

OOP in Drupal

Raphael Schär – Prevision AG

1. Über mich
2. Einführung in OOP
3. Einführung in die Drupal Architektur
4. OOP in Drupal
5. Weiterführende Ressourcen
6. Fragen und Bemerkungen

Code Beispiele von Fast Gallery



1. Über mich

Ein paar Facts

- Master of Science in Wirtschaftsinformatik an der Uni Zürich
- Betreiber von rapsli.ch
- Eigentümer von Schär Webdesign
- Seit 1. Mai bei der Prevision



Rapsli's Drupal World

Über mich | Blog | Snippets | Webdesign | Fortsatz | Drupal Themes | Bilder | FAQ

Suchen

Beliebte Inhalte

- Debuggen mit PHPeclipse
- Objektorientierte Programmierung in Drupal - Drupal Architektur
- Drupal Interface Module - Nodeformaussehen verändern
- Feedback zum Themebuilder gesucht
- Deil Inspiron 510m Ram aufrüsten
- E-commerce vs Ubercart, eine kleine Sammlung
- Patchen mit Windows - Tutorial
- Inagocache erstellt Bilder nicht -> Pfad falsch
- Zu viel Drupal?
- Drupal Themes vs. Wordpress Themes

Mein Twitter

↳ drs.ch is presenting there website

Drupal Media Camp

submitted by rapsli on Di, 05/07/2009 - 20:29

↳ Drupal ↳ dnc ↳ Drupal ↳ drupal media camp ↳ switzerland

Geschrieben am 07. May 2009

Oké, morgen ist es soweit. Ich muss noch die letzten Folien für meine kleine Präsentation machen. Weiss schon jemand, ob er teilnehmen wird? Würde mich interessieren, was so die Hintergründe/Ergebnisse sind und was die Erwartungen sind. Oder kommt vielleicht gar niemand an meine Session? ...? Hoffen wir es mal nicht.

↳ rapsli's blog ↳ Neuen Kommentar schreiben ↳ 34 Abrufe

Objektorientierte Programmierung in Drupal - Drupal Architektur

submitted by rapsli on Mi, 05/05/2009 - 15:06

↳ Drupal ↳ architektur ↳ Drupal ↳ entwicklung ↳ Modulentwicklung ↳ OOP

Geschrieben am 06. May 2009

Die Drupal Architektur in Kürze

Ich muss an dieser Stelle sagen, dass ich auch nicht in-depth Kenntnisse der genauen Vorgehensweise habe, ich bin kein Drupal Core Entwickler. Die Grundlegenden Prinzipien sollten dennoch klar sein. Hier sind eigentlich zwei wichtige Grundfragen:

snippets entwicklung themen Joomla Drupal tutorial PHP review fast gallery how-to theme Modulentwicklung tips statistik neues modu modu more tags

↳ rss

Abonnieren Sie Rapsli's World per E-Mail

Neueste Kommentare

- Hallo Gast ist halt nicht vor 1 Tag 22 Stunden
- CC: als Anfänger empfinde ich vor 6 Tage 21 Stunden
- Wenn man erst einmal vor 6 Tage 21 Stunden
- Die slides werden sicher vor 1 Woche 20 Stunden
- Hoi Rapsli ich werde leider

Generelles

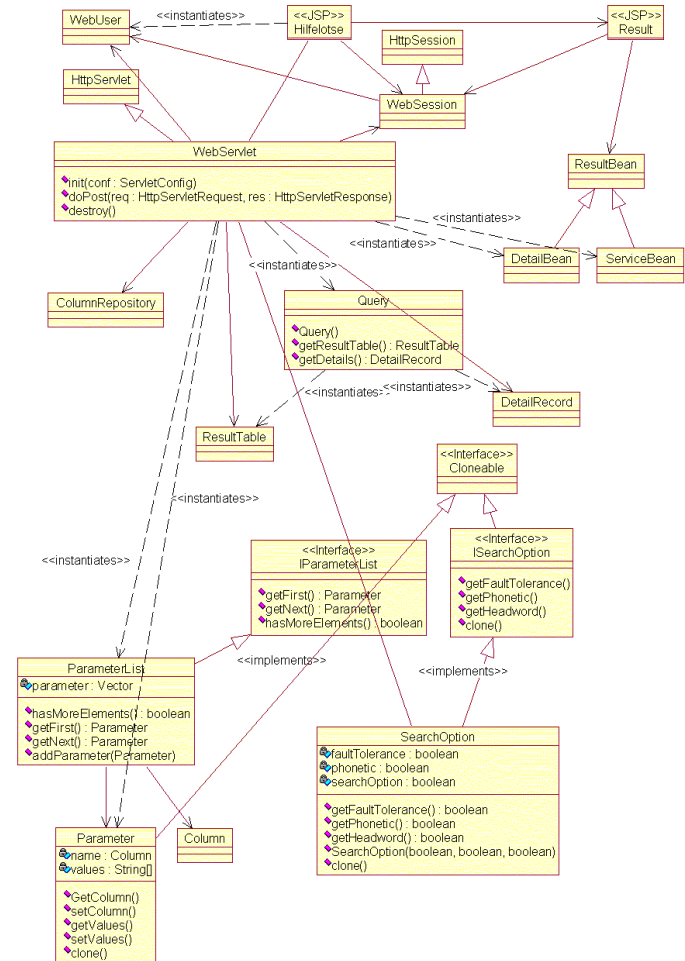
- Was ist der Publikums Background?
- Kommentare Willkommen
- Fragen Willkommen
- -> Ich bin (noch) nicht allwissend ;)
- Der Code ist nicht ganz so sauber wie er sein sollte, da diverse Leute daran gebaut haben.



