Creer son site web en php _ installer xdebug

- Objet : environnement de développement php
- Niveau requis :
  débutant, avisé
- Commentaires : *Contexte d'utilisation du sujet du tuto.*
- Débutant, à savoir : Utiliser GNU/Linux en ligne de commande, tout commence là !.

Introduction

- Dans ce wiki, je pars du principe qu'Apache est déjà installé, ainsi que php7.3

Si ce n'est pas le cas : prendre en main apache2

```
php -v
```

```
PHP 7.3.19-1~deb10u1 (cli) (built: Jul  5 2020 06:46:45) ( NTS )
Copyright (c) 1997-2018 The PHP Group
Zend Engine v3.3.19, Copyright (c) 1998-2018 Zend Technologies
    with Zend OPcache v7.3.19-1~deb10u1, Copyright (c) 1999-2018, by Zend
Technologies
```

Installation de Xdebug sous Apache en mode PHP

* Il est possible d'installer xdebug avec apt, mais ce sera pas forcément la dernière version de xdebug soit avec pecl.



Ne pas faire les deux installations, soit avec apt comme ci-dessus, soit en utilisant pecl, l'utilitaire php.

Installation avec APT

```
apt-get install php-xdebug
```

Installation avec PECL

```
apt install php-pear
```

* Pour utiliser phpize

```
apt install php-dev
```

* Installation de l'extention php xdebug

```
pecl install xdebug
```

- Pendant l'installation :

Le retour de la commande donne une information qu'il faut conserver et qui ressemble à ceci :

```
zend_extension=/usr/lib/[phpversionX]/20131226/xdebug.so
```

Il faut bien copier la ligne qui comporte des chiffres et surtout pas celle de indiquée par l'installateur `zend_extension=xdebug.so`.

Il faudra la copier dans le fichier php.ini de Apache. En effet, pour faire du debugage entre son IDE et son navigateur, il est suffisant de configurer le fichier php.ini du dossier `/etc/php/7.3/apache2/` correspondant à la version de php utilisée.

Configuration de php



à faire aussi bien après l'installation de XDEBUG avec apt qu'après celle avec pecl.

- On édite le fichier php.ini :



Le fichier de php.ini est toujours celui qui est dans le répertoire apache2 de la **version** de php utilisée !

Par exemple ici on utilise la version **7.3** de php, le chemin est donc `/etc/php/7.3/apache2/php.ini`.

```
vim /etc/php/7.3/apache2/php.ini
```

```
[Xdebug]
zend_extension=/usr/lib/php/20180731/xdebug.so
xdebug.mode = debug,develop,trace
xdebug.start_with_request = yes
xdebug.remote_host=localhost
xdebug.remote_port=9003
xdebug.remote_log=/tmp/xdebug.log
```

- À partir de maintenant, la fonction php phpinfo() permet de vérifier que xdebug est bien installé:





On voit bien le résultat de la configuration de xdebug dans php.ini.

xdebug.mode = debug,develop,trace

- debug : cette option permet de faire du pas à pas. Quand on l'ajoute on voit ("*Enabled*") dans le tableau ci-dessous.
- develop: cette option permet d'utiliser la fonctionnalité "*Development Aids*". Utile pour surchager var_dump()
- trace : cette option permet de debuger les fonctions php.

On peut en choisir d'autres. La documentation donne le détail des options possibles pour xdebug.mode ici.



Il faut mettre le port 9003 et non pas 9000 depuis la version 3 de xdebug, sans quoi rien ne marche !

- Il est possible de faire du pas dans son IDE préféré après avoir rechargé apache !

```
service apache2 reload
```

Pour debuger en ligne de commande

Pour ceux qui préfèrent debuger en ligne de commande, il faut ajouter la configuration pour xdebug dans le fichier php.ini de dossier cli car pour l'instant php n'est pas informé de la présence de xdebug.

