Intégration continue avec Jenkins et PHP sous Windows

Date de dernière mise à jour : 08/01/2014 à 16:15

Intégration continue

Jenkins (auparavant appelé Hudson) est un outil d'intégration continue écrit en langage Java. L'intégration continue consiste à vérifier, à chaque changement du code, que l'application en cours de développement ne subit pas de régressions, mais vous permet également de vérifier la qualité du code.

Cet outil (seul ou utilisé conjointement avec d'autres) dispose de nombreux plugins et vous permet notamment d'exécuter les tâches suivantes :

-Automatiser les compilations (appelées **Builds**- dans l'outil)
-Exécuter une suite de tests unitaires
-Effectuer des revues du code (analyse statique) et détecter le code dupliqué
-Générer un rapport de couverture de code (la couverture de code est le taux de code testé d'un programme)
-Générer automatiquement la documentation technique (qui utilisera les commentaires que vous aurez écrit en suivant la syntaxe PHPDoc ou JavaDoc)

Installation des outils PHP pour Jenkins

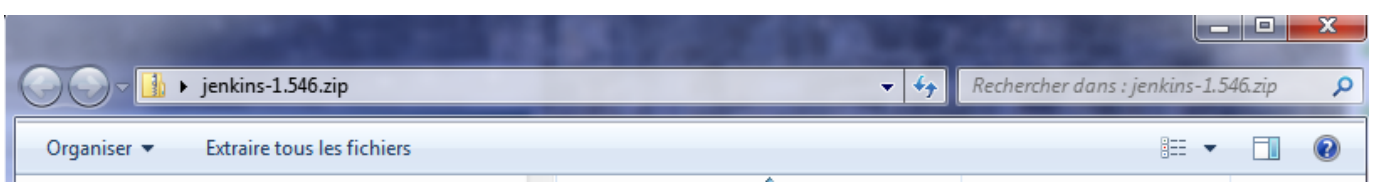
Afin que notre projet PHP soit correctement analysé et intégré à Jenkins, il y a certaines commandes à exécuter. Cliquez sur le bouton démarrer, puis, dans la zone de recherche, tapez **cmd** puis appuyez sur la touche **Entrée**. Tapez successivement l'ensemble des commandes ci-après. Vous devez avoir Pear d'installé. Si ce n'est pas le cas, téléchargez le fichier http://pear.php.net/go-pear.phar puis placez-le dans le répertoire **pear** de Wamp (dans mon cas, cela donne **E:\wamp\bin\php\php5.4.16**) puis rendez-vous dans ce répertoire en ligne de commande, et exécutez la commande suivante : **php -d phar.require_hash=0 go-pear.phar**. Cela devrait être OK !

Exécutez maintenant l'ensemble des commandes suivantes :

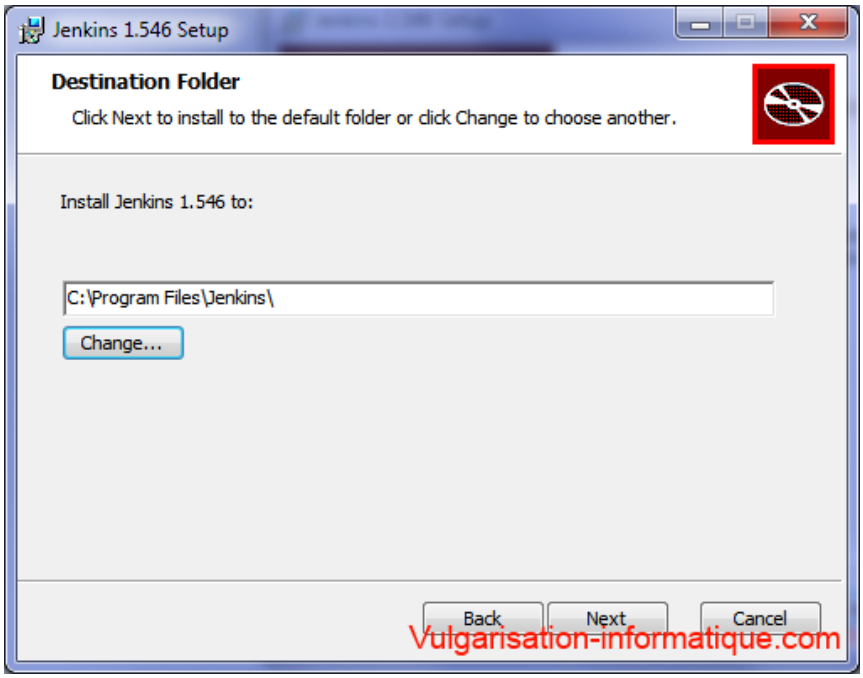
pear config-set auto_discover 1
pear install --onlyreqdeps pear.phpunit.de/PHPUnit- (Installe la gestion des tests unitaires via PHPUnit).
pear install --alldeps pear.phing.info/phing
-PHP Code Sniffer effectue une vérification de la conformité de votre code PHP sur certains standards (Zend et Pear notamment). Vous pouvez créer votre propre standard. Exécutez la commande suivante pour l'installer : **pear install --alldeps –a PHP_CodeSniffer**
-PHP Documentor vous permet de générer de la documentation. Installez-le à l'aide de la commande suivante : **pear install --alldeps PHPDocumentor**
-PHP Copy/Paste Detector est un outil vous permettant de détecter les portions de code dupliquées. Installez-le à l'aide de la commande suivante : **pear install --alldeps phpunit/phpCPD**
-PHP Mess Detector est un outil permettant de détecter certains problèmes dans votre code. Il offre certaines similitudes avec PHP Depend. Installez-le à l'aide de la commande suivante : **pear install -a pear.phpmd.org/PHP_PMD**
-PHP Depend est un outil générant de nombreux graphes pour évaluer la qualité de votre code. Vous pouvez l'installer en exécutant cette commande : **pear install pear.pdepend.org/PHP_Depend**
pear install -a phpunit/PHP_CodeBrowser
pear install -a phpunit/phploc

Installation de Jenkins sous Windows

Maintenant que PHP est configuré comme il faut, nous allons pouvoir installer Jenkins. Pour installer Jenkins sous Windows (avec Wamp), rendez-vous sur le lien Miroirs Jenkins pour choisir la version que vous souhaitez. Si vous souhaitez télécharger la dernière version stable, rendez-vous ici : Télécharger Jenkins stable (Windows). Décompressez ensuite le fichier zip. Celui-ci est composé de deux fichiers : un fichier **setup.exe** et un fichier à l'extension **.msi**.

