

# UtiCMS

**Entwickler Handbuch**

Inhaltsverzeichnis.....	2
Vorwort.....	4
Hello World.....	4
Code Konventionen.....	4
Admin Modul.....	5
Filter.....	6
Einstellungen speichern und laden.....	7
Weiter Hooks.....	7
API Referenz.....	8
user api.....	13
Spellcheck API.....	16
Anti-Spam API.....	16
Datenbankfunktionen.....	16
Kryptographie Funktionen.....	17
file_get_contents_wrapper.....	17
HTML Funktionen.....	18

Logger Funktionen.....	18
String Funktionen.....	18
Version des CMS abfragen.....	19
Templates.....	20
Template API.....	20
Datenbankstruktur.....	26

## VORWORT

Mit UliCMS lassen sich auch in kurzer Zeit funktionale Erweiterungen und Themes entwickeln. Dieses Handbuch soll in die Programmierung in UliCMS einführen.

Es wird vorausgesetzt, dass Sie bereits Kenntnisse in PHP und HTML haben.

## HELLO WORLD

Legen Sie unter **modules/** einen Ordner an, der den Namen Ihres Moduls hat. Es dürfen alle Zeichen vorkommen, die auch in PHP-Funktionsnamen vorkommen dürfen.

Damit ein UliCMS Modul vom System erkannt wird, muss im Modulordner die Datei `name_des_moduls_main` vorhanden sein.

In der Datei muss die `name_des_moduls_render`-Methode vorhanden sein. Diese muss den HTML-Code als String zurückgeben, der ausgegeben werden soll, wenn man das Modul über den Code zum Einbetten in eine Seite einbindet.

Diese Funktion muss auch vorhanden sein, wenn das Modul keine **Frontend-Ausgabe** hat.

```
<?php
function hello_world_render(){
    return „<p>Hallo Welt</p>“;
}
?>
```

## CODE KONVENTIONEN

Statt **mysql\_query** verwenden Sie bitte bei der Entwicklung die Funktion **db\_query**.

Dabei handelt es sich um eine Wrapper-Funktion, die es ermöglicht, für das Debugging alle Datenbankabfragen zu loggen. Fügen Sie folgende Zeile in **die cms-config.php** ein, um den SQL-Logger zu aktivieren:

```
var $query_logging = true;
```

Die SQL-Logdateien werden unter `[DOCUMENT_ROOT]/content/log/db` abgelegt und nach dem heutigen Datum benannt.

## ADMIN MODUL

Module können den Administrationsbereich des CMS mit Funktionen wie z.B. Einstellungsmöglichkeiten erweitern.

Legen Sie dazu bitte die Datei **name\_meines\_moduls\_admin.php** an.

```
<?php
// Name des Moduls als Überschrift
define("MODULE_ADMIN_HEADLINE", "Name meines Moduls");

// Für die Möglichkeit die Rechte für den Zugriff auf dieses Modul per Konfigurations-
variable zu setzen
$required_permission = getconfig("name_meines_moduls_required_permission");

// Wenn keine eigenen Berechtigungseinstellungen gesetzt sind, soll dieses Modul für Re-
dakteure und Admins
// zugreifbar sein. Siehe Administrator Handbuch Abschnitt Benutzerrollen.
if($required_permission === false){
    $required_permission = 40;
}

// Damit setzen wir die Berechtigung für dieses Modul.
define(MODULE_ADMIN_REQUIRED_PERMISSION, $required_permission);

// Diese Methode gibt die Administrationsoberfläche des Moduls aus, sofern der Nutzer
die nötige Berechtigung hat, ansonsten kommt eine Fehlermeldung wie „Zugriff verwei-
gert“.

function name_meines_moduls_admin(){
?>
    <p>Und schon haben wir unser erstes Admin Modul. Hier könnten Sie z.B. ein Formular
mit den Einstellungen des Moduls einfügen.</p>
<?php
}
?>
```

## FILTER

Mit Hilfe von **Filtern** ist es möglich, mit einem Modul den Content einer Seite, den Seitentitel und die Metadaten zu überschreiben.

