

Objectifs du tutoriel : Installer, paramétrer et découvrir l'utilisation de tout un environnement de développement Web Php basé uniquement sur des solutions open-source. Vous allez voir que nous allons paramétrer un environnement de fou que les plus grandes boîtes n'ont pas toujours (et je sais de quoi je parle)! J'espère que ce tutoriel vous donnera envie de laisser tomber votre bloc note...

Ce tutoriel a été réalisé sous Ubuntu mais il peut tout à fait se réaliser sous n'importe quel OS.

Niveau : Débutant à intermédiaire pour quelques points

Pré-requis :

- Quelques notions en html et php pour comprendre le contexte...
- Environnement Java (JRE) car Eclipse est un exécutable Java
- Un serveur Web LAMP, MAMP ou WAMP suivant votre OS
- Un serveur SVN (local ou distant au choix)
- Php > 5.1.0 pour Mantis
- Du temps, car c'est un long tuto...

Plan du tutoriel :

- [Introduction](#)
- [Tutoriel](#)
 - [Installation Eclipse et PDT](#)
 - [Téléchargement Eclipse](#)
 - [Installation de PDT](#)
 - [Création d'un projet Php](#)
 - [Interprétation des scripts Php](#)
 - [Mise en place du débbugger](#)
 - [Intro](#)
 - [Paramétrage Eclipse](#)
 - [Débuggage](#)
 - [La documentation PhpDoc avec doxygen](#)
 - [Intro](#)
 - [Installation du plugin Eclox](#)
 - [Paramétrage et génération de la documentation](#)
 - [Framework de test unitaire : SimpleTest](#)
 - [Intro](#)
 - [Téléchargement de SimpleTest](#)
 - [Installation du plugin pour Eclipse](#)
 - [Création de tests unitaires](#)
 - [Exécution des tests](#)
 - [Outil de suivi des versions \(ou gestion de configuration logicielle, terme plus pompeux...\)](#)
 - [Intro](#)
 - [Installation plugin](#)
 - [Création d'un repository](#)
 - [Utilisation pratique de subversive](#)
 - [Tracker de bugs](#)
 - [Intro](#)

- [Myllyn](#)
- [Installation de Mantis](#)
 - [Téléchargement et installation](#)
 - [Paramétrage de Mantis](#)
- [Installation du plugin Eclipse pour Mantis](#)
- [Utilisation de Mantis avec Mylyn](#)
 - [Report d'un bug](#)
 - [Correction du bug](#)
- [Parallèle avec Synergy \(Outil Telelogic/IBM qui coûte un bras...\)](#)
- [Bilan](#)
- [Aller un peu plus loin avec Eclipse](#)
 - [Coloration syntaxique et thèmes](#)
 - [Annotations dans le code](#)
 - [Complétion automatique](#)
 - [Templates](#)
 - [Folding](#)
 - [Sources déportées](#)
 - [Cas de sources déportées locales](#)
 - [Cas de sources déportées distantes](#)
- [Sources](#)
- [Conclusion](#)

Introduction

Dans ce tutoriel, nous allons voir comment déployer tout un environnement de développement pour faire du Web avec Php avec uniquement des solutions open-source gratuites. Php peut paraître un langage peut rigoureux (pas typé, pas compilé) en comparaison à JavaEE, et je vous l'accorde il a ses défauts. On peut cependant l'utiliser dans les meilleures conditions pour travailler proprement et rester plus agile qu'avec des technos plus lourdes !

Cet environnement que nous allons créer va débiter par la configuration spécifique d'un IDE pour Php (Eclipse avec son plugin PDT). À ce stade, nous pourrions déjà pisser du code (pour ceux qui adorent cette expression...)! Nous installerons et paramètrons XDebug dans la foulée pour pouvoir débiter nos scripts Php. Nous poursuivrons avec l'installation de Doxygen pour sortir des documentations PHP aux petits oignons...

Qui dit code, dit « test unitaire » si on veut faire mieux que son petit neveu de 14 ans ! Pour cela nous allons utiliser le framework SimpleTest. Ce framework (comme beaucoup de framework de test unitaire dans diverses technos) s'utilise de manière similaire à JUnit (framework Java de référence).

Nous sommes certes des supers développeurs mais nous restons des hommes, il nous arrive donc de produire des bugs... Nous installerons donc un trackeur de bug (Mantis) qui nous permettra de tracer tous nos bugs, d'affecter des tâches correctives à des équipiers et d'historiser les résolutions.

Enfin, pour la gestion des releases de notre appli Web, nous utiliserons SVN directement à partir d'Eclipse via le plugin Subversive.

Allez fini le blabla, on s'y colle...

Tutoriel

Installation Eclipse et PDT

Note : Cette première section est volontairement très très détaillée afin que les plus débutants puissent appréhender et découvrir Eclipse. Si vous connaissez déjà cet IDE, rendez-vous directement à la rubrique [suivante](#).

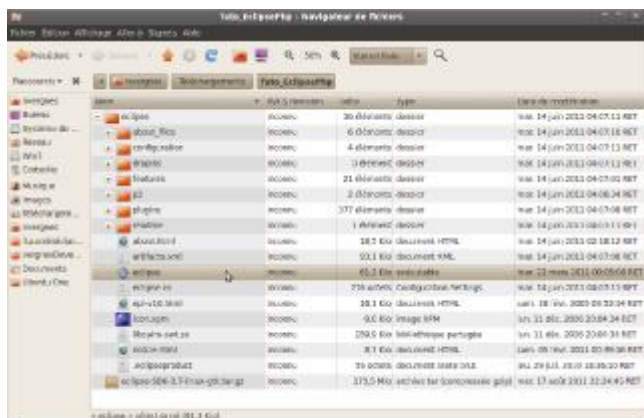
Téléchargement Eclipse

Pour commencer, téléchargeons la dernière release d'Eclipse : Indigo. Cette fois, la fondation a eu la flemme de nous proposer un bundle Eclipse+PDT comme à l'accoutumée. Il faut donc prendre Eclipse Classic (par exemple) et ajouter soi même le plugin PDT. Pour les néophytes, Eclipse Classic est la base de l'IDE qui ne gère que Java. Pour étendre Eclipse à d'autres langages ou technos, il faut ajouter des plugins. PDT est un plugin qui permet l'intégration de la techno Php dans Eclipse, pour le C/C++ c'est CDT, etc.

On va donc commencer par prendre Eclipse Classic [ICI](#). Pensez à bien choisir la version qui correspond à votre OS.



On décompresse l'archive téléchargée. Pour lancer le soft, on rentre dans le dossier d'Eclipse et on cherche l'exécutable qui s'appelle « eclipse », on s'en serait douté...



Il n'y a rien à installer ! Vous déplacez votre répertoire à votre guise sans rien reparamétrer. Vous pouvez ainsi sauvegarder très facilement votre IDE quand vous aurez passé des heures à le customiser après ce tutoriel !

Au lancement, Eclipse vous demande dans quel workspace vous désirez travailler. Tous vos projets seront enregistrés dans votre workspace. Vous pourrez toujours changer de workspace par la suite mais sachez que pas mal d'éléments de configuration d'Eclipse sont enregistrés dans le workspace (vos perspectives par exemple). Du coup, changer de workspace signifie souvent du reparamétrage...

Installation de PDT

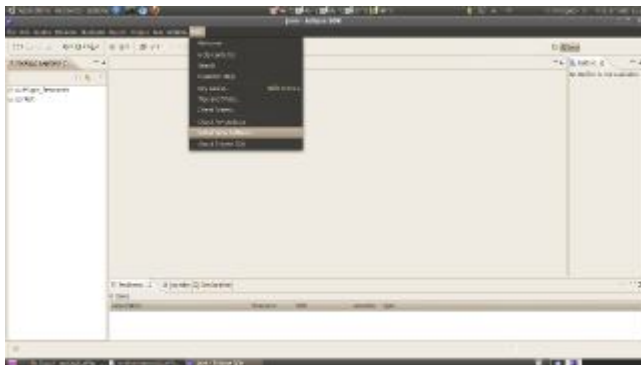
Pour installer un plugin, il existe plusieurs méthodes.

La plus simple est d'utiliser un « Update Site ». On indique simplement une url qu'Eclipse va aller explorer. Ensuite il va vous proposer l'ajout du plugin. C'est la méthode la plus propre car vous pouvez très facilement désinstaller le plugin en allant dans le même menu et en choisissant la désinstallation.

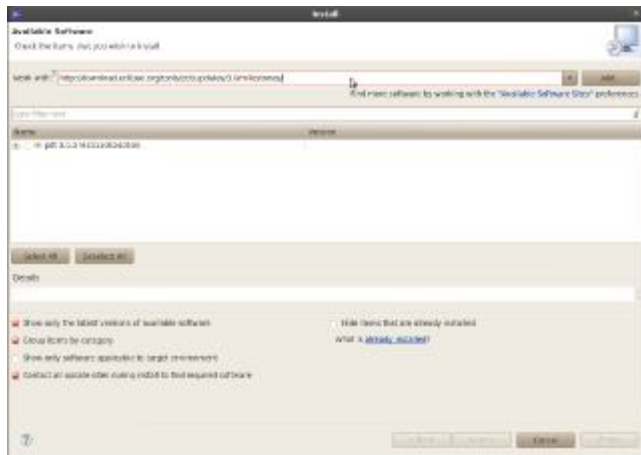
Deuxième méthode : vous téléchargez le plugin sous forme d'archive, vous lancez Eclipse et vous faites comme précédemment en remplaçant l'url par le chemin local de l'archive sur votre PC. On ne pas faire autrement si aucune url Update Site n'est disponible.

Enfin, on peut télécharger le plugin et aller insérer les fichiers directement dans l'arborescence d'Eclipse. Il suffit de fermer et rouvrir le soft pour que les modifications soient prises en compte. Cette méthode est à déconseiller car vous risquez de pourrir votre IDE...

Nous allons utiliser la méthode 1, c'est à dire à partir d'une l'url « Update Site ».



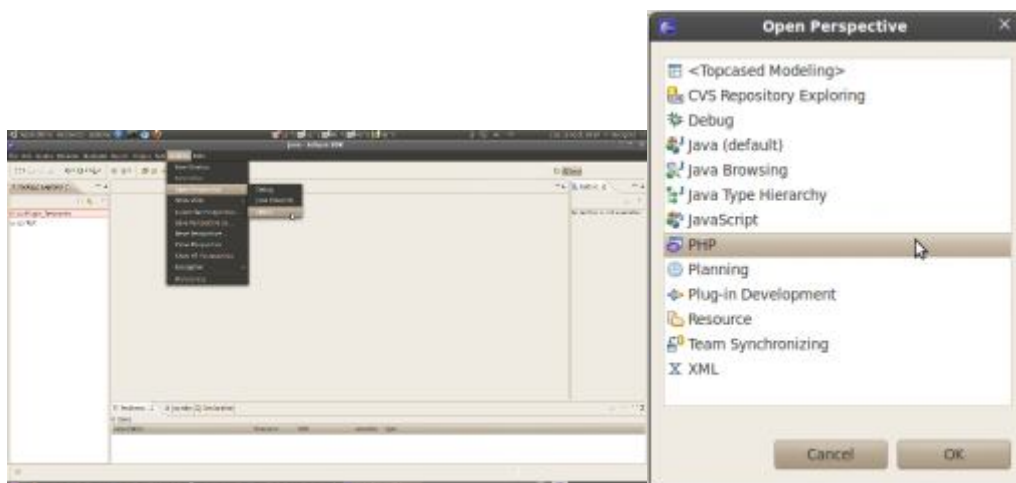
On saisi l'url suivante : <http://download.eclipse.org/tools/pdt/updates/3.0/milestones/> puis on clique sur « Add ». On coche ensuite tous les modules de PDT.



Le téléchargement peut prendre quelques minutes puis l'installation se déroule... Précisez que vous acceptez les termes de la licence pour achever l'installation.

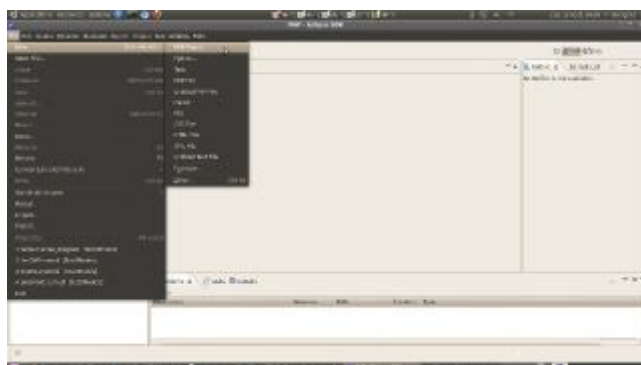
Après chaque modification de la configuration d'Eclipse, il faut fermer et relancer le soft afin que les nouveaux modules soient pris en compte.

À présent Eclipse est capable de gérer le PHP mais pour autant nous sommes encore sur la perspective par défaut dédiée à Java. Ouvrons donc la bonne perspective.



Création d'un projet Php

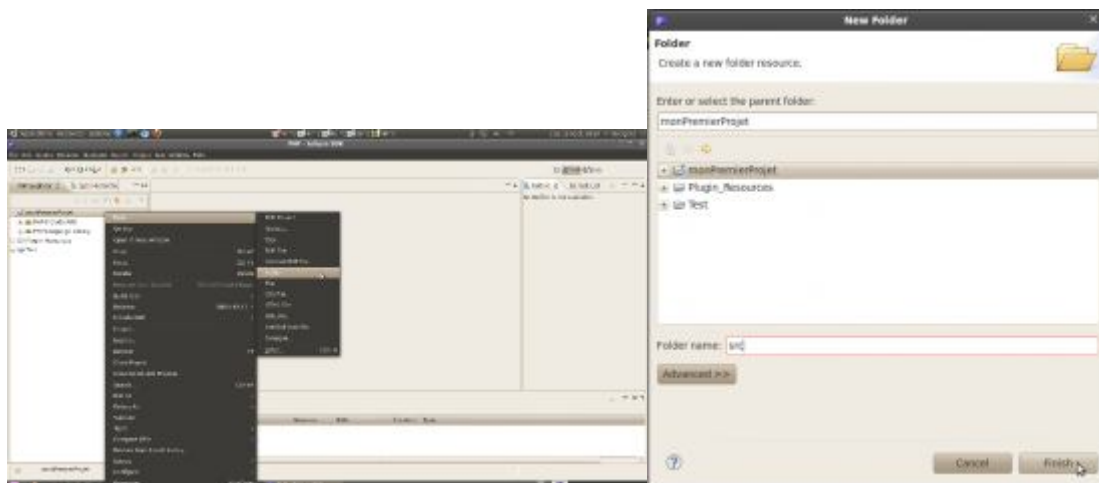
Nous sommes à présent prêt pour commencer à coder en PHP ! Créons un nouveau projet.