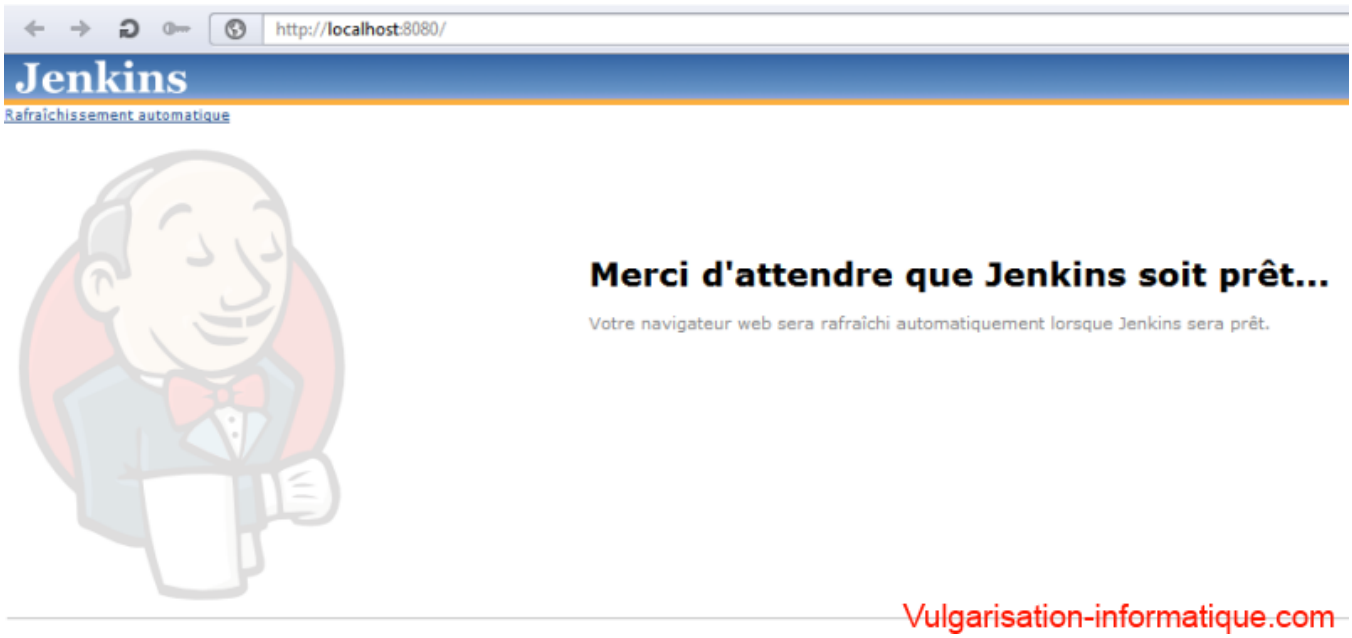


Exécutez le fichier **setup.exe**. L'assistant d'installation s'ouvre. Cliquez une première fois sur **Next**. Vous avez alors la possibilité de choisir le répertoire d'installation de Jenkins :



Lorsque vous aurez choisi le répertoire de destination, cliquez une fois sur **Next**, puis sur **Install**. Si Windows vous affiche un message vous prévenant que le programme jenkins.msi va effectuer des modifications et a besoin de votre accord, répondez par **Oui**. La procédure d'installation de Jenkins est alors en marche. Lorsque celle-ci est terminée, cliquez sur **Finish**.

Jenkins ouvre alors votre navigateur à l'adresse suivante : **http://localhost:8080**. Une page identique à celle-ci s'affiche :



L'attente peut être longue (plusieurs minutes). Si l'affichage de la page est indisponible (que vous n'avez même pas la page ci-dessus), vérifiez

qu'aucun service n'écoute sur le port 8080, puis rafraîchissez la page quelques secondes après avoir arrêté le service correspondant.

Création du projet Jenkins

Téléchargez le fichier template de base qui nous servira pour la suite en cliquant sur le lien suivant : PHP Jenkins Template. Dézippez le contenu de ce fichier dans le répertoire "jobs" de Jenkins (si vous avez installé Jenkins dans le répertoire par défaut, vous copierez le contenu du zip ici : **C:\Program Files\Jenkins\jobs**. Votre répertoire **Jobs** doit contenir maintenant un unique dossier nommé **php-jenkins-template-master**. Renommez ce dossier en **php-jenkins-template**.

Il est nécessaire de relancer le service jenkins. Pour ce faire, cliquez sur le bouton démarrer, puis, dans la zone de recherche, tapez **services.msc** et validez en appuyant sur **Entrée**. Recherchez le service nommé **jenkins**. Quand vous l'aurez trouvé, effectuez un clic avec le bouton droit de la souris puis cliquez sur **Arrêter**. Une fois le service complètement arrêté, refaites un clic droit puis choisissez cette fois **Démarrer**.