Es gibt folgende Filter:

- `name_des_modules_seo_title_filter`
- `name_des_moduls_seo_meta_description_filter`
- `name_des_moduls_blog_seo_meta_keywords_filter`

Legen Sie eine PHP-Datei mit dem Namen eines Filters an und fügen Sie eine Funktion ein, die genauso wie der Filter heißt:

```
<?php

function mein_modul_seo_title_filter($txt){

    return $txt;

}

?>
```

Der erste Parameter (hier **\$txt** genannt) enthält den zu filternden Text (im Beispiel den Seitentitel).

Die Funktion muss als Rückgabewert den gefilterten Text zurückgeben.

Man kann z.B. **Platzhalter im Titel ersetzen** oder den Titel sogar ganz überschreiben.

Filter werden z.B. vom Modul **blog\_seo** genutzt.

## EINSTELLUNGEN SPEICHERN UND LADEN

Es gibt drei verschiedene Methoden in UliCMS um Konfigurationsvariablen zu modifizieren:

```
setconfig(„name_der_einstellung“, „wert“); // Setzt name_der_einstellung auf wert.
```

```
getConfig(„name_der_einstellung“); // gibt entweder den Wert von name_der_einstellung  
aus oder false
```

```
deleteconfig(„name_der_einstellung“); // löscht „name_der_einstellung“
```

Die Werte werden in der Datenbanktabelle **ulicms\_settings** als mediumtext gespeichert. Der Name einer Konfigurationseinstellung darf maximal 255 Zeichen lang sein.

## WEITER HOOKS

### **mein\_modul\_head**

Dieses Template wird beim Aufruf von `base metas()` ausgeführt.  
Es dient dazu Dinge im `<head>` einer Seite einzufügen.

### **mein\_modul\_before\_content**

Wird ausgeführt vor der Ausgabe von `content()`.

### **mein\_modul\_after\_content**

Nach der Ausgabe des Contents. Kann z. B. dafür genutzt werden, eine Bewertungsfunktion oder zum Inhalt des Artikels relevante Dinge auszugeben.

### **mein\_modul\_cron**

Cron-Hook. Dient dafür, um Vorgänge auszuführen, die beim Aufruf jeder Seite ausgeführt werden sollen.  
Diese Hook wird aufgerufen, nach dem der letzte HTML-Code ausgegeben wurde.

## API REFERENZ

### **is\_admin\_dir()**

Befinden wir uns gerade im Admin-Ordner?

**Returns:** Boolean

---

### **checkForUpdates()**

Prüft ob neue Updates da sind.

Gibt entweder einen String mit einem Infotext über die neue Version aus oder false zurück.

**Returns:** String oder Boolean

---

### **getModuleAdminSelfPath()**

gibt den Wert von \$\_SERVER["REQUEST\_URI"] mit entfernten Anführungszeichen zurück

**Returns:** String

---

### **buildCacheFilePath(\$request\_uri)**

Setzt den Pfad zu der Cache-Datei für \$request\_uri zusammen

**Returns:** String



### **SureRemoveDir(\$dir, \$DeleteMe)**

Leert den Inhalt des Ordners \$dir rekursiv

\$DeleteMe ist eine Boolean.

Wenn \$DeleteMe true ist, wird das geleerte Verzeichnis anschließend selbst gelöscht.

**Returns:** Void

---

### **buildSeoUrl(\$page)**

Setzt den relative Pfad zu \$page zusammen.

Wenn \$page nicht gesetzt ist, wird der Pfad zum Sytemnamen der aktuellen Seite zurückgegeben

**Returns:** String

### **getModulePath(\$module)**

Gibt den Relativen Pfad zum Modulordner vom Modul \$module aus.

\$module ist ein String.

**Returns:** String

---

### **getModuleAdminFilePath (\$module)**

Gibt den Pfad zur Admin PHP-Datei vom Modul \$module zurück.

**Returns:** String

---

### **getModuleMainFilePath (\$module)**

Gibt den Pfad zur main-Datei des Moduls \$module zurück.

**Returns:** String

### **getModuleUninstallScriptPath(\$module)**

Gibt den Pfad zum uninstall-Script des Moduls \$module zurück.

Das Uninstall-Script wird bei der Deinstallation eines Moduls vor dem entfernen des Modulordners aufgerufen.

**Returns:** String

---

### **getAllModules()**

Gibt eine Liste aller installierten Module zurück

**Returns:** Array of String

---

### **replaceShortcodesWithModules(\$string)**

Führt die Module die im String \$string per Platzhalter eingebunden sind aus und ersetzt diese mit der Rückgabe der \_render() Methode.

