Développement Web par composants graphiques réutilisables

Nicolas Thouvenin
INIST-CNRS
2 allée du parc de Brabois 54519 Vandœuvre lès Nancy

Stéphane Gully
INIST-CNRS
2 allée du parc de Brabois 54519 Vandœuvre lès Nancy

Résumé

Zend Framework, Symphony, CakePHP sont des frameworks PHP MVC ayant une approche globale pour construire une application Web.

Cette philosophie impose certaines limites que nous avons dépassées en préférant un modèle de type « Hierarchical MVC ».

Pxxo permet de créer des composants graphiques (ou widgets) qui permettent de construire une page Web comme une hiérarchie de composants plus simples. Chaque composant est une micro-application Web pouvant être facilement imbriquée dans une application existante ou même dans un autre composant.

Au final, cette imbrication de composants élémentaires peut constituer une application complète.

Mots clefs

Framework, PHP 5, Widgets, Développement, OpenSource, HMVC

1 Introduction

Depuis 10 ans, l'INIST-CNRS développe et fournit des sites et des services Web. Les technologies utilisées ont rapidement évolué des simples programmes CGI écrits en Langage C vers des outils et des techniques plus modernes. Aujourd'hui, la majorité des développements Web réalisés à l'INIST-CNRS se fait avec le langage PHP. Ce langage est utilisé soit dans des sites Web classiques, soit dans des applications spécifiques développées en interne, soit au travers de logiciels de gestion de contenu tels que DokuWiki et SPIP.

Cette diversité d'usage du langage PHP pose problème lorsque l'on souhaite capitaliser et réutiliser du code source produit dans différents projets. Pour y pallier, deux pistes sont possibles :

- Utiliser partout et tout le temps le même logiciel de gestion de contenu.
- Utiliser un framework.

Dans le premier cas, le choix du logiciel est problématique car il faut trouver un logiciel capable de s'adapter à des besoins divers. Le logiciel SPIP s'est rapidement, et largement, imposé à l'INIST comme dans beaucoup de laboratoires CNRS. Or si ce logiciel est parfaitement adapté à la production de contenu éditorial, il est impossible de l'utiliser comme outil collaboratif ou bien pour gérer des contenus spécifiques (ex : données terminologiques, biologiques…). L'usage d'un logiciel unique est donc utopiste, reste l'usage d'un framework.

Le choix d'un framework pour cadrer ces développements est un choix difficile, surtout si le but principal est de pouvoir réutiliser du code entre différents projets totalement indépendants. On peut facilement trouver des frameworks qui proposent des briques logicielles prêtes à l'emploi (ORM, DBA, etc…) et facilement réutilisables d'un projet sur l'autre. Par contre, très peu de frameworks proposent un cadre de développement permettant de réutiliser, de partager et d'intégrer du code dans des logiciels ou des programmes existants.

À partir de 2004, l'INIST-CNRS a commencé à utiliser, développer et soutenir le projet Pxxo. Il s'agit d'un framework PHP qui propose une approche par composants élémentaires et indépendants. L'idée de base est de permettre la construction d'une page

Web comme une hiérarchie de composants gigognes. Chacun de ces composants, nommés widgets, peut être intégré sans effort dans n'importe quel type d'application écrite en PHP, mais peut également être associé avec d'autres composants pour former une application complète.

2 Pxxo, framework non intrusif

Pxxo (http://www.pxxo.net) est un ensemble de classes écrites en langage PHP. Ces classes sont regroupées dans une librairie (ou package) compatible avec les normes PEAR. Pxxo sert à créer des composants graphiques (ou widgets) qui permettent de construire une page HTML comme une hiérarchie de composants plus simples. Chaque composant est une micro-application Web pouvant être facilement imbriquée dans une application existante ou dans un autre composant. Les micro-applications ainsi développées peuvent autant être un simple sélecteur de date qu'une interface utilisateur plus complexe permettant la modification de structures XML. Pxxo fournit un cadre de programmation, des techniques et des outils pour créer un composant graphique et gérer un ensemble de composants. Pour chaque composant, Pxxo propose une solution simple permettant de séparer la présentation et les traitements. Chaque développeur reste autonome et indépendant, tout en ayant la possibilité de réutiliser des parties d'autres applications sans effort. Il peut aussi très simplement valoriser son travail en mettant à disposition ses développements pour d'autres projets. Les principaux avantages sont :

- Qualité : en imposant une structuration du code, Pxxo améliore la qualité du code PHP produit. Le code produit est plus facilement maintenable et réutilisable.

