ULICMS

Neue Features in UliCMS 9.8.2

Stand: 18.02.2016

Inhaltsverzeichnis

Über dieses Dokument

Dieses Dokument beschreibt die neuen Features in UliCMS 9.8.2.

Zielgruppe

Zielgruppe dieses Dokuments sind Systemadministratoren, Webentwickler und Endanwender von UliCMS.

Übersicht der neuen Features

Diese Version von UliCMS enthält alle Features die in UliCMS 9.8.1 enthalten waren, sowie die folgenden neuen Features.

- Translation Overrides
- Seite erstellen & bearbeiten nutzt nun Tabs
- Die folgenden Klassen wurden hinzugefügt
    - File
    - Path
    - Mailer
    - Database
    - CustomData
- Funktionen zu Template Klasse hinzugefügt
- Skript „shell/sinstall.php“ hinzugefügt
- Mobile_Detect auf Version 2.8.19 aktualisiert
- CKEditor auf Version 4.5.6 aktualisiert
- "Passwort zurücksetzen" Passwortlänge auf 12 Zeichen erhöht
- API Funktion is_tablet() hinzugefügt
- Felder zu Benutzerprofilen hinzugefügt

Detailbeschreibungen

Im Folgenden eine detailierte Beschreibung der neuen Features.

Translation Overrides

Das neue Feature „Translation Overrides“ ermöglicht es, die Übersetzungen aus den Sprachdateien anzupassen. Die Anpassungen überstehen im Gegensatz zu direkten Anpassungen der Sprachdateien auch Updates.

Voraussetzungen

Translation Overrides setzen voraus, dass eine Übersetzung bereits in den normalen Sprachdateien definiert ist.

Translation Overrides haben nur einen Effekt, wenn die anzupassende Übersetzung über eine der folgenden Funktionen eingebunden werden:

- get_translation()
- translate()
- get_translation()

Translation Overrides funktionieren nicht, wenn Übersetzungen direkt aus einer Konstante ausgegeben werden.

Nutzung

1. Legen Sie den Ordner „ULICMS_ROOT/lang/custom“ an, sofern noch nicht vorhanden.
2. Erstellen Sie im Ordner „ULICMS_ROOT/lang/custom“ eine Datei [languagecode].php (z.B. de.php oder en.php), sofern noch nicht vorhanden.
3. Öffnen Sie die angelegte Datei in einem Texteditor.
4. Sie können Translation Overrides wie folgt hinzufügen.

```
<?php

Translation::override(„my_translation“, „Meine eigene Übersetzung“)
Translation::override(„another_translation“, „Eine weitere Übersetzung“)
```

Seiten erstellen & Bearbeiten nutzt nun Tabs

Die Eingabefelder in der Seitenverwaltung sind nun in Tabs sortiert.
Dies macht das Bearbeiten von Seiten für den Anwender übersichtlicher

Es gibt die folgenden Tabs:

- Titel und Überschrift
- Typ
- Menü Eintrag
- Weiterleitung auf externen Link
- Menüpunkt als Grafik & Design
- Sichtbarkeit
- Meta-Daten
- Öffnen In
- Open Graph
- Benutzerdefinierte Daten (JSON)

Feld „Typ“ hinzugefügt

Die Spalte `type` befindet sich schon länger in der Datenbank, jedoch war bisher kein Auswahlfeld in der GUI dafür vorhanden.
Dieses wurde nun hinzugefügt. Derzeit gibt es jedoch noch keine Auswahl. Es ist immer „Seite“ ausgewählt, bzw. steht in der Datenbank „page“.

Zukünftig sollen Auswahlmöglichkeiten hinzukommen.

Neue Klassen und Funktionen

Im folgenden eine Beschreibung der neuen Klassen und Funktionen für Webentwickler.

Klasse „File“ hinzugefügt

Die Klasse **File** wurde hinzugefügt. Diese enthält die folgenden Funktionen:

File::write($file, $content)

Den Text $content in die Datei $file schreiben. Die Datei wird überschrieben, wenn sie bereits vorhanden ist.

File::append($file, $content)

Der Text $content wird an das Ende der Datei $file angehängt.

File::read($file)

Der Inhalt der Datei $file wird zurückgegeben.

File::delete($file)

Die Datei $file wird gelöscht. Ordner können mit dieser Funktion jedoch nicht gelöscht werden.