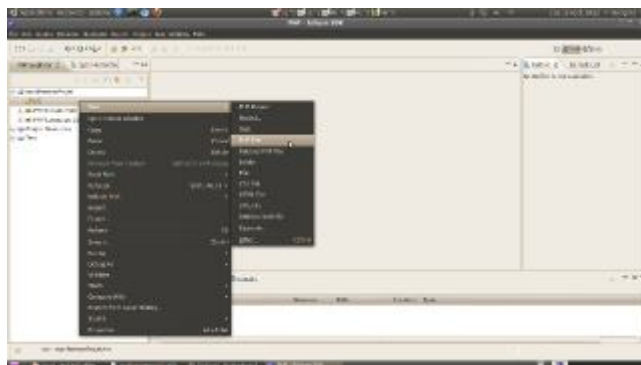
Le projet doit être nommé puis paramétré. Pour l'instant, nous allons choisir un nom puis faire directement « Finish ». Nous aurons donc tous les paramètres par défaut. Nous les modifierons un peu plus tard...



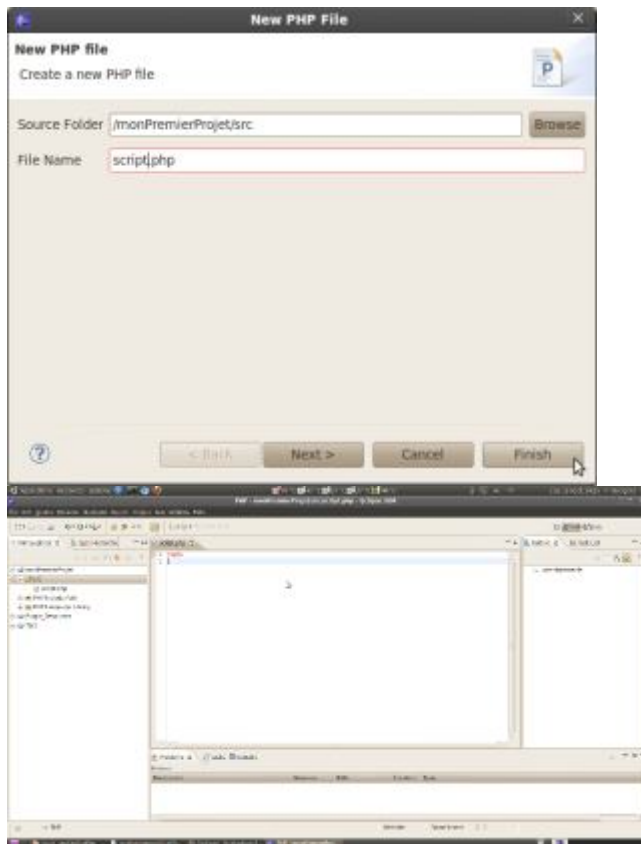
Le projet est alors créé mais il est vide. Ajoutons un répertoire « src » dans lequel nous placerons nos scripts PHP. Pour cela, click droit sur le projet, puis : New / Folder



Créons notre premier script. Pour cela, click droit sur notre nouveau répertoire /src, puis: New / PHP File



On peut éventuellement modifier son emplacement si on s'est trompé lors du click droit. On spécifie un nom de fichier et on termine.



On remarque qu'Eclipse nous a ouvert la balise du PHP mais ne l'a pas fermé... Allez comprendre...

Dans le cadre ce tuto, afin d'avoir un peu de matière à faire joujou, nous allons coder une première classe PHP. Voici le code :

```
1 <?php
2
3 /**
4  * Classe Robot qui permet de simuler le comportement d'un robot.
5  * @author nvergnes
6  * @version 1.0
7  */
8 class Robot {
9     /**
10      *
11      * Age du robot
12      * @var integer
13      * @access private
14      */
15     var $age;
16
17     /**
18      *
19      * Nom du robot
```

```

18     * @var string
19     * @access private
20     */
21     var $nom;
22
23     /**
24     * Constructeur
25     * @param integer $age L'âge du robot
26     * @param string $nom Le nom du robot
27     */
28     function Robot( $age, $nom ) {
29         $this->age = $age;
30         $this->nom = $nom;
31     }
32
33     /**
34     * Méthode pour faire parler le robot.
35     */
36     function parle() {
37         echo "Bonjour, je m'appelle $this->nom. J'ai déjà $this->age ans !
38     }
39
40 // construction de 2 robots
41 $robot1 = new Robot(10, 'R2-D2');
42 $robot2 = new Robot(15, 'C-3PO');
43
44 // à vous les robots, parlez !
45 $robot1->parle();
46 $robot2->parle();
47
48
49
50
51
52
53
54
55

```

Au passage, notez que j'ai tout commenté au standard PhpDoc pour anticiper sur la suite du tuto... Ce formalisme est très proche de la célèbre JavaDoc. Vous pouvez visiter ce site pour en savoir plus : <http://manual.phpdoc.org/HTMLframesConverter/phpdoc.de/>.

Interprétation des scripts Php

Pour tester ce script, on a besoin d'un serveur Web de type LAMP (pour Linux) ou MAMP (pour MAC) ou WAMP (pour Windows). Je suppose que vous en avez déjà installé un sur votre machine, sinon direction Google!

Actuellement notre script se trouve dans le workspace Eclipse. Nous voudrions pourtant pouvoir exécuter notre Php... Pour cela plusieurs solutions. On peut paramétrer Apache pour lui faire interpréter des fichiers dans des répertoires particuliers. Je ne vais pas utiliser cette méthode car Apache n'est pas l'objet du tuto. Nous allons faire au plus simple : créer un lien symbolique de notre répertoire « src » vers notre répertoire « www » qui lui est bien interprété par Apache.

Pour faire un peu geek 😊, on va faire ceci en ligne de commande :

```
1 ln -s /home/nvergues/workspace/monPremierProjet/src/  
/var/www/monPremierProjet
```

Note : nvergues, c'est moi ! Pensez donc à remplacer nvergues par le nom de votre home directory ! 😊

Attention : la commande « ln » nécessite des chemins absolus. Les chemins relatifs ne fonctionnent pas !

Remarque : Pour les autres OS que Linux, nous aurions pu faire d'une autre façon. Nous aurions pu créer le script directement dans le répertoire www et indiquer à Eclipse que les sources étaient déportées. Consultez le paragraphe [Cas de sources déportées locales](#) et vous pourrez ainsi continuer le tuto 😊. Dans tous les cas, n'hésitez pas à poser des questions dans le fil de commentaires au besoin.

Je préfère la méthode que je vous ai montré pour limiter www à des liens symboliques et garder toutes les sources dans ma home directory, ceci sur ma station de développement.

Sur <http://localhost> on peut voir notre répertoire contenant le script.



Il suffit de cliquer sur le fichier « script.php » pour faire interpréter notre code. Voici le résultat :



Félicitation, nous venons de paramétrer Eclipse pour PHP et nous sommes en mesure d'exécuter nos scripts! C'est un bon début...

Mise en place du débogger

Intro

Nous allons mettre en place le débogger Xdebug. Sachez qu'il existe également Zend-debugger. Je n'ai jamais utilisé ce dernier, je ne pourrais donc pas vous en faire un comparo...

Premièrement, il faut installer Xdebug sur son OS. Pour Ubuntu suivez ce tuto : http://wiki.ubuntu-fr.org/eclipse_php_xdebug#installation. Pour les autres : Google ! 😊

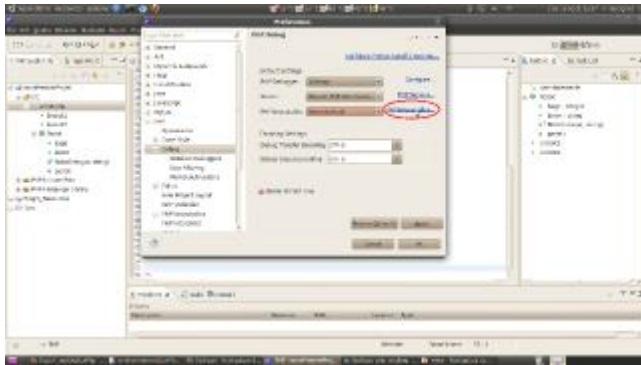
Paramétrage Eclipse

Une fois installé, on va paramétrer Eclipse. Il faut aller dans Window / Preferences. Puis PHP / Debug.

Changez « Zend Debugger » par « Xdebug ».



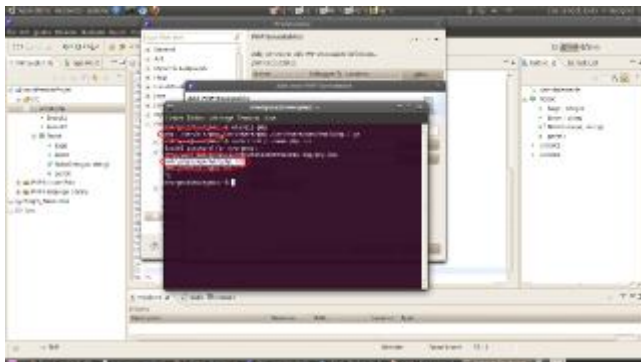
Cliquez ensuite sur « PHP executables »



Configurez alors comme sur la figure suivante (si vous êtes sur un Linux, sinon : direction => Google...) :



Note : Pour les chemins, ils peuvent potentiellement être différents chez vous. Pour vérifier, faites un petit « whereis php » puis un « sudo find / -name php.ini ».



Débuggage

Après validation, on ouvre la perspective dédiée au débbugage : Window / Open Perspective / Debug

On peut placer un breakpoint : double click dans la marge à gauche de la ligne sur laquelle vous souhaitez placer le point d'arrêt.

Ensuite on va dans : Run / Debug Configurations... On sélectionne « PHP Web Page » et on clique sur « New launch configuration » en haut à gauche. Paramétrer ensuite comme sur les figures suivantes :

Puis pensez à valider avec : « Apply »

Note : Adaptez au besoin si vous n'avez pas fait comme moi, ce n'est pas très compliqué. Voici un screenshot du debuggage :

Note : Par défaut, on utilise le navigateur Web interne d'Eclipse. Il est tout à fait possible, ou plutôt souhaitable, de paramétrer votre favori. Pour ma part, je travaille avec Chromium (version Open Source de Chrome, le navigateur de Google) pour développer. J'utilise par contre Firefox pour les tests de sécurité avec des plugins comme « SQL inject me », « Hackbar », etc.

Pour paramétrer votre navigateur : Window / Preferences, puis General / Web Browser.

Bon ce n'est pas si difficile de debugger des scripts PHP?

La documentation PhpDoc avec doxygen

Intro

La documentation est une étape super importante pour pérenniser le projet. Même quand on bosse freelance, il faut souvent se remettre dans de vieux projets. Et là... merci les commentaires ! Si on a bossé avec une doc PhpDoc carrée, on peut explorer tout le projet à partir d'une doc générée html par exemple. On peut voir les graphes d'appel, la hiérarchie des classes, etc. C'est vraiment très utile!

Je vous montre doxygen car je l'ai souvent utilisé pour le C et C++. Il gère pas mal de langages. Je ne sais pas si c'est le mieux pour PHP. À vous de voir s'il vous convient 😊.

On télécharge doxygen ICI : <http://www.stack.nl/~dimitri/doxygen/download.html#latestsrc>.

Il faut ensuite installer doxygen en suivant la procédure et suivant votre OS. Suivez les indications de cette page : <http://www.stack.nl/~dimitri/doxygen/install.html>.

Installation du plugin Eclox

On doit ensuite ajouter un autre plugin à Eclipse : Eclox pour exploiter doxygen directement à partir de l'IDE. On utilise l'update site : <http://download.gna.org/eclox/update>

On choisi Eclox ou Hot-Eclox. Je n'ai pas vu trop de différence entre les 2... Hot-Eclox doit simplement être une version Beta. À vous de voir :-p.

On relance Eclipse pour charger le plugin. Je passe rapidement car on a déjà vu l'installation d'un plugin précédemment...

Paramétrage et génération de la documentation

Vous devez normalement visualiser un nouveau bouton (un @) dans Eclipse.

Justement, on va cliquer dessus . On choisit Yes pour créer le doxyfile. C'est le fichier qui va comporter nos paramètres de la doc.

On édite ensuite le fichier `.doxyfile`.

Ce fichier est au format xml mais le plugin Eclox comportant un xsl, par magie, Eclipse vous affiche le paramétrage sous forme plus sympathique qu'un xml brut 😊. Voici un exemple de paramétrage, à vous de choisir votre format de sortie et vos options. J'ai créé un répertoire « doc » au même niveau que « src ». C'est ce nouveau répertoire qui va recevoir tous les fichiers générés par doxygen.



Note : J'ai choisi le langage Java de la rubrique « Optimize results for ». En effet la JavaDoc et la PhpDoc sont vraiment très proches...

J'ai opté pour du html en sortie. Voici la documentation de notre super classe de ouf !



Vous pouvez naviguer dans tout le projet grâce aux hyperliens. Je trouve cette doc au top ! 😊
(J'en fais pas trop ?)

Framework de test unitaire : SimpleTest

Intro

Bon on a déjà une bonne base mais on va faire encore mieux !

Les framework de tests unitaires permettent d'écrire des tests qui vont être placés directement dans le projet. On va créer un nouveau répertoire à la racine du projet. Il y aura désormais : src, doc et tests.

Les tests feront partie intégrante du projet, ils seront gérés exactement comme le code, c'est à dire : correctement implémentés (pas à l'arrache), gérés en version avec svn (nous verrons ceci plus loin dans le tuto), commentés, etc.

Nous allons utiliser SimpleTest. J'ai choisi ce framework car il ressemble fortement à JUnit (framework de référence Java) et que CakePhp l'utilise également. Il est écrit en Php, du coup rien à installer !

Téléchargement de SimpleTest

On télécharge SimpleTest [ICI](#).

Décompressez l'archive dans votre projet courant (c'est la méthode la plus simple). Vous allez donc avoir l'arborescence suivante :



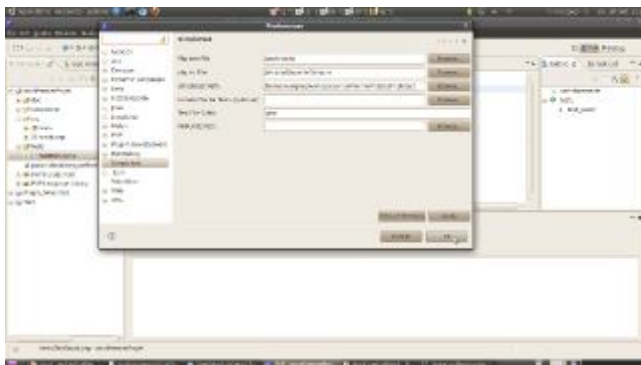
Note : Quand vous modifiez à la main la structure du projet Eclipse, pensez à rafraîchir l'explorateur de projet d'Eclipse. Pour cela, click droit sur le projet, puis: Refresh.

Installation du plugin pour Eclipse

Ensuite, ajoutons le plugin dans Eclipse à partir de l'update site suivant :
<http://simpletest.org/eclipse/> (je passe vite, on a déjà installé plusieurs plugin...)



On paramètre ce plugin : Window / Preferences puis SimpleTest



Création de tests unitaires

Dans notre répertoire « tests », nous allons créer un fichier TestRobot.php qui aura la responsabilité de tester la classe php « Robot ». J'ai choisi cette convention de nommage : TestMaClasseATester.php afin de pouvoir facilement retrouver les fichiers liés aux tests unitaires. Certains préfèrent marquer les fichiers comme suit : MaClasseATester.test.php. À vous de voir.

