Beispielmodule programmieren

Anmerkung

Das Beispiel geht ab Version 17.1 Ein Modul besteht aus mindestens einer PHP-Datei .php für den Code der im Hintergrund ablaufen soll und einer Template-Datei .tpl die für die Darstellung von HTML sorgt. Die PHP-Datei wird zuerst beschrieben, ganz unten befinden sich die Template-Dateien.

PHP-Datei beispielmodul.php

Zuerst wird die Klasse für das Modul erstellt, wie hier das Beispielmodul. Mit der darin enthaltenen, statischen Funktion TableSearch werden alle Live Tabellen für das Modul definiert. Innerhalb von TableSearch() werden mit einem switch alle Live Tabellen aufgezählt. Live Tabellen können beliebig benannt werden. In der Variable $heading ist ein Array mit den Spaltennamen die in der Live Tabelle angezeigt werden. Mit der Variable $width werden die einzelnen Spaltenbreiten in prozentualer Angabe in einem Array festgelegt. Die Variable $findcols beinhaltet ein Array mit den Datenbankspalten die in der Live Tabelle sortierbar sind. $searchsql legt im Array die Datenbankspalten fest, die im Suchfenster durchsucht werden können. In der Variable $menue werden, falls benötigt, die Menüpunkte für die Einträge der Live Tabelle definiert, wie in diesem Beispiel edit und delete. Im dazugehörigen Link von ist hinter action der jeweiligen ActionHandler ersichtlich, der bei einem Klick auf ausgeführt wird. In der Variable $sql wird die SQL Abfrage eingetragen, von welcher das Ergebnis in der Live Tabelle gesehen werden so. Zu beachten ist, dass das WHERE der Abfrage in der Variable $where eingetragen werden muss. Dasselbe gilt für GROUP BY, was in der Variable $groupby steht. $count sorgt für die Darstellung der Anzahl der Einträge die angezeigt werden unterhalb der Live Tabelle.

?php //WAWIFILEONLYON VERSION=ALL

class Beispielmodul { var $app;

static function TableSearch(&$app, $name, $erlaubtevars) { // in dieses switch alle lokalen Tabellen (diese Live Tabellen mit Suche etc.) für dieses Modul switch($name) { case "beispielmodul_list": $allowed['beispielmodul'] = array('list');

```
$heading = array('Adresse', 'Datum', 'Beschreibung', 'Menü');
  $width = array('40%', '30%', '20%', '10%');


  $findcols = array('b.id', 'b.adresse', 'b.datum', 'b.beschreibung');
  $searchsql = array('b.id', 'b.adresse', 'b.datum', 'b.beschreibung');

  $defaultorder = 1;
  $defaultorderdesc = 0;
```

```
    $jahr = date("Y");

    $menu = "<a href=\"index.php?module=beispielmodul&action=edit&id=%value%\"><img

    $sql = "SELECT SQL_CALC_FOUND_ROWS b.id, b.adresse, b.datum, b.beschreibung, b.

    $where = " YEAR(b.datum) < '$jahr'";

    $groupby = "GROUP BY b.datum";


    $count = "SELECT count(DISTINCT b.id) FROM beispielmodul b WHERE $where";
    break;
}

$erg = false;

foreach($erlaubtevars as $k => $v)
{
  if(isset($$v))$erg[$v] = $$v;
}
return $erg;
```

Danach folgt die Funktion __construct() die den Konstruktor darstellt und alle ActionHandler festlegt. Der ActionHandler definiert im ersten Parameter welches Action gemeint ist und im zweiten Parameter die zugehörige Funktion die beim jeweiligen Action ausgeführt werden soll. Die ActionHandler können ebenfalls beliebig benannt werden. Außerdem wird die Funktion Install() aufgerufen.

function __construct($app, $intern = false) { $this->app=&$app; if($intern)return; $this->app->ActionHandlerInit($this); //Ab hier alle ActionHandler definieren die das Modul hat $this->app->ActionHandler("list", "BeispielmodulList"); $this->app->ActionHandler("create", "BeispielmodulCreate"); $this->app->ActionHandler("edit", "BeispielmodulEdit"); $this->app->ActionHandler("delete", "BeispielmodulDelete");

$this->app->ActionHandlerListen($app); $this->Install(); }

Um neue Tabellen und Spalten in der Datenbank anzulegen, wird eine neue Funktion namens Install() angelegt. Innerhalb dieser Funktion werden die Funktionen CheckTable() und CheckColumn() verwendet. CheckTable() prüft, ob die Tabelle bereits in der Datenbank vorhanden ist und legt sie an falls das nicht der Fall ist. CheckColumn() prüft, ob die Spalten in der angegebenen Tabelle bereits existieren und legt sie an falls das nicht der Fall ist.

function Install() { $this->app->erp->CheckTable("beispielmodul"); $this->app->erp->CheckColumn("id", "int(11)", "beispielmodul", "NOT NULL AUTO_INCREMENT");

