FORMATION DRUPAL

Support de cours - Mise à jour 02/11

Présentation de Drupal

Drupal est un CMS libre et open source créé en 2001 par Dries Buytaert.
Il s'adresse à la fois à des débutants ou des programmeurs experts. Sa flexibilité lui permet de répondre à la très grande majorité des besoins du marché: sites institutionnels, blog, annuaire, communautaire, marchand ou intranets.

Rapidité de mise en place
Drupal permet d'obtenir un site fonctionnel et évolutif en quelques clics sans écrire une seule ligne de code.

Modularité
L'on peut étendre les possibilités en ajoutant des modules. Ceux-ci sont très nombreux, proposés par la communauté et toujours sous licence libre GPL.

Cette architecture permet aux programmeurs de réaliser des modules personnalisés afin d'étendre les fonctionnalités sans pour autant modifier le corps de Drupal.

Le code de Drupal
La qualité du code et la robustesse de son interface de programmation (API), font que Drupal est également présenté comme un environnement de développement PHP (Framework). On parle alors de « Content Management Framework ».

Spécificités de Drupal
Drupal est entièrement programmé en PHP. L'ensemble est constitué de modules gravitant autour d'un noyau très léger. Chaque module est en quelque sorte une bibliothèque de fonctions qui enrichit l'application et augmente ses possibilités.

Une des forces de Drupal est la possibilité qu'ont les modules d'interagir entre eux. La contrepartie de cette flexibilité est la complexité, ainsi, Drupal propose souvent une ou plusieurs solutions pour résoudre le même problème. D'autre part, le plus dur est quelquefois de trouver «le» module qui répondra le mieux à vos besoins.

Un autre point qui distingue Drupal des autres CMS est que le site et son interface d'administration sont intimement liés: les administrateurs éditent leur contenu dans le même contexte graphique ou presque que celui du visiteur. Cette fonctionnalité peut-être déroutante au premier abord, mais est par la suite particulièrement productive et intuitive.

Comme tous les CMS, l'architecture d'un site Drupal repose sur un modèle de contenu spécifique qui permet de structurer l'information. Drupal utilise un système de noeud couplé à une taxonomie particulièrement flexible.

Concepts fondamentaux de Drupal

Le node et le type de contenu

Le nœud est l'élément central du modèle de contenu employé par Drupal. Dans sa version la plus simple, un nœud est un objet générique composé d'un titre et d'un contenu de type texte.

Toute la puissance de Drupal réside dans la possibilité de spécialiser un nœud en lui adjoignant des champs supplémentaires. C'est ainsi que l'on va créer ce que l'on appelle un nouveau type de contenu. Ainsi, chaque élément de contenu présent dans un site Drupal est un nœud et chaque nœud appartient à un type de contenu.

La distribution de base vient avec deux types de contenu (article et page). Ce sont des types de contenu génériques qui conviennent à de nombreuses situations. Cependant, vous aurez certainement besoin d'autres types de contenu. Drupal vous permet de les créer soit par le truchement de modules optionnels soit par l'utilisation d'un module indispensable appelé CCK.

Pour différencier un type de contenu, vous pouvez lui associer de nouveaux champs. Grâce à ses modules, Drupal dispose d'une multitude de champs qui vous permettront de stocker des éléments de contenu extrêmement variés : texte, chiffre, date, url, numéro de téléphone, image, vidéo, carte.

Exemple :

Pour votre site d'actualités, vous considérez que le contenu de type Article n'est pas suffisamment structuré, vous pouvez alors créer un nouveau type de contenu Article_Journal et lui adjoindre un sous-titre (champs texte) et une image principale (champ image).

Types de contenu standards

Les types de nœud par défaut de Drupal sont :

Article
Contenu court et/ou traitant d'un point précis (sans lien direct avec d'autres articles). Ils comportent un titre et un corps de texte, mais peuvent être étendus par d'autres modules. L'accroche fait également partie du corps. Les articles peuvent être utilisés dans un blog personnel ou dans un site d'actualités.

Page
Contenu plutôt statique ou page orpheline (typiquement la page de crédit du site, une page de présentation).

Quelle est la différence réelle entre une «page» et un «article» ?

Pour faire court : aucune. En réalité, il y en a une, mais c'est uniquement au niveau de l'affichage par défaut de ces pages sur le site. En effet, au contraire d'un article, une page n'affiche pas les informations de l'auteur et la date de publication, d'autre part, les commentaires sont désactivés par défaut. Ces paramètres d'affichage sont aisément modifiables ce qui fait que au final, d'un point de vue technique il n'y a aucune différence.

L'intérêt est ailleurs, la différentiation des types de contenu est un moyen à la disposition du concepteur pour bâtir un modèle de contenu adapté au besoin. Dans ce cas, cela permet de séparer strictement et de traiter différemment les informations statiques d'une « page » (C.G.V, à propos...) qui sont rarement mises à jour, des informations dynamiques d'un « article » dont la durée de vie est plus faible (nouvelles, promotions...)

Livre
Ensemble de nœuds liés entre eux par des liens hiérarchiques (parent, enfant, frère). Cette structure hiérarchique (un sommaire) est générée automatiquement à l'affichage d'un nœud de ce type. Comme son nom l'indique, ce contenu est adapté à un contenu structuré en chapitre et sous-chapitre, un contenu organisé autour d'une progression logique.

Un livre est un effort collectif d'écriture : les utilisateurs peuvent collaborer à l'écriture des pages du livre, positionner les pages dans le bon ordre, et passer en revue ou modifier les pages écrites précédemment.

Et si j'ai besoin d'autres types de contenu ?

Il est tout à fait possible dans Drupal de rajouter autant de type de contenu qu'on le souhaite. Pour ajouter de nouveaux champs, il faut néanmoins utiliser un module qui n'est pas livré avec la distribution de base, CCK, ou le programmer via un module personnel.

L'utilisation de CCK est intuitive et ne nécessite pas de connaissances en programmation : Il suffit de donner un nom (et quelques autres options) à un nouveau type de contenu pour qu'il soit directement exploitable. On peut lui ajouter autant de champs que nécessaire presque aussi simplement. Bien entendu, comme tous les modules Drupal, CCK est disponible librement sous licence GPL.

Commentaires

Les commentaires sont des éléments classiques d'un CMS communautaire et ont été popularisés par les blogs. C'est la base de l'interaction entre le(s) éditeur(s) d'un site et les lecteurs, car ils permettent à ces derniers de commenter le contenu en question.

Dans Drupal, les commentaires ont pour particularité de ne pas être des nœuds. Ce sont donc des éléments à part dans le modèle de contenu. Cette architecture a été retenue parce que plus performante et donc capable de gérer des milliers de commentaires. Concrètement, cela signifie qu'ils ne seront pas exploitables de la même façon que les autres contenus, mais on peut tout de même les personnaliser comme on le souhaite.

Tous les nœuds de Drupal peuvent recevoir des commentaires et on peut paramétrer la manière dont ceux-ci se comporteront. Ces réglages peuvent être différents pour chaque type de contenu ou même pour chacun des nœuds.

Catégories, vocabulaire, taxonomie, terme

La taxonomie est le concept qui permet la structuration des différents éléments de contenu d'un site Drupal. Très puissant et flexible il est parfois difficile à appréhender, car il utilise des mots qui ne font pas partie du langage courant : terme, vocabulaire et taxonomie

Le « terme » est l'étiquette qui va servir à désigner un regroupement de différents nœuds.

Un « vocabulaire » est un groupe de plusieurs « termes ».

La « taxonomie » va préciser le type de relation qui existe entre les termes d'un même vocabulaire.

* La taxonomie libre : les termes peuvent être choisis sans contrainte
* La taxonomie simple : un seul terme par contenu
* La taxonomie multiple : plusieurs termes peuvent être choisis parmi une liste prédéfinie

Enfin une taxonomie peut définir des relations hiérarchiques entre les termes afin de définir des catégories et sous catégories.

Drupal permet donc de créer un vocabulaire pour chaque besoin de classification qu'il peut avoir dans un site. Par exemple, un site de recettes de cuisine pourra avoir un vocabulaire « Tag » de taxonomie libre pour classer les contributions des lecteurs avec des termes divers comme « épice » ou « facile » et avoir en même temps un vocabulaire « Région » de taxonomie simple pour classer la provenance des recettes avec des termes comme « Auvergne » ou « Bourgogne ».

Bloc

Les blocs sont des éléments de contenu spécifiques. En général, ils sont utilisés pour afficher une information secondaire (commentaires récents), un contenu lié au contenu principal (profil de l'auteur de l'article) ou pour afficher des éléments de navigation (menu, formulaire de connexion). Certains modules définissent leurs propres blocs que vous pouvez ou non activer à votre guise. Vous pouvez également créer de nouveaux blocs via l'interface d'administration.

Chaque bloc possède une page de configuration qui vous permet de régler sa visibilité : selon les pages affichées, selon le visiteur ou encore selon une logique plus complexe régie par un code en PHP.

Le bloc peut être placé dans une région définie par le thème graphique du site. Le nombre et l'emplacement des régions sont fonction du thème utilisé. Par exemple le thème par défaut « Garland » définit les régions suivantes : haut de page, barre latérale gauche, barre latérale droite, contenu principal et pied de page. « Zen », un autre thème populaire, lui, définit des régions supplémentaires comme la partie supérieure du contenu ou inférieure du contenu.

Menu

Un menu est simplement une liste d'hyperliens vers des pages internes ou vers d'autres sites. Cette liste peut-être hiérarchisée afin de permettre de créer des menus et des sous-menus.

En général ils sont utilisés afin de créer des éléments de navigation pour les utilisateurs de votre site. Drupal crée par défaut un « menu primaire » (primary links) souvent utilisé pour la barre de navigation principale et un « menu secondaire » (secondary links) employé pour des liens de moindre importance. Une fois encore Drupal ne vous limite pas et vous pouvez créer autant de menus que votre site nécessite.

Lorsque l'on créé un menu, celui-ci génère automatiquement un bloc. Vous pouvez alors paramétrer l'emplacement de ce bloc pour que votre menu s'affiche à l'écran.

Utilisateur, rôles et droits

Chaque « utilisateur » du site est identifié lors de sa connexion. Les utilisateurs ont un « rôle » qui leur est assigné. Chaque « rôle » dispose d'un certain nombre de « droits ». L'ensemble des « droits » d'un rôle permet de spécifier ce qu'il est autorisé à faire sur le site.

Le nombre de rôle est illimité et peut donc être adapté finement au besoin du site. Par exemple, un site pourrait avoir les rôles suivants :

* visiteur anonyme
* administrateur,
* webmestre,
* rédacteur,
* utilisateur authentifié...

Chaque utilisateur Drupal est associé à un ou plusieurs rôles, dans ce cas, les droits se cumulent.

Pour finir, sachez qu'il existe un certain nombre de modules qui permettent d'étendre les fonctionnalités de gestion des droits et des utilisateurs afin de permettre une gestion des droits d'accès encore plus fine (pour chaque utilisateur par exemple).

Thème

Comme tous les CMS modernes, l'architecture de Drupal est conçue de manière à séparer strictement la logique, le contenu et l'apparence d'un site. L'un des nombreux avantages de cette technique réside dans la possibilité de modifier totalement l'apparence d'un site sans avoir à toucher à la mécanique interne de l'application ou les données du contenu. Il est même tout à fait possible de faire cohabiter plusieurs thèmes dans une même installation et de laisser le choix du thème aux utilisateurs.

Le thème employé par défaut est appelé «Garland». Bien entendu, il existe de nombreux thèmes disponibles librement sous licence GPL. Vous en trouverez une liste presque exhaustive sur http://drupal.org/project/Themes Cependant la plupart de vos projets nécessiteront de créer votre propre thème, dans ce cas vous pourrez soit le créer de toute pièce soit partir d'un des thèmes de base comme « Zen » afin de gagner en productivité et bénéficier d'une structure de base de grande qualité.

Les thèmes Drupal sont des ensembles de fichiers qui modifient l'affichage par défaut de votre site. Pour simplifier le travail de mise en page et en style, Drupal permet d'utiliser plusieurs « moteurs de template » (engines) qui codifient les instructions d'affichage. Par défaut, c'est le moteur PHPtemplate qui est utilisé mais d'autres comme SMARTY sont disponibles si vous le souhaitez.

L'affichage par défaut est intercepté et modifié (override) par le thème sélectionné à deux niveaux :

* La structure XHTML par défaut peut être interceptée afin de modifier la nature, l'ordre et le balisage sémantique des informations affichées
* Les styles CSS par défaut peuvent être interceptés afin de modifier l'apparence et la mise en page de ces mêmes informations

Cœur ou base (Core)

Core est un terme anglophone désignant le cœur, l'élément central...

Dans Drupal, le cœur est en fait la distribution officielle qui est livrée avec un certain nombre de modules et de thèmes standards. Parmi les modules livrés dans le cœur, tous ne sont pas activés par défaut. En fait, seul un petit nombre de modules sont strictement nécessaires au fonctionnement de Drupal. Le cœur de Drupal est donc léger. La contrepartie est qu'il ne permet de réaliser que des applications simples, limitées en nombre de fonctionnalités. Ce fait est d'ailleurs source de confusion pour le débutant qui mesure la richesse fonctionnelle de Drupal à l'un de ces modules de base. En fait une grande partie de la puissance de Drupal est apportée par ses modules optionnels.

Module (module)

On pourrait comparer le concept de module à celui de plugin ou greffon. Un module est une brique logicielle s'ajoutant à votre application pour étendre ses fonctionnalités ou modifier celles existantes.

Il est très simple d'activer ou non un module dans Drupal : une simple case à cocher suffit.

En pratique, un module est une bibliothèque de fonctions php qui s'interfacent avec les modules déjà présents par le biais des hameçons.

Hameçon (Hook)

Un hameçon est un point d'entrée dans le processus logique de votre site. Il permet aux modules de modifier le fonctionnement de l'application sans avoir à modifier les fichiers standards.

Cela présente un énorme avantage : lorsque une mise à jour de Drupal est disponible, vous pouvez l'installer sans perdre vos modifications.