Sachez que si vous souhaitez aller encore plus loin avec de l'intégration continue, il est plus simple de pouvoir parser des fichiers à l'aide de conventions de nommage...

Voici le code de test :

```
1 <?php
2
3 // Il faut inclure ce fichier pour utiliser SimpleTest
4 // Attention de bien spécifier le chemin relatif par rapport au
5 répertoire courant, c'est à dire /tests
6 require_once '../simpletest/autorun.php';
```

```

7
8 // Il faut inclure le fichier à tester
9 require_once '../src/script.php';
10
11 /**
12 *
13 * Classe de test de la classe Robot
14 * @author nvergues
15 */
16 class TestRobot extends UnitTestCase {
17     /**
18     *
19     * Test du constructeur. On s'assure de la bonne initialisation.
20     */
21     function test_constructeur() {
22         // Création d'un robot nommé R2-D2 âgé de 10 années
23         $robot1 = new Robot( 10, 'R2-D2' );
24
25         // On vérifie la bonne allocation de l'instance de Robot
26         $this->assertNotNull( $robot1 );
27
28         // On teste si l'âge est OK
29         $this->assertEqual( 10, $robot1->age );
30
31         // On teste si le nom est OK
32         $this->assertEqual( 'R2-D2', $robot1->nom );
33     }
34 }
35
36
37

```



Exécution des tests

À ce stade si tout se passait dans le meilleur des mondes, on suivrait la procédure du site Web de SimpleTest et on aurait la même simplicité qu'avec JUnit. Malheureusement, le plugin est buggué et on ne parvient pas à exécuter un Run Eclipse de type SimpleTest. C'est ultra dommage car les résultats s'afficheraient dans une vue de la perspective PHP plutôt pratique: « Result View ». (Voir le bas de la capture d'écran précédente).

On obtiendrait les résultats exactement comme sur la figure suivante où l'on peut voir un exemple de JUnit : <http://www.cs.put.poznan.pl/dweiss/site/projects/junit-tomcat/tests.gif>.

Il va donc falloir faire un peu plus à l'ancienne 😊 !

Nous allons créer une config de Run de type « PHP Web Page » et s'en contenter...

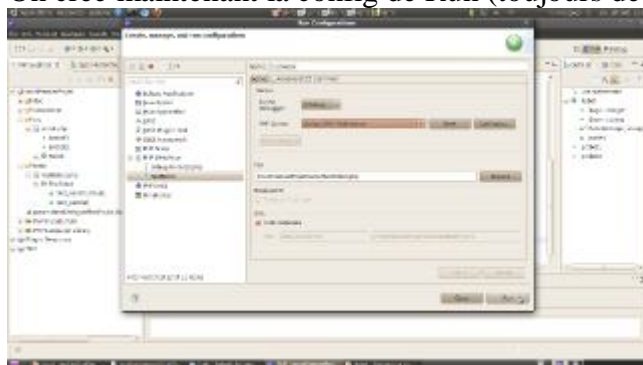
Nous avons le même problème qu'au départ : les tests ne se trouvent pas dans le répertoire par défaut du serveur Apache, à savoir www. Nous allons refaire le lien symbolique précédent de manière un peu différente :

```
1 rm /var/www/monPremierProjet # On détruit l'ancien lien symbolique
2 ln -s /home/nvergnes/workspace/monPremierProjet/ /var/www/monPremierProjet #
   On crée un lien de tout le projet vers www
```

On peut voir la différence sur la figure suivante. Ce n'est plus le contenu de /src qui est visible dans www mais c'est l'ensemble du projet Eclipse (le code utile, les tests, le framework, la doc)...



On crée maintenant la config de Run (toujours de type PHP Web Page) :



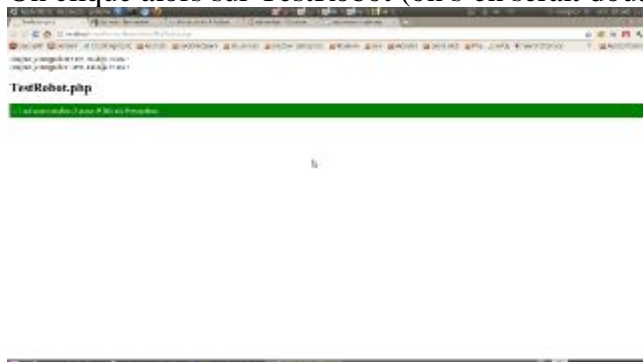
On lance le Run () à partir du picto vert qui ressemble à un « play » :



Il faut cliquer juste à sa droite sur le triangle pour dérouler le menu suivant :



On clique alors sur TestRobot (on s'en serait douté!). Voici les résultats :



On remarque tout en haut (avant le rapport de SimpleTest) un affichage parasite qui provient de l'interprétation du fichier script.php. Et oui, il ne faut pas oublier que pour tester la classe Robot, nous avons inclus script.php dans TestRobot.php ! Or script.php contient la définition d'une classe mais aussi des appels à la fin du fichier. Dans un cas normal, le fichier ne contiendrait que la définition de la classe, on aurait donc directement le rapport.

Ensuite vient le compte rendu de SimpleTest. Nous voyons que nous avons joué un cas de test unitaire comportant 3 tests réussis : d'où la couleur verte.

Je ne vais pas faire un long blabla sur les tests unitaires car ce n'est pas l'objet du tuto. Allez voir la doc [ICI](#).

Note : Dans l'exemple, je vous l'accorde le test ne sert pas à grand chose en l'état. Mais imaginez que dans le futur vous deviez ajouter un constructeur à 3 arguments. Php permettant un nombre d'arguments variable, vous allez sans doute intervenir dans l'ancien code de la

classe Robot. Dès lors, vous n'avez aucune garantie de ne pas fracasser l'ancien code ! Cet ancien code n'est peut être même pas de vous ! Votre seul garde fou sera les test unitaires qui ont été joints au projet à l'époque. Vous allez rejouer les tests pour être certain de ne pas avoir introduit de régression sur le code de votre collègue. J'espère que vous en comprenez mieux l'importance maintenant 😊.

Outil de suivi des versions (ou gestion de configuration logicielle, terme plus pompeux...)

Intro

Je sais que la mode est aujourd'hui à Git mais je vais quand même vous faire utiliser le bon vieux SVN. Git est un bon outil mais très spécifique et je ne suis pas certain que tous ses utilisateurs aient une justification à utiliser git plutôt que svn... Je préfère avoir mon repository bien centralisé sur un seul serveur... SVN me semble vraiment moins casse-gueule mais c'est juste mon ressenti perso...

Ok, nous avons déjà un environnement bien complet pour bien travailler! Cela dit si nous travaillons à plusieurs, il nous faut bien s'organiser. Dans tous les cas, il faudra bien gérer les versions de notre appli Php (et ses éventuels forks). Ça tombe bien, SVN nous apporte la solution 😊.

Je ne vais pas vous faire un cours sur SVN, je vous invite à lire ceci au besoin : http://dev.nozav.org/intro_svn.html ou <http://svnbook.red-bean.com/nightly/fr/svn-book.pdf>.

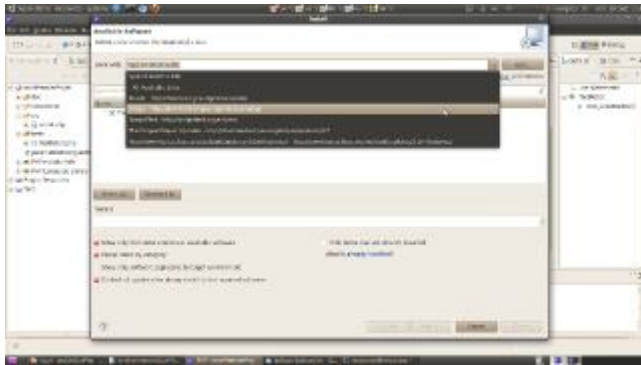
Petit rappel tout de même sur le jargon SVN :

- **trunk (tronc)** : On y travaille normalement, c'est ici que se trouve les sources en cours de dev
- **branch (branche)** : On y travaille pour tester quelques modifs scabreuses pour ne pas pourrir le tronc. On fusionnera les modifs si la travail abouti et si la solution testée est retenue. On y travaille également si on fait un fork du logiciel par exemple.
- **tag (étiquette)** : C'est un point marquant dans le projet (release, milestone, jalon, etc.)

Je suppose que vous avez un serveur SVN sous la main, soit sur votre machine, soit sur une machine distante. Je vais utiliser un serveur SVN local dans le cadre de ce tutoriel. En pratique, je travaille via ssh sur un serveur distant. Si vous souhaitez plus d'infos, posez moi des questions dans le fil de commentaires à la fin de ce tuto 😊.

Installation plugin

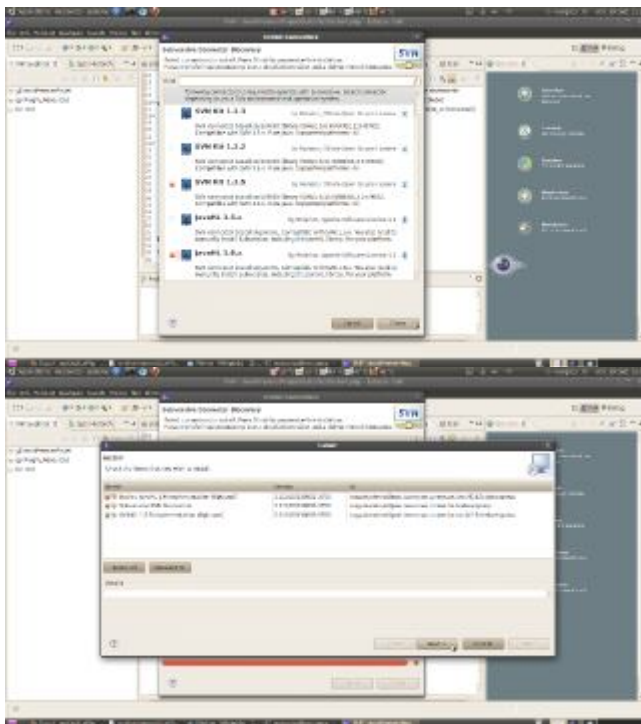
Commençons par installer le plugin svn pour Eclipse. J'ai choisi Subversive plutôt que Subclipse car il m'a semblé moins limité (notamment sur les schéma des historiques des fichiers et les graphes). On va se rendre dans les dépôts officiels de Eclipse Indigo : Help / Install New Software...



On déroule alors la rubrique Collaboration. On va cocher plusieurs modules :

- Mylyn Context Connector : Team Support
- Subversive revision graph
- Subversive SVN Integration for the Mylyn Project
- Subversive SVN Team Provider

Après installation et redémarrage nécessaire d'Eclipse, on vous propose d'installer SVN Kit et JavaHL. Paramétrez comme sur les figures suivantes :



Redémarrage à nouveau d'Eclipse... On va ouvrir d'autres perspectives que Php et Debug afin de pouvoir utiliser SVN : Window / Open Perspective / Other...

Vous devriez voir de nouvelles perspectives : SVN Repository Exploring et Team Synchronizing



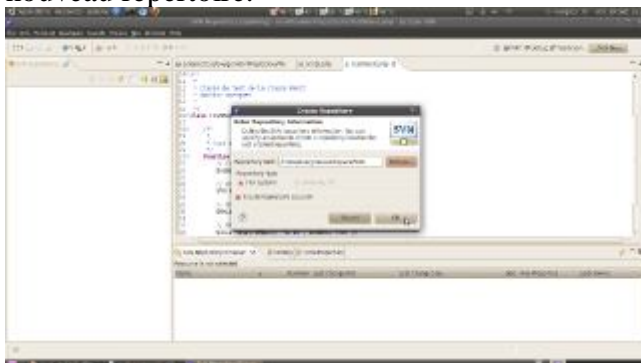
Ce sont ces perspectives qui nous permettront de faire les synchronisations entre fichiers locaux du projet et les dépôts svn, de gérer les connexions à des dépôts, etc.

Création d'un repository

On va créer un nouveau repository (dépôt) en local. Pour cela on ouvre la perspective Repository Location, et on choisit « New Repository ».



On choisit l'emplacement de ce dépôt. J'ai créé un nouveau répertoire qui va accueillir notre dépôt svn dans /workspace qui contient déjà notre projet Eclipse. J'ai nommé « SVN » ce nouveau répertoire.



On peut alors voir le nouveau dépôt, la révision courante et son contenu.



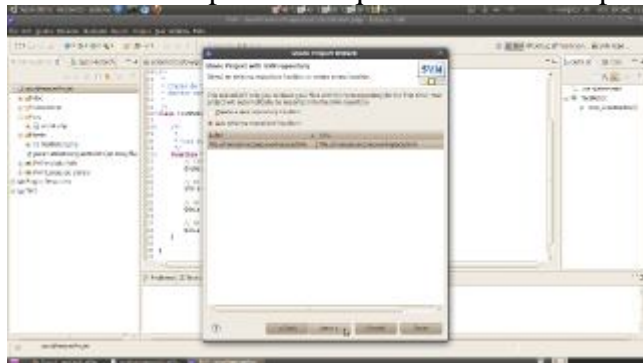
On va devoir importer notre projet /workspace/monPremierProjet dans ce dépôt. Pour cela, on retourne sur la perspective Php, on fait un click droit sur la racine de notre projet Php. On choisit Team / Share Project



On choisit le type de gestionnaire : SVN ou CVS. On utilise SVN.



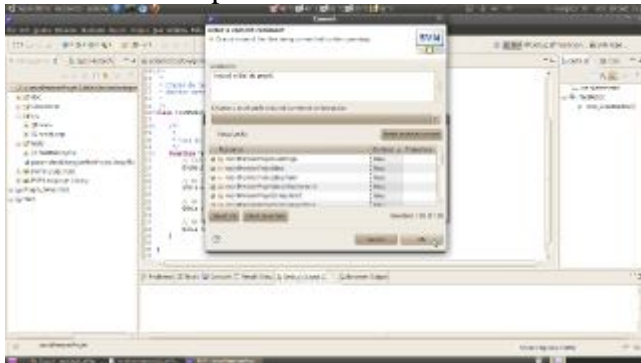
On choisit le dépôt dans lequel on souhaite importer notre projet et tout son contenu.



On s'authentifie si nécessaire. Dans notre cas, nous n'avons rien protégé par mot de passe ou clé RSA, donc on passe cette étape en cliquant sur OK sans rien remplir. Quand on travaille avec ssh+svn (c'est à dire sur un dépôt distant), il faut saisir ses paramètres d'authentification. N'hésitez pas à me questionner sur ce point si nécessaire...



On pense à toujours commenter son action puisqu'elle est tracée et qu'on doit pouvoir comprendre ce qui a été modifié dans le repository ! On coche tous les fichiers puisqu'on souhaite tout importer.



Note : Dans un vrai cas de figure, nous aurions pu choisir de ne pas importer la doc du projet dans le repository car elle est générée automatiquement... À vous de voir 😊...