$this->app->erp->CheckColumn("adresse", "int(11)", "beispielmodul", "NOT NULL"); $this->app->erp->CheckColumn("datum", "DATE", "beispielmodul", "NOT NULL"); $this->app->erp->CheckColumn("beschreibung", "VARCHAR(255)", "beispielmodul", "NOT NULL"); }

Als nächstes folgen die einzelnen Funktionen die durch die ActionHandler aufgerufen werden. BeispielmodulList() sorgt dafür, das alles angezeigt wird, was der Benutzer beim Aufruf des Moduls sehen soll. Die Funktionen MenuEintrag fügen neue Menüpunkte zur Navigation des Moduls hinzu. TableSearch gibt an, welche Live Tabelle zur Darstellung der Daten verwendet werden soll, in diesem Falle die Live Tabelle beispielmodul_list. Ohne die Angabe einer Template-Datei mit der Funktion Parse, kann kein Inhalt dargestellt werden

Angabe einer Template-Datei mit der Funktion Parse, kann kein Inhalt dargestellt werden.

function BeispielmodulList() { $this->app->erp->MenuEintrag("index.php?module=beispielmodul&action=create","Neuer Eintrag"); $this->app->erp->MenuEintrag("index.php?module=beispielmodul&action=list","Übersicht");

```
 $this->app->YUI->TableSearch( 'TAB1','example_module_list', "show","","",basename(__F
$this->app->Tpl->Parse( "PAGE", "example_module_list.tpl");
```

Innerhalb der Funktion BeispielmodulEdit() wird der Code definiert, der bei einem Klick auf das Edit Symbol eines Eintrags der Live Tabelle ausgeführt werden soll. Zuerst wird mit MenuEintrag neue Menüpunkte zum Modul hinzugefügt. Einer der auf die aktuelle Seite verweist und einer der zurück auf die Live Tabelle verlinkt. Über GET wird in die Variable $id die ID von dem zu bearbeitenden Auftrag übertragen. Mit POST wird der Wert des Speichern Buttons geholt. Außerdem wird ein Array erstellt, dass über GetInput (weiteres weiter unten) die Werte aus den Eingabefeldern holt. Ist die ID gesetzt und eine Zahl und der Speichern Button wurde gedrückt, wird geprüft ob alle Pflichtfelder befüllt wurden. Ist dies nicht der Fall, erhält die Variable $error eine jeweilige Fehlermeldung. Ist die Variable $error nicht leer, sprich nicht alle Pflichtfelder wurden befüllt, wird eine Fehlermeldung im Templatefeldbereich MESSAGE angezeigt. Andernfalls wird der bisherige Eintrag mit UPDATE aktualisiert und eine Erfolgsmeldung wird angezeigt. Desweiteren wird das derzeitige für diesen Eintrag hinterlegte Datum aus der Datenbank geholt. Selbiges gilt für die Beschreibung. Mit Hilfe von Set werden diese Werte in die Eingabefelder voreingetragen, damit die Person, wenn sie die Einträge bearbeiten will sieht, welche Werte genau sie bearbeitet. Parse gibt zum Schluss an welche Template-Datei verwendet werden soll.

function BeispielmodulEdit() { $this->app->erp->MenuEintrag("index.php?module=beispielmodul&action=edit&id=$id","Details"); $this->app->erp->MenuEintrag("index.php?module=beispielmodul&action=list","Zurück zur Übersicht");

```
 $id = (int)$this->app->Secure->GetGET( 'id');
$store = $this->app->Secure->GetPOST( 'store');

$input = array();

$input = $this->GetInput();

if(is_numeric($id) && $store!=''){

 $error = '';

 if($input['date']=='') $error .= 'Please enter a date.<br>';
 if($input['description'] == '') $error .= 'Please enter a description.<br>';

 if($error!=''){
  $this->app->Tpl->Set( 'MESSAGE',"<div class=\"error\">$error</div>");
 }else{
  if($error == ""){
   $this->app->DB->Update( "UPDATE example_module SET date ='{$input['date']}', desc

   $this->app->Tpl->Set( 'MESSAGE', "<div class=\"success\">The settings were success
  }
 }
}
$date = $this->app->DB->Select( "SELECT date FROM example_module WHERE id = '$id'"
$description = $this->app->DB->Select( "SELECT description FROM example_module WH
```

```
$this->app->Tpl->Set( 'DATE', $date);
$this->app->Tpl->Set( 'DESCRIPTION', $description);

$this->app->Tpl->Parse( 'PAGE', "example_module_edit.tpl");
}
```

BeispielmodulDelete() löscht den gewählten Eintrag aus der Live Tabelle bei Klick auf das Delete Symbol. Mittels der Variable $_SERVER['HTTP_REFERER'] kann ermittelt werden, von welcher Seite der Benutzer kam, also in diesem Fall die aktuelle Seite mit der Live Tabelle. Über GetGET wird die ID des zu löschenden Eintrages geholt und mit einem MySQL Statement wird der Eintrag anschließend gelöscht. Zum Schluss wird auf die bisherige Seite weitergeleitet.