Formulaire (form)

Il y a plusieurs types de formulaires dans Drupal :

* formulaires de saisie de contenu
* formulaires d'administration des modules

Pour la saisie de contenu, le formulaire est généré automatiquement lorsque vous créez un nœud, il permet à l'éditeur d'écrire les contenus de son site. Les formulaires d'administration des modules permettent aux administrateurs de paramétrer le fonctionnement de Drupal et de ses modules.
Tout formulaire Drupal peut être étendu / modifié par d'autres modules et également avoir son propre gabarit.

Vue (view)

Les vues ne font pas partie de la distribution de base de Drupal mais sont disponibles sous la forme d'un module optionnel : Views. Nous évoquons tout de même ce concept ici car Views est l'un des modules qui constituent la panoplie indispensable de tout administrateur Drupal.

Une vue permet d'afficher une liste de nœuds, d'utilisateurs ou de commentaires en fonction de critères de tris, de filtres ou encore d'arguments plus complexes. Par exemple, vous pouvez créer une vue pour afficher « les derniers articles », « les derniers inscrits » ou, plus compliqué « 12 commentaires au hasard parmi ceux qui ne sont pas validés et qui sont associés à des contenus de type X de l'auteur Y ».

Le module, qui peut-être complété par de nombreux autres modules, permet de représenter les vues de multiples façons : vue complète du nœud ou simplement son accroche, tableau, liste, calendrier, carte géographique...

Le tout avec une interface claire et intuitive, elle permet d'obtenir une vue sans programmation et très rapidement.

Panneau (panel)

Comme pour Views, Panels fait parti des modules indispensables de Drupal.

Avec ce module, il est possible de réunir dynamiquement, sans écrire une ligne de code, différents éléments de contenus sur une seule page... Par exemple, vous pouvez faire une page d'accueil complexe comme sur les sites portails en quelques clics.

Un panneau peut contenir des nœuds, des profils d'utilisateur, des commentaires, des blocs, des vues et même d'autres panneaux. Comme il définit des contextes propres à chaque panneau, Panel sait gérer l'affichage de contenus connexes en fonction de l'auteur, la taxonomie, le groupe qui est affiché...

Site officiel de Drupal France

http://drupalfr.org/

Le site est géré par des bénévoles de l'association Drupal France et Francophonie.

- On y trouve notamment :
- Documentation en français
- Aide à l'installation de Drupal
- Forums
- Liens pour le téléchargement de modules et des dernières versions de Drupal

http://drupal.org/

Structure identique mais documentation plus complête en anglais, support, possibilité de contribuer à la communauté (code, debugging, traduction, documentation, etc)

Installer et paramétrer Drupal

Installer Drupal en local

Wamp
Télécharger puis installer Wamp (http://www.wampserver.com/)
La dernière version de WampServer comprend :
- Apache 2.2.11
- MySQL 5.1.36
- PHP 5.3.0

* Apache : c'est ce qu'on appelle un serveur web. Il s'agit du plus important de tous les programmes, car c'est lui qui est chargé de délivrer les pages web aux visiteurs. Cependant, Apache ne gère que les sites web statiques (il ne peut traiter que des pages HTML). Il faut donc le compléter avec d'autres programmes.
* PHP : c'est un plug-in pour Apache qui le rend capable de traiter des pages web dynamiques en PHP. En clair, en combinant Apache et PHP, notre ordinateur sera capable de lire des pages web en PHP.
* MySQL : c'est le logiciel de gestion de base de données dont je vous ai parlé en introduction. Il permet d'enregistrer des données de manière organisée (comme la liste des membres de votre site). Nous n'en aurons pas besoin immédiatement, mais autant l'installer de suite.

Lors de l'installation, un dossier «www» est créé, généralement dans c:\wamp\www.
Nous placerons notre copie de Drupal à l'intérieur de ce dossier.

L'interface de gestion de Wamp est disponible à cette adresse : http://localhost

Télécharger Drupal
Rendez vous sur la page d'accueil du site http://drupal.org/, vous y trouverez un bloc sur la droite intitulé «Download».

Décompresser l'archive

Copier l'intégralité de l'archive dans votre dossier web
Le dossier web est l'emplacement, sur votre serveur HTTP, où sont stockés les sites web que vous hébergez. Ici, il s'agit du dossier «www» créé par Wamp à l'emplacement c:\wamp\www.

Créer la base de donnée dans le phpMyAdmin de Wamp
Penser à noter l'identifiant de l'utilisateur et son mot de passe.
Par défaut root/root

Saisir l'url dans la barre d'adresse du navigateur
Drupal détecte que votre installation n'est pas configurée. Vous êtes donc redirigés automatiquement vers l'application d'installation sur url/install.php?profile=default.

Pour installer la version française
Choisissez «Learn how to install Drupal in other languages» puis cliquez sur «Determine if a translation of this Drupal version».
Recherchez la version de votre choix, intitulée «French translation». Cliquez sur le lien «Download» à côté de la version Drupal que vous vous apprêtez à mettre en place.


Drupal 6 translation status

The Drupal 6 translation is nearly complete. A release will be published within the next few weeks.

Version	Date	Links	Status
6.x-1.x-dev	2008-Jun-30	Download · Release notes	Development snapshot
5.x-1.1	2008-Jan-09	Download · Release notes	Recommended for 5.x
4.7.x-	2008-Jan-22	Download · Release notes	Development snapshot

Décompressez l'archive et placez les fichiers qu'elle contient au même endroit que ceux de la distribution standard de Drupal. Si une alerte vous indique que des dossiers ou les fichiers existent déjà, ce n'est pas grave. Au besoin, autorisez le remplacement des fichiers.

Retournez sur votre page d'installation et actualisez la.

Vous avez maintenant le choix entre:

English (built-in)
French (Français)

Configurer les informations de la base de données

Options de base

* «Nom de la base de données»: le nom de la base de données que vous avez créé auparavant pour ce site. Attention : Il ne s'agit pas de «MySQL» ou «PostgreSQL», mais bien du nom de la base dans laquelle seront placées les tables de données.
* «Nom d'utilisateur» : l'identifiant de l'utilisateur à qui vous avez donné l'accès à cette base de données.
* «Mot de passe» : le mot de passe qui permet d'accéder à cette base de données.

Options avancées

* «Serveur de base de données» : par défaut «localhost» (le SGBD se trouve sur la même machine que le serveur HTTP), si votre base se situe sur un autre serveur, indiquez son adresse IP ou le nom fourni par votre administrateur.
*«Port»: Par défaut vide (équivalent à 3306 pour MySQL). Si votre administrateur vous a indiqué un port différent de 3306, indiquez le ici.
*«Préfixe de table»: Cette option (vide par défaut) est utile dans le cas où vous ne disposez que d'une seule base de données pour vos différents sites. Dans ce cas, indiquez par exemple le nom de votre site (sans espaces ni caractères spéciaux) ou tout autre appellation qui vous permettra de ne pas mélanger les tables de données des différentes applications utilisant cette base de données. Astuce: Si vous avez un doute, mettez un préfixe («drupal_» par exemple), cela ne perturbera en rien votre installation et vous serez sûr de ne pas abîmer vos autres sites.

Dupliquer le fichier default.settings.php et le renommer en settings.php

Déployer son site web Drupal en ligne

Prérequis pour installer Drupal sur un serveur distant

* Un serveur HTTP comme «Apache» avec le module «PHP».
* Un système de gestion de base de données (SGBD) comme «MySQL».

Créer une archive .tar.gz du dossier drupal
Télécharger Putty ou un autre client ssh pour windows.

Quelques commandes ssh importantes :

pour se connecter en ssh :
ssh login@nomdedomaine.com

pour décompresser un .tar.gz :
tar -xzvf cmsmadesimple-1.6-base.tar.gz

pour créer un .tar.gz
tar -zcvf nom_du_fichier_compresse.tar.gz fichier_a_compresser

Exporter la base de donnée du site grace au phpMyAdmin de Wamp

Copier puis décompresser l'archive du site sur le serveur http

Importer la base de donnée grace au phpMyAdmin du serveur http

Outils pour l'environnement de développement Drupal

Un bon éditeur de fichiers

Un éditeur de texte comme Bloc-notes peut suffire.
Le mieux reste d'utiliser un logiciel spécialisé qui colore votre code et qui numérote vos lignes. Il existe de nombreux logiciels gratuits.

Sous Windows
Notepad++ (gratuit)
E-texteditor (Textmate pour Windows)

Sous Mac
TextWrangler (gratuit)
Textmate

Drush

Drush n'est pas un module pour Drupal, c'est un outil à installer sur votre serveur qui va vous permettre d'administrer vos sites en ligne de commandes.
Parmi les possibilités qu'offre Drush vous allez pouvoir via une simple ligne de commande installer Drupal, activer ou désactiver des modules ou encore télécharger des thèmes.
(plus d'informations ici : http://drupalfr.org/node/9762)

Remise à niveau PHP

Les fondamentaux en PHP

Sites statiques et sites dynamiques

Les sites statiques sont réalisés uniquement à l'aide des langages (X)HTML et CSS.
Les sites dynamiques utilisent d'autres langages en plus de (X)HTML et CSS, tels que PHP et MySQL. Le contenu de ces sites web est dit «dynamique» parce qu'il peut changer sans l'intervention du webmaster. La plupart des sites web d'aujourd'hui sont des sites dynamiques.

internet est un réseau composé d'ordinateurs. Ceux-ci peuvent être classés en deux catégories :

- Les clients : ce sont les ordinateurs des internautes comme vous. Votre ordinateur fait donc partie de la catégorie des clients. Chaque client représente un visiteur d'un site web.
- Les serveurs : ce sont des ordinateurs puissants qui stockent et délivrent des sites web aux internautes, c'est-à-dire aux clients.

Lorsque le site est statique, le schéma est très simple. Cela se passe en deux temps :

- Le client demande au serveur à voir une page web.
- Le serveur lui répond en lui envoyant la page réclamée.

Lorsque le site est dynamique, il y a une étape intermédiaire : la page est générée.

- Le client demande au serveur à voir une page web.
- Le serveur prépare la page spécialement pour le client.
- Le serveur lui envoie la page qu'il vient de générer.

La page web est alors générée à chaque fois qu'un client la réclame.

XHTML et CSS

XHTML : c'est le langage à la base des sites web. Il ressemble beaucoup au HTML mais impose quelques règles un peu plus strictes.

CSS : c'est le langage de mise en forme des sites web. Tandis que le XHTML permet d'écrire le contenu de vos pages web et de les structurer, le langage CSS s'occupe de la mise en forme et de la mise en page. C'est en CSS que l'on choisit notamment la couleur, la taille des menus, etc.

PHP et MySQL

PHP : c'est un langage que seuls les serveurs comprennent et qui permet de rendre votre site dynamique. C'est PHP qui «génère» la page web sous forme de code XHMTL.

MySQL : c'est ce qu'on appelle un SGBD (Système de Gestion de Base de Données). Son rôle est d'enregistrer des données de manière organisée. Le langage qui permet de communiquer avec la base de données s'appelle le SQL.

Balise PHP

<?php /* Le code PHP se met ici */ ?>

La balise PHP peut s'insérer n'importe où dans une page XHTML, y compris dans l'en-tête de la page.

Afficher du texte

Les instructions echo et print

<?php echo «Ceci est du texte»; ?>

<?php print «Ceci est du texte»; ?>

Ces deux instructions affichent : «Ceci est du texte». La seule différence est que print retourne une valeur.

Inclure des balises XHTML

<?php echo «Ceci est du texte»; ?>

Le mot «texte» sera affiché en gras grâce à la présence des balises et

Les commentaires monolignes

```
<?php
echo «Hello World»; // Ceci est un commentaire

// Ceci est un commentaire aussi
echo «Hello World»;
?>
```

Les commentaires multilignes

```
<?php
/* Pour écrire un commentaire sur plusieurs lignes
il faut utiliser cette syntaxe */
echo «Hello World»;
?>
```

Les variables

Une variable est une information stockée en mémoire temporairement. En PHP, la variable existe tant que la page est en cours de génération. Dès que la page PHP est générée, toutes les variables sont supprimées de la mémoire. Ce n'est donc pas un fichier qui reste stocké sur le disque dur mais une information temporaire présente en mémoire vive.

Une variable est toujours constituée de deux éléments :

* Son nom : Par exemple age_du_visiteur.
* Sa valeur : Par exemple : 17.

Les différents types de variables
Les variables sont capables de stocker différents types d'informations. On parle de types de données. Voici les principaux types à connaître :

* **Les chaînes de caractères** (string) : les chaînes de caractères sont le nom informatique qu'on donne au texte. En PHP, ce type de données s'appelle : string.
Exemple : «Je suis un texte».
Une chaîne de caractères est habituellement écrite entre guillemets ou entre apostrophes (on parle de guillemets simples) : 'Je suis un texte'.
* **Les nombres entiers** (int) : ce sont les nombres du type 1, 2, 3, 4, etc. On compte aussi parmi eux les nombres relatifs : -1, -2, -3...
Exemple : 42
* **Les nombres décimaux** (float) : ce sont les nombres à virgule, comme 14,738. Attention, les nombres doivent être écrits avec un point au lieu de la virgule (c'est la notation anglaise).
Exemple : 14.738
* **Les booléens** (bool) : Permet de retenir si une information est vraie ou fausse. On écrit true pour vrai, et false pour faux.
Exemple : true
* **Rien** (NULL) : Pour dire qu'une variable ne contient rien. On indique donc qu'elle vaut NULL. Ce n'est pas vraiment un type de données, mais plutôt l'absence de type.

Affecter une valeur à une variable

```
<?php
$age_du_visiteur = 17;
?>
```

Avec ce code PHP, on vient de créer une variable :
* Son nom est age_du_visiteur
* Sa valeur est 17

On ne peut pas mettre d'espaces dans un nom de variable. A la place, utilisez un underscore _.
Évitez aussi les accents, les cédilles et tout autre symbole pour le nom.
Le symbole Dollar ($) précède toujours le nom d'une variable.

Modifier la valeur d'une variable

```
<?php
$age_du_visiteur = 17; // La variable est créée et vaut 17
$age_du_visiteur = 23; // La variable est modifiée et vaut 23
$age_du_visiteur = 55; // La variable est modifiée et vaut 55
?>
```

La variable $age_du_visiteur va être créée et prendre pour valeur, dans l'ordre : 17, 23, puis 55.

Afficher le contenu d'une variable

```
<?php
$age_du_visiteur = 17;
echo $age_du_visiteur;
?>
```

Cette instruction affiche «17».