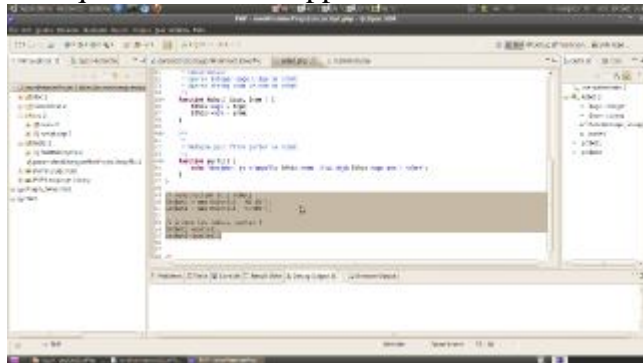
Retour dans la perspective Php, on voit alors dans l'explorateur de projet que tous nos fichiers possèdent une release ! Le projet est de plus identifié par le nom du repository. Pas mal non?



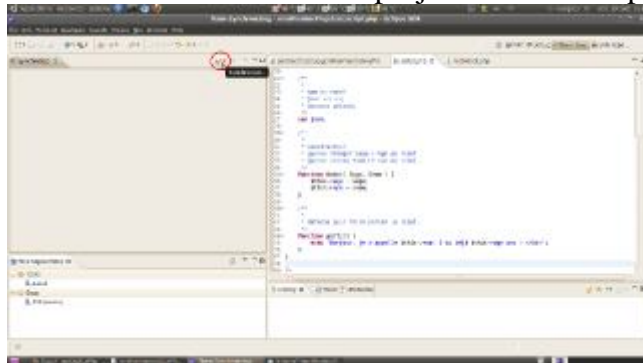
Utilisation pratique de subversive

Nous allons maintenant utiliser Subversive au travers d'un exemple pour mieux comprendre son fonctionnement.

Nous allons ouvrir notre script.php et retirer les appels après la fin de définition de la classe Robot. Nous ne conservons que la classe Robot dans ce fichier. La figure qui suit montre le bloc que nous allons supprimer:



Après cette modification, on peut visuellement voir que le fichier n'est plus synchronisé avec son dépôt. Le signe « > » apparaît devant les répertoires et fichiers désynchronisés. Allons dans la perspective « Team Synchronizing » qui permet justement de tester ce qui a été modifié entre les fichiers du projet et ceux du dépôt. On clique sur « Synchronize ».



Puis SVN / Next. On laisse tout coché puisqu'on souhaite tester la synchro pour tout le projet, on termine...



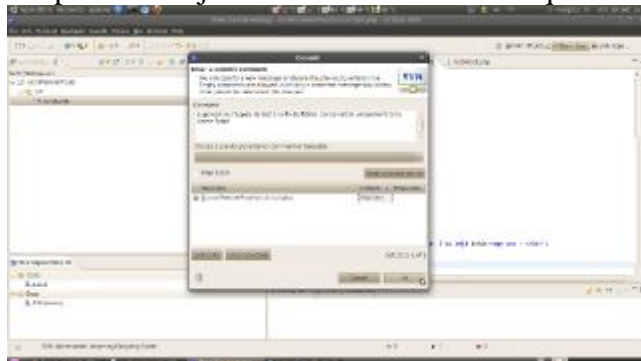
On voit alors apparaître le fichier qui n'est plus en phase avec le dépôt !



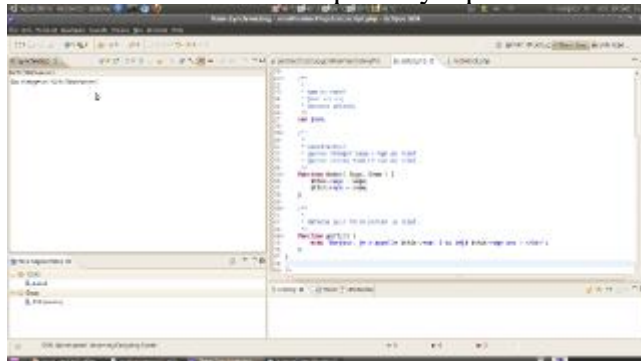
On a donc faire le « commit » pour accepter les modifications et les importer dans le dépôt.
On fait click droit sur le fichier, et on fait le commit.



On pense à toujours commenter la modif qui va intégrer le dépôt !



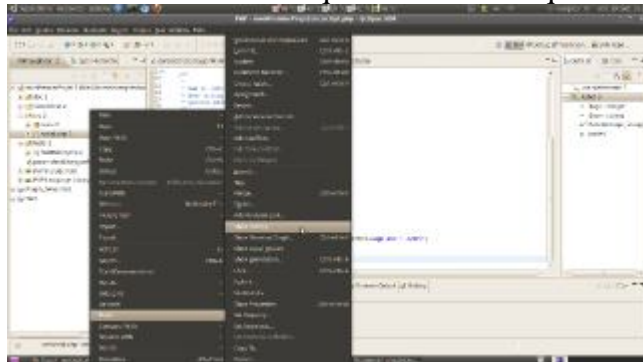
On valide. On voit alors qu'il n'y a plus de fichiers désynchronisé.



Retour sur la vue Php, on peut alors voir le nouvel indice de révision du fichier script.php. Il est passé à la révision 3.



Si on fait clic droit sur le fichier script.php puis : Team / Show History... puis OK, on peut voir une vue de l'historique du fichier en question.

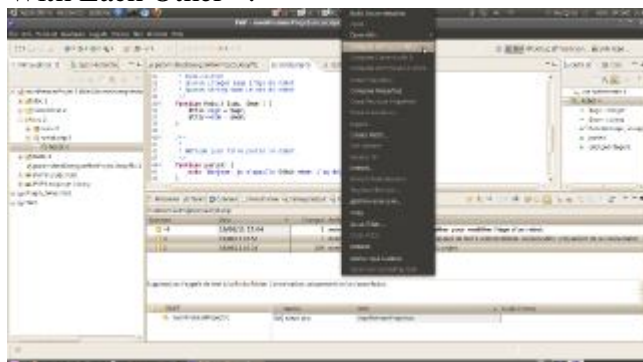


Voici l'historique :



On voit ici l'importance des commentaires !

Dernière opération très courante que nous allons voir : la comparaison de différentes versions d'un fichier. Pour cela, On peut afficher l'historique comme nous venons de le voir, puis on sélectionne 2 versions du fichier que l'on souhaite comparer, puis click droit et « Compare With Each Other ».



On voit alors un éditeur en colonne s'afficher :



On a à droite la révision 2 et à gauche la révision 3. Le plus récent est donc à gauche. On voit bien que nous avons supprimé le bloc d'appels de tests après la définition de la classe.

Nous n'allons pas aller plus loin avec subversive. Nous avons vu les bases d'une utilisation standard. Je vous laisse le soin de faire joujou afin de retrouver toutes les commandes SVN que vous connaissez déjà (dériver vers une branche, fusionner une branche dans le tronc, faire un checkout pour récupérer un projet dans un état donné, faire des tags, etc).

Note : Je vous conseille fortement de conserver tous vos plugins en anglais car sinon il est très pénible de s'y retrouver entre la traduction et le terme original. J'ai déjà testé subversive en français et je ne savais jamais si un import (terme français) était un checkout (terme anglais), un commit (terme anglais), un import (terme anglais), etc.

Je pense vous avoir montré le côté friendly de subversive. Ce plugin permet d'utiliser SVN avec une extrême simplicité. On n'a plus envie de réutiliser SVN en ligne de commande ! Maintenant, il vous faudra un peu de pratique pour ne pas perdre de temps dans les menus assez denses...

Tracker de bugs

Intro

Des bugs ? Comment ? Nous sommes d'excellents codeurs !

Certes mais l'erreur étant humaine, nous allons quand même installer un trackeur ! Et puis si vous êtes si exceptionnel, peut être vos collègues le sont-ils moins ?

J'ai choisi d'utiliser Mantis (<http://www.mantisbt.org/>) car c'est un soft écrit en Php. Encore du Php ??? Et oui, c'est pratique quand on n'est pas root sur un serveur !

Si vous souhaitez déployer votre tracker sur un mutu, vous n'avez pas les droits d'admin et vous êtes limités par la config. En général vous avez quand même toujours Php de dispo. Vous voyez l'intérêt ? 😊

Sinon, vous pouvez regarder ses concurrents: Bugzilla (écrit en Perl), Trac (écrit en python),

...

Mylyn

Avant d'aller plus loin, je dois faire une parenthèse sur Mylyn. Lorsque nous avons installé Subversive, nous avons installé les modules suivants :

- Mylyn Context Connector : Team Support
- Subversive SVN Integration for the Mylyn Project

Mylyn permet d'intégrer à Eclipse une gestion de tâches, ceci indépendamment des outils connectés. Par exemple, si aujourd'hui vous travaillez avec Mantis, demain vous passez à Trac, vous aurez toujours la même gestion de vos tâches puisque Mylyn ne changera pas ! De plus, tout est intégré et peut interagir : SVN, Mantis, etc...

Mylyn va nous permettre de se connecter au repository de Mantis. On pourra alors, comme dans Subversive, synchroniser ces dépôts.

Installation de Mantis

Téléchargement et installation

On télécharge Mantis [ICI](#).

Je prend la 1.2.6 qui est la dernière version à l'heure où je rédige ce tuto.

On extrait l'archive dans notre répertoire interprété par Apache : /www pour moi. J'ai renommé le répertoire en « mantis » (j'ai enlevé la version du nom).

On modifie les droits du répertoire et de tout son contenu (donc de manière récursive) :

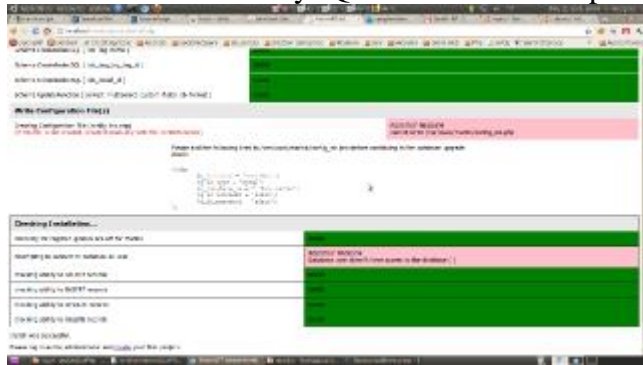
```
1]chmod -R 755 /var/www/mantis/ #pour les Linuxiens
```

On se rend sur <http://localhost/mantis> pour commencer l'install... On paramètre.



Puis : Intall/Upgrade Database

La base de données MySQL se crée et se remplit de tables mais une erreur apparaît.



Il faut créer un fichier à la main car nous avons laissé 755 en permission. Il aurait fallut utiliser 777 pour laisser toutes les libertés à Mantis...

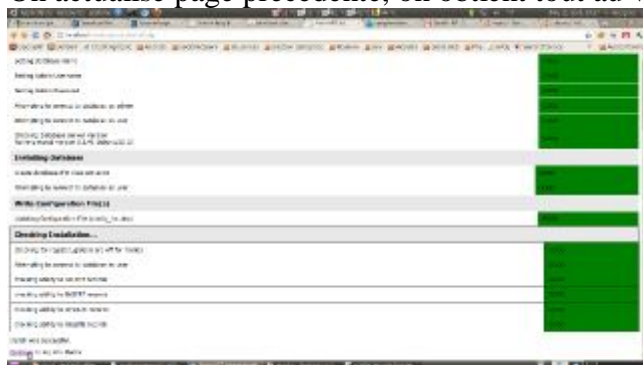
On crée le fichier : config_inc.php dans /mantis

Dedans, il faut modifier le premier bloc :

```
1# --- Database Configuration ---
2$g_hostname = 'localhost';
3$g_db_username = 'root'; # Le login MySQL
4$g_db_password = 'MySQLwinner'; # Le mot de passe MySQL
5$g_database_name = 'bugtracker';
6$g_db_type = 'mysql';
```

Remarque : Attention, dans le screenshot précédent, Mantis nous indique le bloc à copier et coller dans le fichier config_inc.php. Les variables \$g_db_username et \$g_db_password qu'il nous indique ne sont pas bonnes ! Ce sont les paramètres de votre serveur MySQL qu'il faut utiliser !

On actualise page précédente, on obtient tout au vert !



On clique en bas à gauche sur : Continue to log into Mantis



L'administrateur par défaut a été crée : login : administrator, password : root
On arrive alors sur l'admin de Mantis !



Paramétrage de Mantis

À ce stade, il serait impératif de changer le mot de passe par défaut du compte « administrator ». Pour cela : My Account... Pour le tuto, je ne vais pas faire ceci pour gagner 2min...

On va débiter par la création d'un nouvel utilisateur : Manage / Manage Users
Puis : Create New Account





Remarque : Je passe rapidement sur Mantis car notre but est de l'utiliser avec Eclipse. En cherchant, vous trouverez à tout paramétrer sans problème. Posez moi des questions si vous rencontrez des problèmes, mais je ne suis pas expert Mantis quand même 😊.

Il faudrait paramétrer le serveur mail smtp afin que Mantis puisse envoyer des notifications aux utilisateurs. Je passe cette étape...

On crée ensuite un projet : Manage / Manage Projects / Create New Project



Nous en avons fini pour le paramétrage pour l'instant. Je vous rappelle notre but : ne pas utiliser Mantis via ce client Web (que nous sommes en train d'utiliser) mais l'utiliser via Eclipse. Nous reviendrons sur ce client Web pour vérifier qu'Eclipse parvient bien à créer des « reports » Mantis.

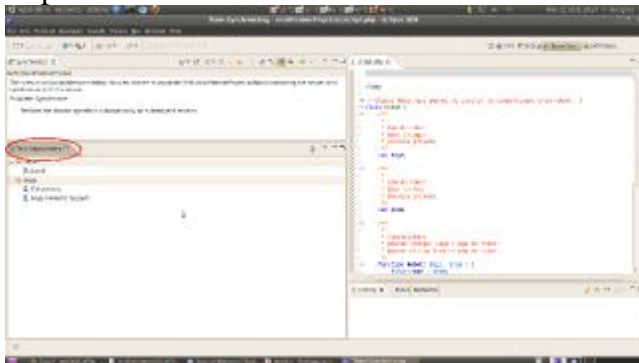
Installation du plugin Eclipse pour Mantis

Nous utilisons l'update site suivant : <http://mylyn-mantis.sourceforge.net/eclipse/update>

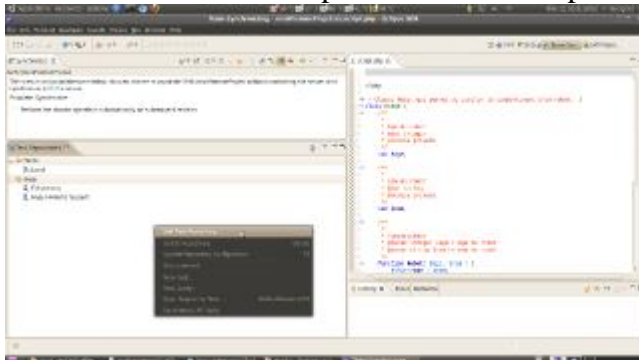


Comme d'hab, on redémarre Eclipse...