- Modularité : le respect du principe de décomposition en hiérarchie de composants permet aux développeurs Pxxo de modifier simplement et rapidement l'apparence et l'ergonomie des IHM (Interfaces Homme Machine) produites. Cette modularité permet également de réaliser des IHM "patchwork" qui réutilisent des composants développés auparavant.

- Performance : Pxxo intègre plusieurs techniques qui améliorent nettement la vitesse de chargement des pages Web.

Ses principaux inconvénients sont :

- Une philosophie en rupture par rapport à la majorité des développements PHP : Pxxo propose une approche par composants alors qu'habituellement l'approche par page est privilégiée.

- Un coût initial d'appropriation existe, même si Pxxo est fortement porté par la philosophie KISS, "Keep It Simple and Stupid".

2.1 Présentation

Pxxo est basé sur le paradigme de la programmation orientée objet. À ce titre, il implémente plusieurs motifs de conceptions : Fabrique, Vue composite, Décorateur, Patron de méthode, Singleton ainsi que le motif HMVC (Hierarchical-Model-View-Controller) adapté au Web. Le développement d'un widget Pxxo se fait par dérivation d'une classe générique dans laquelle on définit des traitements auxquels on associe des fichiers de modèles triés par thème. La construction d'une hiérarchie d'objet se fait :

- soit par la création d'objets fils, un objet pouvant posséder N fils.
- soit par l'insertion séquentielle d'objets parents, un objet pouvant posséder un seul et unique père.

L'organisation, le nommage et la structuration des méthodes, des fichiers et des répertoires se font en respectant des conventions. Un widget accepte une liste de paramètres venant de son père ou directement de l'application (variables globales). Pour permettre aux widgets de s'intégrer au mieux dans des environnements graphiques divers, Pxxo propose un mécanisme de thèmes basé sur la surcharge. Chaque fichier (images, javascript, css, html) utilisé pour produire l'interface pourra être redéfini localement. Pour réutiliser et distribuer simplement les composants développés, Pxxo propose de livrer chaque widget sous la forme de package PEAR. Ceci rend extrêmement simple et pratique le déploiement, le suivi, le partage, et la centralisation du code.

2.2 Usage

Utiliser un widget Pxxo est extrêmement simple. Il s'utilise comme n'importe quelle classe d'une librairie PHP : On inclut dans un premier temps son fichier de description pour créer un objet avec l'instruction *new*.

Dans un souci de simplicité et d'homogénéisation entre tous les widgets, quelle que soit leur nature, les paramètres de création d'un widget se présentent toujours comme un tableau associatif de clé et de valeur. Ce choix délibéré permet d'une part de paramétrer un widget à l'aide d'une documentation claire sous forme d'un tableau descriptif, et d'autre part l'intégration des widgets dans d'autre programmes via une description XML (exemple : plugin dokuwiki).

Avant d'exécuter et d'afficher le widget, on peut facilement lui ajouter de nouvelles fonctionnalités en le décorant avec un ou plusieurs autres widgets.

```php
<?php
require_once 'MonWidget.php';
$p = array('title' => 'Mon widget');
$o = new MonWidget($p);
$o->addDecorator('Pxxo_Widget_Core');
$o->main();
$o->dump();
?>
```

Figure 1 - *Exemple d'utilisation d'un widget*

2.3 Squelette d'un widget

La structure d'un widget se décompose en deux parties :

- La classe du widget
- Les templates du widget

La classe est chargée d'effectuer les actions demandées par l'utilisateur via l'interface Web. A chaque type d'action correspond une méthode de la classe (cf figure 2) :

- La méthode *index* est obligatoire, elle est chargée de définir le comportement par défaut du widget. Dans l'exemple, elle sera chargée de créer un formulaire.
- La méthode *resultats* est elle une méthode spécifique à notre exemple. Elle sera appelée pour effectuer une recherche dont les paramètres seront envoyés par le formulaire de la méthode index.