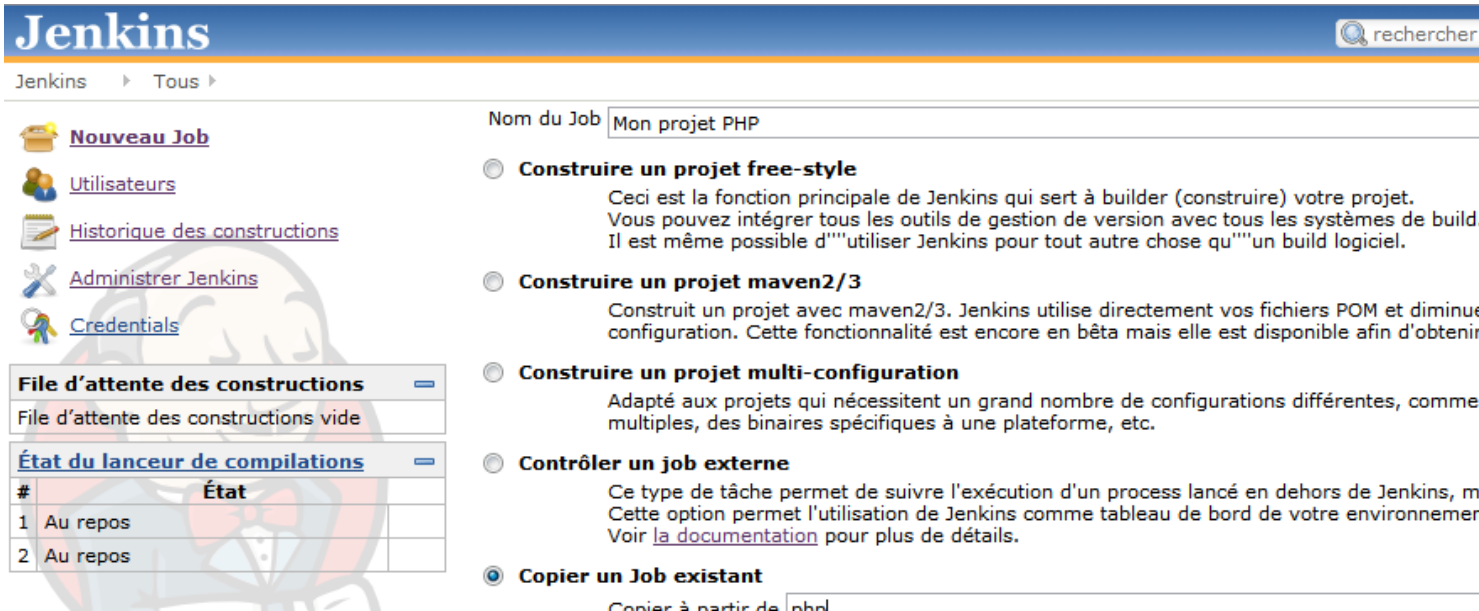
Lorsque Jenkins a fini de générer la page, vous allez vous retrouver devant l'écran suivant :

Si vous en êtes à la même étape, c'est que tout est OK ! Vous pouvez alors passer à la suite.

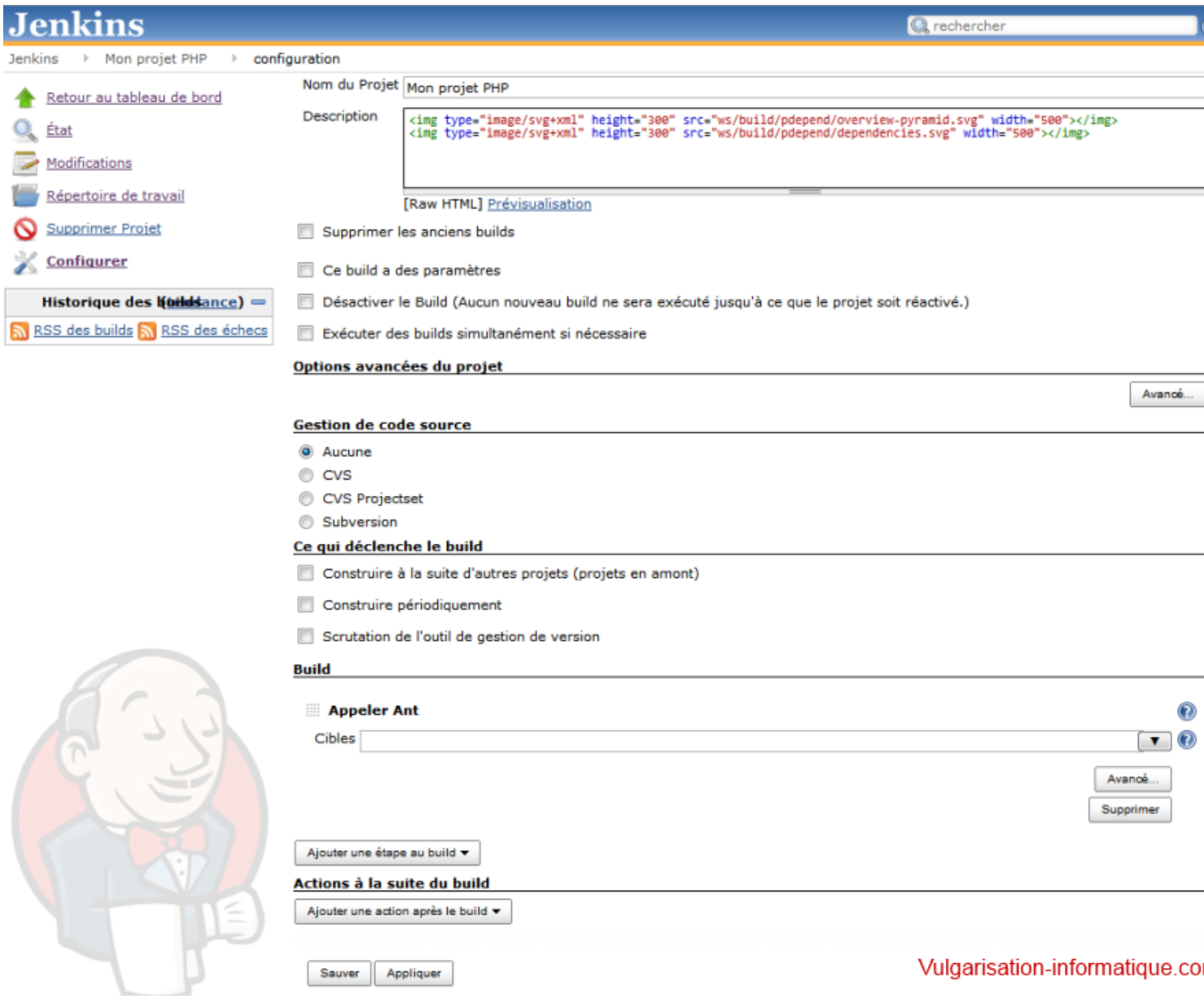
Configuration de Jenkins pour PHP et Wamp