On ouvre la perspective « Team Synchronizing », et on a normalement une vue « Task Repositories »



Click droit sur cette vue puis : Add Task Repository



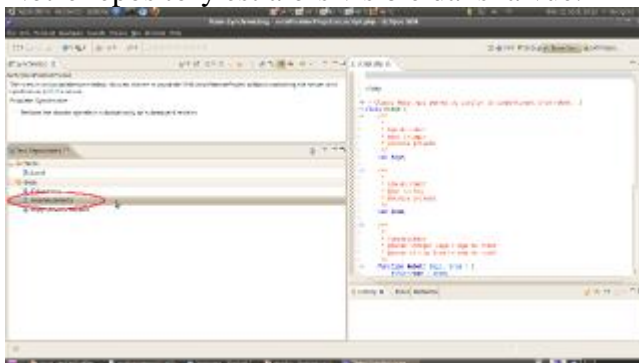
On voit qu'on a bien installé le connecteur pour Mantis, donc c'est OK.



En principe, je devrais utiliser par exemple « codeweblog » comme utilisateur mais le mot de passe n'a pas pu être envoyé car je n'ai pas paramétré le serveur mail. Du coup pour l'exemple, j'utilise « administrator »...



Notre repository est alors visible dans la vue.



Utilisation de Mantis avec Mylyn

Bon nous allons maintenant créer un bug manière d'avoir à utiliser notre tracker !

On modifie notre constructeur de la façon suivante :

```
1
2 /**
3  *
4  * Constructeur
5  * @param integer $age L'âge du robot
6  * @param string $nom Le nom du robot
7  */
8 function Robot( $age, $nom ) {
9     $this->age = 0;
10    $this->nom = $nom;
11 }
```

On s'est planté dans l'initialisation de l'âge...

Report d'un bug

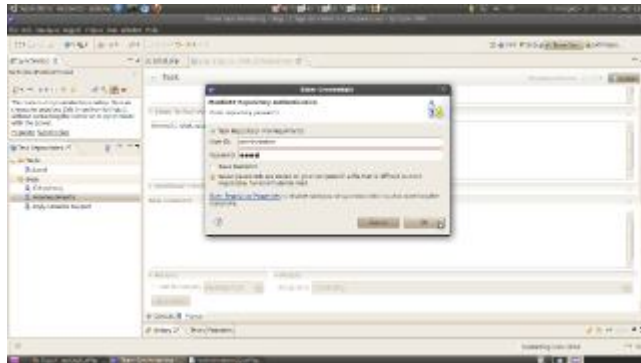
Un utilisateur d'un robot nous contacte en nous disant que le robot lui maintien qu'il a toujours 0 ans. Il nous pourri car il n'est pas très content d'avoir payé aussi cher pour un produit buggué ! Bon ça lui passera avec le temps...

[illegible]


The screenshot shows the 'New repository' dialog box in TortoiseSVN. The 'Name' field is empty, and the 'URL' field contains 'http://svn.apache.org/repos/asf/ant/trunk/'. The 'Repository type' is set to 'Apache Subversion'. The 'Create new repository' button is highlighted.

The image contains two screenshots of the AWS IAM console, specifically the 'Groups' page for the 'AWS-ReadOnlyAccess' group. The top screenshot shows the 'Groups' page with a table of groups and a 'Create New Group' button. The bottom screenshot shows the 'Groups' page with a table of groups and a 'Create New Group' button.

On fait « Submit ». On valide son identité :



Correction du bug



```

def add(x, y):
    return x + y

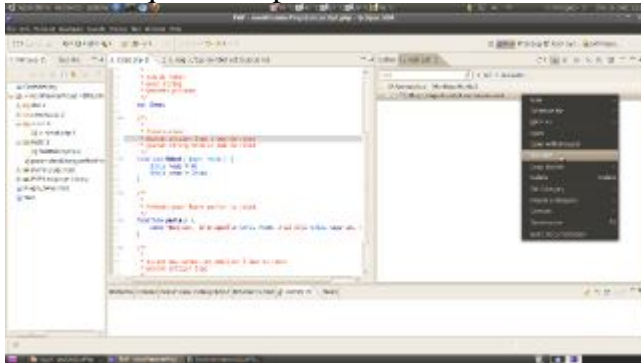
x = input("Enter first number: ")
y = input("Enter second number: ")

result = add(x, y)

print("The sum of " + x + " and " + y + " is " + str(result))

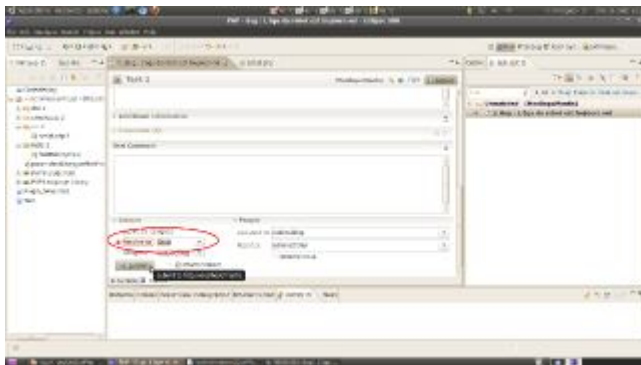
```

Pour résoudre le bug, on va activer la tâche. Pour cela, on fait click droit puis Activate, ou bien on clique sur le petit cercle devient l'intitulé de la tâche.



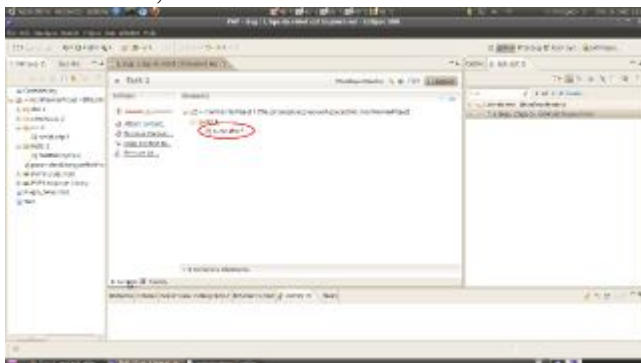
Une fois activée, le cercle se remplit en bleu pour indiquer visuellement qu'elle est active. On peut donc toucher notre code pour résoudre le problème.

Une fois corrigé, on va taguer la tâche comme : Fixed (c'est à dire traitée). On ouvre la tâche par un double click et on scrolle tout en bas. On coche alors « Resolved as » puis on sélectionne « fixed »



Remarque : On voit alors le nom de la tâche se rayer pour indiquer visuellement que la tâche est terminée. On peut tout de même la réactiver au besoin (même si ce n'est pas conseillé).

Ce qui est assez génial c'est que la tâche a agrégé un contexte. Pour le voir, on choisit la tab « Context », on arrive alors sur la tab suivante :



On peut alors voir quels fichiers ont été modifiés dans le cadre de la résolution ! Pas mal non 😊.

Retournons dans le client Web de Mantis voir ce qui a été fait... Voici le dashboard Mantis de l'utilisateur « administrator » :



Si on fait View Issues on retrouve notre report de bug :

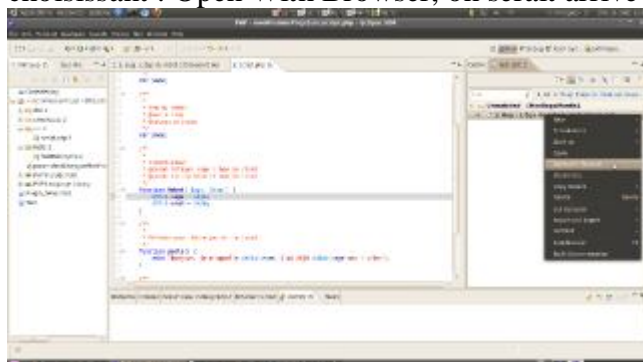


On clique dessus, on retrouve bien la description exhaustive :



On voit bien au passage le statut « Resolved » de notre bug!

On aurait pu faire plus simple, en faisant click droit sur la tâche sous Eclipse et en choisissant : Open With Browser, on serait arrivé sur la même description !



Parallèle avec Synergy (Outil Telelogic/IBM qui coûte un bras...)

Pour ceux qui connaissent Synergy (outil de versioning), on l'utilise avec des tâches.

Le chef de projet affecte des tâches contenant une description, une durée, une ressource allouée, etc. Ensuite le développeur travaille en activant sa tâche et toutes les modifs sont tracées au sein de cette tâche. On peut donc envisager de travailler de la même façon avec Subversive/svn et Mylyn/Mantis grâce au contexte qui est agrégé.

Et tout ceci GRATUITEMENT !!! Ceci était un clin d'oeil à mon ancien employeur 😊.

Bilan

Je n'ai certes pas été exhaustif mais j'espère que vous aurez perçu l'intérêt de Mylyn sous Eclipse... Je trouve cette intégration très réussie et très pratique. On peut utiliser Mantis ou Trac de la même façon sous Eclipse ce qui s'avère sécurisant en cas de changement d'outil!

Le concept de tâche permet de plus d'agréger un contexte, une planif, de savoir qui a résolu (ou pas d'ailleurs) pour le triquer s'il a fait n'importe quoi ! C'est donc très sécurisant pour le projet.

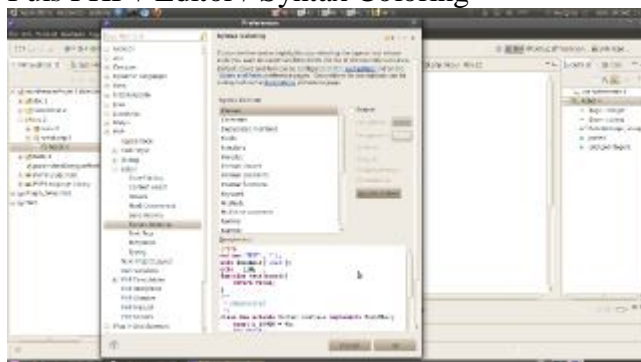
Aller un peu plus loin avec Eclipse

Avant de terminer et de vous laisser souffler, quelques derniers tuyaux concernant Eclipse : coloration syntaxique, annotations listées, complétion automatique, templates, folding, utilisation de sources déportées (distances ou locales), ...

Coloration syntaxique et thèmes

Vous avez pu le voir depuis le début du tutoriel, le thème par défaut est à fond clair. Je sais que pas mal de développeurs préfèrent les thèmes à fond sombres (ou même parfois obscurs ! lol). Du coup, il peut être pratique de modifier les couleurs de notre très cher IDE.

Pour cela, Window / Preferences
Puis PHP / Editor / Syntax Coloring

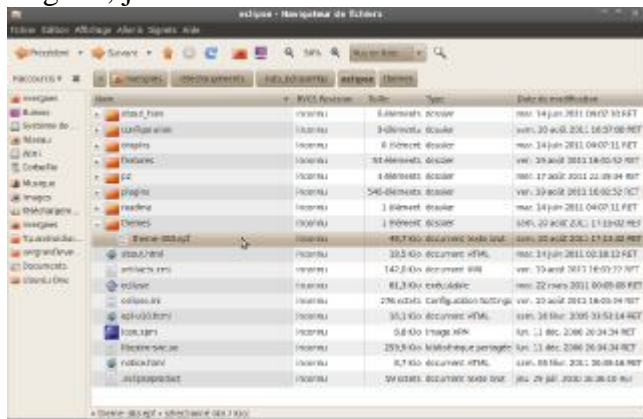


Vous pouvez ainsi tout modifier et vous refaire un thème de A à Z. Mais bon courage... Enfin disons que ce n'est pas ma passion ce genre d'opération. On va faire mieux !

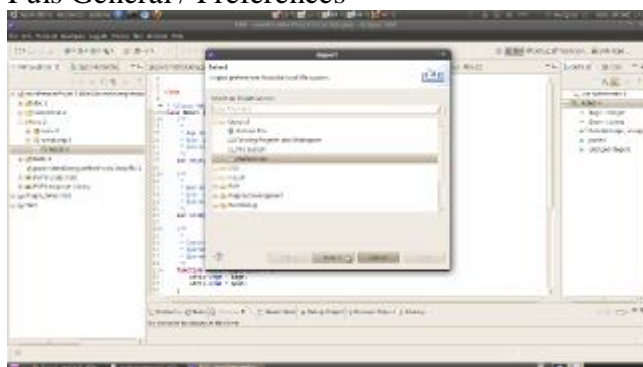
On se rend sur <http://www.eclipsecolorthemes.org/?list=toppicks&lang=php>, on cherche un thème Php qui nous plaît. Par exemple je vais vous montrer un exemple avec celui-ci : <http://www.eclipsecolorthemes.org/?view=theme&id=383> même si je ne supporte pas les fonds foncés 😞. C'est juste pour faire plaisir à certains 😊.

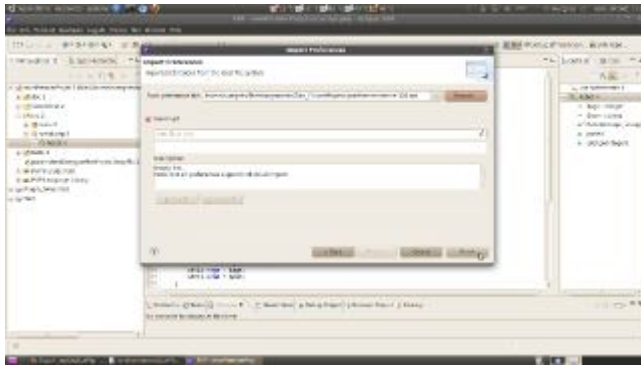


On télécharge le fichier au format *.epf, on le place où on veut. Le plus logique est de le placer dans un dossier que l'on crée dans le répertoire d'Eclipse. Par exemple, pour faire original, je vais créer un dossier /themes dans /eclipse.

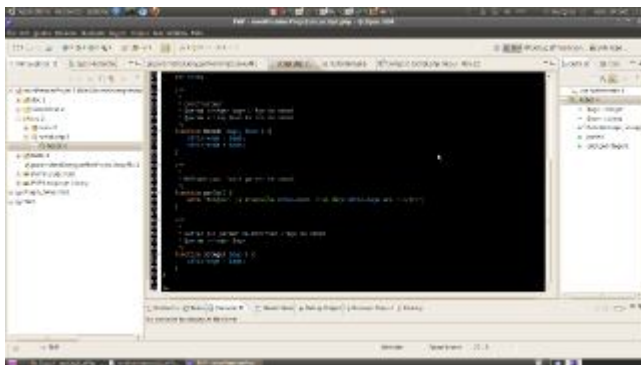


Ensuite : File / Import...
Puis General / Preferences





Et voici notre nouveau thème en action ! Je suis sur que vous allez adorer pouvoir changer de thème aussi facilement 😊!



Annotations dans le code

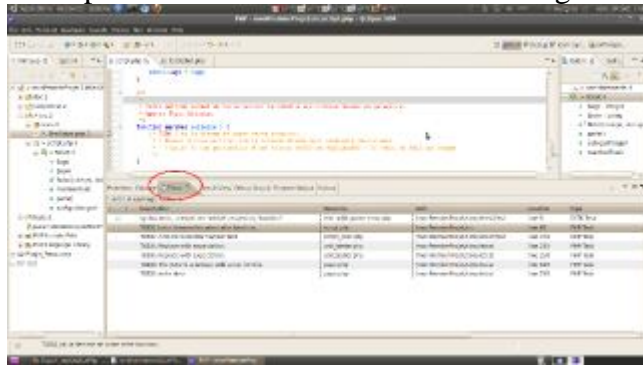
Une fonctionnalité très intéressante d'Eclipse (valable pour tous les langages d'ailleurs, pas seulement Php), est de pouvoir ajouter des annotations à votre code.