La concaténation

- avec des guillemets doubles

```
<?php
$age_du_visiteur = 17;
echo «Le visiteur a $age_du_visiteur ans»;
?>
```

- avec des guillemets simples

```
<?php
$age_du_visiteur = 17;
echo 'Le visiteur a ' . $age_du_visiteur . ' ans';
?>
```

Ces 2 instructions affichent «Le visiteur a 17 ans».

Les opérations de base : addition, soustraction...

- avec une variable

```
<?php
$nombre = 2 + 4 - 1; // $nombre prend la valeur 5
$nombre = 3 * 6 / 2; // $nombre prend la valeur 9
$nombre = 10 % 5; // $nombre prend la valeur 0 car la division tombe juste
?>
```

- avec plusieurs variables

```
<?php
$nombre = 10;
$resultat = ($nombre + 5) * $nombre; // $resultat prend la valeur 150
?>
```

Les conditions

Symboles à connaitre

==	Est égal à	!=	Est différent de
>	Est supérieur à	<	Est inférieur à
>=	Est supérieur ou égal à	<=	Est inférieur ou égal à

If

```
<?php
$heure = 12;

if ($heure >= 17)
{
   echo «Bonsoir !»;
}
?>
```

If... Else

```
<?php
$heure = 12;

if ($heure >= 17)
{
   echo «Bonsoir !»;
}
else // SINON
{
   echo «Bonjour !»;
}
?>
```

If... Elseif...Else

```
<?php
$heure = 9;

if ($heure >= 17)
{
   echo «Bonsoir !»;
}
elseif ($heure >= 14) // SINON SI
{
  echo «Bon après-midi !»;
}
else // SINON
{
   echo «Bonjour !»;
}
?>
```

Les booléens

```
<?php
if ($autorisation_entrer == true)
{
   echo «Bienvenu»;
}
elseif ($autorisation_entrer == false)
{
   echo «Autorisation refusée, désolé !»;
}
?>
```

- ou en version condensée

```
<?php
if ($autorisation_entrer)
{
   echo «Bienvenu»;
}
else
{
   echo «Autorisation refusée, désolé !»;
}
?>
```

- ou encore (version condensée négative)

```
<?php
if (! $autorisation_entrer)
{
   echo «Autorisation refusée, désolé !»;
}
else
{
   echo «Bienvenu»;
}
?>
```

Autre syntaxe

```
if ($a == $b) :
print «it's equal!»;
endif;
```

Les structures ternaires
La condition testée est $age >= 18. Si c'est vrai, alors la valeur indiquée après le point d'interrogation (ici TRUE) sera affectée à la variable $majeur. Sinon, c'est FALSE qui sera affectée à $majeur.

```
<?php
$age = 24;

$majeur = ($age >= 18) ? true : false;
?>
```

Switch
On préfèrera utiliser switch dans le cas d'une série de conditions à analyser.

```
<?php
$note = 10;

switch ($note) // on indique sur quelle variable on travaille
{
case 20:  // dans le cas où $note vaut 20
echo «Excellent travail, c'est parfait !»;
break;

case 15;  // dans le cas où $note vaut 15
echo «Vous êtes bon»;
break;

case 10: // etc etc
echo «Vous avez la moyenne»;
break;

case 5:
echo «Vous êtes mauvais»;
break;

case 0:
echo «C'est une catastrophe !»;
break;

default:
echo «Désolé, je n'ai pas de message à afficher pour cette note»;

}
?>
```

Conditions multiples

Mot-clé	Signification	Symbole équivalent
AND	Et	&&
OR	Ou	||

AND

```
<?php
if ($heure >= 8 AND $jour != «dimanche»)
{
   echo «Debout !»;
}
elseif ($heure >= 8 AND $jour != «dimanche»)
{
   echo «Vous pouvez vous recoucher !»;
}
?>
```

OR

```
<?php
if ($jour == «dimanche» OR $jour == «lundi»)
{
   echo «Désolé, nous sommes fermés !»;
}
else
{
   echo «Bienvenu !»;
}
?>
```

Les boucles

Qu'est ce qu'une boucle ?
Une boucle est une structure qui fonctionne sur le même principe que les conditions (if... else). Concrètement, une boucle permet de répéter plusieurs fois des instructions.
Quel que soit le type de boucle (while ou for), il faut indiquer une condition. Tant que la condition est remplie, les instructions sont réexécutées. Dès que la condition n'est plus remplie, on sort de la boucle.

Une boucle while

```
<?php
while ($continuer_boucle == true)
{
   // instructions à exécuter dans la boucle
}
?>
```

Une boucle while avec incrémentation
Ici je dois copier 100 fois «Je n'utiliserai plus Joomla».

```
<?php
$nombre_de_lignes = 1;

while ($nombre_de_lignes <= 100)
{
   echo 'Je n\'utiliserai plus Joomla.<br />';
   $nombre_de_lignes++; // $nombre_de_lignes = $nombre_de_lignes + 1
}
?>
```

Autre exemple : Ici le texte affiché s'incrémente à chaque fois.

```
<?php
$nombre_de_lignes = 1;

while ($nombre_de_lignes <= 100)
{
echo 'Ceci est la ligne n°' . $nombre_de_lignes . '<br />';
$nombre_de_lignes++;
}
?>
```

Une boucle for
Ici les 3 éléments nécessaires à l'incrémentation (initialisation, condition, incrémentation) sont condensés en une ligne.

```
<?php
for ($nombre_de_lignes = 1; $nombre_de_lignes <= 100; $nombre_de_lignes++)
{
echo 'Ceci est la ligne n°' . $nombre_de_lignes . '<br />';
}
?>
```

Les fonctions

Qu'est ce qu'une fonction ?
Une fonction est une série d'instructions qui effectue des actions et qui retourne une valeur.
On donne en entrée à la fonction un paramètre sur lequel elle va faire des calculs et la fonction nous retourne en sortie le résultat.

Appeler une fonction

```
<?php
calculCube();
?>
```

Souvent, les fonctions acceptent un ou plusieurs paramètres.

```
<?php
fonctionImaginaire(17, 'Vert', true, 41.7);
?>
```

Cette fonction recevra 4 paramètres : 17, le texte «Vert», le booléen vrai et le nombre 41,7.

Récupérer la valeur de retour de la fonction
Il y a 2 types de fonctions :
* Celles qui ne retournent aucune valeur (ça ne les empêche pas d'effectuer des actions)
* Celles qui retournent une valeur

Si la fonction retourne une valeur, on la récupère dans une variable comme ceci :

```
<?php
$volume = calculCube(4);
?>
```

Un exemple de fonction native de PHP: date
Les fonctions de PHP sont répertoriées ici : http://fr.php.net/manual/fr/funcref.php
La fonction date renvoie l'heure et la date.

Paramètre	Description
H	Heure
i	Minute
d	Jour
m	Mois
Y	Année

Pour afficher la date complète et l'heure :

```
<?php
$jour = date('d'); // On enregistre les informations de date dans des variables
$mois = date('m');
$annee = date('Y');
$heure = date('H');
$minute = date('i');
// Maintenant on peut afficher ce qu'on a recueilli
echo 'Nous sommes le ' . $jour . '/' . $mois . '/' . $annee . 'et il est ' . $heure. ' h ' . $minute;
?>
```

Créer ses propres fonctions

```
<?php
$nom = 'Sandra';
echo 'Bonjour, ' . $nom . ' !<br />';


$nom = 'Patrick';
echo 'Bonjour, ' . $nom . ' !<br />';


$nom = 'Claude';
echo 'Bonjour, ' . $nom . ' !<br />';
?>
```

La création d'une fonction nous permet d'éviter une tache fastidieuse et répétitive.

```
<?php
function DireBonjour($nom)
{
   echo 'Bonjour ' . $nom . ' !<br />';
}

DireBonjour('Sandra');
DireBonjour('Patrick');
DireBonjour('Claude');

?>
```

Autre exemple

```
<?php
// Ci-dessous, la fonction qui calcule le volume du cône
function VolumeCone($rayon, $hauteur)
{
  $volume = $rayon * $rayon * 3.14 * $hauteur * (1/3); // calcul du volume
  return $volume; // indique la valeur à renvoyer, ici le volume
}

$volume = VolumeCone(3, 1);
echo 'Le volume d\'un cône de rayon 3 et de hauteur 1 est de ' . $volume;
?>
```

Attention, sans l'instruction return, la fonction ne renvoie rien !

Les tableaux

Qu'est ce qu'un tableau ?
Un tableau (array) est une variable.
il est possible d'enregistrer de nombreuses informations dans une seule variable grâce aux tableaux. On distingue deux types de tableaux :

* Les tableaux numérotés
* Les tableaux associatifs

Les tableaux numérotés
Ces tableaux sont très simples à imaginer. Regardez par exemple ce tableau, contenu de la variable $prenoms :

Clé	Valeur
0	François
1	Michel
2	Nicole
3	Véronique
4	Benoît
...	...

$prenoms est un array : c'est ce qu'on appelle une variable «tableau». Elle n'a pas qu'une valeur mais plusieurs valeurs (vous pouvez en mettre autant que vous voulez).
Dans un array, les valeurs sont rangées dans des «cases» différentes. Ici, nous travaillons sur un array numéroté, c'est-à-dire que chaque case est identifiée par un numéro. Ce numéro est appelé clé.

Attention ! Un array numéroté commence toujours à la case n°0 !

• créer un tableau numéroté
Pour créer un tableau numéroté en PHP, on utilise généralement la fonction array.

```
<?php
// Cet exemple vous montre comment créer l'array $prenoms :
$prenoms = array ('François', 'Michel', 'Nicole', 'Véronique', 'Benoît');
?>
```

Vous pouvez aussi créer manuellement le tableau case par case :

```
<?php
$prenoms[0] = 'François';
$prenoms[1] = 'Michel';
$prenoms[2] = 'Nicole';
?>
```

ou encore (ici PHP sélectionne automatiquement le numero de case)

```
<?php
$prenoms[] = 'François'; // Créera $prenoms[0]
$prenoms[] = 'Michel'; // Créera $prenoms[1]
$prenoms[] = 'Nicole'; // Créera $prenoms[2]
?>
```

• afficher un tableau numéroté
Pour afficher un élément, il faut donner sa position entre crochets après $prenoms.

```
<?php
echo $prenoms[1];  //Affiche «Michel»
?>
```

Les tableaux associatifs
Le principe est identique, sauf qu'au lieu de numéroter les cases, on va les étiqueter en leur donnant à chacune un nom différent.

Par exemple, supposons que l'on veuille, dans un seul array, enregistrer les coordonnées de quelqu'un (nom, prénom, adresse, ville etc...). Si l'array est numéroté, comment savoir que le n°0 est le nom, le n°1 le prénom, le n°2 l'adresse ?... C'est là que deviennent utiles les tableaux associatifs.

• construire un tableau associatif
Pour le créer, on utilisera également la fonction array, mais on mettra «l'étiquette» devant chaque information :

```
<?php
// On crée notre array $coordonnees
$coordonnees = array (
   'prenom' => 'François',
   'nom' => 'Dupont',
   'adresse' => '3 Rue du Paradis',
   'ville' => 'Marseille');
?>
```

ou encore (on crée ici le tableau case par case)

```
<?php
$coordonnees['prenom'] = 'François';
$coordonnees['nom'] = 'Dupont';
$coordonnees['adresse'] = '3 Rue du Paradis';
$coordonnees['ville'] = 'Marseille';
?>
```

On obtient un tableau de ce type :

CLÉ	VALEUR
prenom	François
nom	Dupont
adresse	3 Rue du Paradis
ville	Marseille

• afficher un tableau associatif
Il suffit d'indiquer le nom de l'élément à afficher entre crochets, ainsi qu'entre guillemets ou apostrophes puisque l'étiquette du tableau associatif est un texte.

```
<?php
echo $coordonnees['ville']; //Pour extraire la ville
?>
```

Parcourir un tableau
Il existe 3 moyens d'explorer un array :
* La boucle for
* La boucle foreach
* La fonction print_r (utilisée principalement pour le débuggage)

La boucle for

```
<?php
// On crée notre array $prenoms
$prenoms = array ('François', 'Michel', 'Nicole', 'Véronique', 'Benoît');

// Puis on fait une boucle pour tout afficher :
for ($numero = 0; $numero < 5; $numero++)
{
   echo $prenoms[$numero] . '<br />'; // affichera $prenoms[0], $prenoms[1] etc...
}
?>
```

La boucle foreach

```
<?php
$prenoms = array ('François', 'Michel', 'Nicole', 'Véronique', 'Benoît');

foreach($prenoms as $element)
{
   echo $element . '<br />'; // affichera $prenoms[0], $prenoms[1] etc...
}
?>
```

A chaque tour de boucle, la valeur de l'élément suivant est mise dans la variable $element.
L'avantage de foreach est qu'il permet aussi de parcourir les tableaux associatifs.

```
<?php
$coordonnees = array (
   'prenom' => 'François',
   'nom' => 'Dupont',
   'adresse' => '3 Rue du Paradis',
   'ville' => 'Marseille');

foreach($coordonnees as $element)
{
   echo $element . '<br />';
}
?>
```

Foreach va mettre tour à tour dans la variable $element le prénom, le nom, l'adresse et la ville contenus dans l'array $coordonnées.
On met donc entre parenthèses :
- D'abord le nom de l'array (ici $coordonnees)
- Ensuite le mot-clé as (qui signifie quelque chose comme «en tant que»)
- Enfin le nom d'une variable que vous choisissez qui va contenir tour à tour chacun des éléments de l'array (ici $element).

Afficher un array avec print_r

```
<?php
$coordonnees = array (
   'prenom' => 'François',
   'nom' => 'Dupont',
   'adresse' => '3 Rue du Paradis',
   'ville' => 'Marseille');

echo '<pre>';
print_r($coordonnees);
echo '</pre>';
?>
```

L'instruction print_r permet d'afficher raidement le contenu d'un array.
La balise HTML <pre> nous permet ici d'avoir un affichage plus correct.

Introduction à la programmation orientée objet

Il est possible de programmer en PHP de nombreuses façons différentes.
La programmation orientée objet, également appelée POO, est une technique de programmation qui va vous aider à mieux organiser votre code, à le préparer à de futures évolutions et à rendre certaines portions réutilisables pour gagner en temps et en clarté.

