

Joomla! 1.0 Extensions zu Joomla! 1.5 kompatibel machen

Sonntag, 17. September 2006

Original: http://dev.joomla.org/.../tips:make_your_extensions_fit

Dies ist ein erster Entwurf ...

Obwohl Joomla 1.5 mit einer Rückwärtskompatibilität zu Joomla 1.0 im Sinn gestaltet wurde, gibt es nur sehr wenige Elemente, die nicht rückwärtskompatibel sind und minimale Anpassungen erfordern. Dieses Dokument fasst die Änderungen zusammen die für Joomla 1.0 Erweiterungen gemacht werden müssen, damit sie unter Joomla 1.5 zu laufen.

{tab=Komponenten}

Globaler Gültigkeitsbereich

Um die Sicherheit zu erhöhen, und eine bessere Kapselung zu erreichen, werden die Komponenten nicht mehr einfach nur über die Haupt-index.php eingefügt (inkludiert), sondern von einer Funktion aufgerufen. Daher ist der Gültigkeitsbereich der Variablen nicht länger Global. Dies bedeutet das Variablen die im Hauptteil der Komponente erstellt wurden, und vor ihrer Erstellung nicht als "global" deklariert wurden, zur Hauptroutine der Komponente gehören und damit nicht von der Funktion der Komponente als global lesbar sind.

Die Adaption ist trivial: Fügen Sie einfach eine Zeile am Anfang der Komponente hinzu: "global" für jede globale Variable Ihrer Komponente.

Die Parameter REQUEST / POST

Per Default werden URL Parameter nicht mehr in globale Variablen des selben Namens übertragen, sondern sollten normal abgefragt werden. Auch hier ist die Umstellung trivial: statt direkt auf, zum Beispiel, \$form zuzugreifen, verwenden Sie zuerst:

Eine Requestvariable in Joomla! 1.0 einlesen (funktioniert auch in Joomla! 1.5)

```
$form = mosGetParam( $_REQUEST, 'reportform');
```

Neue Methode um eine Requestvariable in Joomla! 1.5 einzulesen

```
$form = JRequest::getVar('reportform');
```

Dies verhindert nebenbei SQL Injection Attacken, da sowohl mosGetParam als auch JRequest die Variablen für gefährlichen SQL Code escapen (überschreiben).

Veränderungen an der ACL haben begonnen.

Die Tabellenstruktur der benutzten phpGACL hat sich auf Grund eines Updates an der neuestes phpGACL leicht geändert. Die Methode \$acl->.... wird Rückwärtskompatibel bleiben, daher sind Komponenten die die ACLtabellen nicht stark benutzen OK. Bitte melden Sie Inkompatibilitäten damit sie im core behoben werden können.

{tab=Module}

Für Module gelten die gleichen Änderungen wie für Komponenten weiter oben beschrieben. Zusätzlich gilt:

Pfad des Moduls

Jedes Modul wird, wie auch die Komponenten, in seinem eigenen Unterverzeichnis installiert. Das bedeutet das wenn es Bilder oder andere Dateien enthält, sich auch der Pfad ändert.

{tab=Joomla! Plugins}

(früher Mambots)
Wechsel des Namens

Die Mambots wurden in Joomla! Plugins umbenannt. Auch das Verzeichnis. Daher gelten dort die gleichen Anmerkungen.

Diese Anmerkung gilt für Content Plugins vs Content Mambots.

Anmerkung #1

Gemäß den allgemeinen Änderungen ist die altbekannte Methode um den Zugriff von direkten externen aufrufen zu unterbinden:

Die alte Methode der Mambots (funktioniert auch noch in Joomla! 1.5) `defined('_VALID_MOS')` or die('Direct Access to this location is not allowed.');

Die neue Methode der Joomla! 1.5 Plugins `defined('_JEXEC')` or die('Restricted access');

Anmerkung #2

Das Ereignis eintragen

Die alte Methode der Mambots
`$_MAMBOT->registerFunction('onPrepareContent', 'botMybot');`

Die neue Methode der Joomla! 1.5 Plugins
`$mainframe->registerEvent('onPrepareContent', 'botMybot');`

Anmerkung #3

Es wurde eine neue Coreklasse entwickelt die uns im Umgang mit Plugins helfen soll, diese Klasse ist `JPluginHelper`.

Sie kann benutzt werden (unter anderem) um festzustellen, ob ein Plugin veröffentlicht wurde.

```
$plugin =& JPluginHelper::getPlugin('content', 'mybot');
if (!$plugin->published){
    //plugin nicht veröffentlicht
}else {
    //plugin veröffentlicht
}
```

Anmerkung #4

Es wurde eine neue Coreklasse entwickelt die uns im Umgang mit Plugins helfen soll, diese Klasse ist `JParameter`.

Sie kann benutzt werden, um die Parameter des Plugins auszugeben. Hier ein Beispiel:

```
$pluginParams = new JParameter( $plugin->params );
```

Bitte beachten Sie, dass wir dem Konstruktor die Daten übergeben, die vorher als Instanz in Anmerkung #3 deklariert wurde. Hierüber können wir die Parameter des Plugins steuern.

Nehmen wir zum Beispiel an, das Plugin hätte die Parameter: P1, P2, P3. Das Abfragen der Parameter ist einfach:

```
$pluginParams->def( 'P1', 0 );
```

Weitere Änderungen

folgt...

{/tabs}