**Returns:** String

---

### **getPageIDBySystemname(\$systemname)**

Gibt die ID eines content-Datensatzes anhand vom Systemnamen

**Returns:** Integer oder Null

### **getPageSystemnameByID(\$id)**

Gibt den Systemnamen einer Seite anhand deren ID zurück, ansonsten „-“

**Returns:** String

### **getAllSystemNames()**

Gibt eine sortierte Liste der Systemnamen aller Seiten zurück

**Returns:** Array of String

---

### **getAllLanguages()**

Gibt ein Array der Sprachcodes aller Sprachen zurück.

**Returns:** Array of String

---

### **file\_extension(\$filename)**

Gibt die Endung der Datei \$filename zurück

**Returns:** String

---

### **containsModule(\$page, \$module = false)**

Prüft ob \$module in der Seite mit dem Systemnamen \$page vorhanden ist.

Wenn \$module fehlt wird die aktuelle Seite geprüft.

**Returns:** boolean

---

### **uninstall\_module(\$name)**

Ruft das Uninstall-Script vom Modul \$name auf.

### **is\_logged\_in()**

Prüfen, ob der Nutzer eingeloggt ist.

**Returns:** Boolean

**Alias:** logged\_in()

---

### **has\_permissions(\$mod)**

Prüft ob die Berechtigung des aktuellen Nutzers >= \$mod ist.

Informationen über das Rechtesystem in UliCMS finden Sie im Administrator Handbuch

**Returns:** Boolean

---

### **tbname(\$name)**

Setzt den Datenbank Prefix vor \$name.

Der Datenbank Prefix wird bei der Installation des CMS festgelegt und in der cms-config gespeichert.

**Returns:** String

---

### **cms\_version()**

Gibt die Versionbezeichnung des CMS zurück (z.B. „2014R6“)

Hinweis: Für Prüfung der Version verwenden Sie bitte nicht diese Art der Versionsnummer sondern die internalVersion in der ulicms\_version-Klasse

**Returns:** String

### **is\_admin()**

Ist der angemeldete Nutzer Admin?

**Returns:** Boolean

## **USER API**

Die Funktionen zur Userverwaltung befinden sich in der Datei users\_api.php.  
Diese muss includet werden, bevor sie genutzt werden können

### **getUsers()**

Gibt eine Liste aller Usernamen zurück. Alphabetisch sortiert.

**Returns:** Array of String

---

### **changePassword(\$password, \$id)**

Setzt das Passwort vom Nutzer mit der ID \$id auf \$password.  
Gibt True zurück wenn es geklappt hat, ansonsten false.

**Returns:** Boolean

---

### **getCurrentURL()**

Gibt die vollständige URL zur aktuellen Seite inklusive Protokoll, Port [falls nicht 80] und GET-Parameter aus, so wie Sie in der Adresszeile des Browser steht

**Returns:** String

### **getUserByName(\$name)**

Gibt die Datenbankzeile des Users mit dem Namen \$name als assoziatives Array oder false zurück.

**Returns:** Array oder False

---

### **getUserById(\$id)**

Gibt die Datenbankzeile des Users mit der User ID \$id als assoziatives Array oder false zurück.

**Returns:** Array oder False

---

### **adduser(\$username, \$lastname, \$firstname, \$email, \$password, \$group)**

Legt einen neuen User an.

**Achtung:**

Diese Funktion prüft nicht, ob der User \$username schon existiert. Bitte vorher prüfen.

**Returns:** void

**register\_session(\$user, \$redirect = true)**

Registriert die Sitzung eines Nutzers.

\$user muss der Rückgabe von **getUserByName** oder **getUserById** entsprechen

Wenn \$redirect = true ist, erfolgt eine automatische Weiterleitung.

Wenn \$\_GET[„go“] gesetzt ist, wird zu der URL \$\_GET[„go“] umgeleitet.]

**Returns:** Void

---

**validate\_login(\$user, \$password)**

Prüft einen Login mit Usernamen und Password.

Diese Funktion ist abwärtskompatibel zum Hashing das in früheren Versionen von UliCMS verwendet wurde.

## SPELLCHECK API

Diese Funktionen befinden sich in der Datei `spellcheck.php` die erst includet werden muss.

### **autocorrect\_common\_typos(\$text)**

Korrigiert häufige Rechtschreibfehler in der Deutschen Sprache.