Cliquez sur le lien (en haut à gauche) intitulé **Nouveau job** pour démarrer. En PHP, vous aurez la plupart du temps à lancer des scripts pour effectuer des tests et génération de **builds**. Un build est un projet "compilé" (que l'on peut assimiler à un package) stocké sur le serveur Jenkins. En général, un build correspond à une version compilée de votre programme. Dans le cas d'un site web dynamique PHP, le langage n'étant pas compilé, vous pouvez comparer un build à une version figée de votre site à l'instant de création du build.

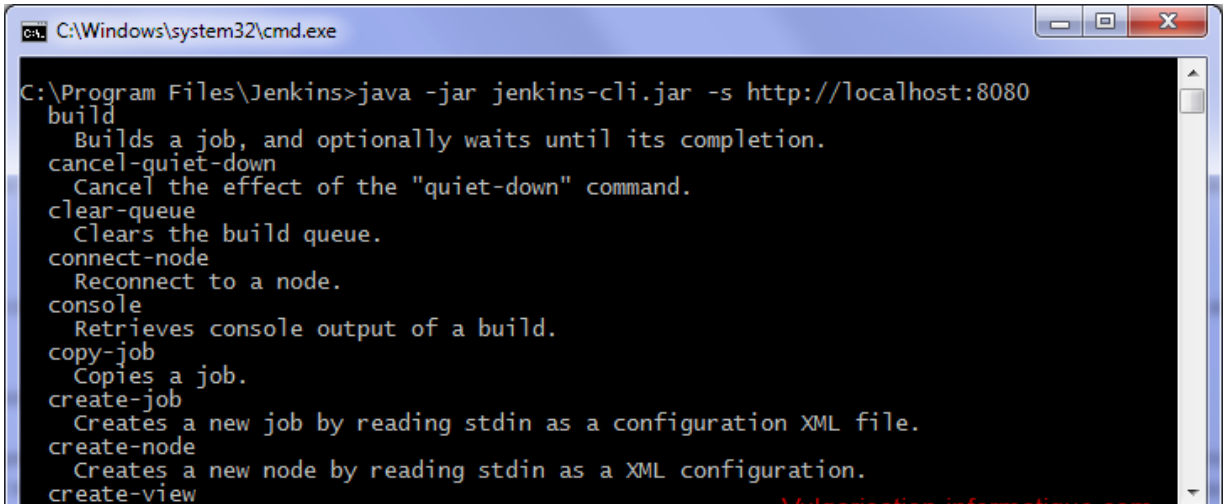
Donnez un nom à votre projet. Nous allons nous baser sur le template téléchargé plus haut, choisissez l'option **Copier un job existant** et tapez dans la zone **php** puis cliquez sur **php-jenkins-template**.



Contrôlez que vous ayez bien les mêmes informations que sur la capture suivante, puis cliquez sur **Sauver** :



Téléchargez **jenkins-cli.jar** à l'adresse suivante : http://localhost:8080/jnlpJars/jenkins-cli.jar puis enregistrez-le dans le répertoire racine de jenkins. Ouvrez une console (cmd) puis placez-vous dans le répertoire jenkins. Exécutez ensuite la commande suivante : **java -jar jenkins-cli.jar -s http://localhost:8080**. Vous devriez avoir un long message comme ceci :

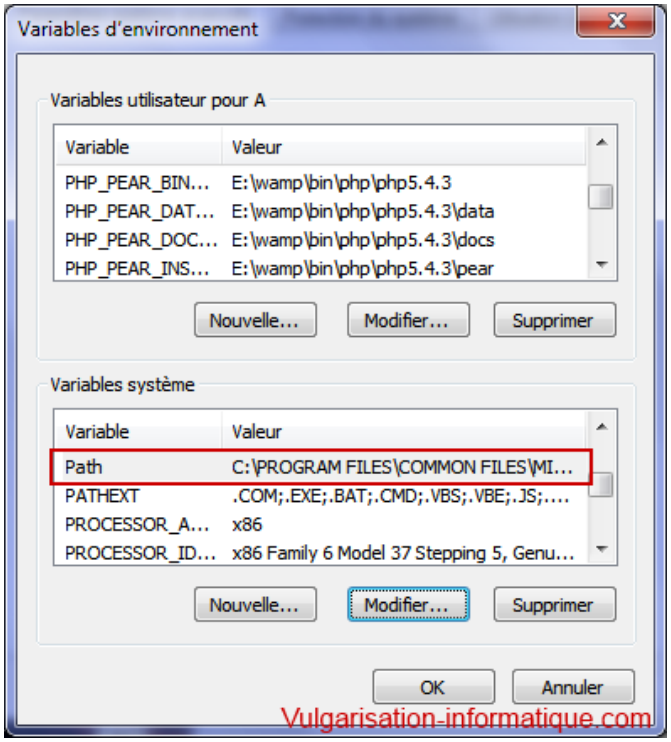


Téléchargez Apache ANT (lien direct vers le zip version 1.9.3 : Apache ANT 1.9.3) puis décompressez ce zip dans le répertoire dans le répertoire **bin** de Wamp (cela donne pour mon exemple **E:\wamp\bin**). Vous devez avoir dans le répertoire bin de Wamp, le répertoire d'apache Ant.