Ces deux méthodes ne sont pas chargées d'effectuer l'affichage de la page HTML. Elles vont seulement communiquer des informations (données saisies par l'utilisateur, résultat d'une recherche dans la base de données) à des templates qui eux seront chargés de présenter au format HTML ces mêmes informations. A chaque méthode correspond un template du même nom :

- Le template *index.php.html* est chargé d'afficher le formulaire de recherche
- Le template *resultats.php.html* est chargé d'afficher la liste des résultats

Ces templates définissent la structure de la page HTML correspondant à chaque action. Le style graphique et l'ergonomie du widget sont contrôlés par des feuilles de styles (CSS), des images (JPEG, PNG...), et par des scripts écrits en Javascript (s'appuyant éventuellement sur la librairie JQuery gérée nativement). Toutes ces ressources sont regroupées de façon cohérentes dans un répertoire.

L'existence de ce répertoire nous amène à la notion de thème. Le thème contrôle entièrement l'apparence visuelle et l'ergonomie du widget. Chaque widget possède un thème par défaut qui est représenté par un répertoire nommé *default*. Ce thème peut alors être dérivé pour changer complètement l'apparence et l'ergonomie d'un widget. Ceci permet d'intégrer un même widget dans

différents applicatifs ayant des harmonies visuelles différentes. La création d'un thème consiste à créer un nouveau répertoire (son nom sera le nom du thème - exemple : *blueprint*) et à surcharger les ressources du thème *default* :

- Le fichier *style.php.css* va remplacer celui du thème *default*
- Le fichier *background.png* est un nouveau fichier utilisé par le fichier *style.php.css*
- Les fichiers *index.php.html*, *resultats.php.html* et *logo.png* sont issus du thème *default* et donc réutilisés par le thème *blueprint.*

De façon plus générale, la création d'un thème passe par l'héritage du thème par défaut. Ceci signifie que seuls les fichiers à personnaliser sont redéfinis et éventuellement complétés par des fichiers supplémentaires.

Séparer clairement les templates du code permet de répartir plus facilement les taches entre les développeurs et les graphistes.

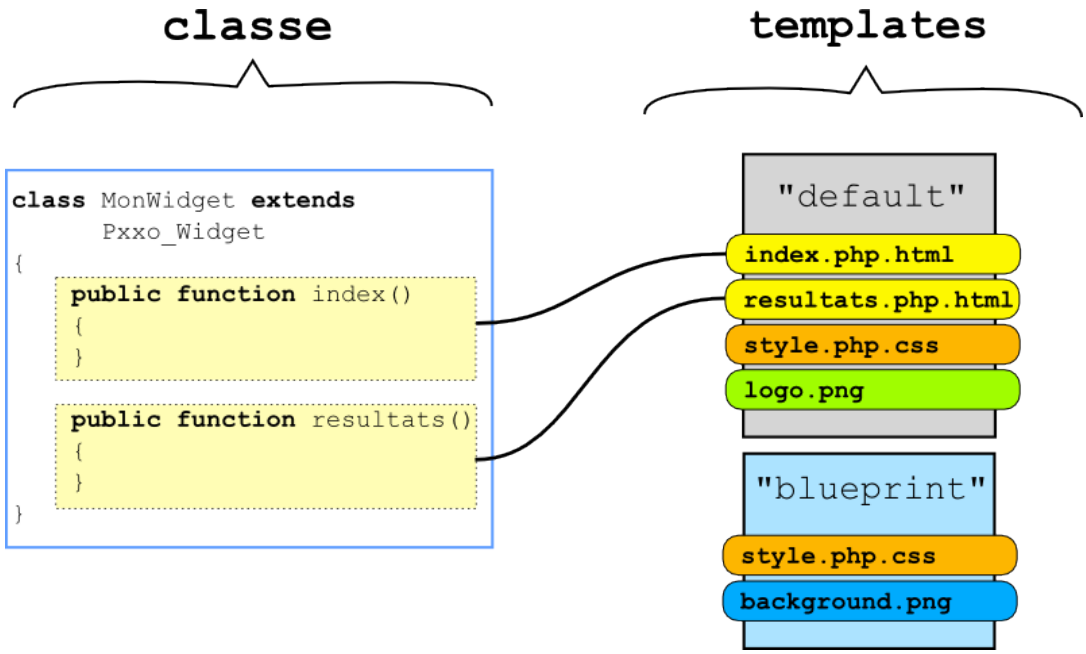

Figure 2 - *Vue schématique d'un widget*

2.4 Hierarchical-Model-View-Controller

Comme vu précédemment, Pxxo propose une séparation claire entre le code PHP et le code HTML, ce découpage correspond au modèle et à la vue du motif de conception classique Modèle, Vue, Contrôleur. Contrairement au motif classique, Pxxo n'impose pas l'usage d'un contrôleur général dédié à une application mais intègre au sein de chaque widget un contrôleur local et indépendant.