Die Funktion basiert auf folgenden Wikipedia-Artikel:

[http://de.wikipedia.org/wiki/Liste\\_h%C3%A4ufiger\\_Rechtschreibfehler\\_im\\_Deutschen](http://de.wikipedia.org/wiki/Liste_h%C3%A4ufiger_Rechtschreibfehler_im_Deutschen)

**Returns:** String

## ANTI-SPAM API

Anti-Spam Funktionen finden sich in der Datei **`antispam-features.php`**.

### **isCountryBlocked()**

Prüft anhand der Endung von `gethostbyaddr($_SERVER["REMOTE_ADDR"])` ob der Besucher aus einem Land in dem die Kommentarfunktion gesperrt ist kommt

**Returns:** Boolean

## DATENBANKFUNKTIONEN

Datenbankfunktion befinden sich in der Datei **`lib/db_functions.php`**.

### **db\_query(\$sql\_string);**

Wrapper um `mysql_query()`. Ermöglicht das mitloggen von Datenbankabfragen.  
Sollte statt `mysql_query` verwendet werden.

**Returns:** MySQL Result



## KRYPTOGRAPHIE FUNKTIONEN

Funktionen zur Verschlüsselung befinden sich **unter lib/encryption.php**

### **hash\_password(\$password)**

Setzt den Salt vor \$password um gibt einen SHA1-Hash zurück.

**Returns:** String

## FILE\_GET\_CONTENTS\_WRAPPER

**lib/file\_get\_contents\_wrapper.php** enthält Funktionen zum Download von URLs in einen String.

### **is\_url(\$url)**

Gibt zurück ob \$url eine URL ist.

**Returns:** Boolean

---

### **file\_get\_contents\_wrapper(\$url)**

Wenn allow\_url\_fopen aktiviert ist, wird \$url mit file\_get\_contents runtergeladen.

Wenn nicht dann wird file\_get\_contents\_curl aufgerufen.

Wenn CURL nicht installiert ist dann false.

**Returns:** String oder False

---

### **file\_get\_contents\_curl(\$url)**

Ähnlich wie file\_get\_contents lediglich das CURL statt fopen zum Download verwendet wird.

**Returns:** String oder False

## HTML FUNKTIONEN

Diese Funktionen dienen der Verarbeitung von HTML-Codes.

### **htmlRemoveTagByName(\$html, \$tag)**

entfernt aus dem HTML-Code \$html sämtliche Vorkommen von \$tag.

\$tag ist die Bezeichnung eines HTML-Tags ohne Klammern. (z.B. „script“).

**Returns:** String

## LOGGER FUNKTIONEN

Logger-Funktion werden für die Fehleranalyse benötigt.

### **log\_db\_query(\$query)**

Schreibt die Datenbankabfrage \$query in content/log/db/YYYY-MM-DD.log.

Wird nur ausgefüllt, wenn **\$config->query\_logging** auf **true** ist.

**Returns:** Boolean

## STRING FUNKTIONEN

### **getExcerpt(\$str, \$startPos=0, \$maxLength=100)**

Gibt den Ausschnitt von \$startPos bis maximal \$maxLength, Schneidet keine Wörter ab.  
Setzt ans ende des String drei Punkte.

**Returns:** String

### **isEmpty(\$str)**

Wie empty(), nur das vorher noch trim() auf \$str angewendet wird.

**Returns:** Boolean

---

### **decodeHTMLEntities(\$str)**

Dekodiert kodierte HTML Entities. Die Funktion ist UTF-8 kompatibel.

**Returns:** String

---

### **keywordsFromString(\$str)**

Ermittelt die häufigsten Wörter in \$str

Gibt ein sortiertes Assoziatives Array zurück

**Returns:** Array

## **VERSION DES CMS ABFRAGEN**

```
$version = new ulicms_version();
```

```
$version->getVersion(); // 2014R6
```

```
$version->getInternalVersion(); // Array(6, 2)
```

```
$version->getDevelopmentVersion() // true oder false
```

## TEMPLATES

Mit UliCMS können Sie recht schnell gut aussehende Templates erstellen.

Die grundlegende Templatestruktur.

**oben.php** – Header, Menüs

Content wird danach ausgegeben.