function BeispielmodulDelete() { $ref = $_SERVER['HTTP_REFERER']; $id = $this->app->Secure->GetGET("id"); $this->app->DB->DELETE("DELETE FROM beispielmodul WHERE id = '$id' LIMIT 1");

```
 header("Location: $ref");
exit;
```

Die Funktion BeispielmodulCreate() legt einen neuen Eintrag in der Datenbanktabelle beispielmodul an. Zuerst wird mit MenuEintrag ein neuer Menüpunkt zum Modul hinzugefügt, der zurück auf die Live Tabelle verweist. Mit der Funktion GetInput werden die Werte aus den Eingabefeldern geholt. Die Erklärung zu dieser Funktion erfolgt weiter unten. Über GetPOST wird der Wert des Speichern Buttons geholt. Wenn dieser Wert nicht leer ist, also gedrückt wurde, wird geprüft, ob die Pflichtfelder bei der Eingabe ausgefüllt wurden. Falls diese nicht ausgefüllt wurden, erhält die Variable $error eine jeweilige Fehlermeldung. Enthält die Variable $error nun also einen Wert, wird die Fehlermeldung im Templatefeldbereich MESSAGE ausgegeben. Andernfalls wird der neue Eintrag hinzugefügt und die Meldung des erfolgreichen Hinzufügens wird angezeigt. Ist der Speichern Button nicht gedrückt worden, werden die Eingabefelder mittels der Funktion SetInput (wird weiter unten erklärt) mit den bisherigen eingegebenen Werten gefüllt. BeispielmodulCreate() verweist am Ende auf ein anderes Template, da die Oberfläche eine andere ist als bei BeispielmodulList(), jedoch kann das selbe Template wie bei BeispielmodulEdit() verwendet werden, da die Oberflächen bei beiden gleich ist.

function BeispielmodulCreate() { $this->app->erp->MenuEintrag("index.php?module=beispielmodul&action=list","Zurück zur Übersicht");

```
$input = $this->GetInput();
$speichern = $this->app->Secure->GetPOST( "speichern");

if($speichern != ""){

 $error = '';

 if($input['datum']=='') $error .= 'Geben Sie bitte ein Datum ein.<br>';
 if($input['beschreibung']=='') $error .= 'Geben Sie bitte eine Beschreibung ein.<br>';

 if($error!=''){
  $this->app->Tpl->Set( 'MESSAGE', "<div class=\"error\">$error</div>");
 }else{
  $this->app->DB->Insert( "INSERT INTO beispielmodul (datum, beschreibung) VALUES (
  $newid = $this->app->DB->GetInsertID();
  $msg = $this->app->erp->base64_url_encode( "<div class=\"success\">Der Eintrag wu
  header("Location: index.php?module=beispielmodul&action=edit&id=$newid&msg=$m
  exit;
```

```
    exit;
  }
}

$this->SetInput($input);

$this->app->Tpl->Parse( 'PAGE', "beispielmodul_edit.tpl");
```

Die Funktion GetInput erstellt eine Variable namens $input, welche ein Array beinhalten wird. Am Index datum des Arrays wird der Wert mittels POST aus der Variablen datum geholt. Selbiges geschieht für den Index beschreibung. Zum Schluss wird das Array bei return zurückgegeben.

function GetInput(){ $input = array(); $input['datum'] = $this->app->Secure->GetPOST('datum'); $input['beschreibung'] = $this->app->Secure->GetPOST('beschreibung');

```
  return $input;
```

Die Funktion SetInput erhält als Parameter ein Array. Mittels der Funktion Set werden die Textfelder für Datum und Beschreibung an der Stelle DATUM und BESCHREIBUNG gefüllt.

[en-us] function SetInput($input){ function SetInput($input){ $this->app->Tpl->Set('DATUM', $input['datum']); $this->app->Tpl->Set('BESCHREIBUNG', $input['beschreibung']); }this->app->Tpl->Set('DATE', $input['date']); $this->app->Tpl->Set('DESCRIPTION', $input['description']); }

Template-Datei beispielmodul_list.tpl

```
<div id="tabs-1">

  [MESSAGE]
  [TAB1]
  [TAB1NEXT]
</div>
```

Template-Datei beispielmodul_edit.tpl

```
<div id="tabs-1">
  [MESSAGE]
  <form action="" method="post">
    <fieldset>
      <legend>Land</legend>
      <table width="100%" border="0" class="mkTableFormular">
        <tr><td>Datum: </td><td><input type="text" name="datum" value="[DATUM]"
      <tr><td>Beschreibung: </td><td><input type="text" name="beschreibung" value
      </table>
    </fieldset>
    <input type="submit" name="speichern" value="Speichern" style="float:right"/>
  </form>
</div>
```

War dieser Beitrag hilfreich?

Ja Nein

Haben Sie Fragen? Anfrage einreichen

Verwandte Beiträge

API Dokumentation

Eigene Module für Xentral programmieren

Belege überladen

Wichtige Funktionen

Arbeiten mit dem Development Programm

Verwandte Community Beiträge