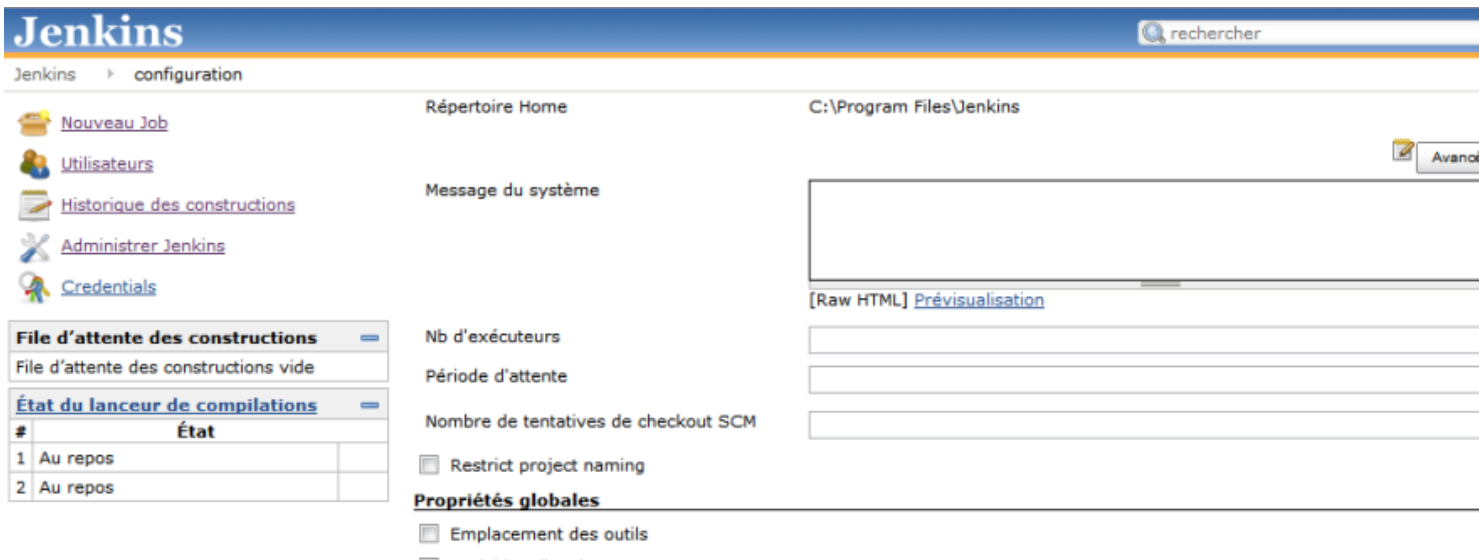
Définissez de nouvelles variables d'environnement. Pour ce faire, effectuez un clic avec le bouton droit sur l'icône **Ordinateur** (située sur le bureau), puis cliquez sur **Propriétés**, puis sur **Paramètres système avancés** et enfin sur le bouton **Variables d'environnement**.

Dans la zone **Variables système**, cliquez sur le bouton **Nouvelle**, puis créez les deux variables suivantes successivement :

JAVA_HOME- ayant pour valeur **C:\Program Files\Java\jdk1.6.0_39**- (si la version de votre jdk est 1.6.0_39, sinon remplacez par le répertoire qui existe sur votre ordinateur).
ANT_HOME- avec la valeur correspondant au répertoire **bin**- du répertoire d'Apache ANT, dans mon cas : **E:\wamp\bin\apache-ant-1.9.3**
-Repérez la variable d'environnement nommée **PATH**- et ajoutez à la suite la valeur **;C:\Program Files\Java\jdk1.6.0_39\bin**- (n'oubliez pas le point-virgule qui sert à séparer les autres valeurs déjà présentes, et même chose concernant la version du jdk : indiquez le répertoire correspondant à votre emplacement.)



Fermez toutes les options Windows puis retournez dans Jenkins. Cliquez à gauche sur **Administrer Jenkins** puis sur **Configurer le système**. Vous vous retrouvez devant un écran ressemblant à celui-ci :



Cochez la case **Variables d'environnement** puis cliquez sur le bouton **Ajouter** qui apparait à droite de la mention **Liste des paires clé-valeur**.

Dans la zone **Nom** indiquez la valeur suivante : **PATH** puis renseignez dans la valeur la même valeur que la variable d'environnement Windows nommée **PATH** et vue précédemment dans ce tutoriel, puis cliquez sur le bouton **Enregistrer**.

Arrêtez puis relancez le service Jenkins.