Les widgets Pxxo sont par nature prévus pour être assemblés hiérarchiquement. Chaque widget :

- Est attaché ou non à un widget parent : son père.
- Peut utiliser un ou plusieurs sous-widgets : ses fils.
- Peut ajouter un ou plusieurs widget ascendants : ses décorateurs.

Cette organisation hiérarchique se rapproche de celle que l'on peut trouver dans des interfaces utilisateur graphiques non Web (Qt, Swing, wxWidget).

L'organisation hiérarchique de plusieurs schémas MVC locaux forme ce que l'on appel le motif de conception Hierarchical-Model-View-Controller (HMVC).

Ce type d'architecture permet de construire des applications complètes en imbriquant des widgets spécifiques et des widgets génériques réutilisables dans différents contextes.

Par exemple la figure 3 montre un assemblage de widgets totalement indépendants d'un contexte applicatif (widget générique) :

- *Pxxo_Widget_Form*
- *Pxxo_Widget_Input_Suggest*
- *Pxxo_Widget_Input_Select*
- *Pxxo_Widget_Input_Submit*

Et de widgets dédiés à l'application (widget spécifique) :

- *MonWidget*
- *ListeResultats*

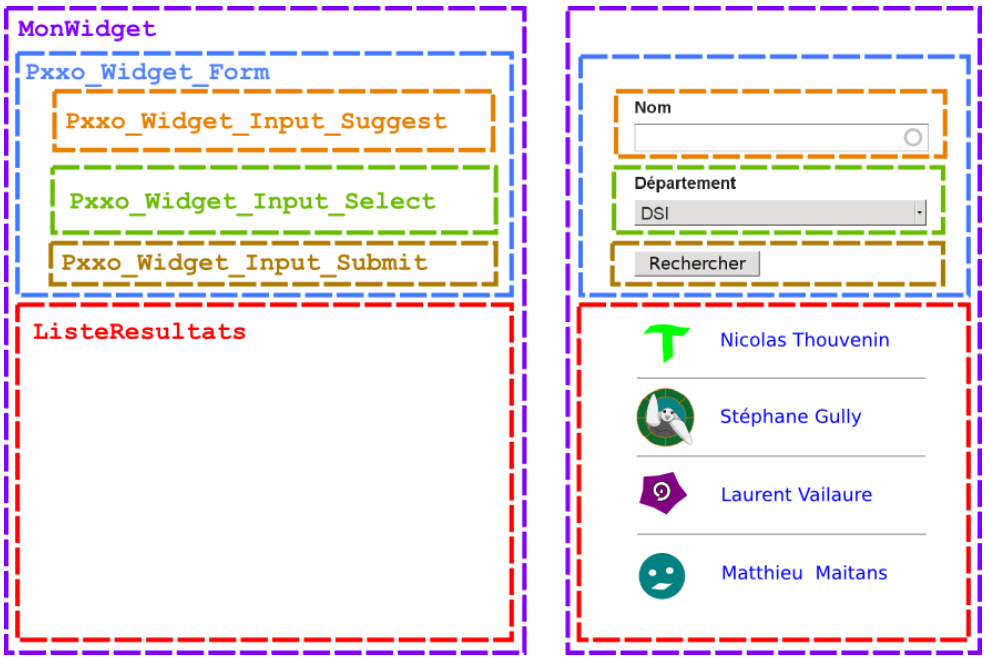

Figure 3 - *Plan d'assemblage de widgets*

2.5 Distribution

Chaque widget Pxxo est par nature indépendant, il embarque le code PHP et ses ressources associées. Ils peuvent donc être emballés et numérotés (versioning) facilement. Pour gérer la distribution de paquet, Pxxo s'appuie sur les technologies PEAR (http://pear.php.net), ce qui permet de créer des dépôts PEAR locaux alimentés en paquets par des channels. Le channel PEAR officiel bien sûr mais également le channel Pxxo qui propose plusieurs dizaines de widgets génériques prêts à l'emploi.

Grâce à la gestion de paquets, on bénéficie pour des composants graphiques de la même souplesse d'utilisation qu'avec les autre système de gestion de paquets (rpm, deb, etc …) :

- Mise à jour automatique, une garantie de sécurité et de qualité.
- Une gestion de dépendance entre les différents paquets et ou différentes versions
- Une installation simplifiée

À noter, qu'un dépôt local n'est pas directement accessible au travers du protocole HTTP. On peut très bien utiliser le dépôt PEAR fournit en standard dans les distributions. C'est à dire installer Pxxo au niveau système pour en faire profiter plusieurs applicatifs indépendants.