File::rename($old, $new)

Benennt bzw. verschiebt die Datei $old zu $new.

File::getLastChanged($file)

Gibt den Zeitpunkt der letzten Änderung der Datei $file als Unix Timestamp zurück.

File::lastChanged($file)

Gibt den Zeitpunkt der letzten Änderung der Datei $file als Unix Timestamp aus.

File::getExtension($filename)

Gibt die Dateiendung hinter dem letztem Punkt von $filename zurück.

Klasse „Path“ hinzugefügt

Die Klasse **Path** wurde hinzugefügt.
Diese enthält die folgenden Funktionen.

Path::resolve($path)

Einen Pfad auflösen.
Dabei werden die folgenden Konstanten vom System durch den entsprechenden Pfad ersetzt:

- ULICMS_ROOT
- ULICMS_TMP
- ULICMS_CACHE

Außerdem werden unter Windows Backslashes durch Slashes ersetzt und Slashes am Ende des Strings werden entfernt.

Klasse „Mailer“ hinzugefügt

Die Klasse **Mailer** wurde hinzugefügt.
Die darin enthaltenen Funktionen ersetzen zukünftig die globalen Funktionen zum Mailversand aus der Datei. **lib/mailer.php**

Mailer::send($to, $subject, $message, $headers = "")

Diese öffentliche Funktion versendet E-Mails je nach Einstellung entweder über die mail() Funktion oder per PEAR Mail.

Mailer::splitHeaders($headers)

Diese Funktion konvertiert einen String mit E-Mail Headern in ein Array.
Sie wird nur intern von der Funktion **Mailer::sendByPEAR()** genutzt.

Mailer::sendbyPEAR($to, $subject, $message, $headers = "")

Sofern der Mailversand per PEAR Mail erfolgen soll, wird diese Funktion genutzt.

Diese Funktion kann nicht direkt aufgerufen werden, sondern wird von der Funktion **Mailer::send()** genutzt.

Klasse „Database“ hinzugefügt

Die Klasse „Database“ wurde hinzugefügt.
Diese Klasse enthält Funktionen zum Zugriff auf die Datenbank und soll zukünftig die globalen Funktionen, die in der Datei lib/db_functions.php definiert sind, ersetzen..
Statt „Database“ kann man auch das Alias „DB“ nutzen.

Die Klasse enthält die folgenden Funktionen:

Database::query($sql)

Führt **$sql** aus und gibt ein mysqli Result Set zurück.

Database::getPDOConnectionString()

Generiert den PDOC Connection String.

Database::getServerVersion()

Gibt die Version des MySQL Servers zurück.

Database::getClientInfo()

Gibt die MySQL Version der Client Libray zurück.

Database::preparedQuery($sql, $typeDef = FALSE, $params = FALSE)

Führt eine SQL Abfrage mit prepared Statements durch.

$typeDef ist ein Array mit den Typen für die Werte, die im Array **$params** angegeben werden. Diese werden automatisch maskiert, um SQL Injections zu verhindern.

Database::escapeName($name)

Maskiert eine Spaltenbezeichnung durch einschließen von $name mit den entsprechenden Sonderzeichen.

Database::getLastInsertID()

Gibt die ID des zuletzt eingefügten Datensatzes zurück.

Database::getInsertID()

Alias für **Database::getLastInsertID()**

Database::fetchArray($result)

Eine Zeile aus dem Ergebnis **$result** als Array zurückgeben und den Zeiger auf den nächsten Datensatz setzen.

$result wird von den Funktionen **Database::query()** und **Database::preparedQuery()** zurückgegeben

Database::fetchArray($result)

Die Namen der zurückgegeben Datenbankspalten aus **$result** as Array zurückgeben.

$result wird von den Funktionen **Database::query()** und **Database::preparedQuery()** zurückgegeben.

Database::fetchAssoc($result)

Eine Zeile aus dem Ergebnis **$result** als assoziatives Array zurückgeben und den Zeiger auf den nächsten Datensatz setzen.

$result wird von den Funktionen **Database::query()** und **Database::preparedQuery()** zurückgegeben.

Database::fetchAll($result)

Alle Ergebniszeilen aus **$result** als Array von assoziativen Arrays zurückgeben.

