurückgesprungen
- Hooks werden in der Datei *config.php* eines Moduls registriert
- Das Handbuch unter http://dev.typolight.org zeigt genau, welche Hooks wie im Core zur Verfügung stehen.
- Module können eigene Hooks enthalten, um selbst erweiterbar zu werden.

Kleine Änderungen

- Kleine Änderungen sind auch ohne eigene Module machbar
- Updatesichere Speicherung im Ordner TL_ROOT/system/config/
- Werden als letztes Element eingelesen, können ergo z.B. alle globalen Variablen überschreiben
- Empfehlung: Wirklich nur für kleine Änderungen nutzen
- Nutzbare Dateien:
  - *localconfig.php*
    - Ggf. benötigte allgemeine Konfigurationsvariablen
  - *dcaconfig.php*
    - Änderungen an bestehenden DCA
    - Bsp:
-Tag im Seitentitel (für's Menü) erlauben:

      $GLOBALS['TL_DCA']['tl_page']['fields']['title']['eval']['preserveTags'] = true;
  - *langconfig.php*
    - Einzelne sprachenspezifische Änderungen
    - Bsp: Die Text-Ausgabe eines anderen Moduls zu ändern

Ressourcen für Entwickler



- Entwicklerserver *http://dev.typolight.org*
  - Handbuch
  - Developers Guide
    - Table Configuration
      - Genaue Definition aller in den DCA benutzbaren Optionen und Einstellungen
    - Hooks
      - In TL selbst enthaltene Hooks mit Erläuterungen
    - API
      - Hier kann man sich alle Core-Klassen und Methoden im Detail ansehen.
  - User Tutorials
- Das Forum *http://typolight.org/board.html*
- SelfPHP http://selfphp.de

Auch nicht vergessen: Bestehende Erweiterungen & Kollegen!

de·la·Haye
KOMMUNIKATIONSDESIGN

Vielen Dank für Eure Aufmerksamkeit!

Kontakt

de la Haye Kommunikationsdesign
Inh. Christian de la Haye
Karl-Seepe-Str. 12
41747 Viersen

0 21 62 - 50 12 25
service@delahaye.de

TYPOlight

Usertreffen 2009