Qu'est ce qu'un objet ?

L'idée de la programmation orientée objet est de manipuler des éléments que l'on appelle des «objets» dans son code source. En résumé, un objet un mélange de plusieurs variables et fonctions.

La classe est un plan, une description de l'objet. Imaginez qu'il s'agit par exemple des plans de construction d'une maison.

L'objet est une instance de la classe, c'est-à-dire une application concrète du plan.
Un objet possède les comportements de la classe à laquelle il appartient.

Pour reprendre l'exemple précédent, l'objet est la maison. On peut créer plusieurs maisons basées sur un plan de construction. On peut donc créer plusieurs objets à partir d'une classe.

les objets ont des propriétés :
$robot->couleur
$robot->matière
$robot->finition

les objets ont des méthodes :
$robot->action()

Dans le cadre du thème c'est surtout les propriétés qui vont nous intéresser

Nous utiliserons notamment des instructions comme :
print $node->links{'blog_usernames_blog']['href'];

- ici le nœud est un objet
- links est une propriété de l'objet nœud
- links est un tableau que l'on parcourt en donnant une première clé, puis une deuxième

Démarrer avec Drupal

Installation et activation des modules Drupal

Les modules du core

Drupal est livré avec une série de modules préinstallés ou non. Vous trouverez ces modules dans le dossier modules du core. Certains de ces modules sont indispensables à la construction d'un site avec Drupal (node, block, menu, path, taxonomy, user...). D'autres pourront être activés ou désactivés selon vos besoins (blog, comment, forum, search...).

Les modules contribués par la communauté

Drupal dispose de plus de 6000 modules contribués par la communauté.
Ces modules contribués sont téléchargeables sur http://drupal.org/project/modules.
Certains de ces modules, même si ils ne sont pas intégrés au core de Drupal 6 sont devenus incontournables lors de la construction d'un site Drupal (Content Construction Kit, Views, Pathauto, Imageapi, ImageField et ImageCache, Administration Menu, Devel...).

Voici une liste non exhaustive de quelques-uns des modules les importants.

1. Content Construction Kit (CCK)
L'un des plus importants modules de Drupal. À tel point qu'il est désormais présent dans le cœur de Drupal 7 sous le nom de Field Api. Il nous permet de créer des types de contenus structurés et personnalisés. Avec ce module nous pouvons ajouter des champs dates, des liens, des images et des listes de sélection à nos types de contenu afin de les enrichir. Il existe des modules dédiés à CCK pour augmenter nos possibilités lors de la création de nos types de contenus.

2. Views
Si avec CCK nous pouvons créer des contenus à la structure complexe, Views nous permet de les montrer, les lister sous tous les angles possibles selon des critères de tri et des filtres complexes. Il génère dynamiquement les requêtes (queries) dont nous avons besoin, comme par exemple la liste des derniers commentaires. Mais Views produit aussi des pages et des blocks sur base des requêtes ainsi générées. Le tout sans une seule ligne de programmation grâce à sa puissante interface graphique.

3. Pathauto
Ce petit module nous permet de créer automatiquement des URL (URL friendly) pour que les moteurs de recherche puissent indexer votre contenu avec un indice (page rank) élevé. N'oublions pas que les moteurs de recherche donnent beaucoup d'importance aux mots qui se trouvent dans le nom de domaine mais aussi dans l'URL.

4. Ckeditor
Permet aux rédacteurs d'utiliser un éditeur de texte (WYSIWYG) assez complet pour créer ou modifier n'importe quel champ de type texte. Plus besoin de connaitre le HTML pour mettre le texte en italique ou créer une liste. Couplé à IMCE, ce module nous permet de télécharger vers le serveur des images ou des animations flash depuis l'éditeur de texte et des les insérer dans le contenu.

5. Administration menu
Un petit module qui nous permet d'atteindre une page de l'interface d'administration sans devoir faire 15 clics pour y arriver grâce à un menu qui reste en permanence en haut de nos pages. Simple mais indispensable.

6. Devel
La boite à outils du développeur. Ce module nous permet de maintenir nos caches, simuler des utilisateurs, déboguer notre code (dsm() et autres), voir les variables ou les hooks utilisés.

Du côté de la performance, il nous permet d'enregistrer et de voir des statistiques comme le temps ou la mémoire nécessaire pour générer une page. Par ailleurs ce module est très utile pour connaitre en détails toutes les requêtes SQL nécessaires pour générer une page, identifier les requêtes les plus lentes pour pouvoir les optimiser par la suite.

Il se couple d'un sous-module, Theme developer, qui est l'outil indispensable au themer ou designer, pour mettre en place le layout, choisir le nom de ses fichiers .tpl et identifier les variables qui sont passées à ces derniers.

7. Imagefield
Ce module nous permet d'ajouter un champ à nos types de contenus pour pouvoir télécharger des images vers le serveur. Son ami de toujours, ImageCache, nous permet lui de redimensionner automatiquement les images ainsi envoyées grâce à des dimensions (presets) que nous pouvons configurer. Ce module ainsi que ImacheCache sont désormais présent dans le core de Drupal 7.

Installation et activation des modules contribués

Télécharger le module et en extraire les fichiers

Copier le dossier du module dans sites/all/modules
Vous devrez créer ce dossier modules au préalable. Tous les modules contribués devront être installés dans ce dossier (et non dans le dossier module présent à la racine de votre dossier Drupal). Ceci à pour avantage de permettre la mise à jour du core de Drupal sans écraser les modules contribués que vous aurez installés.

Activer le module
Allez dans Administer > Site building > Modules. Cochez le bouton 'enabled' du module concerné puis cliquez le bouton 'Save Configuration'.

Gérer les permissions
Certains modules nécessitent de changer les permissions pour les faire fonctionner.
Pour cela, allez dans Administer > User management > Permissions, vérifiez si le module concerné figure dans la liste et donnez aux différents rôles les autorisations souhaitées.

Configurer le module
La plupart des modules ont leur propre page de réglages, généralement accessibles depuis le Menu d'Administration

Attention: Vous ne pouvez avoir qu'une seule copie de chaque module du même nom dans un site Drupal. Le nom du module est déterminé par le nom du fichier .module, et non par le nom du dossier.

Installation et activation d'un thème Drupal

Les thèmes du core

5 thèmes sont livrés à l'intérieur du core de Drupal. Vous les trouverez dans le dossier themes présent à la racine de votre dossier Drupal.
Garland est le thème par défault. Il n'est pas recommandé d'utiliser Garland comme point de départ pour construire votre propre thème. Il reste cependant un trés bon thème pour l'administration de votre site Drupal.

Les thèmes contribués par la communauté

Les thèmes contribués sont téléchargeables à cette adresse : http://drupal.org/project/themes.
Certains d'entre-eux peuvent être utilisé tels quels. D'autres sont typiquement des thèmes de bases que vous pourrez utiliser comme point de départ ou comme thème parent de votre propre thème (on parle alors de sous-thème).
Parmi ces thèmes de base, on peut citer Zen, Fusion, Basic, etc...

Installation et activation d'un thème contribué

Télécharger le thème et en extraire les fichiers

Copier le dossier du thème dans sites/all/themes
Vous devrez créer ce dossier themes au préalable. Tous les themes contribués devront être installés dans ce dossier (et non dans le dossier themes présent à la racine de votre dossier Drupal). Ceci à pour avantage de permettre la mise à jour du core de Drupal sans écraser les thèmes contribués que vous aurez installés.

Activer le thème
Allez dans Administer > site building > themes. Cochez le bouton 'enabled' du thème concerné puis cliquez le bouton 'Save Configuration'. Ici vous pouvez également définir le thème par défaut de votre site.
Si vous souhaitez définir un thème différent pour l'administration, c'est possible en allant dans Administer > site configuration > administration theme.

Construire un site avec Drupal

Créer et structurer le contenu

Nœuds, types de contenu et champs

Chaque élément de contenu de votre site est un node. Un node a plusieurs champs par défault. Un node est défini par un type de contenu.
Vous pouvez créer vos propre champs (custom fields) et définir vos propres types de contenu, notamment avec l'aide du module contribué CCK (Content Construction Kit). Ce module sera intégré au core de Drupal 7 sous le nom Field UI.

CCK
Par défaut, Drupal 6 vient avec des types de contenu comme page et story.
Par ailleurs certains modules du core ou modules contribués créent automatiquement leurs propres types de contenu quand vous les activez (Blog, Forum, etc).

CCK vous permet de créer vos propres types de contenus et d'y mettre exactement les champs dont vous avez besoin pour un type de contenu donné.

De nombreux modules complémentaires sont disponibles pour enrichir CCK en fonction de vos besoins spécifiques (video field, audio field, etc...).

Enfin CCK ajoute dans l'interface de gestion des types de contenu de nouvelles tabulations comme Manage fields (l'endroit où vous créez de nouveaux champs) et Display fields (l'endroit où vous décidez comment ils vont être affichés).

La taxonomie

Drupal permet de classer le contenu dans des catégories. Cela permet par exemple, de regrouper des contenus de différents types sous une même dénomination. Il est donc possible de mettre une image, une page, un article ou un billet de blog dans la même catégorie. La gestion des catégories sous drupal est gérée par le module taxonomy.

Sous drupal, les catégories sont définies par deux types d'éléments : les vocabulaires et les termes. Un vocabulaire est un ensemble de termes liés entre eux.

Selon les cas, un vocabulaire peut être constitué de termes créés par les utilisateurs (folksonomy) ou de termes créés par les administrateurs du site.

Un exemple de taxonomie hiérarchique qui classe la musique en termes et sous termes.

Vocabulary = Music
term = classical
  sub-term = concertos
  sub-term = sonatas
  sub-term = symphonies
term = jazz
  sub-term = swing
  sub-term = fusion
term = rock
  sub-term = soft rock
  sub-term = hard rock

Créer des catégories

Comme nous l'avons vu, créer des catégories sous drupal revient à créer un vocabulaire qui contiendra des termes. Dans Administer > taxonomy, nous avons la liste des termes et des vocabulaires qui sont créés sur notre site. Bien entendu, pour l'instant il n'y a rien.

Pour créer un vocabulaire, il suffit de cliquer sur l'onglet ajouter un vocabulaire, puis de donner un nom à notre vocabulaire et de sélectionner le type de contenu qui pourra être classé avec ce vocabulaire.

Une fois validé, le vocabulaire sera listé sur la page suivante. Il suffit alors d'ajouter des termes à ce vocabulaire en cliquant sur le lien ajouter des termes et de donner un nom à notre terme.

Nous voilà avec des catégories que nous pouvons désormais utiliser.

Classer son contenu

Lors de la création d'un nouveau contenu, un nouveau bloc d'option nommé catégories apparaît et nous propose de choisir un des termes parmis ceux que nous avons créés.

Bien entendu, il est possible d'assigner plusieurs termes d'un même vocabulaire ou des termes de plusieurs vocabulaires à un même contenu.

Utiliser la taxonomie dans les menus

La taxonomie peut être utilisée pour afficher les éléments qui correspondent à un terme spécifique. Dans ce cas, les termes correspondent à des catégories.

Quand vous créez un terme, Drupal lui assigne un nombre, ce nombre est visible quand on survole le nom du terme dans la liste.

Vous pouvez maintenant aller dans Administer > site building > menus et créer un nouvel élément auquel vous attribuerez par exemple le path taxonomy/term/1, où 1 est le numero de notre terme.

Si vous utilisez une taxonomie hierarchique, vous pouvez également utiliser un path du type taxonomy/term/2/2, le second paramètre est alors la profondeur avec laquelle les enfants du terme seront explorés (taxonomy/term/2/all pour tous les enfants du terme).

Les menus

L'une des premières choses que l'on souhaite faire pour organiser son contenu est de pouvoir créer des menus et ainsi proposer une navigation à l'usager.

Avant de continuer, il faut vérifier que le module menu est bien activé. Il l'est par défaut.

La mise en place des menus comporte trois étapes :

- créer un menu,
- associer une entrée de menu à un contenu,
- afficher le menu.

Créer un menu

Pour créer un menu, nous allons nous rendre dans administrer > menus. La page de gestion des menus s'affiche.

Nous remarquons qu'il existe déjà trois menus sur notre site. Ils ont été créés par le système lors de l'installation. Le premier, Navigation, correspond au menu de l'utilisateur. Le deuxième, Primary links (main menu dans D7), correspond aux liens primaires. Les liens primaires sont des menus «spéciaux» dans le sens où l'on peut les afficher en les appelant directement depuis un thème. Il y a également un menu Secondary Links (secondary menu dans D7) qui peut amener aux sous-pages tributaires du menu Primary.

Pour créer un nouveau menu, il suffit tout simplement de cliquer sur l'onglet ajouter un menu. Le page d'ajout d'un menu s'affiche, et il ne nous reste plus qu'à donner un nom à notre menu en remplissant le champ Titre. Une fois le formulaire validé, nous pouvons alors voir apparaître notre menu dans la liste des menus.

Associer un contenu à un menu

Il existe deux manières d'associer un contenu à un menu. La première consiste à le faire au niveau du contenu. Si par exemple je veux une entrée de menu pour une page bien précise, il me suffit de l'éditer et de me rendre dans le bloc d'option Paramètres du menu. Là, il me faut remplir le champ Titre qui donnera son nom au lien dans le menu, le champ Description qui s'affichera dans la bulle d'information au survol de la souris au dessus du lien. La liste déroulante Elément parent permet de définir à quel endroit du menu se trouvera mon entrée. C'est en sélectionnant le parent que l'on définit les menus et sous-menus. Le Poids sert à définir l'ordre d'affichage des liens du menu lorsque plusieurs entrées ont le même parent.

La deuxième méthode consiste à se rendre dans la partie administration > menu de drupal. Il suffit ensuite de cliquer sur l'onglet Ajouter un élément de menu. Nous devons renseigner les mêmes informations que précédemment, ainsi que le chemin qui permet d'accéder au contenu auquel l'entrée sera associée.

Afficher le menu

La dernière étape consiste à afficher le menu. Dès que l'on crée un menu comme à l'étape 1, Drupal va automatiquement créer un bloc qui contiendra notre menu. Pour afficher notre menu, il nous suffit donc simplement d'activer le bloc et de le positionner là où l'on veut. Rendez-vous dans administrer/blocs, repérez la ligne correspondant à votre menu (elle porte le même nom que votre menu), cochez la case dans la colonne Activé puis validez le formulaire.