Par exemple, vous devez coder une fonction relou mais vous avez la flemme de la faire aujourd'hui, et oui on reporte souvent à demain 😊. Vous mettez alors un commentaire pour y penser, puis vous mettez un post-it todo-list sur votre station, puis il s'envole ou bien vous le voyez tellement souvent que vous n'y faites même plus attention...

Il y a mieux ! Vous mettez votre commentaire précédé de l'annotation TODO. Puis vous ajouter une vue à votre perspective Php courante : Window / Show View / Tasks



Vous pouvez alors voir une liste de tâches grâce aux annotations de votre projet.



On retrouve bien mon commentaire « J'ai la flemme de coder cette fonction... »

Pour infos, les autres TODO de la liste sont dans le framework de test SimpleTest qui fait partie du projet. (D'ailleurs ça ne rassure pas trop sur l'utilisation de SimpleTest...)

Par défaut, Eclipse reconnaît les annotations suivantes : FIXME, TODO et XXX

Vous pouvez en ajouter des personnelles et même définir des priorités : Window / Preferences
Puis : General / Editors / Structured Text Editors / Tasks Tags



En cochant « Enable searching for Task Tags », vous avez la possibilité d'en ajouter. Je n'ai pas poussé l'investigation plus loin car TODO et FIXME me suffisent amplement !

Alors terminés les post-it?

Complétion automatique

Un grand classique des IDEs, la complétion automatique... Le principe est d'assister la saisie en vous proposant une liste de choix.

Il suffit de commencer un mot, la complétion est automatique par défaut. Si jamais elle n'était pas automatique, il faut la forcer en faisant : CTRL+SPACE

Un exemple, je saisi : \$t

Immédiatement Eclipse me propose : \$this



Je tape ENTER, Eclipse me propose alors une liste de possibilités.



Il me suffit de choisir une possibilité et de faire ENTER. Pratique, non ?

Templates

Note inutile : Ce paragraphe est dédié à Lanzorg 😊.

Les templates permettent d'aller plus loin avec la complétion auto. Vous pouvez vous définir des blocs entiers de code qui s'afficheront lors d'un raccourci.

Il existe un bon nombre de templates déjà définis par défaut. Ils appartiennent à un langage donné bien sur. Vous pouvez en créer pour Java, pour Php et ce ne seront pas nécessairement les mêmes.

Pour les voir et peut être en ajouter : Window / Preferences

Puis : PHP / Editor / Templates

The screenshot shows the Windows Task Manager Performance tab. The CPU usage is at 100%. The 'Processes' tab is also visible, showing a list of running processes. The 'Background processes' section is expanded, showing various system services like 'System Idle Time', 'smss.exe', 'svchost.exe', etc.

Si on examine ce template de plus près, on voit qu'il est constitué de macros : `\dollar` par exemple, ou encore `\index`.

Ces macros rendent la saisie intelligente : on déplace le curseur dans le template (de macro en macro) grâce à la touche TAB. Eclipse vous affiche les zones à saisir dans un encadré, avec un nom qui vous indique une sémantique. Par exemple, dans le for, on nous indique que le premier champ à compléter est un index qui doit être précédé d'un dollar (spécificité Php de variables).

1. Je saisi : for, Eclipse me propose alors « for – for statement » entre autres, je le sélectionne et fais ENTER.
2. Je tape : \$monIndex puis TAB
3. Je tape : 10 puis TAB (en supposant que 10 est ma borne max)
4. Le curseur est positionné dans le corps du for, pile où je dois coder!

- Je ne suis pas expert en macros, je ne vous en dirais pas tellement plus mais je vous invite à lire ceci : <http://www.programmez.com/tutoriels.php?tutoriel=82&xtor=EPR-106> qui traite des templates Java avec Eclipse.

Sachez que c'est un outil puissant pour gagner du temps même si je dois avouer ne quasi jamais les utiliser... Mais je vais tâcher de m'y mettre dans mon prochain projet !

Le folding est une fonctionnalité toute simple qui est disponible sur de nombreux éditeurs mais elle est tellement pratique que je ne peux pas la zapper !

Le folding est parfois traduit par une expression à la xxx du type « pliage de code ». Ce procédé permet de modifier visuellement la granularité du code. On peut par exemple réduire tous les corps des boucles pour voir le code à un niveau plus haut. On peut réduire les longs commentaires par exemple...

Le folding est parfois traduit par une expression à la xxx du type « pliage de code ». Ce procédé permet de modifier visuellement la granularité du code. On peut par exemple réduire tous les corps des boucles pour voir le code à un niveau plus haut. On peut réduire les longs commentaires par exemple...

Dans Eclipse, folding activé, on peut réduire ou agrandir les blocs avec les – et + dans la marge à gauche du code.

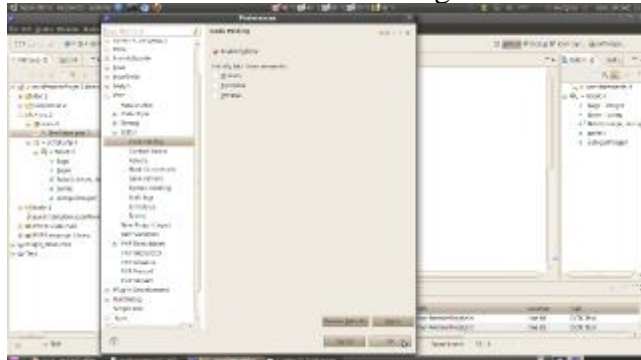
Exemple de tous les commentaires et méthodes réduits pour voir toute la classe sur un seul écran :



Sans le folding, je suis loin de voir tout le code :



Pour modifier les paramètres du folding : Window / Preferences
Puis : PHP / Editor / Code Folding



Le paramétrage est assez pauvre pour PHP, mais bon c'est suffisant.

Sources déportées

Nous avons vu en première partie du tuto que les sources devaient se trouver dans le répertoire interprété par Apache (en général /www). Eclipse possède pourtant son propre workspace contenant ses projets.

J'ai donc choisi de placer les sources dans le projet Eclipse sur lequel j'ai travaillé et de créer un lien symbolique vers www. Cette solution peut déplaire et ne pas fonctionner sur Windows

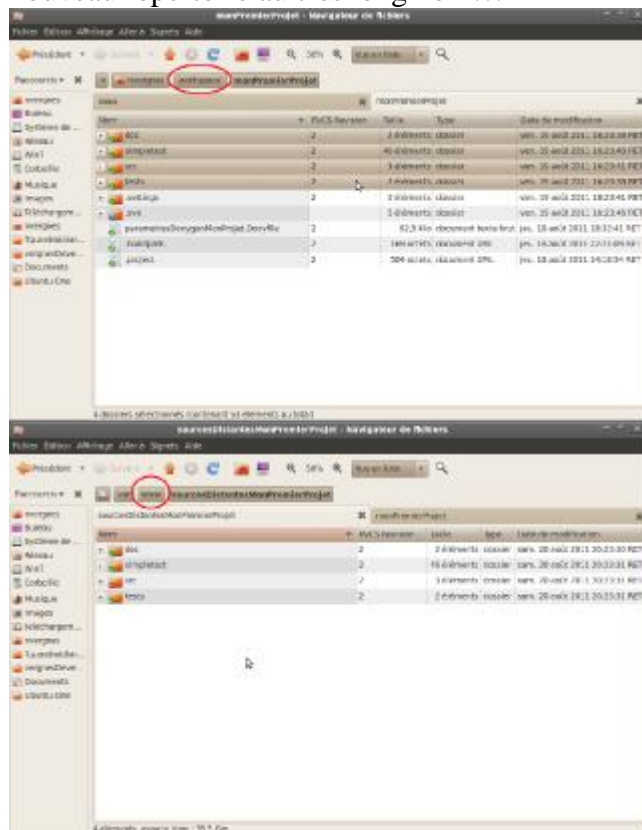
(je suis un peu une quenelle en Windows, donc à voir s'il existe l'équivalent du lien symbolique Unix...).

Le problème se pose également si on travaille sur un serveur distant. Il est alors nécessaire de pouvoir éditer/modifier ses sources. Pour cela on utilisera ftp (sans cryptage donc pas super) ou mieux sftp (basé sur le protocole ssh2).

Eclipse permet donc de travailler sur des sources déportées, c'est ce qui permet d'apporter une solution à ces 2 problèmes énoncés.

Cas de sources déportées locales

Je vais commencer par créer un répertoire /sourcesDistantesMonPremierProjet dans /www. Ensuite je place les répertoires /doc, /SimpleTest, /src, et /tests de mon projet Eclipse dans ce nouveau répertoire au très long nom...



Je supprime mon ancien lien symbolique...

[illegible][illegible]

The screenshot shows the 'Windows Firewall - Rule Editor' window. The 'Rule Name' is 'Program'. The 'Action' is 'Block this program from the Internet'. The 'Programs' list contains 'C:\Program Files\Internet Explorer\IEXPLORE.exe'. The 'Protocol' is 'TCP' and the 'Ports' are '80, 443'. The 'Direction' is 'Outgoing'. The 'Scope' is 'All IP addresses'. The 'Logging' checkbox is checked. The 'Advanced' tab is selected, showing 'Local address' as 'Any IP address' and 'Remote address' as 'Any IP address'. The 'General' tab is also visible, showing the rule is 'Enabled' and 'Active'.



The screenshot shows the Visual Studio IDE with the 'Properties' window open. The 'Debug' tab is selected, and the 'Command Line' field is highlighted with a red circle. The 'Command Line' field contains the path to the 'Debug' directory of the project.

Avec cette méthode, quelque soit votre OS, vous pourrez déporter vos sources où bon vous semble 😊.

Cas de sources déportées distantes

Note importante : Ce cas de figure n'est pas un cas d'utilisation recommandé ! Il peut être cependant utile, si on ne travaille pas avec un outil de versioning comme SVN, d'aller modifier un fichier directement sur un serveur en production sur un petit site. En général, il faut plutôt faire la modif sur son repository SVN, puis synchroniser avec le serveur en production. Cette étape s'automatise avec des outils d'intégration continue comme Jenkins (avec Phing). Je n'aborderais pas ce point dans ce tuto.

Pour ce cas d'exemple, je vais créer un nouveau projet Php qui sera initialement vide. Je vais ensuite utiliser sftp pour synchroniser mes sources distantes avec mon Eclipse.

Pourquoi sftp et pas ftp ? Par ce que rien n'est crypté avec ftp. Du coup vos mots de passe transitent en clair. Il suffit d'utiliser un sniffeur pour récupérer toutes vos données ! Donc n'utilisez jamais ftp ! Sftp quant à lui est basé sur ssh donc crypté (cryptographie asymétrique : clé privé, clé publique)...

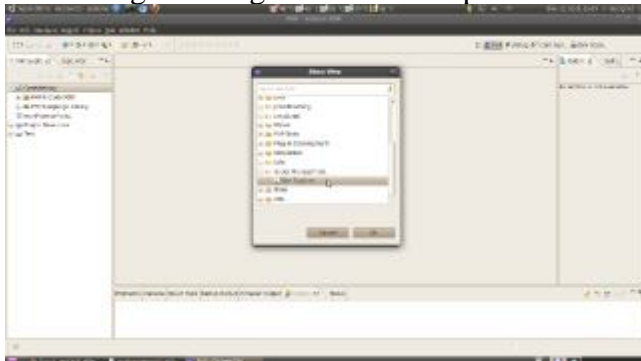
Il faut au préalable installer un plugin : <http://www.jcraft.com/eclipse-sftp/>. L'update site étai le suivant : <http://eclipse.jcraft.com/>



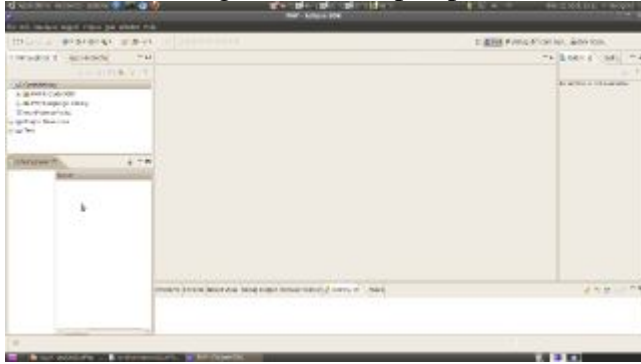
On redémarre Eclipse... On va ensuite ajouter une nouvelle vue dans une de nos perspectives.

Pour cela : Window / Show View / Other...

Puis : Target Management / Site Explorer



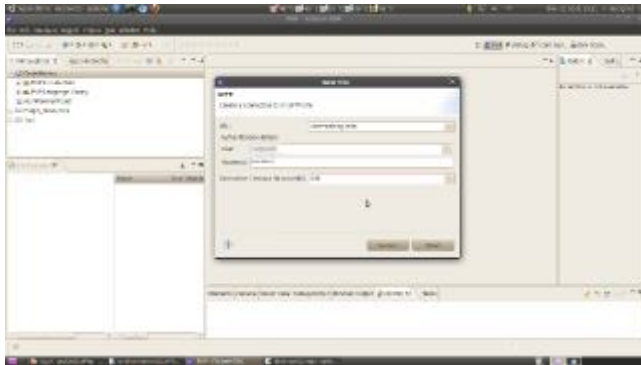
Voici la vue intégrée dans ma perspective de base PHP :



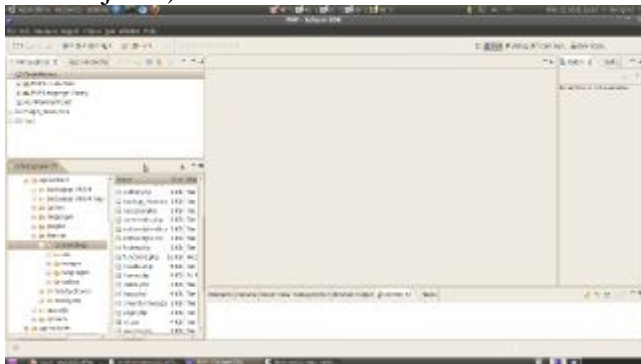
On fait (dans cette vue) : click droit puis New / Target Site

Vous entrez ensuite vos données personnelles pour ssh puis « Finish » (attention : elles sont différentes des données d'accès ftp!).

Pour le timeout, à vous de voir. Votre connexion se fermera automatiquement après ce timeout...

