

---

# Drupal Développeur

## Theming et développement pour Drupal

*Une formation [T@hitiClic](#)*

*Formateur : Fabien Crépin*

# Introduction

---

- Objectif : savoir développer un module et un thème
- 4 demi-journées
- Ce qu'on verra en théorie et en pratique :
  - Environnement logiciel
  - Architecture de Drupal
  - Les grands principes
  - Modifier des éléments graphiques
  - Créer un thème
  - Créer un module

---

# Environnement logiciel

# Environnement logiciel

---

- Le core : PHP4, peu d'objets
- Les modules : PHP4, parfois 5, parfois l'utilisation de véritables classes, assez rare
- Donc Drupal n'est pas codé orienté objet
- Cependant, il utilise des objets ponctuellement
- Procédural, documentation format JavaDOC
- API explorable à <http://api.drupal.org>

# Environnement logiciel

---

- Le core est maintenu et très guidé par une équipe dédiée dont le créateur de Drupal
- Il existe des normes d'écritures
- Le code est versionné via CVS, tout le monde peut contribuer
- *Devel* permet d'afficher des variables de façon sympathique
- *Coder* permet de mettre en évidence les soucis syntaxiques

# Configuration de travail

---

- Eclipse installé avec le plugin PDT
- Si vous travaillez avec javascript ou ExtJS, installez aussi SpKet ([www.spket.com](http://www.spket.com))
- Mettez en forme par défaut selon les préconisations Drupal, notamment remplacez les tabulations par des espaces
- Créez un projet qui pointe vers le répertoire *sites/all* de votre site

# Systeme de fichiers

---

- Allez à la racine de votre installation et regardez les fichiers
- *files* : fichiers utilisateurs
- *includes* : fichiers des fonctions principales
- *modules* : modules du core, utilisent les includes
- *misc* : js, icones, png
- *profiles* : profils d'installation
- *sites* : les données propres aux sites
- *themes* : les themes du core
- *tmp* : fichiers temporaires

# Systeme de fichiers

---

- Points d'entrée :
- *cron.php* : invoque les tâches de maintenance, à appeler via crontab
- *index.php* : le point d'entrée principal
- *install.php* : à supprimer après installation
- *xmlrpc.php* : utile pour mettre en place des services, on lui préférera le module *Services*

---

# Thème

# Thème : les bases

---

- <http://drupal.org/theme-guide/6>
- Contenu d'un thème :
  - Fichier .info
  - template.php
  - \*.tpl.php
  - Assets : css, js, images, etc
- En général, les modules produisent le rendu au travers de fonction *theme\_\**
- Les thèmes se basent sur des moteurs, en général *phptemplate*

# Surcharge

---

- On peut surcharger les fichiers de template en copiant les fichiers *.tpl.php* dans le répertoire du thème
- On peut surcharger les fonctions *themable* dans le fichier *template.php*
- Les fichiers les plus utiles sont *page.tpl.php*, *node.tpl.php*, *block.tpl.php*, *comment.tpl.php*
- On peut particulariser des templates pour des zones ou pages, par exemple *node-recette.tpl.php*, *page-front.tpl.php*, *block-sidebar\_left.php*

# Contenu d'un fichier template

---

- On utilise surtout des variables fournies au thème qui sont juste rendues via *print*
- Quelques fonctions utilisées parfois : *check\_url* ou *theme*
- Le reste est de la mise en forme HTML et CSS
- Les variables utiles :
  - *\$language, \$head, \$scripts, \$styles*
  - les régions : *\$sidebar\_first, \$content, \$closure, etc*

# Le fichier .info

---

- Ce fichier contient
  - les infos sur le thème
  - les positions offertes
  - les styles utilisés
  - scripts utilisés
  - les caractéristiques configurables
  - les styles Skinr