Travailler avec les blocs et les régions

Qu'est ce qu'un bloc ?
Les blocs sont des boites de contenu (comme «User Login» or «Who's online») qui peuvent être affichés dans les régions (comme le footer ou la sidebar) de votre page.

Certains blocs deviennent disponibles sur votre site en activant des menus. Une fois le bloc créé, on peut modifier son apparence, sa position ainsi que les pages sur lesquelles il apparait.

Les régions
C'est le thème de votre site qui définit les régions disponibles. Par défaut, 5 régions sont disponibles dans le core de Drupal. Vous pouvez en supprimer ou en ajouter autant que vous le souhaitez (voir Introduction à la création d'un thème).

Les blocs sont placés dans les régions via la page d'administration des blocs (Administer > Site building > Blocks), Ce placement est toujours relatif à un thème. Les blocs peuvent donc être placés différemment pour chaque thème.

L'ordre des blocs à l'intérieur d'une région peut être défini en ajustant leur poids (weight) ou en déplaçant les blocs (drag and drop).

Configuration et création des blocs
Chaque bloc posséde une page de configuration qui permet de paramétrer son affichage.
Vous pourrez notamment :
- montrer/cacher le bloc sur certaines pages
- permettre à l'utilisateur de montrer/cacher le bloc
- définir le fonctionnement du bloc (par exemple décider de masquer le bloc si celui-ci est vide)

L'inferface de gestion des blocs permet également d'ajouter des blocs et d'en définir le contenu. Chacun de ces blocs consiste en un titre, une description et un corps (body).

Travailler avec Views

Le module Views permet aux administrateurs et aux créateurs de sites de créer, gérer et afficher des listes de contenu. Chaque liste gérée par le module «Views» est appellée «vue» (view) et son résultat est appellé un affichage (display). Les affichages peuvent être de type bloc ou page, et chaque vue peut avoir plusieurs affichages. Des aides à la navigation, comme le chemin système et un élément de menu peuvent être définis par les affichages de type page. Par défaut, on peut créer des vues pour lister des noeuds (vue de type Node), des révisions de contenu (vue de type Révision du nœud) ou des utilisateurs (vue de type Utilisateur). On peut restreindre l'accès à une vue à des rôles spécifiques.
L'affichage d'une vue peut prendre la forme d'une liste (list view), d'un tableau (table view) ou encore d'une liste de teasers (teaser view).

On ajoute, supprime ou modifie une vue via la page d'administration de Views.

Toutes les vues se basent sur un cadre conceptuel (conceptual framework) qui inclut:

Fields
Les Champs sont les plus petits morceaux de contenu qui puissent être affichés. En ajoutant par exemple les champs Node: Titre, Node: Type et Node : Date de publication à une vue de type node, on obtiendra une liste incluant, pour chaque noeud, son titre, son contenu et sa date de publication.

Relationships
Les Relations indiquent comment les données sont liées les unes aux autres. Si une «relation» existe telle que les informations fournies par un champ CCK node référence, le contenu d'un noeud lié peut être inclus dans la vue.

Arguments
Les Arguments sont des paramètres additionnels qui permettent de restreindre les résultats dynamiquement, via l'url. Par exemple en ajoutant un argument de type Node: Type à une vue de type node et en définissant son chemin comme «contenu», on filtre dynamiquement le contenu par type de contenu. Par exemple en accédant à la vue à l'adresse «http://www.example.com/contenu/story» on liste tous les contenu de type article alors que «http://www.example.com/contenu» liste tous les contenus, quels que soient leur type.

Sort criterias
Les critères de tri déterminent l'ordre dans lequel les données seront affichées. Par exemple, en ajoutant le critère de tri Node: date de publication (en ordre descendant), on trie la liste par date de publication, du plus récent au plus ancien.

Filters
Les Filtres permettent de limiter les résultats affichés. Par exemple en ajoutant le filtre Node : Publié = oui empêche les éléments non publiés d'apparaître dans les résultats.

Displays
Les Affichages contrôlent comment le résultat sera affiché. Chaque vue a un affichage par défaut qui, dans l'absolu ne génère aucun affichage mais qui est utilisé pour stocker les paramètres par défaut et pour afficher des vues par la programmation lorsqu'aucun autre affichage n'est définit. Les affichages de type page qui donnent à la vue une url qui et permette à la vue d'être le contenu principal d'une page, et ceux de type bloc qui permet d'afficher le résultat dans un bloc de contenu secondaire sont beaucoup plus utiles pour l'utilisateur.

Tutoriel Views

Le module Views offre une quantité incroyable de fonctionnalités, en particulier lorsqu'il est associé à une utilisation intelligente des champs Référence de noeuds. Lorsque vous mettez les nodes de votre site en relation avec des champs Référence de noeuds, cette relation peut facilement mise à profit pour créer des views très utiles.

A titre d'exemple, je vais réaliser une vue pour un site de musique. Dans ce site, il y aura trois types de contenus liés les uns aux autres : nodes Groupes (« Beatles », « AC/DC », etc), nodes Album (« Back in Black », « Revolver », etc), et Événements (concerts, télévision, passages, etc).

Ces différents types de contenus ne disposent chacun que des champs minimaux pour la démonstration des fonctionnalités de Views dont il sera question dans cet article. Par exemple, le type de contenu « Événements » ne dispose pas de champs pour les dates ou les lieux - ils pourront facilement être ajoutés par la suite.
Pour associer les trois types de contenus, les deux types de contenus « Album » et « Événement » disposent d'un champ Référence de noeud qui pointent vers le node « Groupe ». Par exemple, quand un nouveau node « Album » est créé, l'utilisateur est obligé de choisir le groupe adéquat.

Pour l'utilisation des ces types de contenus, je crée plusieurs nodes de chaque type et j'ai donc quelques données pour travailler avec lors de la création de la vue.

Le but de cet exercice est de pouvoir créer une page qui affichera un seul node d'un groupe avec la liste de ses albums et événements. Voici un court exemple de ce que je souhaite obtenir :

Infos sur le Groupe

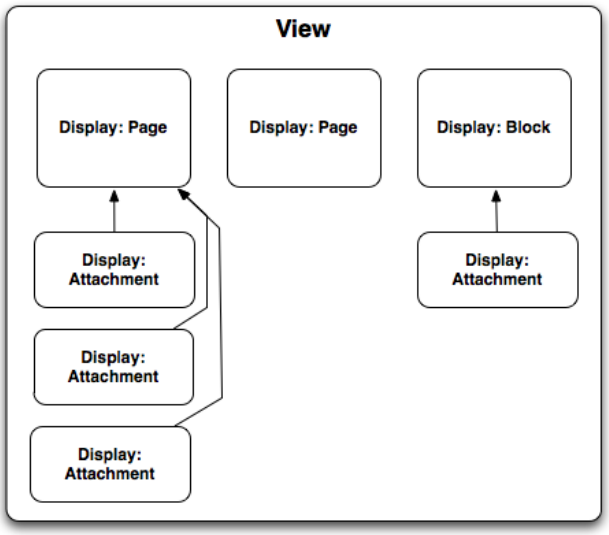
[Nom du groupe]

Albums
[Album 1]
[Album 2]
[Album 3]

Événements
[Événement 1]
[Événement 2]
[Événement 3]

Il y a plusieurs façons d'obtenir cela - on pense à Panels (http://www.drupal.org/project/panels) (context) et aux blocs (visibilité) - mais je vais employer une méthode qui n'utilise que les fichiers attachés de Views. Car quelle que soit la méthode choisie, on est presque certains au final de devoir créer des views, alors pourquoi utiliser du code ou des modules supplémentaires ?

Un fichier attaché Views est simplement un mode d'affichage de Views qui est rattaché à un autre affichage Views (habituellement un affichage de page ou de bloc). En simplifiant les choses, cela pourrait ressembler à ceci :



Ce schéma montre qu'un affichage de type Page ou Bloc (ainsi que d'autres types d'affichages apportés par des modules tiers) peuvent avoir n'importe quel nombre d'affichages associés. Sur une page, ces affichages associés peuvent se situer physiquement avant ou après l'affichage auquel ils sont rattachés.
Pour cet exemple, je vais créer une view qui a un affichage de type Page (appelé « Band Page ») et deux Affichages Associés - fichiers attachés - (appelés « Album Attachment » et « Events attachments »). Les deux Affichages Associés seront attachés à la fin de la page « Band » et nous aboutirons ainsi au résultat souhaité indiqué précédemment.

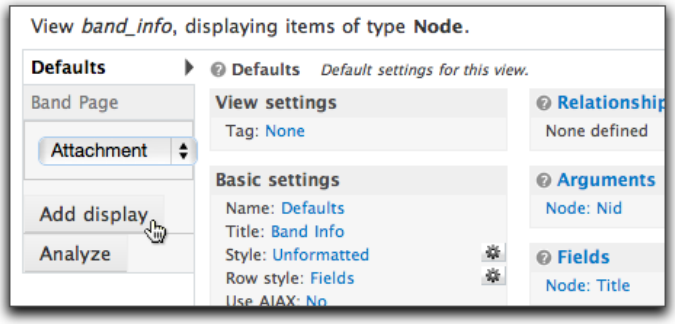
Pour la première étape de création de la view, je suis allé sur la page admin/build/views/add et j'ai créé une view de type Node appelée « band_info » contenant les valeurs suivantes pour l'affichage par défaut :

Paramètres de base
Titre : Band Info
Arguments
Node : Nid (Action à mener si l'argument est absent : Cacher la vue / Page non trouvée (404))
Champs
Node : Titre
Filtres
Node : Publié Oui
Node : Type = Band

Ensuite j'ai crée un nouvel affichage de type Page, je l'ai nommé « Band page » et j'ai paramétré son chemin à band/%. Rappelez-vous, le % est juste un paramètre substituable pour l'argument Node : Nid paramétré pour l'affichage par défaut.

À ce stade vous pouvez enregistrer la view et la vérifier dans la Prévisualisation en direct en indiquant dans le
champ Arguments l'ID de node d'un node « Band ». Cette view ne devrait afficher que le nom d'un groupe dont l'id de node est celui indiqué comme argument.

Maintenant, les choses intéressantes. Je vais créer un fichier attaché qui affiche tous les nodes « Albums » du groupe dont l'ID de node est indiqué comme argument. Rappelez-vous, il n'y a rien de particulier en ce qui concerne les fichiers attachés (Attachments displays). Ce sont juste des views comme les autres, sauf que vous pouvez les associer à d'autres affichages. Pour créer l' « Attachment Albums », je commence par sélectionner Fichier attaché dans la liste déroulante au-dessus du bouton Nouvel affichage puis je clique sur ce bouton Nouvel Affichage.

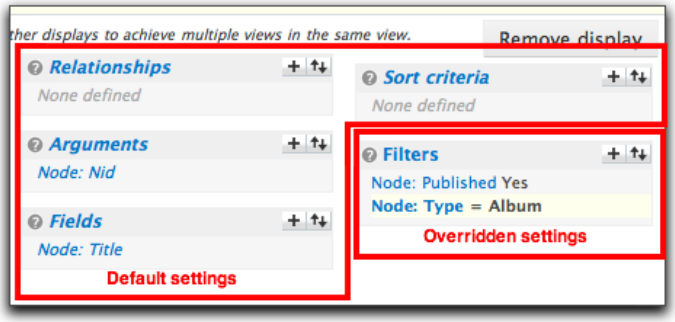


Le paramétrage initial pour le nouvel affichage est simple :

Paramètres de base
Nom : Album Attachment
Paramètres de fichiers attachés
Position: Après
Attacher à: Band Page

Puisque nous voulons afficher une liste de nodes « Albums », je veux paramétrer Node : type sur Album. Pour cela, je dois faire attention à ne pas modifier les paramètres par défaut de toute la view, je veux juste modifier les paramètres pour cet affichage-là. Pour cela, dans le bloc Filtres, je clique sur Node : Type puis je clique sur le bouton Supplanter afin de ne modifier que l'affichage courant et non toute la view.

Une fois que ce paramètre est supplanté, je peux le modifier à « Album » sans risque de perturber les autres affichages. Il est facile de voir si un paramètre est supplanté : il n'est plus affiché en italiques dans le récapitulatif d'affichage :



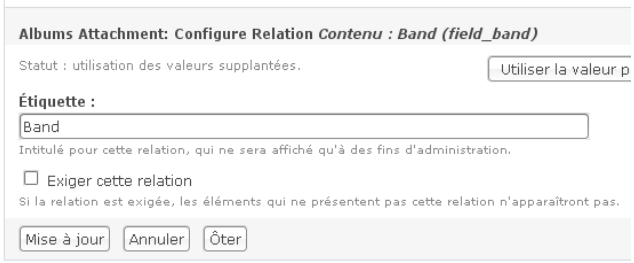
On aborde à présent le point délicat : puisque notre affichage ne montrera désormais que les nodes « Albums », comment allez-vous le filtrer avec l'ID des nodes « Groupes » ? Avec une relation, bien sûr ! Souvenez-vous, les types de contenus « Groupe » et « Album » sont reliés via un champ CCK Référence de noeud. Nous pouvons utiliser cette relation pour filtrer les nodes Albums. La première étape consistera à ajouter une nouvelle relation à l'affichage Album Attachment. Dans ce cas, je vais ajouter une relation Contenu : Band.

Mais avant cela, je dois m'assurer que j'ajoute cette relation uniquement à l'affichage Albums attachments et pas à la vue entière.

C'est une source d'erreurs fréquente. Observez la zone Relations, vous remarquez qu'elle est en italique ? Cela veut dire que quel que soit ce que je lui ajoute, ce sera ajouté à l'affichage par défaut - et à tous les affichages qui héritent de l'affichage par défaut. Je dois d'abord cliquer sur le lien Relations puis cliquer sur le bouton Supplanter.

Une fois que c'est fait, je sais que je suis en train d'ajouter la relation au seul affichage Albums Attachments. Je vais ajouter la relation qui lie un contenu « Album » à un contenu « Groupe » - dans cet exemple, il s'agit du champ Référence de noeud Contenu : Band.