$result wird von den Funktionen **Database::query()** und **Database::preparedQuery()** zurückgegeben.

Database::close()

Schließen der Datenbankverbindung.

Database::connect($server, $user, $password)

Verbindung mit Datenbankserver **$server** mit den angegebenen Zugangsdaten aufbauen.

Database::select($schema)

Die Datenbank $schema zur aktiven Datenbank machen.

Database::getNumFieldCount($result)

Anzahl der Spalten des Ergebnis $result zurückgeben.

$result wird von den Funktionen **Database::query()** und **Database::preparedQuery()** zurückgegeben.

Database::getAffectedRows()

Anzahl der Zeilen, die von der letzten Datenbankabfrage betroffen waren zurückgeben.

Database::fetchObject($result)

Eine Zeile aus dem Ergebnis **$result** als Objekt zurückgeben und den Zeiger auf den nächsten Datensatz setzen.

$result wird von den Funktionen **Database::query()** und **Database::preparedQuery()** zurückgegeben.

Database::fetchRow($result)

Eine Zeile aus dem Ergebnis **$result** als Objekt zurückgeben und den Zeiger auf den nächsten Datensatz setzen.

$result wird von den Funktionen **Database::query()** und **Database::preparedQuery()** zurückgegeben.

Database::getNumRows($result)

Die Anzahl der Zeilen aus dem Ergebnis $result zurückgeben.

$result wird von den Funktionen **Database::query()** und **Database::preparedQuery()** zurückgegeben.

Database::getLastError()

Gibt den zuletzt aufgetretenen Fehler bei einer SQL Abfrage zurück.

Database::error()

Alias für Database::getLastError()

Database::getAllTables()

Gibt eine Liste aller Tabellen der Datenbank als Array zurück.

Database::escapeValue($value, $type = null)

Maskiert $value um SQL Injections zu verhindern.

Ein Variablentyp von **$value** kann als **$type** mitgegeben werden.

Mögliche Werte sind:

- DB_TYPE_INT
- DB_TYPE_FLOAT
- DB_TYPE_STRING
- DB_TYPE_BOOL

Wenn **$type** = **null** ist, wird versucht, den Typ von $value automatisch zu erkennen.

Klasse „CustomData“ hinzugefügt.

Die Klasse CustomData wurde hinzugefügt.
CustomData enthält Funktionen zum Zugriff auf die benutzerdefinierten Daten (JSON) der Seiten.

Sie enthält die folgenden Funktionen:

CustomData::get($page = null)

Gibt ein assoziatives Array mit den derzeit gesetzten Custom Data Variablen zurück.
Wenn **$page** = null ist, werden die Variablen der derzeit geöffneten Seite zurückgeben.

CustomData::set($var, $value, $page = null)

Ändert die benutzerdefinierte Variable **$var** in **$value**.

Wenn **$page** = null ist, wird die Variable der derzeit geöffneten Seite in der Datenbank geändert.

CustomData::delete($var = null, $page = null)

Löscht die benutzerdefinierte Variable $var in der Datenbank.

Wenn **$var** gesetzt ist, wird nur die Variable $var gelöscht.
Wenn $var = null ist, werden alle benutzerdefinierten Daten auf der Seite $page gelöscht.

Wenn **$page** = null ist, wird die Variable der derzeit geöffneten Seite in der Datenbank gelöscht.

Funktionen zur „Template“ Klasse hinzugefügt

Die folgenden Funktionen wurden zur Klasse **Template** hinzugefügt:

Template::getEscape($value)

Kodiert bzw. maskiert HTML-Sonderzeichen in $value und gibt den kodierten Text zurück.

Template::escape($value)

Kodiert bzw. maskiert HTML-Sonderzeichen in $value und gibt den kodierten Text aus.

Template::homepageOwner()

Gibt den „Inhaber der Website“ aus.

Template::getHomepageOwner()

Gibt den „Inhaber der Website“ zurück.

Template::poweredByUliCMS()

Gibt den Text „Diese Seite läuft mit UliCMS 9.8.2“ aus.

API Funktion is_tablet() hinzugefügt

Die API Funktion **is_tablet()** wurde hinzugefügt.
Diese gibt **true** zurück, sofern die Anfrage von einem Tablet aus erfolgte, ansonsten wird **false** zurückgegeben.