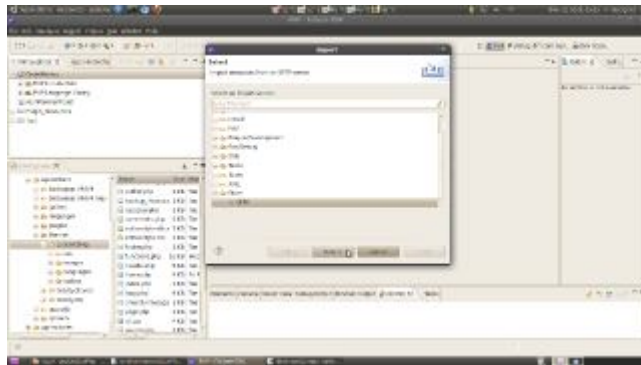


Vous pouvez à présent explorer votre serveur grâce à sftp (dans la nouvelle vue que nous avons ajouté)



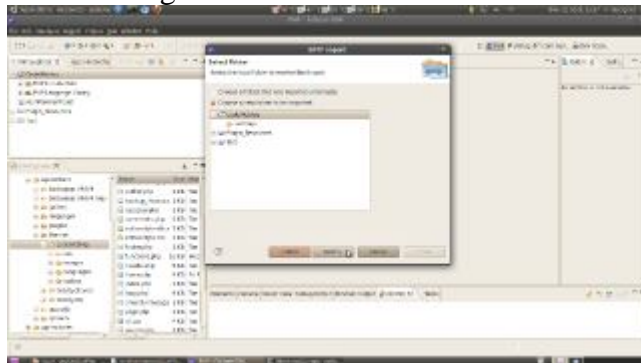
Ensuite : File / Import...

Puis : Other / SFTP



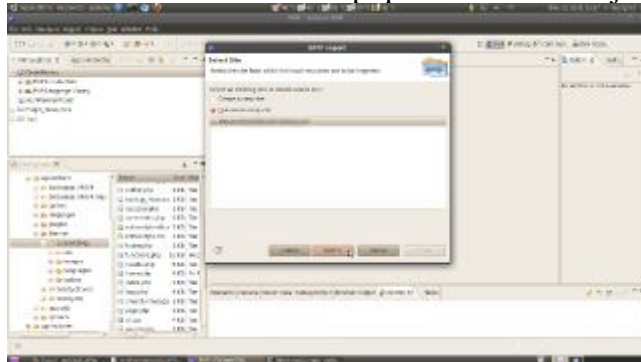
Puis : Next

On choisit le répertoire qui va recevoir nos sources, donc pour moi mon nouveau projet CodeWeblog.



Puis : Next

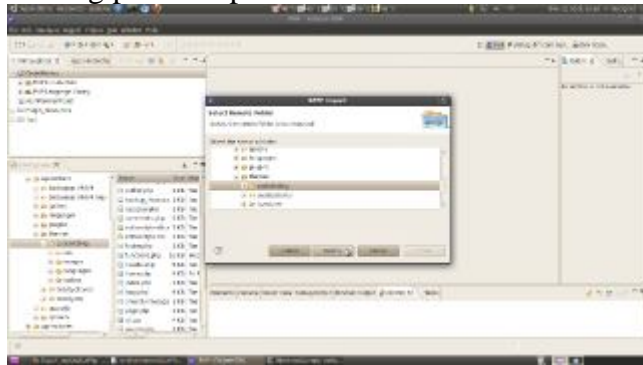
On choisit la connexion sftp que nous avons déjà paramétrée.



Puis : Next

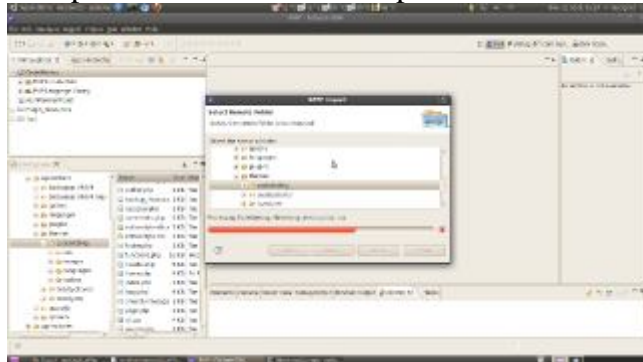
On explore alors le serveur et on choisit la partie à récupérer, pour moi mon thème WordPress

du blog par exemple.

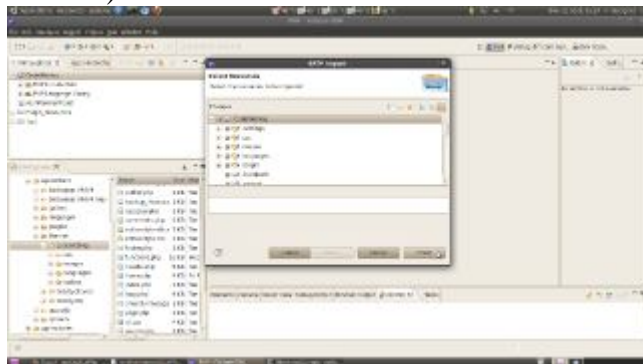


Puis : Next

L'import de la structure des répertoires commence...

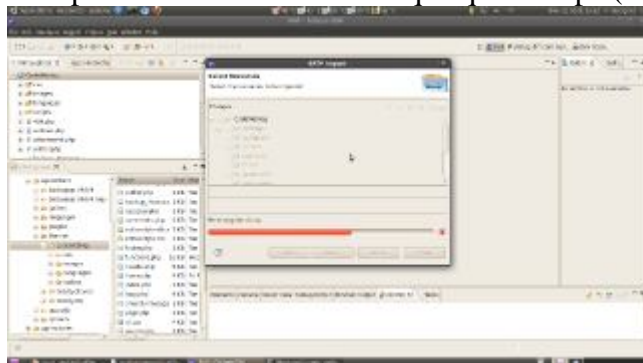


On choisit ensuite les fichiers qui doivent être importés (ça permet ainsi d'en exclure certains).

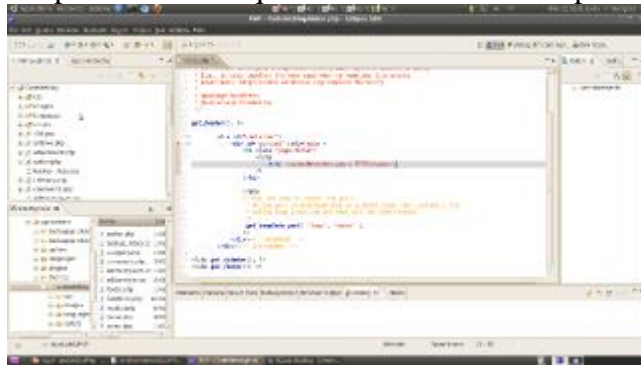


Puis : Finish

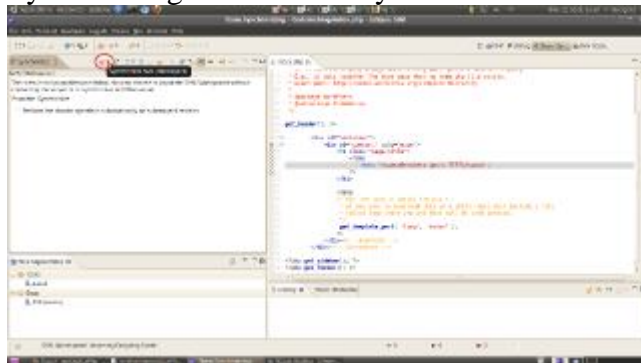
L'import commence... et dure quelques temps (suivant la taille de vos données)...



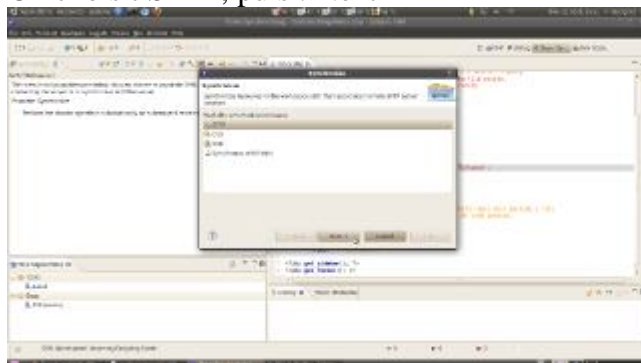
On peut alors voir que les sources ont été copiés dans notre projet Eclipse.



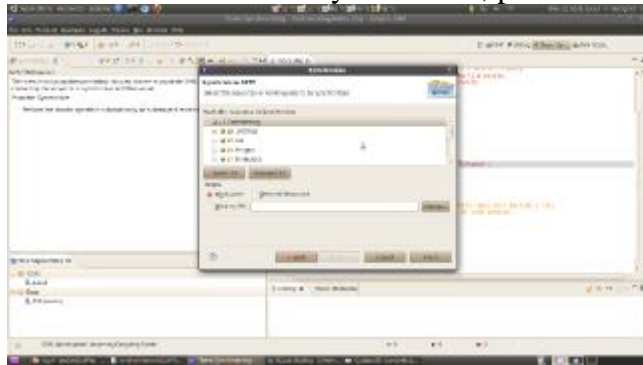
Je modifie index.php de mon thème WordPress par exemple. Ces modifs se font sur ma copie locale, dans mon workspace Eclipse. Il me faut alors synchroniser pour que les modifs se propagent sur mon serveur de production. Pour cela, on va dans la perspective « Team Synchronizing ». On fait : « synchronize ».



On choisit SFTP, puis : Next

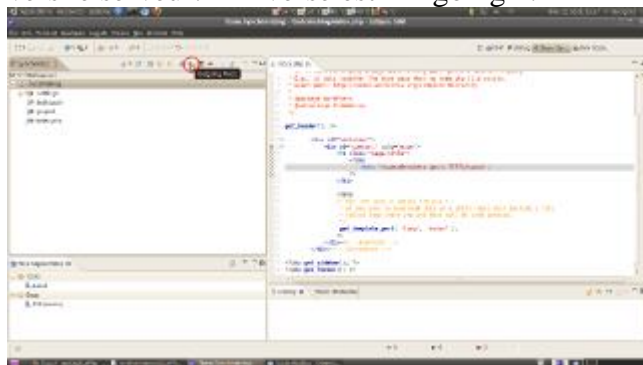


On choisit les sources à synchroniser, puis Next.

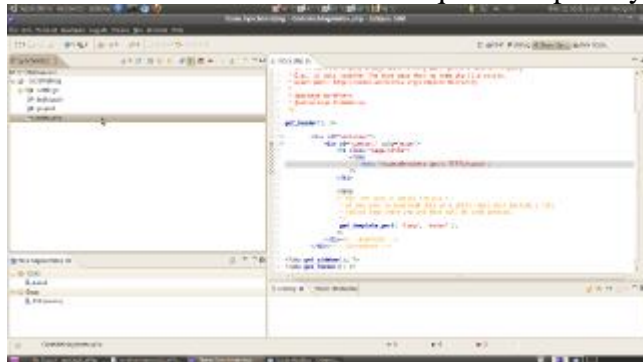


Puis : Finish

On choisit ensuite le mode, pour nous : « outgoing », c'est à dire pour envoyer une donnée vers le serveur. L'inverse est « ingoing ».

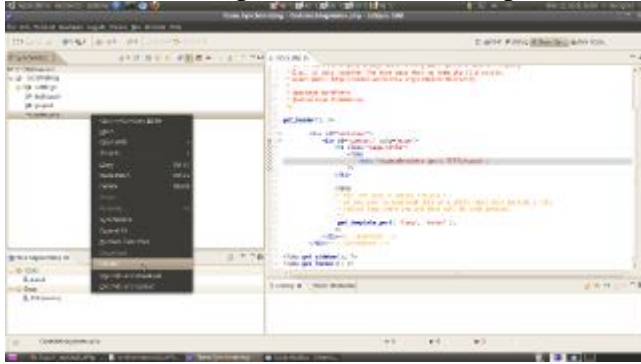


On voit alors la liste des fichiers que l'ont peut synchroniser (index.php dans mon cas).



Remarque : Les autres fichiers possibles dans la synchro sont les fichiers générés par Eclipse pour la gestion du projet, je n'ai donc pas intérêt à les mettre en prod!

On choisit un (ou plusieurs) fichier(s), puis click droit / Upload



L'upload se déroule et propage votre modif 😊.

Voilà, nous avons fait le tour des possibilités usuelles pour travailler sur des sources déportées. L'avantage de tout intégrer dans Eclipse vous permet de toujours travailler de la même façon. Que vous soyez avec SVN, SVN+SSH ou avec SFTP, vous utiliserez toujours la perspective « Team Synchronizing » pour propager vos modifs. Excellent non ? 😊

Sources

- SFTP : <http://onfaitduweb.com/programmation/ftp-et-sftp-depuis-eclipse>
- Templates Eclipse : <http://www.programmez.com/tutoriels.php?tutorial=82&xtor=EPR-106>
- Mantis : <http://manual.mantisbt.org/manual.installation.php>
- LAMP sur ubuntu : <http://doc.ubuntu-fr.org/lamp>
- Phing : <http://www.phing.info/trac/>
- jcraft : <http://www.jcraft.com/eclipse-sftp/>
- Mylyn : <http://fr.wikipedia.org/wiki/Mylyn>
- SVN : <http://doc.ubuntu-fr.org/subversion>, <http://subversion.tigris.org/>
- Subversive : <http://www.eclipse.org/subversive/downloads.php>
- SimpleTest : <http://www.onpk.net/index.php/2005/01/12/254-tutoriel-simpletest-decouvrir-les-tests-unitaires>,
http://simpletest.sourceforge.net/fr/extension_eclipse.html
- Doxygen : <http://www.stack.nl/~dimitri/doxygen/install.html>
- Eclox : <http://home.gna.org/eclox/>
- Xdebug : http://wiki.ubuntu-fr.org/eclipse_php_xdebug,
<http://xdebug.org/docs/remote>, <http://blog.pascal-martin.fr/post/xdebug-debugging-graphique-eclipse-pdt>
- PhpDoc : <http://manual.phpdoc.org/>
- PDT : <http://www.zend.com/community/pdt?ecl=EclipseZend>

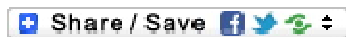
Conclusion

J'espère que ce tutoriel vous aura intéressé et vous aura permis d'améliorer votre environnement de développement. Il peut être fastidieux de déployer tout ce que nous avons vu mais la qualité du travail en dépend...

Sachez qu'un équivalent de cet environnement constitué par des outils propriétaires payants coûterait des milliers d'euros par poste ! J'ai eu l'occasion de travailler avec ce genre d'outils. Ce n'est pas toujours mieux et les coûts étant tellement élevés, les mises à jours ne sont pas toujours effectuées... Merci l'Open Source 😊.

N'hésitez pas à me poser des questions ou à me signaler des erreurs si vous en voyez. Il peut y en avoir quelques unes...

Dernier truc, je ferais sans doute des tutos vidéo dans la foulée sur la manipulation d'Eclipse et de tout cet environnement. Mon aménagement à la Réunion étant bien avancé, je reprend du service 😊! À bientôt dans un prochain tutoriel!



[« Conférence E1 – Saison 2 du 10 juin à Toul... Apprendre les bases d'Eclipse pour développe... »](#)

31 commentaires

1.



lanzorg

[le 23 août 2011 à 10 h 09 min](#)

Alors là ! Ca va faire des heureux et moi le premier ^^.

[Répondre](#)

2.



[Nicolas](#)

[le 23 août 2011 à 10 h 13 min](#)

@Lanzorg, je t'ai dédié un paragraphe! Si tu cherches bien... 😊

[Répondre](#)

3.



lanzorg

[le 23 août 2011 à 11 h 20 min](#)

Voilà, première lecture effectuée !