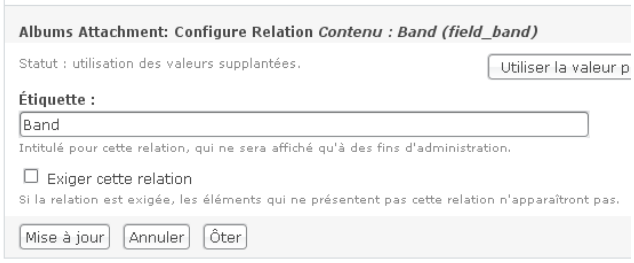
Après l'ajout de cette relation, j'ai deux paramètres à configurer. L'étiquette n'est utilisée que pour identifier la relation telle qu'elle apparaît dans cette view spécifique. Lorsqu'on ajoute plusieurs relations à une view, il est utile de leur donner des noms explicites. Heureusement, Views propose des noms par défaut pertinents, je m'en tiendrai à eux.



Dans le cas présent, le paramètre Exiger cette relation n'est pas nécessaire parce que je sais que l'argument filtrera tout sauf les contenus Albums que je veux voir.
La dernière pièce du puzzle concerne l'argument Node : Nid. Puisque l'affichage Albums Attachment hérite de l'argument de l'affichage Band Page, je dois m'assurer qu'il est utilisé correctement. Dans son état actuel, l'affichage essaiera de filtrer la liste des Albums avec l'argument Node ID fourni. Ce ne sera pas beaucoup de travail puisque l'argument fourni est le Node ID d'un Groupe. Je dois dire à la view d'utiliser cet argument pour filtrer les nodes Albums renvoyés par le Groupe - en utilisant la relation que je viens de paramétrer. Ce qui est plutôt facile à obtenir.

Tout d'abord, je dois cliquer sur l'argument Node : Nid pour le modifier, puis cliquer sur le bouton Supplanter pour que les modifications n'affectent que cet affichage. Ensuite, dans la liste déroulante Relation je dois choisir Band.

Cela filtrera effectivement les nodes Albums renvoyés par le Node ID associé du Groupe.



La dernière chose que j'ai faite à Albums attachment pour le rendre un peu plus user-friendly a été de surcharger le paramètre Entête (bloc Paramètres de base, lien Entête) pour lui ajouter ceci : Albums
Cela affichera un bel entête au-dessus des Albums.

Jetons un rapide coup d'oeil à la Prévisualisation en direct avec un node ID de Groupe correct comme argument, vous obtiendrez quelque chose comme ceci :

À ce stade, l'affichage Album Attachment est pratiquement terminé. Vous pouvez ajouter quelques champs
supplémentaires, des tris et des options d'affichage, mais notre but est atteint.
L'affichage Events Attachment fonctionne exactement de la même façon. La différence principale est que le filtre Node: Type utilise le type de contenu Events, mais tout le reste est semblable à l'affichage Albums Attachment.

En rassemblant le tout, la view finale ressemblera à ceci :

Il n'est pas difficile d'imaginer comment vous pouvez désormais utiliser ces outils pour créer toutes sortes de views intéressantes des mêmes données. Peut-être qu'une page Album ou Événement qui affichera aussi des infos sur le node Groupe associé. Peut-être ajouter une nouveau type de contenu pour les dates d'évènements, vous pouvez alors construire une view qui affiche tous les événements par lieux de tournées avec des infos sur le Groupe pour chaque date. La méthode indiquée ici est assez souple pour vous permettre d'aller aussi loin que vous le voulez avec vos données.

Installation de Drupal en multisite

Drupal a été conçu dés le départ pour qu'il puisse simplement être utilisé en multisite.

* une seule fois le code PHP de Drupal sur le serveur
* en cas de mises à jour : une seule mise à jour upgrade Drupal pour tous les sites.
* thèmes et modules supplémentaires téléchargés ou modifiés par vous, disponibles pour tous les sites
* malgré l'installation multisite, il reste possible de n'autoriser un thème ou module qu'à un seul site

En terme de performance il n'y a aucune raison qui puisse impliquer de meilleures résultats en multi-site qu'en multi-installation. Chaque site disposant de son propre espace de cache et de sa propre base de donnée, le gain en ce domaine est donc nul.

Les inconvénients
D'un point de vue pratique, sauf si les sites sont de véritables clones, la mutualisation peut s'avérer plus néfaste qu'autre chose :

* quand on fait évoluer un module, nécessité de systématiquement tester l'impact sur tous les multisites, et ce même si on ne le change que pour un site.
* gérer le paramétrage statique (settings.php) quand on a plusieurs noms de domaines pour un multisites donné. Ce point a été largement amélioré en Drupal 7 avec les alias de sites, mais c'est un vrai problème dans Drupal 6.

Mise en place
Pour créer un site utilisant un code partagé, suivre les étapes suivantes :

1. Créer une nouvelle base de donnée pour le site
2. Créer un sous-dossier du dossier site avec comme nom le nom du nouveau site
3. Copier le fichier sites/default/default.settings.php dans le sous-dossier précédemment créé. Le renommer en settings.php
4. Régler les permissions du dossier du nouveau site, authoriser le fichier de configuration en écriture
5. Créer un lien symbolique du type exemple.com/subdir (et non un sous domaine du type subd.exemple.com)
6. Taper l'URL du nouveau site dans le navigateur et suivre la procédure standard d'installation

Il peut être nécessaire de modifier le fichier de configuration de votre webserver (httpd.conf pour Apache) pour authoriser Drupal à overrider les réglages d'Apache. C'est valable pour toutes les installations de Drupal, ça n'est pas spécifique au multisite.

Pour plus d'informations concernant la configuration du webserver :
http://drupal.org/node/36628

Modules et thèmes spécifiques
Chaque site peut avoir ses propres modules et thèmes en plus de ceux installés dans le dossier sites/all. Il suffit de créer des dossiers modules et thèmes à l'intérieur du dossier du site.

Par exemple le site sub.exemple.com a un custom thème et un custom module qui ne doivent pas être accessibles aux autres sites :
sites/sub.exemple.com/settings.php
sites/sub.exemple.com/themes/custom_theme
sites/sub.exemple.com/modules/custom_module

Créer son propre thème

Vue d'ensemble des fichiers de thèmes

Un thème est un ensemble de fichiers qui définit la mise en page du site. Vous pouvez également créer un ou plusieurs « sous-thèmes », ou variations à partir d'un thème. Le seul fichier obligatoire est le fichier .info, mais beaucoup de thèmes et sous-thèmes utiliseront d'autres fichiers.

.info (obligatoire)
Tout ce qui nécessaire à Drupal pour voir votre thème doit être référencé dans le fichier .info.

Si le thème en a besoin, les feuilles de styles, le code Javascript, les régions, etc, doivent être spécifiés dans ce fichier. Tout le reste est facultatif.

Le nom «interne» du thème découle également de ce fichier. Par exemple, s'il est nommé «drop.info» alors Drupal verra «Drop» comme nom de thème.

Fichiers gabarits (template files) (.tpl.php)
Ces gabarits contiennent des balises XHTML et des variables PHP.
Chaque fichier .tpl.php gère l'affichage des données. Dans certains cas, plusieurs fichiers .tpl.php peuvent être gérés par le biais de suggestions.
Ces fichiers sont facultatifs, si votre thème n'en possède pas c'est l'affichage par défaut qui sera utilisé. Évitez toute structure complexe ou programmatique dans ces fichiers, ils ne devraient contenir que du balisage xHTM et des variables PHP.

Vous trouverez des exemples de ces fichiers .tpl.php dans les dossiers des modules installés.

Les copier dans votre dossier de thèmes forcera Drupal à utiliser votre version.

Remarque : le registre des thèmes place les informations sur les thèmes disponibles en cache. Vous devez vider ce cache chaque fois que vous modifiez votre fichiers de thèmes.

template.php
Le fichier template.php reçoit toute la logique conditionnelle et le traitement des données affichées. Il n'est pas indispensable mais pour une bonne organisation des fichiers.tpl.php il peut être utilisé pour les fonctions de pré-traitement , la génération des variables avant qu'elles ne soient fusionnées dans les fichiers tpl.php.

Les fonctions personnalisées, la surcharge des fonctions de thèmes, ou toute autre personnalisation de l'affichage doit être faite dans ce fichier.

Il débute par la balise d'ouverture PHP <?php, la balise de fermeture n'est pas nécessaire et il est même recommandé que vous l'omettiez.

Sous-thèmes
A première vue, les sous-thèmes se comportent comme n'importe quel autre thème.
La seule différence est qu'ils héritent des ressources de leurs thèmes-parent.

Pour créer un sous-thème, un paramètre «base theme» doit être spécifié dans le fichier .info.
A ce moment, ils héritent des ressources de leur thème-parent. Il peut y avoir plusieurs niveaux d'héritage. Un sous-thème peut, par exemple, déclarer un autre sous-thème comme base de départ.

Création du dossier du thème et du fichier .info

La première étape dans la création d'un thème est de créer le dossier où il sera enregistré, et créer le fichier .info pour informer Drupal de son existence.

Pour créer le dossier

- Créez ce dossier dans le dossier /sites/all/themes
- Nommez-le votretheme, tout en minuscules

Pour créer le fichier .info

- Creéz un document dans un éditeur de texte et collez-y les lignes suivantes (en les modifiant là où ce sera nécessaire) :

```
name = votretheme
description = Description de votre thème
core = 6.x
engine = phptemplate
stylesheets[all][] = style.css
stylesheets[all][] = layout.css
regions[left] = Left sidebar
regions[right] = Right sidebar
regions[content] = Content
regions[header] = Header
regions[footer] = Footer
```

- Sauvegardez ce fichier dans le dossier créé précédemment et nommez-le votretheme.info.

Les feuilles de style
Vous pouvez ajouter plusieurs feuilles de styles CSS à votre thème,
Le «all» entre crochets indique que la feuille sera utilisée quelque soit le media (screen, print...).

Les régions
Les régions disponibles pour un thème doivent être spécifiées avec la clé regions suivie du nom machine de la région entre crochets et du nom compréhensible comme valeur. Par exemple : regions[theRegion] = Le nom de ma région . Le nom comprehensible sera utilisé pour l'intitulé de la région dans la page d'administration des blocs dans Administer > Site Building > Blocks.

Si aucune region n'est définie, ce sont les valeurs par défaut qui sont utilisées.

Ajouter une région personnalisée désactive les régions par défaut. Si vous voulez conserver les régions par défaut en plus des régions personnalisées, ajoutez-les manuellement.

Le nom interne d'une région est automatiquement converti en variable region dans le template page.tpl.php. La region [left] affichera tous les blocs qui lui sont affectés par le biais de la variable $left. Les noms internes doivent respecter les contraintes de nommage des variables PHP. Ils doivent seulement contenir des caractères alphanumériques ou le signe souligné mais ne peuvent débuter par un chiffre.

La mise en cache
Le contenu du fichier .info est mis en cache dans la base de données, il faut donc penser à vider le cache sans quoi sa modification ne sera pas remarquée par Drupal.

Conversion d'un thème statique en thème Drupal

Le gabarit principal de Drupal est appelé page.tpl.php. Faites une copie du gabarit de votre site et appelez-la page.tpl.php. La plupart du PHP nécessaire à la conversion de votre gabarit pourra être prélévée à l'intérieur du fichier page.tpl.php d'un des thèmes par défaut ou du page.tpl.php du module system (modules/system)

La section Head de votre site

```
<head>
  <title><?php print $head_title; ?></title>
      <?php print $head; ?>
      <?php print $styles; ?>
      <?php print $scripts; ?>
</head>
```

Nous utilisons ici les variables de l'API Drupal disponibles dans page.tpl.php pour l'impression des données du HEAD de votre page.

- $head_title: le titre de votre page, à utiliser dans la balise TITLE.
- $head: meta tags, keyword tags, etc...
- $styles: permet l'importation des feuilles de style déclarées dans le fichier .info.
- $scripts: permet le chargement des scripts de la page (Javascript...).

Les sidebars

Beaucoup de sites ont une barre latérale (sidebar), à gauche ou à droite de la page, placée à côté de la zone de contenu principale. C'est probablement le cas de votre gabarit. Ces barres latérales affichent différentes fonctions ou informations (par exemple, la form de connexion, les menus de navigation, les commentaires récents, etc).

Dans Drupal, ces zones sont appelées blocs et vous pouvez les personnaliser en allant dans le menu Administer > Site building > Blocks. Pour que Drupal sache où afficher ces blocs, vous devez ajouter des placeholders dans votre thème (Drupal appelle ces placeholders des régions). Il y a plus d'un placeholders disponible (et vous pouvez aussi créer les vôtres) mais il est important d'ajouter en premier lieu la barre latérale, puisque le menu d'administration y est affiché. Ajoutez les régions suivantes à votre barre latérale :

```
<?php print $left; ?>
```

...et/ou...

```
<?php print $right; ?>
```

Les menus

Dans Drupal, le menu de navigation par défaut de votre site s'appelle Primary Links. Il y a également un menu Secondary Links qui peut amener aux sous-pages tributaires du menu Primary. Lorsqu'il est dépouillé de toutes ses images et de ses styles, un menu Drupal n'est rien de plus qu'une imbrication de listes non ordonnées, en HTML brut, avec un lien pour chaque élément de la liste. Vous pouvez utiliser les CSS pour personnaliser un menu à votre façon; modifier son apparence, sa disposition, horizontale, comme des onglets, ou verticale, etc.

Pour afficher dans votre site les liens de menu Primary,ou Secondary, collez le code suivant soit dans la partie supérieure de votre thème (par exemple, si votre menu est stylisé en tant qu'onglets ou boutons), soit dans la barre latérale (dans le cas où vous souhaitez y afficher votre menu verticalement).

```
<div id=»primary»>
<?php print theme('links', $primary_links); ?>
</div> <!-- /#primary -->

<div id=»secondary»>
<?php print theme('links', $secondary_links); ?>
</div> <!-- /#secondary -->
```

Le contenu principal et dynamique de la page

A ce stade, vous avez ajouté tout le cadre de mise en page de tout votre site. Nous allons maintenant inclure la zone du gabarit où le contenu courant sera affiché (un contenu différent selon la page vue).