2. Einführung in die OOP

Wichtige Begriffe der OOP

- OOP (Objekt Orientierte Programmierung)
- Klassen
- Objekte
- Methoden
- Variablen



Vorteile von OOP

- Stärkere Kapselung möglich (Objekt = BlackBox)
- Vererbungsmöglichkeit: Erlaubt eleganter & schlanker Code
- Wiederverwendung erleichtern durch Kapselung
- Leichtere Wartung von Code (aufgrund Kapselung)
- Programmierung gegen Interfaces

Klasse

- Ist der Bauplan für ein Objekt
- Vorgabe um ein Objekt zu erstellen
- Klasse \neq Objekt



Beispiel Klasse

```
class FastGalleryCache {
  /**
   * Creates an image resource from a path. Supported filetypes are GIF,
   * JPEG, and PNG.
   * $param path
   * Path to the image.*/
  function imagecreatefromfile($path) {
    //some smart code
    //...
    return $types[$info[2]] ($path);
  }
  /**
   * Removes all the thumbs.
   * @param path
   * Path of directory where thumbs are located.
   * @param recursive
   * Specify whether to recurse through subdirectories.*/
  public function flushThumbs($path = '', $recursive = FALSE) {
    // Get all .thumb files from given path
    $files = glob($path . '*.thumb');
    //....
  }
}
```

Objekt

- Ein Objekt ist die Instanz der Klasse
- Die Klasse ist abstrakt und wird nicht verwendet, das Objekt ist konkret und „macht“ etwas.
- Jedes Objekt muss vor der Verwendung initialisiert werden:

```
$cache = new FastGalleryCache ();  
$cache->createthumb („image.jpg“, 200, 100);
```

Methoden

- Gibt dem Objekt Funktionalität.
- Jede Methode hat einen Sichtbarkeitsbereich:
 - Public: Kann von einem Benutzer verwendet werden
 - Private: Kann nur vom Objekt verwendet werden
 - Static: benötigt keine Instanzierung
- Beim Refactoring können die privaten Methoden beliebig verändert werden. Nicht so bei den Public Methoden, denn diese sind die Schnittstellen.

Variablen

- Gibt dem Objekt einen Zustand bzw. Eigenschaften.
- Wiederum: public, private, static
- Grundsätzlich sollte Variablen immer auf private gesetzt werden und der Zugriff über sog. Accessor Methoden erfolgen

```
class FastGalleryCache{  
    private $state = 0;  
    public function getState(){  
        return $state;  
    }  
}
```



3. Die Drupal Architektur

Front-Controller Pattern

- Sehr beliebt in Web-Anwendungen
- Es wird nicht eine einzelne Seite aufgerufen
- index.php (im Normalfall) lädt das ganze System (Bootstrap Phase)
 - content/ein-test (landet bei index.php, wo entschieden wird, was für eine Seite rauskommt)

Das Hook System



- Drupals Stärke! Hook = Haken
- Erlaubt es Modulen mit Drupal zu interagieren
- Arbeitsschritt läuft wie folgt ab:
 - Ein Hook wird aufgerufen, z.B. hook_menu
 - Drupal Durchläuft alle Module, welche diesen hook implementieren
 - Nächster Hook, wieder das Gleiche.
 - Das Gewicht eines Moduls bestimmt, wie früh oder spät ein hook aufgerufen wird.

Hook System Teil II

- Extrem Flexibel!!!
- Modifikationen sind möglich ohne Core zu ändern.
 - Daher „don't hack Drupal“.
- Es lassen sich auch eigene Hooks erstellen (views macht das z.B.)
- Theming folgt den gleichen Prinzipien.

```
print theme('image','mein-bild.jpg');
```



4. OOP in Drupal

Grundsätzliches



- Nicht nötig alles auf OOP umzustellen! Hook-System bietet bereits ein sehr flexibles System!
- Nicht die Architektur umbauen, sondern die Entwicklung von eigenen Modulen verbessern.
- Views 2 verfolgt einen solchen Ansatz.

Struktur eines OOP Moduls

- sites/all/modules
 - myModule
 - myModule.module
 - myModule.info
 - evtl. myModule.css
- sites/all/modules
 - myModule
 - myModule.module
 - myModule.info
 - evtl. myModule.css
 - **myModule.class.php**

.module ist Anbindung an Drupal API
.class.php ist die eigentliche Modul Logik

Klasse Instanzieren

- Über eine Funktion kann das Cache Objekt geholt werden.
- Singleton Pattern einsetzen, da eine Instanz ausreichend ist.

```
function _fast_gallery_get_object() {  
    include_once ('fast_gallery.class.php');  
    $fg = FastGallery :: getInstance();  
    return $fg;  
}
```

Singleton Pattern

```
class FastGalleryCache {
  static private $instance = null;
  /**
   * We are implementing a singleton pattern
   */
  private function __construct() { }

  public function getInstance() {
    if (is_null(self :: $instance)) {
      self :: $instance = new self;
    }
    return self :: $instance;
  }
}
```

LIVE CODE

&

DEMO

http://www.rapsli.ch/drupal/fast_gallery/



5. Ressourcen zu OOP

Die folgenden Seiten behandeln das Thema

- <http://dc2009.drupalcon.org/session/objectifying-drupal-introduction-oop> (Larry Garfield)
- <http://www.php.net/manual/en/language.oop5.php>
- http://www.sallyahmed.com/portal/articles_76_OOP-in-PHP-Drupal-CMS.html
- -> Gibt einige Diskussionen dazu (Google)
- Und natürlich www.rapsli.ch



6. Fragen und Bemerkungen



Herzlichen Dank

Raphael Schär

rschaer@previon.ch

Previon AG

Bahnhofplatz

CH-4800 Zofingen

Telefon +41 0848 840 180

www.previon.ch