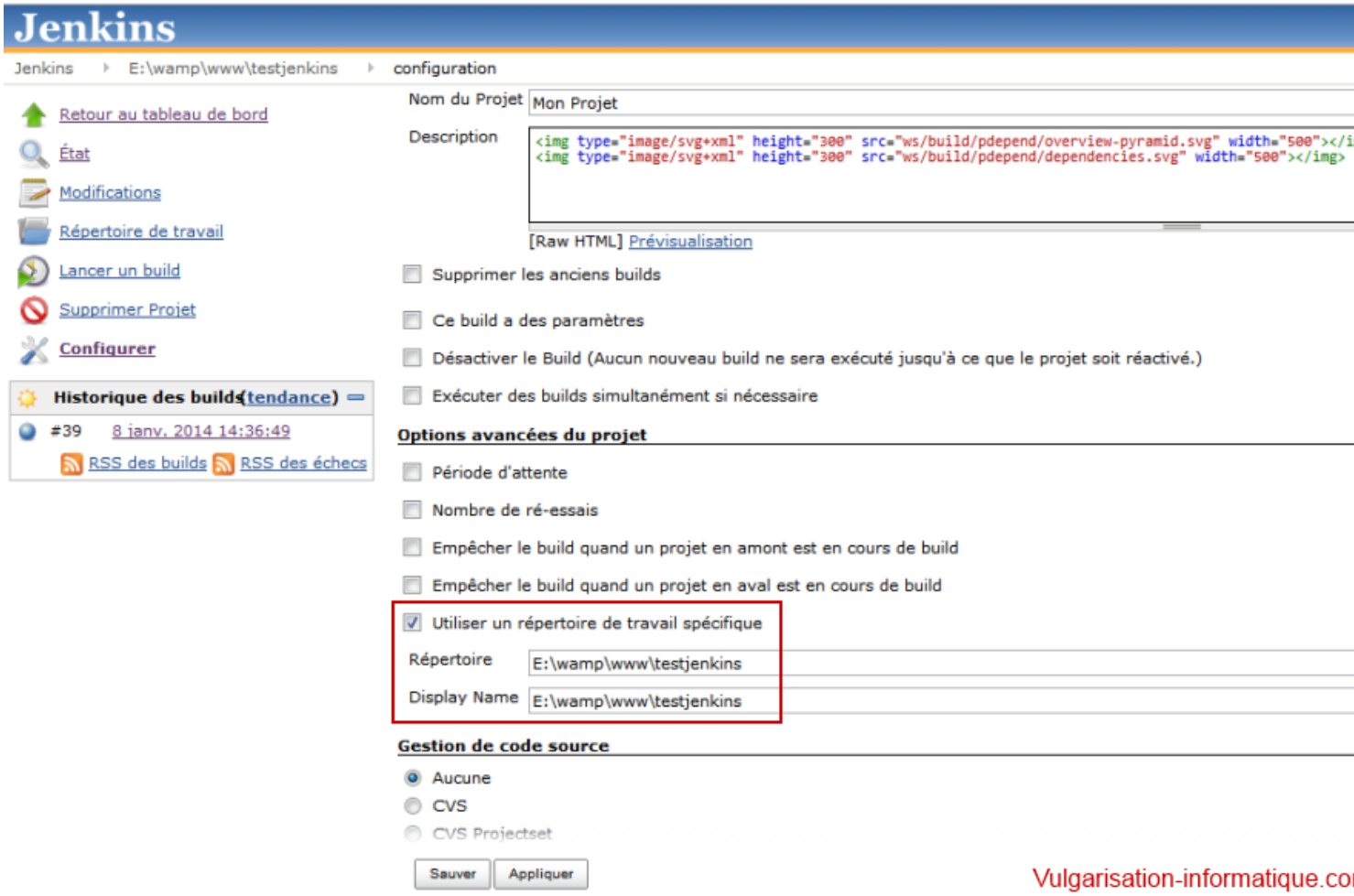
Téléchargez le fichier build.xml puis placez-le a la racine de votre projet PHP. Si vous analysez bien ce fichier, vous y trouverez les traces des commandes qui seront exécutées par Jenkins. Vous pouvez d'ailleurs les exécuter une par une dans une console pour voir le résultat !

Téléchargez le fichier phpmd.xml puis placez-le dans le répertoire **build** de votre projet PHP. Si le répertoire build n'existe pas, créez-le.

Téléchargez le fichier phpcs.xml puis placez-le dans le répertoire **build** de votre projet PHP.

Téléchargez le fichier phpunit.xml puis placez-le dans le répertoire **tests** de votre projet PHP. Si ce répertoire n'existe pas, créez-le à la racine de votre projet.

Il vous faut modifier le répertoire de travail pour qu'il pointe sur le répertoire racine de votre projet. Pour ce faire, dans Jenkins, cliquez à gauche sur **Configurer** lorsque vous vous situez dans votre projet. Cliquez sur le bouton **Avancé** puis renseignez le répertoire racine de votre projet (dans mon cas : **E:\wamp\www\testjenkins**).