Localisez simplement l'endroit de votre gabarit où le contenu principal doit être affiché et copiez-y le code suivant :

```
<?php if ($breadcrumb): ?><div id=»breadcrumb»><?php print $breadcrumb; ?></div><?php endif; ?>

<div id=»content»>
<?php if ($title): ?><h1 class=»title»><?php print $title; ?></h1><?php endif; ?>
<?php if ($tabs): ?><div class=»tabs»><?php print $tabs; ?></div><?php endif; ?>
<?php if ($messages): print $messages; endif; ?>
<?php if ($help): print $help; endif; ?>
<?php print $content; ?>
</div>
```

Nous utilisons ici les variables de l'API Drupal disponibles dans page.tpl.php pour l'impression du contenu de votre page.

- $breadcrumb: Le breadcrumb de la page.
- $title: Le titre de la page.
- $help: Texte d'aide dynamique, essentiellement utiliser dans les pages d'administration.
- $messages: Utilisé notamment pour l'affichage des messages d'erreur.
- $tabs: Utilisé notamment pour l'affichage des onglets view et edit d'un node
- $content: Le contenu de la page courante.

Le footer

```
<?php if ($footer_message || $footer) : ?>
<div id=»footer-message»>
   <?php print $footer_message . $footer;?>
</div>
<?php endif; ?>
```

La balise de cloture de votre thème

Collez le code suivant à la fin de votre gabarit page.tpl.php. Vous devez placer cette balise en bas de page, juste avant la balise </body>.
Elle inclura dynamiquement tout script devant apparaître en bas de page et elle fermera toute balise qui aurait été laissée ouverte :

```
<?php print $closure; ?>
</body>
</html>
```

Créer un sous-thème

Un thème oblige à spécifier ce que vous voulez obtenir, alors qu'un sous-thème vous permet de ne spécifier que ce que vous voulez de différent.

Les sous-thèmes héritent de toutes les feuilles de style, de tous le javascript et de tous les gabarits (fichiers .tpl.php) du thème sur lequel ils sont basés. Ce qui vous permet de le utiliser rapidement et proprement - et tout cela sans avoir à hacker le code existant du thème, ce qui signifie que lorsqu'une nouvelle version du thème est publiée, le vôtre hérite automatiquement de toutes ses améliorations !

Création du dossier du thème et du fichier .info

- Dans /sites/all/themes,créez un dossier qui porte le nom de votre sous thème
- Dans ce dossier, créez un fichier votresoustheme.info qui contiendra les lignes suivantes:

```
name = votresoustheme
description = description de votre sous thème
core = 6.x
base theme = nom du dossier de n'importe quel thème ou sous-thème, du core ou contribué, par ex. bluemarine
stylesheets[all][] = votresoustheme.css
```

Toujours dans le dossier de votre sous-thème, créez un fichier vide nommé votresoustheme.css

Activation du sous-thème

- Allez dans Administer > Site building > Themes pour activer votre thème.
- Sélectionnez-le dans la liste en cochant le bouton-radio Par défaut puis cliquez sur Enregistrer la configuration.

Personnaliser l'affichage dans Drupal

Le theming dans Drupal

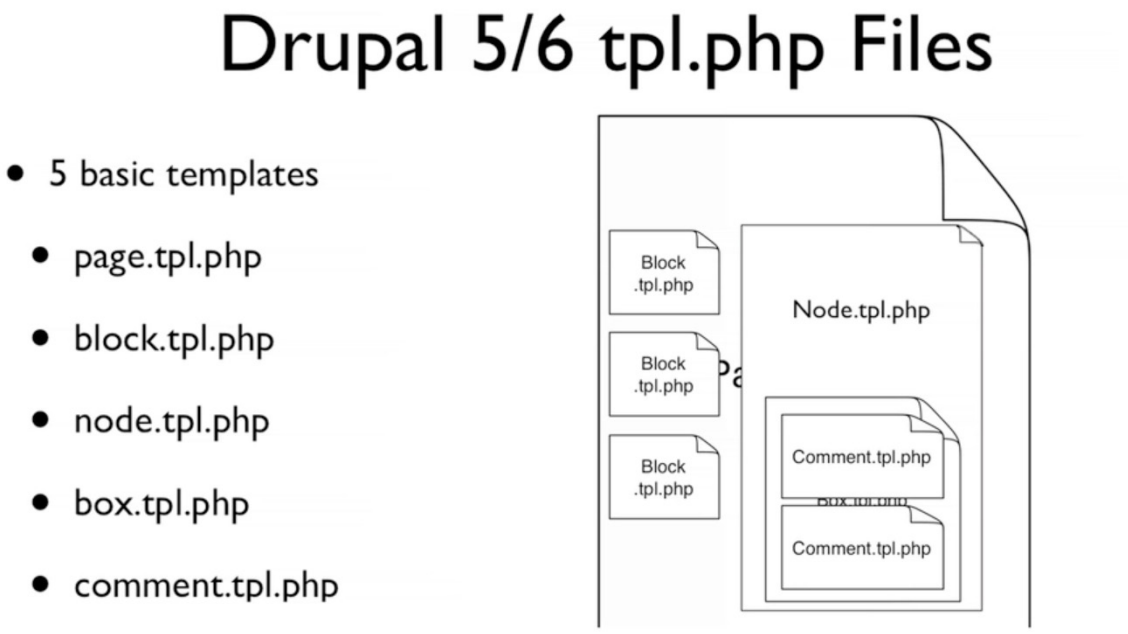
Attention : L'API Drupal 7 pour les thèmes est encore en développement.
Les principaux changements sont listés ici : http://drupal.org/update/theme/6/7

Le theming dans Drupal comporte principalement 4 étapes :
- la création du fichier .info
- le travail sur les fichiers de templates
- la surchage (overriding) des variables de templates
- la surcharge (overriding) des fonctions de thème

Pour les 2 dernières étapes, c'est à dire la surcharge des fonctions de thème et des variables tout le travail se fait à l'intérieur du fichier template.php du thème.

Les templates

Les templates de bases



Drupal utilise 5 templates de base

- page.tpl.php
- node.tpl.php
- block.tpl.php
- box.tpl.php
- comment.tpl.php

Les 3 premières, page.tpl.php, node.tpl.php et block.tpl.php sont les plus importantes.
Elles contiennent respectivement la structure XHTML des pages, nodes et blocs de votre site.

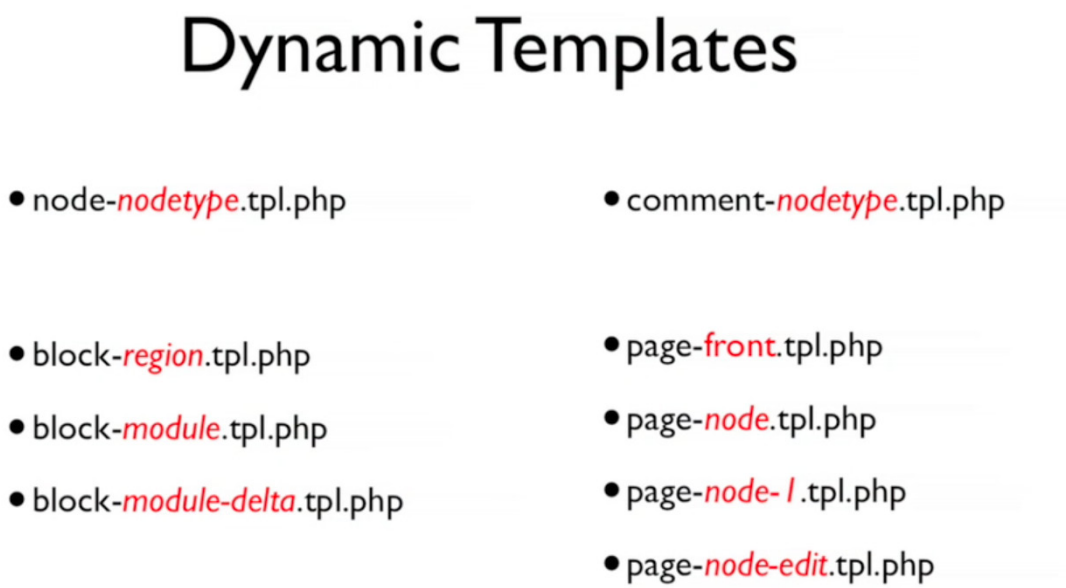
Pour modifier cette structure, il vous suffit de trouver la template responsable de l'affichage que vous souhaitez modifier et de la recopier à la racine de votre thème.

Vous avez ainsi la possibiité d'intervenir globalement au niveau de la structure de la page ou plus précisément au niveau de la structure de chaque node, bloc ou commentaire.

Les templates dynamiques

Pour cibler plus précisément une page ou un élement de contenu particulier, vous avez également la possibilité d'utiliser des templates dynamiques.

Ces templates vous permettent par exemple de cibler uniquement un certain type de nodes ou une page particulière (la homepage est le cas le plus fréquent).



Attention, pour qu'une template dynamique soit prise en compte, il est en général nécessaire de copier dans le thème la template de base correspondante.

Les templates de module

Les modules du cœur de Drupal ainsi que les modules contribués génèrent eux aussi leur propre type de templates, sur lesquels il est également possible d'intervenir afin d'adapter l'affichage de ces modules à vos besoins.

Pour cela, il suffit de suivre la même procédure, à savoir localiser la template concernée, la recopier dans votre thème et y apporter tous les changements souhaités.

Les variables de templates

L'API Drupal

L'API de Drupal fournit la liste des variables disponibles dans chacunes des templates de base de Drupal.

Elle fournit également la liste d'un certain nombre de variables disponibles dans toutes les templates

On peut notamment citer :

• $directory
Le chemin du dossier du thème.
Exemple: sites/all/themes/my_theme

• $is_admin
Indique si l'utilisateur a accés à l'administration du site.

• $is_front
Indique si la page actuelle est la page d'accueil.

• $logged_in
Indique si l'utilisateur est loggé ou non.

• $user
Contient l'objet User de l'utilisateur.

Travailler avec les variables de templates

On remplit progressivement la structure HTML des templates en affichant le contenu des variables dans les templates à l'intérieur des balises appropriées.

On peut aussi avoir besoin d'effectuer des réglages plus précis, par exemple au niveau de l'affiche des nodes ou d'un type de node particulier.

Dans ce cas, on peut afficher le contenu de la variable $node au moyen de la fonction dsm() du module devel.

Cette fonction affiche le contenu de la variable dans un bloc spécial (appelé Krumo).
On peut ensuite afficher une propriété particulière de la variable.

Par exemple :
<?php print $node->title; ?>

La surcharge des variables et des fonctions de thème

Template.php

Le fichier template.php permet de garder toutes les fonctions du thème en dehors des templates.
Il ne contient que du php. Il fournit les variables pour les fichiers templates.
Il est également utilisé pour thèmer de petits morceaux de page (theme_username)

Créer de nouvelles variables

Les fonctions de preprocess permettent d'intervenir sur les variables de template.

Il existe 2 types de fonctions de preprocess :
- theme_preprocess_page(&$vars)
- theme_preprocess(&$vars, $hook)

Quelques exemples :

```
function montheme_preprocess_page(&$vars){
$vars['ma_nouvelle_variable'] = 'une valeur';
}

function montheme_preprocess_node(&$vars) {
$vars['node];
$vars['date_day'] = format_date($node->created, 'custom', 'j');
$vars['date_month] = format_date($node->created, 'custom', 'M');
$vars['date_year'] = format_date($node->created, 'custom', 'Y');
}
```

Cette fonction récupère la date associée au node et crée 3 nouvelles variables contenant le jour, le mois et l'année.

```
function montheme_preprocess(&$vars, $hook){
$vars['classes'] = $hook;
```

Cette fonction crée une variable $classes qui contient le nom du hook courant (page, node, etc...)

Remplacer des variables

```
function montheme_preprocess_page(&$vars){
if($vars['site_slogan'] == '<twitter>'){
$response = drupal_http_request('http://search.twitter.com/search.json?q=barackobama');
$data = json_decode($response->data);
$tweet = $data->results[array_rand($data->results)];
$vars['site_slogan'] = check_plain($tweet->text);
}
}
```

Surcharger les fonctions de thème

Les fonctions de thème

Les fonctions de thème de Drupal sont toutes documentées dans l'API Drupal.
http://api.drupal.org/api/group/themeable

Surcharger une fonction de thème, 3 étapes

• trouver la fonction de thème que nous souhaitons overrider dans le fichier .module
exemple :

```
function theme_username($object){
....
return $output;
}
```

• la copier dans le fichier template.php

• renommer le prefixe de la fonction (thème_ devient montheme_)

exemple :

```
<?php
function theme_breadcrumb($breadcrumb) {
  if (!empty($breadcrumb)) {
    return '<div class=»breadcrumb»>'. implode(' » ', $breadcrumb) .'</div>';
  }
}
?>

fonction montheme_breadcrumb($breadcrumb){
if (!empty($breadcrumb)){
$breadcrumb[]= drupal_get_title;
return '<div class=»breadcrumb»>'.implode('>', $breadcrumb).'</div>';
}
}
```

Nous avons remplacé le séparateur et ajouté le titre de la page courante.

Drupal utilisera dorénavant la nouvelle fonction (hook) au lieu de la fonction par défaut.

Comment la surcharge de fonctions de thème fonctionne t'elle ?

Les fonctions de thème ne sont jamais appelées directement.
Pour savoir quelle fonction de thème utiliser, Drupal utilise la fonction theme().
On passe à cette fonction le nom de la fonction de thème recherchée

Par exemple :
theme ('breadcrumb')
• la fonction cherche si un override de la fonction existe dans le thème
• si elle n'en trouve pas elle cherche si un override existe dans le moteur de template (en général php_template)
• si elle n'en trouve pas elle cherche la fonction à l'intérieur du module

Notions avancées

Comment créer un module

Démarrer

Nommer le module
• Choisir un nom abrégé. Il sera utilisé dans tous les fichiers et dans tous les noms de fonctions du module. Il doit donc débuter par une lettre et ne doit contenir, selon les conventions de nommage de Drupal, que des lettres en minuscule et des signes soulignés. pour cet exemple, nous choisirons monmodule comme nom abrégé.

Créer un dossier et un fichier pour le module
• Créez un dossier dans votre installation Drupal : sites/all/modules/monmodule.
• Créez ensuite un fichier PHP et sauvegardez-le sous le nom monmodule.module dans le dossier sites/all/modules/monmodule.