**unten.php** – Alles was nach den Menüs ausgegeben wird.

style.css – CSS-Codes

**maintenance.php** – Wartungsseite

**404.php** – Fehlerseite für 404 Not Found

**403.php** – Fehlerseite für 403 Forbidden

## TEMPLATE API

**language\_selection()**

Gibt eine Sprachauswahl in Form einer HTML-Liste aus.

**CSS Klassen:** language\_selection

**Returns:** void

---

**random\_banner()**

Gibt einen zufällig ausgewählten Banner aus.

**Returns:** void

### **logo()**

Gibt das Logo aus, sofern es nicht deaktiviert ist.

**Returns:** void

### **year()**

Gibt das aktuelle Jahr aus. **Z.B.** 2013

**Returns:** void

---

### **homepage\_owner()**

Gibt den Eigentümer der Website aus.

**Returns:** void

---

### **homepage\_title()**

Gibt den Titel der Homepage aus.

**Returns:** void

---

### **check\_status()**

Gibt den HTTP Status der aktuellen Seite als String zurück. Z.B. „404 Not Found“

**Returns:** void

### **meta\_keywords()**

Gibt den Meta-Keywords Tag aus.

**Returns:** void

---

### **meta\_description()**

Gibt den Meta-Description Tag aus.

**Returns:** void

---

### **title()**

Gibt den Titel der Seite aus.

**Returns:** void

---

### **Import(\$systemname)**

Gibt die Seite mit dem Systemnamen \$systemname aus.

**Returns:** void

---

### **apply\_filter(\$text, \$type)**

Wendet die Filter des Typs \$type aller Module auf \$text an.

**Returns:** String

### **motto()**

Moto der Homepage ausgeben.

**Returns:** Void

---

### **get\_requested\_pagename()**

Gibt den Systemnamen der aktuellen Seite zurück.  
Ansonsten den Systemnamen der Startseite.

**Returns:** String

---

### **is\_frontpage()**

Befinden wir uns gerade auf der Startseite?

**Returns:** Boolean

---

### **is\_200() is\_403() is\_404()**

Prüft auf spezifische http Status-Codes

**Returns:** Boolean

## **menu(\$name)**

Gibt das Menü \$name aus.

### **Mögliche Werte für \$name:**

left  
right  
top  
bottom  
hidden

\$name ist vom Typ String

**Returns:** Void

---

## **base metas()**

Gibt die Meta-Tags aus. Führt außerdem die \_head Funktionen der Module aus.  
Diese Funktion muss in jedem Theme im <head> verwendet werden.

### **Konfigurationsvariablen:**

hide\_meta\_generator  
hide\_meta\_keywords  
hide\_meta\_description

**Returns:** Void

---

## **autor()**

Den Autor dieser Seite ausgeben

**Returns:** Void



## **content()**

Gibt den Content der aktuellen Seite aus.

Muss nicht manuell aufgerufen werden, da es nach dem inkludieren von **oben.php** automatisch aufgerufen wird.

**Returns:** Void

**ulicms\_settings**

id INT(11)
name VARCHAR(255)
value MEDIUMTEXT
Indexes

**ulicms\_languages**

id INT(11)
name VARCHAR(50)
language_code VARCHAR(6)
Indexes

**ulicms\_content**

id INT(11)
notinfeed TINYINT(1)
systemname VARCHAR(255)
title VARCHAR(255)
target VARCHAR(255)
content LONGTEXT
language VARCHAR(6)
active TINYINT(1)
created BIGINT(20)
lastmodified BIGINT(20)
autor INT(11)
category INT(11)
lastchangeby INT(11)
views INT(11)
comments_enabled TINYINT(1)
redirection VARCHAR(255)
menu VARCHAR(10)
position INT(11)
parent_old VARCHAR(300)
valid_from DATE
valid_to DATE
access VARCHAR(100)
meta_description TEXT
meta_keywords TEXT
deleted_at BIGINT(20)
parent INT(11)
Indexes

**ulicms\_banner**

id INT(11)
name TEXT
link_url TEXT
image_url TEXT
Indexes

**ulicms\_admins**

id INT(11)
username VARCHAR(255)
lastname VARCHAR(255)
firstname VARCHAR(255)
email VARCHAR(255)
password VARCHAR(255)
group INT(11)
old_encryption TINYINT(1)
skype_id VARCHAR(32)
icq_id VARCHAR(20)
avatar_file VARCHAR(40)
about_me TEXT
last_action BIGINT(20)
Indexes