Skript „shell/sinstall.php“ hinzugefügt

Das Skript ULICMS_ROOT/shell/sinstall.php wurde hinzugefügt.

Dieses ist nur auf der Kommandozeile (Shell) ausführbar.

Dieses Skript dient dazu ein SimpleInstall Paket im tar.gz zu installieren.
Es unterstützt nur einen Parameter, [file]. Den Pfad zu einem UliCMS SimpleInstall Paket.

Beispielaufruf:

```
$ php /var/www/html/shell/sinstall.php /home/myuser/fortune-0.0.2.tar.gz
```

```
Package /home/myuser/fortune-0.0.2.tar.gz was successfully installed.
```

Felder zu Benutzerprofilen hinzugefügt

Feld für Twitter Profil hinzugefügt

Es wurde ein Feld zum hinterlegen des Twitter Profils hinzugefügt.
Dort kann der Benutzername aus der Twitter URL hinterlegt werden z.B. twitter.com/**myprofile**

Feld für Homepage hinzugefügt

Ein Feld für die Eingabe der Homepage des Users wurde hinzugefügt.

Hooks hinzugefügt und verschoben

Die Hooks „before_edit_button“ und „after_edit_button“ wurden hinzugefügt.
Die Hook „after_content“ wurde verschoben.

Sicherheit

"Passwort zurücksetzen" Passwortlänge auf 12 Zeichen erhöht

Die zufällig generierten Passwörter, die die „Passwort zurücksetzen“ Funktion generiert sind nun 12 Zeichen lang. Zuvor waren diese nur 8 Zeichen lang.

„Passwort anzeigen“ Checkbox zu Installations-Assistent hinzugefügt

Es wurde eine Checkbox zum Installations-Assistenten hinzugefügt, welche ermöglicht, die Passwortfelder zwischen Klartext und Sternchen umzuschalten.

XSS in Banner Management System behoben

Es gab eine Cross-Site-Scripting (XSS) Sicherheitslücke im Banner-Management-System, die ermöglicht, HTML-Code zur Ausführung zu bringen, durch Eingabe von HTML-Codes in die Textfelder. Diese Sicherheitslücke wurde von dem IT-Sicherheitsforscher Manuel Garcia entdeckt.

Mehrere Sicherheitslücken in den Spamfilter Einstellungen

Es gab zwei SQL Injections, in den Spamfilter Einstellungen.
Diese Sicherheitslücke wurde von dem IT-Sicherheitsforscher Manuel Garcia entdeckt.

Dabei wurde ein weiterer Fehler entdeckt und behoben:
Beim Speichern der Einstellungen des Spamfilters fehlte die Prüfung, ob der derzeit angemeldete Benutzer überhaupt die Zugriffsrechte darauf hat.
Die Prüfung wurde ergänzt

Aktualisierung von Komponenten

Mobile_Detect aktualisiert

Mobile_Detect wurde auf Version 2.8.19 aktualisiert.
Diese Klasse dient der Erkennung von mobilen Geräten wie Smartphones und Tablets.

Details zu den Änderungen in dieser Version finden Sie unter der folgenden Adresse:
http://mobiledetect.net/

CKEditor aktualisiert

CKEditor wurde auf Version 4.5.6 aktualisiert.
Diese Open Source Komponente ermöglicht die grafische Bearbeitung von HTML-Codes.
Falls nach dem Upgrade auf UliCMS 9.8.2 Probleme mit dem Seiteneditor auftreten sollten, leeren Sie bitte Ihren Browsercache

Für eine Liste der Änderungen schauen Sie bitte unter dem folgendem Link:

http://ckeditor.com/blog/CKEditor-4.5.6-Released

Sonstige Änderungen

Änderungen der Datenbankstruktur

In dieser Version wurden keine Änderungen an der Datenbank vorgenommen.

Das update-Skript im Upgrade-Paket dient lediglich dazu, die Liste der installed_patches zurückzusetzen.

Änderungen der API

Es wurden die weiter oben in dieser Dokumentation genannten API-Funktionen hinzugefügt.
Diese Version von UliCMS enthält keine Änderungen, die die Kompatiblität brechen könnten.
Alle Module und Themes die unter UliCMS 9.8.1 funktionieren, werden auch unter UliCMS 9.8.2 funktional sein.