Lancement du build

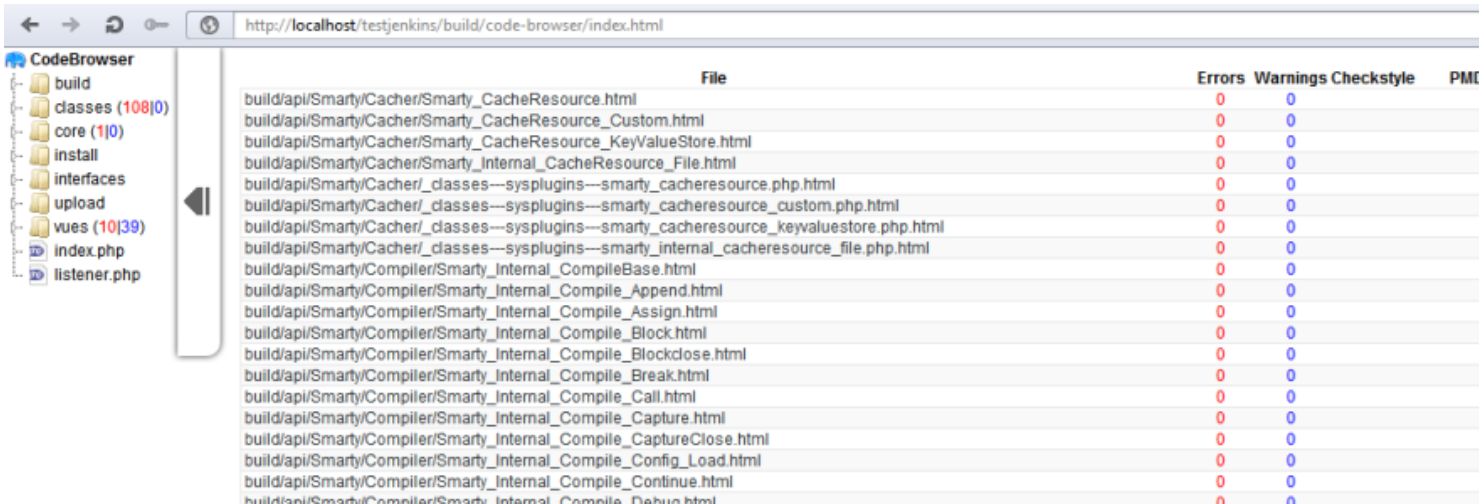
Rendez-vous dans votre projet, puis cliquez sur **Lancer un build**. Patientez quelques minutes (en fonction de la taille de votre projet).

Cliquez sur le build puis, dans le menu à gauche, cliquez sur **Sortie console**. Vous verrez ce qu'il se passe lorsque votre build sera généré.

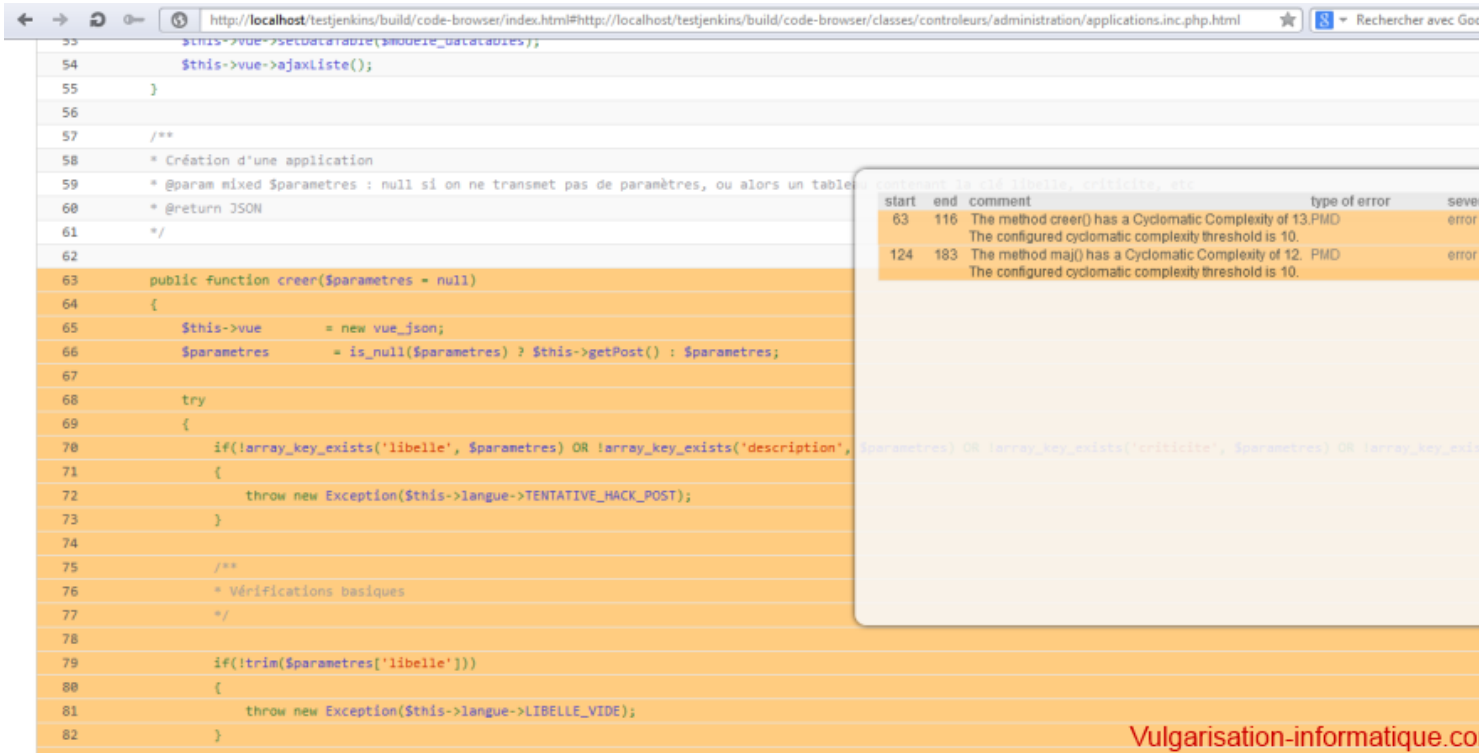
Vous allez maintenant pouvoir voir les résultats de votre code. Pour ce faire, vous avez tout d'abord accès à plusieurs choses situées dans le répertoire **build** :

Documentation générée par PHPDoc
Dans le répertoire **api** se trouve l'ensemble de la documentation de votre projet générée automatiquement par PHPDoc. Vous pouvez la consulter dans votre navigateur en vous rendant à l'adresse de votre projet suivi de **/build/api**. (cela donne chez moi ceci : **http://localhost/testjenkins/build/api**) :