# T.P. : Créons notre thème

---

- Créer un thème revient donc à fournir le fichier *.info* et le fichier *template.php*, au moins
- Dans *sites/all/themes*, créez un nouveau dossier *my\_theme*
- En vous inspirant d'un thème existant, créez un fichier *my\_theme.info*
- Ajoutez un fichier *template.php*
- Activez votre nouveau thème et mettez le par défaut
- Le rendu est produit par *modules/system/\*.tpl.php*

# T.P. : Créons notre thème

---

- Copiez les fichiers *node.tpl.php*, *page.tpl.php* et *block.tpl.php* depuis le thème garland vers le vôtre
- Créez un fichier *styles.css* et insérez le dans votre thème
- Amusez-vous un peu à modifier les styles

# Preprocessing

---

- Avant de sortir les variables vers le thème (templates ou fonctions), il y a des étapes de *preprocessing*
- <http://drupal.org/node/223430> donne l'ordre d'appel des différentes fonctions, attention les modules eux mêmes ont un poids et donc ordre
- Les fonctions de preprocessing s'empilent et modifient les variables successivement
- On garde ainsi la logique métier bien séparée de la présentation par le thème

# T.P. : Preprocessing

---

- Activez le module *Devel*
- Dans le fichier *template.php*, ajoutez ceci :

```
<?php
```

```
function my_theme_preprocess_page(&$vars){
```

```
  dsm($vars);
```

```
}
```

```
?>
```

- Choisissez une variable (*\$vars['left']* par exemple) et modifiez la dans la même fonction

# Theming à la demande avec *Skinr*

---

- *Skinr* est un module assez récent
- Il permet de spécifier des styles qui pourront être utilisées individuellement par les blocs
- Il sera sans aucun doute rapproché du Core à terme
- Pour l'utiliser, on définit des styles dans le fichier *.info*, voir le module *Acquia Prosper* pour un bon exemple

---

# Modules : Grands principes

# Hooks

---

- Les hooks sont des points d'entrée permettant aux modules d'enrichir des traitements
- Par exemple, la page d'aide est le résultat de l'invocation des différents *hook\_help*
- Les hooks très utiles :
  - *hook\_help*
  - *hook\_init* : permet de charger des fichiers au démarrage
  - *hook\_theme* : permet de préciser les fonctions themables
  - *hook\_menu* : permet d'enregistrer des chemins et menus
  - *hook\_block* : permet de créer un bloc
  - *hook\_perm* : permet de gérer les droits d'accès
  - *hook\_form\_alter* : permet de modifier un formulaire
  - *hook\_nodeapi* : permet de travailler sur les nœuds

# Menus

- Grâce à *hook\_menu*, on va pouvoir intégrer nos modules dans le menu d'administration et créer des menus locaux
- Une insertion dans un menu :

```
$menu['admin/build/visual_admin'] = array(  
  'title' => t('Visual Admin'),  
  'description' => t("Provides a way to define admin buttons."),  
  'page callback' => 'visual_admin',  
  'access arguments' => array('administer visual_admin'),  
  'file' => 'visual_admin.admin.inc',  
);
```

*On peut ajouter un type : MENU\_LOCAL\_TASK, MENU\_CALLBACK*

# Les formulaires

---

- [http://api.drupal.org/api/drupal/developer--topics--forms\\_api.html](http://api.drupal.org/api/drupal/developer--topics--forms_api.html)
- On les utilise pour enregistrer des contenus ou pour l'administration
- Un formulaire est :
  - Un tableau `$my_form`
  - Des éléments `$my_form['my_element']`
  - Des attributs `#attribut` (du formulaire ou des éléments)
- On le récupère et rend via `drupal_get_form('nom_de_la_fonction_qui_genere');` ou `drupal_render_form($my_form);`

# Les formulaires

---

- Pour chaque formulaire on peut définir les fonctions :
  - *nom\_du\_formulaire\_validate* : permet de contrôler la saisie
  - *nom\_du\_formulaire\_submit* : permet de définir ce qui se passe à la soumission
- *system\_settings\_form(\$my\_form)* permet de créer des formulaires à vocation d'enregistrer des valeurs dans la table *variables*
- *hook\_form\_alter* vous permet de modifier un formulaire depuis une autre fonction