Voici mes impressions :

- J'ai encore beaucoup à apprendre.
- Je comprends mieux le mot « IDE » ^^.

Merci pour la dédicace.

Sous Netbeans, ça s'appelle « Code Templates » et je n'avais pas fait le rapprochement avec « Templates ».

Bon j'attaque le tuto en mode pratique dans une VM pour pas foutre le bordel ^^.
Encore merci pour ce mastodonte et bon courage pour la vidéo.

[Répondre](#)

4. Ping : [Code Weblog](#)
5. Ping : [Code Weblog](#)



6.

Cam

[le 21 septembre 2011 à 9 h 39 min](#)

Merci pour ce très bon tuto ! 😊

[Répondre](#)



7.

bruno

[le 15 novembre 2011 à 19 h 19 min](#)

bonsoir,

bon cela y est. Je suis sur eclipse.

j'ai pas bien compris comment lié eclipse à localhost.

quand je faisais RUN, il lançait localhost/monprojet/foldername/www/index.php.
alors que mon index.php est sous localhost/index.php.
Après 10 minutes de tripotage, maintenant en faisant RUN, il me met bien

localhost/index.php.

Mais je ne sais pas comment j'ai fais !!!

bon ma page s'affiche, c'est du php + javascript + ajax.

je clique sur mon bouton GO, ok cela fonctionne quelque chose s'affiche, bien.

le clique sur GO déclenche une boucle qui de lance un php toutes les 30s.

mais ! par ce que il y a un mais, le même message s'affiche alors qu'il devrai y avoir un nouveau message. j'ai testé cette petite appli sur mon hébergeur et cela fonctionne. je l'ai testé en direct avec wamp, idem.

Y'a t il une histoire de cache à paramétrer dans eclipse ?

A+ pour de nouvelles zaventures de dummyphp.

[Répondre](#)



8.

latessa Marc

[le 8 décembre 2011 à 23 h 47 min](#)

Merci beaucoup pour ce tutoriel qui me permet de progresser dans ma façon de coder.

Vraiment tout est très bien détaillé et tout fonctionne !

Et pourtant je suis sous Windows (j'ai un peu honte!)

Bravo encore

Marc

[Répondre](#)



o

[Nicolas](#)

[le 9 décembre 2011 à 11 h 21 min](#)

Pas de honte d'être sous Windows 😊. Chacun son OS par rapport à ce qu'on en fait. Pendant mes études j'étais très réticent alors que maintenant il faudrait me payer cher pour retourner à Windows!

En tout cas un grand merci pour le feedback Marc 😊.

[Répondre](#)

9.



Greg

[le 20 février 2012 à 15 h 07 min](#)

Super tuto, plein de ressources très détaillées! Un must en français
Merci à toi.

[Répondre](#)



o

Nicolas

[le 20 février 2012 à 15 h 32 min](#)

Merci à toi Greg pour ton retour!



[Répondre](#)

10.



Beluxp

[le 4 mars 2012 à 10 h 00 min](#)

Merci pour ce **SUPER** travail sur ce tutoriel de qualité.

[Répondre](#)



o

Nicolas

[le 4 mars 2012 à 13 h 56 min](#)

Merci Beluxp! 😊

[Répondre](#)

11.



Fayssal

[le 13 mars 2012 à 17 h 44 min](#)

Bonjour,
Merci pour ce travail intéressant.
J'aimerais bien avoir votre avis sur la problématique suivante:

J'ai un projet php sous SVN, que je veux déployer sur un serveur distant linux en utilisant le plugin sftp de Eclipse.
En fait chaque développeur aura son propre répertoire sur le serveur linux.
Comment combiner tous cela?
avoir d'un côté un répertoire de travail qui soit à la fois synchronisé avec SVN et déployable sur le serveur distant via sftp.

Merci de votre réponse.

[Répondre](#)



○

Nicolas

[le 14 mars 2012 à 5 h 31 min](#)

Bonjour Fayssal

Je ne comprend pas tout à fait votre problématique. Pourquoi chaque développeur aura son propre répertoire ? Le projet n'est-il pas commun ? Les développeurs ne seront-ils pas amenés à travailler sur les mêmes fichiers ?

En général, les développeurs travaillent sur le même projet. Ils ont une copie locale du projet et une fois les modifs validées, ces modifs sont propagées vers dépôt (via commit SVN). Les autres développeurs pourront alors faire un update pour récupérer les travaux des autres. Du coup pas besoin de plusieurs répertoires sur le serveur...

Schéma général :

- - SVN est déployé sur un serveur (sinon, il faut utiliser GIT qui est décentralisé). Sur ce même serveur, on peut déployer un environnement de test. Cet environnement de test récupère toutes les modifs après chaque commit de n'importe quel développeur. Il est souhaitable d'automatiser cette tâche avec ANT ou Phing (qui permettent de construire un projet grâce à un système de script un peu à la manière des makefiles du C). Pour apporter de la qualité, on peut lui ajouter un outil d'intégration continue comme Jenkins par exemple (test unitaires joués après chaque modif du projet de test, vérification de style, vérification de duplication de code, génération de la documentation).
- - Après cet environnement de test, vient l'environnement en production qui peut se situer sur le même serveur ou bien mieux sur un autre (pour des raisons de sécurité et de performance par exemple)... Pour construire le projet en production, on peut utiliser un outil comme Phing. Le projet se construit automatiquement et peut se déployer tout seul via sftp.
- - Dans tous les cas, il faut automatiser au maximum. Du coup, je ne pense pas qu'utiliser le plugin Eclipse SFTP soit une solution confortable pour déployer en production.

Je n'ai sans doute pas répondu à votre problématique. Si tel est le cas, pouvez-vous détailler votre contexte ?

[Répondre](#)



Fayssal

[le 14 mars 2012 à 10 h 02 min](#)

En fait je me suis mal exprimé.

Il s'agit uniquement de l'environnement de DEV.

Le choix de mettre tous sur un seul serveur vient du fait que l'application interagit avec plusieurs autres applications et le fait d'ouvrir des flux pour chaque poste est très coûteux.

Pour le serveur SVN il existe déjà mais sur une machine à part.

J'ai besoin de SFTP pour synchroniser mon répertoire en local avec le serveur de DEV, comment y arriver sachant que le flux est ouvert dans un seul sens de la machine du développeur vers le serveur de DEV.

Merci

[Répondre](#)



■ [Nicolas](#)

[le 14 mars 2012 à 10 h 51 min](#)

Savez-vous qu'on peut utiliser svn via ssh (commande : « svn+ssh »)? Ça ne répondrait pas à votre problème par hasard? En faisant un update ou un commit via le tunnel ssh, vous pouvez ainsi synchroniser votre poste local avec votre dépôt, ou l'inverse, poster des modifs vers le dépôt. Dans ce cas, sftp n'est pas utile.

Consultez ces liens : [lien 1](#), [lien 2](#)

Autre piste si j'ai répondu hors sujet ci-dessus : ne pouvez-vous pas faire un montage NFS de votre serveur de dev sur vos machines clientes qui assurent le dév?

Je ne suis pas certain d'avoir compris à 100%, j'ai peut être encore répondu à côté... 😊

Je ne suis pas très calé en système/réseaux, je crains de ne pas pouvoir vous aider tellement plus... Désolé...

Bon courage.

[Répondre](#)

12.



[macsim](#)

[le 20 avril 2012 à 21 h 10 min](#)

Franchement bravo, c'est détaillé, c'est bien expliqué. Bonne continuation

[Répondre](#)



○

[Nicolas](#)

[le 21 avril 2012 à 6 h 33 min](#)

Merci beaucoup pour ce feedback qui me fait bien plaisir! 😊

[Répondre](#)



anoriel

[le 4 mai 2012 à 9 h 22 min](#)

ça fait 2 ans que je veux me mettre sur eclipse et jusque là les tutos que j'avais testés m'avaient refroidis!
j'ai tout compris a présent et je vois enfin la puissance d'eclipse entre mes mains...
Merci beaucoup pour cette page référence!

[Répondre](#)



[Nicolas](#)

[le 4 mai 2012 à 15 h 35 min](#)

Ouah! Merci pour le feedback! 😊

Content qu'Eclipse t'ait ouvert ses portes vers un nouveau monde, plus simple, plus performant, moins pénible! 😊 Bonne continuation.

[Répondre](#)



phillylovepark

[le 22 juin 2012 à 14 h 15 min](#)

Merci beaucoup pour ce tuto, très complet 😊

[Répondre](#)



15.

[jef](#)

[le 1 juillet 2012 à 14 h 56 min](#)

Bonjour Nicolas,

Bravo pour ce site en général et surtout les tutoriaux qui sont très bien (formateur moi-même, j'en apprécie d'autant plus la qualité 😊).

Petite question Après avoir présenté **Aptana** qui est basé sur Eclipse, vous parlez d'**Eclipse PDT**. Est ce que cela signifie que PDT est « plus abouti » qu'Aptana ?

[Répondre](#)



o

[Nicolas](#)

[le 1 juillet 2012 à 16 h 57 min](#)

Bonjour Jef.

Merci pour votre retour! 😊

Concernant PDT et Aptana, honnêtement, il est très difficile de les départager... En fait, il y aurait 3 comparaisons possibles:

- - Aptana standalone
- - Eclipse + plugin Aptana
- - Eclipse + plugin PDT

Dans la version standalone, je n'aime pas la castration qui a été faite à Eclipse. Se basant sur un socle aussi riche et abouti, je ne comprend pas pourquoi ils ont supprimé des fonctionnalités de base de l'IDE. Du coup, j'ai vite laissé tomber cette solution. En plus elle était assez instable lors de mon test...

Concernant les plugins, Aptana est excellent pour le CSS (complétion automatique, coloration syntaxique réussie, parseur spécifique, etc). Il est aussi pratique pour faire du FTP.

PDT est meilleur dans d'autres domaines, mais je vous avouerais ne plus me souvenir exactement lesquels. Je me souviens par contre avoir un jour installé une version très instable qui m'avait un peu vacciné...

J'ai longtemps rêvé de pouvoir trouver un plugin alliant les côtés positifs de PDT et d'Aptana. J'étais même passé sous Netbeans pour le PHP. Je le trouvais plus stable et globalement meilleur, excepté l'allure générale sous Linux qui n'est pas super sexy... Je suis par contre toujours resté sur Eclipse pour Java/C/C++/Python.

Désolé de ne pas pouvoir vous en dire d'avantage... Ce qui est certain, c'est qu'il reste encore pas mal de progrès à apporter à ces 2 plugins pour être à la hauteur de Java ou CDT par exemple. Il manque encore par exemple de la stabilité et des fonctionnalités de refactoring (quasi inexistantes à ce jour).

Je vous invite donc à tester ces 2 plugins pour vous faire votre avis, et pourquoi pas nous donner un retour en commentaire de ce billet? Ce serait gentil.

Bonne continuation!

[Répondre](#)



[*ief*](#)

[le 2 juillet 2012 à 9 h 43 min](#)

Merci Nicolas pour cette réponse.

J'ai développé il y a quelques années en Java sous Eclipse et j'avoue que cela m'a laissé un goût mitigé : de bonnes choses mais aussi une certaine lourdeur et une instabilité parfois déroutante.

Il y a aussi la config. de base qui doit être costaud : sous XP, 1Go de RAM est trop juste et les processeurs un peu anciens ont bien du mal à suivre le rythme...

Le système des plugins est intéressant mais aussi source de problèmes à cause de l'incompatibilité, des menus qui changent et du manque parfois de documentation claire et complète.

Aujourd'hui je travaille en PHP et j'avoue avoir du mal à trouver l'IDE idéal pour le Web (HTML, CSS, Javascript, SQL et PHP).

Pour l'instant, j'ai PsPad qui me plaît bien mais qui est trop limité pour un travail quotidien.

[Répondre](#)



[Nicolas](#)

[le 2 juillet 2012 à 16 h 12 min](#)

Je comprend Jef. J'ai souvent entendu un discours similaire au tiens concernant la lourdeur de ces IDE. Je n'ai pas connu ce temps. Aujourd'hui, je ne pense pas que ce soit encore instable et si lourdaud. En ce moment j'utilise un Eclipse auquel j'ai ajouté Subversive, PDT, PyDev, CDT, un plugin pour script shell. Bref, plus lourd tu meurs... Et pourtant c'est plutôt stable et sur une workstation moderne, ça réagit vite. Bon sur une vieille machine, je pense que ce sera insupportable, on est d'accord...

Sinon je sais que la tendance actuelle des développeurs Web est d'utiliser SublimeText. C'est pas un IDE mais quand même un outil assez avancé et surtout extrêmement modulaire et customisable. Sinon, si vous avez des ressources correctes sur votre machine, tentez quand même l'essai de Netbeans. C'est vraiment bien je trouve pour PHP.

Continuons de rêver d'un IDE léger et performant pour le Web... 😊

[Répondre](#)



16.

Seb

[le 15 juillet 2012 à 0 h 16 min](#)

Merci pour ce tuto, votre vidéo sur aptana m'avait convaincu mais maintenant j'avoue hésité entre la version Aptana standalone et eclipse + PDT. Si vous retrouvez les avantages de PDT sur le plugin aptana je suis preneur 😊

J'ai testé quelques plugin très sympa Egit, Sql Explorer c pratique aussi.

Sans oublier zen Code.

Merci pour tout ça.m'aide vraiment, pour me configurer un super espace de dev 😊

[Répondre](#)



o

[Nicolas](#)

[le 15 juillet 2012 à 6 h 52 min](#)

Bonjour Seb.

Je vais essayer de faire un comparatif entre les solutions suivantes:

- - Eclipse + plugin PDT
- - Eclipse + plugin Aptana
- - Netbeans

Je ne regarderai par contre pas les « simples » éditeurs comme SublimeText qui est ultra à la mode en ce moment. Je souhaite rester sur de l'IDE. Je ne testerai pas non plus les versions standalone car j'ai besoin de customiser mon environnement, et je préfère pour cela partir sur de bonnes bases vierges! 😊

D'ici quelques jours je vais refaire un peu d'appli Web à coup de PHP et JS. Je vais donc essayer de me fixer un cadre de test. Je vais me constituer un projet comportant: php (code interne + dépendance externe vers un framework), css, js (code interne + dépendance externe vers un framework). Je vais ensuite configurer les 3 solutions pour voir laquelle correspond le mieux à ce que je recherche (svn, complétion auto, coloration syntaxique, vérification syntaxes, etc).

Je ferais donc un billet avec mes résultats de ce test d'ici peu de temps.

A+

[Répondre](#)



17.

Olivier

[le 4 septembre 2012 à 16 h 59 min](#)

Bonjour,

Je ne pouvais pas quitter ce post sans dire un « bravo » et un « merci » : des infos claires et précises pour les installations/configurations et autres, mais aussi une très bonne synthèse d'ensemble sur les outils et les possibilités.

Olivier

[Répondre](#)



18.

Olivier

[le 4 septembre 2012 à 17 h 00 min](#)

Bonjour ,

Je ne pouvais pas quitter ce post sans dire un « bravo » et un « merci » : des infos claires et précises pour les installations/configurations et autres, mais aussi une très bonne synthèse d'ensemble sur les outils et les possibilités.