Rapport d'erreurs généré par Code Browser
Dans le répertoire **code-browser** se trouve une page web recensant l'ensemble des erreurs (faibles ou critiques) de vos fichiers PHP :



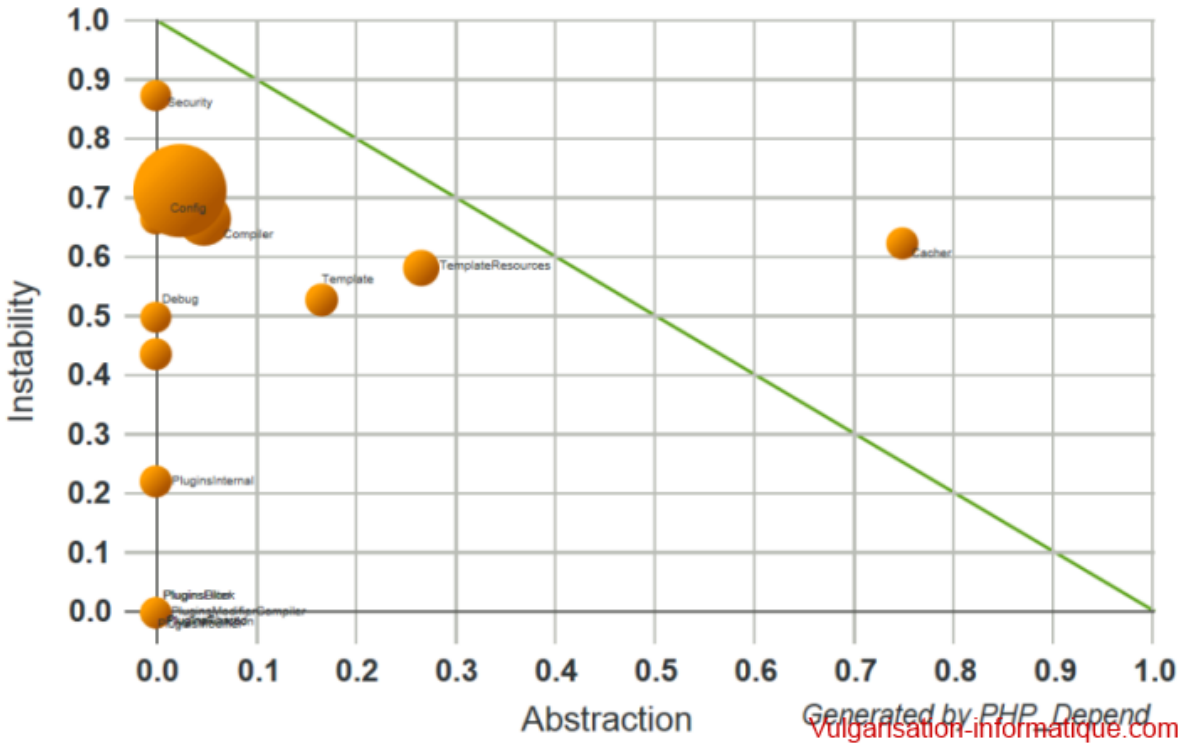
Les erreurs faibles (warning) apparaissent en bleu, les erreurs fortes en rouge. Vous pouvez via l'arborescence de gauche cliquer sur les répertoires et fichiers incriminés pour consulter le détail des erreurs :



Diagrammes de dépendances, stabilité et pyramide générés par PHP_Depend

Dans le répertoire **pdepend** se trouvent deux fichiers .svg (que vous pouvez directement ouvrir avec votre navigateur Web pour peu qu'il supporte le SVG).

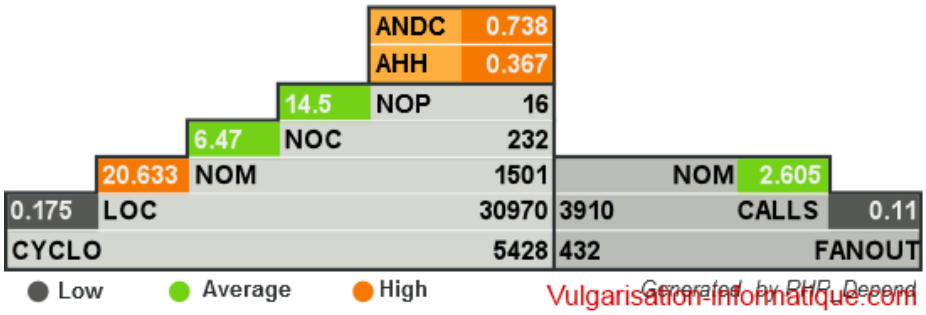
La première image se nomme **dependencies.svg** et est constituée d'un graphique à deux axes :



-En abscisses, se trouve le niveau d'abstraction. Plus il est élevé, plus votre projet est constitué de classes abstraites et d'interfaces.
-En ordonnées, se trouve le niveau d'instabilité, calculé notamment en fonction du nombre de dépendances entre classes. Plus il y a de dépendances, plus il est élevé (pour simplifier).

Le cas idéal serait celui où vos fichiers se situeraient près de la ligne verte. En pratique, qu'ils soient en dessous est déjà une bonne chose. Pour plus d'informations, vous pouvez consulter le site PHP Depend.

La seconde image se nomme **overview-pyramid.svg**. Ouvrez-la de la même manière :



Elle est un peu plus complexe à analyser. Je vous encourage à consulter ce lien http://manuel-pichler.de/archives/31-Using-the-Overview-Pyramid.html si vous souhaitez plus d'informations (en anglais).

Ce tutoriel touche à sa fin. Vous devriez voir cet aperçu lorsque vous consultez le dernier build de votre projet :