---

Module

# Le contenu d'un module

---

- Fichier *.info* : contient la désignation du module, les dépendances
- Fichier *.module* : le fichier principal du module
- En général, on trouve aussi :
  - Fichier *.admin.inc* : fonctions de l'administration du module
  - Fichier *.[DB].inc* : fonctions d'accès à la base
  - Fichier *.theme.inc* : fonctions themables
  - Fichier *.install* : schéma de la base, procédure d'installation
  - Dossier *Translations* : contient les fichiers *\*.po*
  - D'autres fichiers : js, css, images, classes, etc

# Le fichier .info

---

- C'est en gros le manifeste du module
- Typiquement :

`visual_admin.info`

`; $Id$`

`name = Visual Admin`

`description = Provides a visual administration page`

`core = 6.x`

`version = "6.x-0.3"`

`dependencies[] = menu`

# Le fichier .info

---

- D'autres informations peuvent y figurer et sont ajoutées lors du commit (date notamment)
- On retrouve certaines informations sur la page des modules

# Le fichier .module

---

- Il est nécessaire mais ne contient pas forcément tout
- On préférera séparer le code sur plusieurs fichiers
- Une bonne pratique est de créer les hooks Drupal en 1er, et notamment *hook\_help*

# Fonctions Utiles : générales

---

- *t* : rend la chaîne traduisible
- *theme* : invoque une fonction de mise en forme
- *drupal\_set\_message* : permet d'afficher des messages
- *variable\_get* : permet de récupérer des variables enregistrées
- *drupal\_add\_css/js* : insère des fichiers où il doivent aller
- *path\_to\_theme* : renvoie le chemin du thème
- *db\_query* : exécute une requête
- *db\_fetch\_\** : diverses opérations sur les résultats
- *check\_url*, *check\_plain* : contrôles

# Fonctions Utiles : particulières

---

- Pour les autres modules, il faut aller voir les modules en question soit sur la doc en ligne, soit directement dans votre projet Eclipse
- Par exemple peuvent servir suivant les projets :
  - *taxonomy\_get\_vocabularies*
  - *taxonomy\_get\_tree*
  - *menu\_get\_item/menu\_set\_item*
  - *menu\_get\_menus*

# T.P. : créons notre module

---

- Nous allons créer un module qui affiche des titres de nœud dans un bloc
- Nous paramètrerons le nombre et le titre via une page d'administration
- Puis nous créerons une page affichant ces nœuds en utilisant des fonctions de thème

## T.P. : créons notre module

---

- Commencez par créer le `.info` et le `.module`
- Ajoutez les hooks de base : `hook_help`, `hook_perm`
- Ajoutez ensuite le `hook_block`
- Googlez `hook_block` pour les infos précises, il faut retenir qu'on peut définir plusieurs blocs (`$delta`) et gérer les diverses opérations liées aux blocs : `list`, `view`, `configure`, `save`

# T.P. : créons notre module

---

- Ajoutez une interface de configuration pour votre module
- Séparez la collecte des informations du rendu graphique
- Surchargez dans votre thème le template ainsi créé

# Liens utiles

---

- <http://www.drupal.org>
- <http://api.drupal.org>
- <http://api.audean.com/api>
- <http://views.doc.logrus.com/>
- Images :
  - <http://www.sxc.hu>
- Thèmes :
  - <http://www.themebot.com>
  - <http://www.themegarden.org>
  - <http://www.templatemonster.com>
  - <http://www.osskins.com>

# Liens utiles

---

## Blogosphère :

- <http://www.kolossaldrupal.org>
- <http://www.ineation.com>
- <http://www.tahiticlic.com/blog>
- <http://www.drupalistic.net>