Les fichiers modules débutent avec la balise d'ouverture PHP immédiatement suivie d'un balise ID CVS placée dans un commentaire inline. La balise ID CVS, Id, est un token que le système de contrôle de version de Drupal.org complétera avec des informations de révision et d'auteur. Il vaut mieux prendre l'habitude d'écrire cette balise dans votre module, même si vous ne pensez pas le diffuser sur Drupal.org ou pas. Une balise ID similaire est placée dans tous les autres fichiers associés à votre module.

```
<?php
// $Id$
?>
```

Cette balise ID n'est pas une syntaxe spéciale du PHP et elle n'est là que pour le système de contrôle de versions. Aussi, elle doit être placée dans une ligne de commentaire. On le répète, cette balise sera automatiquement complétée et vous pouvez désormais l'oublier.

Normes de programmation
Omettez la balise PHP fermante ?>. La mettre pourrait causer un comportement erratique sur certaines configurations de serveur (notez que les exemples du manuel montrent cette balise fermante uniquement pour des raisons de formatage de texte, vous ne devez pas la mettre dans votre code à vous).

Toutes les fonctions de votre module utilisées par Drupal sont nommées {nom_du_module}_{hook}, où hook est un suffixe de nom de fonction pré-déclaré. Drupal appellera ces fonctions pour obtenir des données spécifiques, les déclarer selon ce modèle indique à Drupal où regarder.

Communiquer votre module à Drupal

Un module se compose, au minimum de deux fichiers monmodule.info et monmodule.module.

Création d'un fichier .info pour votre module, et implémentation de hook_help()
Tous les modules doivent avoir un fichier nom_du_module.info qui contient des méta-informations sur le module.

Le format habituel est le suivant :
; Id
name = Mon module
description = Mon premier module
package = Mes Modules - optional
core = 6.x

Détails du fichier Info

• name (obligatoire)
Il doit respecter les normes de mise en majuscules de Drupal 6 : seule la première lettre du premier mot doit être en majuscule («Exemple module», pas «Exemple Module»)

• description (obligatoire)
Soyez le plus concis possible. Ce champ est limité à 255 caractères.
Notez que si la description comporte des caractères spéciaux, ils doivent être remplacés par leurs entités HTML.
Si la description comporte des guillemets simples ou des apostrophes vous pouvez simplement mettre la chaîne de caractères dans des doubles guillemets.

• core (obligatoire)
Le fichier .info doit indiquer pour quelle version du core Drupal le module ou le thème ont été écrits.

• dependencies (facultatif)
Il y a quelques options supplémentaires qui peuvent apparaître dans le fichier .info, l'une d'elles sont les dépendances du module. Si un module requiert l'activation d'un ou plusieurs autres modules, énumérez les nom de fichiers de ces modules avec la syntaxe suivante :
dependencies[] = taxonomy
dependencies[] = comment

Si des dépendances sont assignées à un module, Drupal ne permettra pas son activation tant qu'elles ne seront pas satisfaites.

• package (facultatif)
Si un module fait partie d'un groupe, la page Administrer » Construction du site » Modules l'affichera avec les autres modules du même groupe. Si aucun groupe n'est attribué, il sera listé dans le groupe Autres. Ne pas assigner de groupe à votre module est tout à fait possible; en règle générale les groupes sont utilisés pour les modules distribués ensemble ou faits pour être utilisés ensemble. Dans le doute, laissez ce champ vide.

Le fichier utilise le format INI et peut avoir un ; Id pour que le CVS insère les données ID au fichier.

Dans Drupal 7

Voici un exemple de fichier .info dans Drupal 7

```
; $Id$
name = Mon module
description = Mon premier module
package = Mes Modules - optional
core = 7.x
files[] = monmodule.module
files[] = monmodule_types.inc
files[] = monmodule.admin.inc
configure = admin/config/monmodule
```

Quelques changements à signaler :

- fichiers (obligatoire)

Les fichiers .info des modules doivent désormais mentionner explicitement tous les fichiers de code chargeables. Drupal supporte maintenant un registre de code chargeable dynamiquement. Pour son fonctionnement, tous les modules doivent maintenant déclarer, dans le fichier .info, les fichiers de code-source contenant des déclarations de classes ou de fonctions. Comme ceci :

```
name = Module Exemple
description = «Un module Exemple.»
...
files[] = exemple.module
files[] = exemple_types.inc
files[] = exemple.admin.inc
```

Lorsqu'un module est activé, Drupal parcourra tous les fichiers déclarés et indexera toutes les fonctions et toutes les classes qu'il trouvera. Cela veut dire que n'importe quelle classe ou fonction qui peut être appelée indirectement (comme un hook) peut se trouver en dehors du fichier .module et sera chargée en cas de besoin. Les classes seront chargées automatiquement par PHP lors de leur premier accès.

- configure (facultatif)

Le path vers la page d'administration principale du module. Si le module est activé, des liens « Configure » et « Droits d'accès » apparaîtront. Ce qui suit sera le chemin du lien « Configure » affiché pour le module dans la page des modules.

```
configure = admin/config/example
```

Spécifier les droits disponibles

Dans Drupal 6
La prochaine fonction à écrire est la fonction des droits d'accès, implémentant la fonction Drupal hook_perm(). C'est là que vous déclarerez les noms des droits de votre module - il ne s'agit pas d'accorder des droits ou de déclarer la portée des droits, il s'agit simplement d'indiquer quels droits sont disponibles pour le module.

Nous implémentons hook_perm() en créant une fonction appelée monmodule_perm() dans le fichier module monmodule.module.

Le rôle de cette fonction sera juste de renvoyer une liste énumèrant les noms des droits d'accès utilisés par le module; une fois que vous aurez déclaré les droits dans l'implémentation de hook_perm(), un administrateur pourra définir les rôles ayant ces droits dans la page Administrer » Gestion des utilisateurs » Droits d'accès.

Par exemple, pour créer un droit appelé «access monmodule content » :

```
<?php
/**
* Valid permissions for this module
* @return array An array of valid permissions for the monmodule module
*/
function monmodule_perm() {
  return array('access monmodule content');
} // function monmodule_perm()
?>
```

Le contenu de la chaîne de caractères qui décrit le droit est libre mais elle doit être unique parmi celles des autres modules. Sinon, une occurrence d'un droit pourra être pris pour un autre. Pour cette raison, la chaîne décrivant le droit devrait comporter le nom du module afin d'éviter des conflits avec d'autres modules. Les conventions de nommage proposent action_verbe_nomdumodule. Utilisez l'exemple suivant comme modèle pour tous les modules que vous développerez:

```
<?php
function monmodule_perm() {

return array('access newmodule', 'create newmodule', 'administer newmodule');

} // function newmodule_perm
?>
```

Dans Drupal 7
Dans Drupal 7, hook_perm() devient hook_permission()

Pour plus d'informations
http://api.drupal.org/api/function/hook_permission/7

Déclarer et générer le bloc de contenu

Block modules : modules qui créent essentiellement du contenu de bloc (module menu)
Node modules : modules qui génèrent du contenu de page (modules blog ou forums).

Dans Drupal 6
Le hook pour créer les blocs s'appelle hook_block. Pour l'implémenter, créez une fonction appelée monmodule_block() dans le fichier monmodule.module.

```
<?php
/**
* Implementation of hook_block
* @param string $op one of «list», «view», «save» and «configure»
* @param integer $delta code to identify the block
* @param array $edit only for «save» operation
**/
function monmodule_block($op = 'list', $delta = 0, $edit = array()) {

  // VOTRE CODE DE GENERATION DE BLOC ICI

} // function monmodule_block
?>
```

La fonction prend ces trois paramètres :
• $op (ou operation : les fonctions hook_block() sont appelées pour réaliser quatre opérations différentes, qui sont list,, view, save et configure. Ce paramètre indique à la fonction quelle opération est appelée. Nous parlerons de l'opération list ci-dessous et de l'opération view dans la prochaine page du tutoriel. Les opération configure et save permettent à votre bloc d'avoir une page de configuration et de sauvegarder les paramètres - nous ne les utiliserons pas dans ce tutoriel.
• $delta : votre module peut définit plus d'un bloc dans son opération list. Chaque bloc a un code delta, qui est normalement un nombre, et Drupal passe ce nombre pour les autres opérations afin que vous puissiez identifier le bloc du module sur lequel effectuer l'opération. Notre module exemple n'aura qu'un seul bloc. Le module user du core Drupal est un exemple de module qui a plusieurs blocs : bloc de login user, bloc utilisateurs récents et bloc qui est en ligne.
• $edit : utilisé avec l'opération save seulement. Non abordé dans ce tutoriel.

La première opération est list, qui énumère les blocs fournis par le module, et déclare comment ils seront vus dans la page Administrer » Construction du site » Blocs (le module Blocks appellera cette fonction avec $op='list' lorsqu'il construit la liste des blocs pour la page d'administration des blocs).

Pour un exemple complet se reporter à block_example.module dans l'API Drupal
http://api.drupal.org/api/examples/block_example--block_example.module/6/source

Dans Drupal 7
Dans Drupal 7 le hook_block a été remplacé par une nouvelle série de hooks plus spécifiques : hook_block_info(), hook_block_configure(), hook_block_save(), and hook_block_view()

Pour un exemple complet se reporter à block_example.module dans l'API Drupal
http://api.drupal.org/api/examples/block_example--block_example.module/7/source

Pour plus d'informations sur la conversion d'un module de Drupal 6 à Drupal 7
http://drupal.org/update/modules/6/7#remove_op

Exemple :

monmodule.module (Drupal 6)

```
function monmodule_block ($op = 'list', $delta = 0, $edit = array()){
   // Enumeration des blocs disponibles
   if ($op == 'list') {
      $blocks['monmodule'] = array(
            'info' => t('Mon bloc'));
      return $blocks;
   // Opération de configuration
   } else if ($op == 'configure') {
      $form['monmodule_var'] = array(
         '#type' => 'textfield',
         '#title' => t(«Info»),
         '#default_value' => «info»,
      );
         return $form;
   // Opération de configuration
   } else if ($op == 'save') {
      variable_set ('monmodule_var_value', $edit['monmodule_var']);
   // Opération d'affichage
   } else if ($op == 'view') {
       $block = array('subject' => t('monmodule'),
       'content' => variable_get ('monmodule_var_value', 'Foo'));
       return $block;
   }
}
```

monmodule.module (Drupal 7)

```
function monmodule_block_info() {
   $blocks['monmodule'] = array(
   'info' => t('Mon bloc'));
   return $blocks;
}

function monmodule_block_configure($delta = '') {
      $form['monmodule_var'] = array(
         '#type' => 'textfield',
         '#title' => t(«Info»),
         '#default_value' => «info»,
      );
         return $form;
}

function monmodule_block_save($delta = '', $edit = array()) {
      variable_set ('monmodule_var_value', $edit['monmodule_var']);
}

function monmodule_block_view($delta = '') {
       $block = array('subject' => t('monmodule'),
       'content' => variable_get ('monmodule_var_value', 'Foo'));
   return $block;
}
```

Création de la page de configuration du module

Créer la fonction de paramétrage

• Nous utiliserons $form qui est un tableau de l'API Forms de Drupal dans lequel on définit des éléments de formulaire. Chaque élément du tableau correspond à un élément du formulaire

• Nous n'avons à définir que les éléments pour le paramétrage -- la fonction system_settings_ form() s'occupera de créer la page du formulaire, d'ajouter le bouton Submit et de la sauvegarde des paramètres.

• Drupal tient à jour une base des variables ou des paramètres; chaque paramètre doit donc avoir un nom unique, pour cela on utilise le nom du module comme préfixe de nom de variable.

• Tout le texte affiché par le formulaire passe par la fonction de traduction t(), ainsi, si des sites tournent en d'autres langues, ils peuvent utiliser notre module.

• Reportez-vous au Forms API Reference et au Forms API Quickstart Guide de Drupal pour plus d'informations sur les opérations possibles avec l'API Forms de Drupal.

Ajouter la page à hook_menu

Une fois que vous aurez créé la fonction pour le formulaire de paramètres, vous devrez définir une URL Drupal pour votre page de paramètres. Cela se fait en implémentant un hook_menu. Dans cette implémentation de hook_menu, nous retournons un tableau qui indique à Drupal l'URL à utiliser, le titre à afficher, la fonction à appeler pour générer la page et les droits requis.

Pour un exemple complet se reporter à block_example.module dans l'API Drupal http://api.drupal.org/api/examples/page_example--page_example.module/function/page_example_menu/7

Exemple :

```
<?php
function monmodule_menu() {
  $items['admin/config/monmodule'] = array(
    'title' => 'Mon module',
    'type' => MENU_NORMAL_ITEM,
    'page callback' => 'monmodule_menu_config_page',
    'access arguments' => array('access content'),
    'access callback' => TRUE,
    'expanded' => TRUE,
  );
 $items['admin/config/monmodule/page1'] = array(
  'title' => 'Réglages 1',
  'type' => MENU_NORMAL_ITEM,
  'page callback' => 'monmodule_menu_config_page1_page',
  'access arguments' => array('access content'),
  'access callback' => TRUE,
  'expanded' => TRUE,
);
 $items['admin/config/monmodule/page2'] = array(
  'title' => 'Réglages 2',
  'type' => MENU_NORMAL_ITEM,
  'page callback' => 'monmodule_menu_config_page2_page',
  'access arguments' => array('access content'),
  'access callback' => TRUE,
  'expanded' => TRUE,
);
  return $items;
}

function monmodule_menu_config_page($content = NULL) {
  $base_content = t(
          'Ceci est la page de configuration de Mon Module'
  );
  return '<div>' . $base_content . '</div><div>' . $content . '</div>';
}

function monmodule_menu_config_page1_page($content = NULL) {
  $base_content = t(
          'Première page de configuration'
  );
  return '<div>' . $base_content . '</div><div>' . $content . '</div>';
}

function monmodule_menu_config_page2_page($content = NULL) {
  $base_content = t(
          'Deuxième page de configuration'
  );
  return '<div>' . $base_content . '</div><div>' . $content . '</div>';
}
?> 1. Mauvais 2. Passable 3. Moyen 4. Bon 5. Excellent NC : NON CONCERNE
1/Objectifs 1 2 3 4 5 NC
```