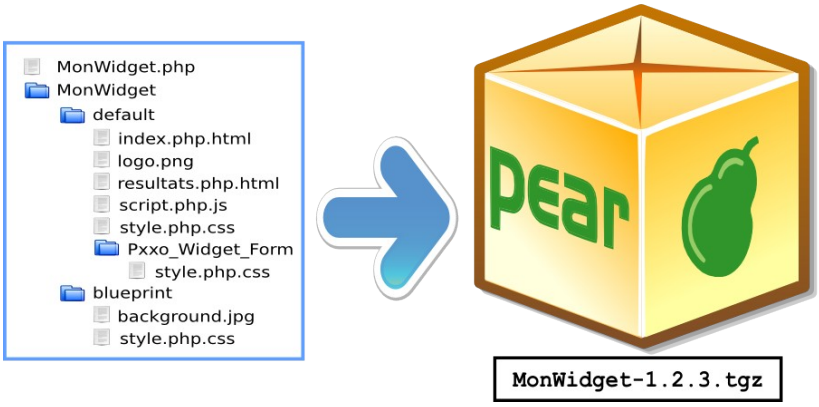

Figure 4 - *Package pear d'un widget*

3 Cas d'utilisations

Refdoc (http://www.refdoc.fr), le nouveau site proposé par l'INIST-CNRS (ouverture en janvier 2010) permettra la commande ou la consultation d'un fond de 35 millions d'articles, ouvrages, rapports, actes de congrès en Science, Technologie, Médecine, Sciences Humaines et Sciences Sociales, de 1847 à nos jours. Pxxo a été utilisé pour la réalisation des interfaces Web du « frontend » et du « backoffice ».

TermSciences (http://www.termsciences.fr/), portail terminologique développé par l'INIST en association avec le LORIA et l'ATILF a pour but de valoriser et de mutualiser les ressources terminologiques (lexiques, dictionnaires, thésaurus) des organismes publics de recherche et d'enseignement supérieur pour aboutir à la constitution d'un référentiel terminologique commun. Le site TermSciences est un très bon exemple d'application Web ne pouvant pas être réalisée avec un CMS éditorial tel que SPIP. En effet, les données traitées sont très spécifiques au domaine de la terminologie et les interfaces de navigation et d'édition de ces données arborescentes ont nécessité des développements particuliers. Le framework Pxxo a été utilisé pour réaliser la totalité des interfaces homme machine.

Cependant, du fait de la modularité des widgets Pxxo, il est tout-à-fait possible d'intégrer des développements Pxxo dans des CMS existants. Par exemple, les formulaires d'inscriptions du "Festival du film de chercheur" (http://www.filmdechercheur.eu/) ont été réalisés avec le framework Pxxo. Cette tâche aurait été impossible en utilisant uniquement SPIP qui ne gère pas la construction de formulaires. Le site du "Corpus de la parole" du ministère de la culture (http://corpusdelaparole.culture.fr/) est également basé sur le CMS SPIP et intègre plusieurs widgets développés en Pxxo pour naviguer dans la base de données très spécifique des corpus oraux (format SMIL : XML+Audio).

Par ailleurs, il existe un plugin Pxxo pour DokuWiki (http://www.dokuwiki.org/plugin:pxxo). Ce plugin permet une intégration poussée des widgets Pxxo au sein même d'une page wiki au moyen d'un balisage de description des paramètres en XML. Un exemple concret d'utilisation est le site du réseau de développeurs DevelopR6 (http://developr6.dr6.cnrs.fr). Ce site est basé sur Dokuwiki et utilise des widgets pxxo pour gérer l'identification des utilisateurs, l'inscription aux listes de discussions et la rubrique revue de Web.

4 Perspectives

Avec maintenant plus de 6 ans d'expérience et d'utilisation en production, Pxxo est stable et mature. Le cœur du programme évolue au rythme des évolutions du langage PHP. Les nouveaux développements concernent plutôt la création et l'amélioration de widgets génériques. Plus précisément, nous cherchons à suivre les tendances du Web, les widgets de création de formulaires en sont un bon exemple.

Pxxo est un logiciel libre, il est donc ouvert à toutes personnes intéressées souhaitant l'utiliser, y contribuer, ou participer à ses évolutions.