On édite le php.ini du dossier cli : `vim /etc/php/<version_de_php>/cli/php.ini`. Par exemple dans ce cas :

```
vim /etc/php/7.3/cli/php.ini
```

```
[Xdebug]
zend_extension=/usr/lib/php/20180731/xdebug.so
xdebug.mode = debug,develop,trace
xdebug.start_with_request = yes
xdebug.remote_host=localhost
xdebug.remote_port=9003
xdebug.remote_log=/tmp/xdebug.log
```

Installation de Xdebug sous Apache en mode FPM

Installation avec APT

```
apt-get install php-xdebug
```

Configuration

- **Si on est en mode php-fpm** :

On édite le fichier `20-xdebug.ini` ou `/etc/php/8.2/fpm/php.ini`.

```
nano /etc/php/<version_de_php>/fpm/conf.d/20-xdebug.ini
```

- **Si on est en mode apache** :

On édite le fichier /etc/php/<version_de_php>/apache2/php.ini

Quelque soit le mode on vérifie qu'on a bien ajouté à la fin les lignes suivantes :

```
zend_extension=debug.so
xdebug.mode = debug,develop,trace
xdebug.start_with_request = yes
xdebug.remote_host=localhost
xdebug.remote_port=9003
xdebug.remote_log=/tmp/xdebug.log
```

On recharge la configuration si les lignes de configuration ci-dessus n'y étaient pas.

- **Si on est en mode php-fpm** :

```
systemctl restart php<version_de_php>-fpm
```

- **Si on est en mode apache** :

```
systemctl restart apache2
```

Verification



C'est une autre méthode qu'en utilisant la fonction php `<?php phpinfo() ?>`

Et maintenant on peut voir que l'installation est faite en faisant :

```
php -i | grep -i "xdebug.mode"
```

```
xdebug.mode => debug,develop,trace => debug,develop,trace
```

Utilisation avec l'EDI CODIUM

Installation de l'ide

```
apt update
```

```
apt install software-properties-common apt-transport-https curl
```

```
curl -sSL https://packages.microsoft.com/keys/microsoft.asc | gpg --dearmor
> /usr/share/keyrings/vscode.gpg
```



ça créé le fichier `vscode.gpg` dans `/usr/share/keyrings/`

```
echo "deb [signed-by=/usr/share/keyrings/vscode.gpg]
https://packages.microsoft.com/repos/vscode stable main" >
/etc/apt/sources.list.d/vscode.list
```

```
apt update
```

```
apt install code
```

Installation de l'extension php debug de Felix Becker

php debug

Paramètres du fichier launch.json



C'est la même configuration avec l'ide svcode.

```
{
    // Use IntelliSense to learn about possible attributes.
    // Hover to view descriptions of existing attributes.
    // For more information, visit:
https://go.microsoft.com/fwlink/?linkid=830387
    "version": "0.2.0",
    "configurations": [
        {
            "name": "Listen for XDebug",
            "type": "php",
            "request": "launch",
            "port": 9003
        },
        {
            "name": "Launch currently open script",
            "type": "php",
```

```
            "request": "launch",
            "program": "${file}",
            "cwd": "${fileDirname}",
            "port": 9003
        }
    ]
}
```

paramètres du fichier /etc/php/8.2/fpm/php.ini ou etc/php/8.2/apache2/php.ini

C'est comme pour tout autre EDI ! Un petit rappel :)

```
[xdebug]
;; xdebug 3
zend_extension=xdebug.so
xdebug.mode=debug,develop,trace
xdebug.mode=debug
xdebug.start_with_request = yes
xdebug.client_host=localhost
xdebug.client_port=9003
```



On veut déboguer avec le serveur interne de symfony, il faut le lancer avec `sudo symfony server:start –no-tls –port=80`.

Il faut alors bien vérifier que apache est éteint : `systemctl stop apache2`



On veut déboguer avec le serveur interne de symfony, en restant sur le port par défaut, on lance comme d'habitude `symfony server:start –no-tls` .

On peut laisser apache tourner `systemctl start apache2` sur le port 80.

Mais dans ce cas, il faut indiquer dans `launch.json` le port à écouter en ajoutant la ligne `url“: “http://127.0.0.1:8000`

```
    "configurations": [
        {
            "name": "Listen for Xdebug",
            "type": "php",
            "request": "launch",
            "port": 9003,
            "url": "http://127.0.0.1:8000"
        },
```

Utilisation avec PHPSTORM EDI

- Aller dans l'onglet Run
- Choisir `Edit configurations…`
- On clique sur + (en haut à gauche), PHP web Page
- Dans `Name` : on met ce qu'on veut, par exemple `local_dev`
- Dans `Server`: on a cliqué sur les trois petits points … qui ouvre une nouvelle fenêtre qui crée un serveur.

- Dans `Name` on met le même nom que précédemment, par exemple `local_dev`
- Dans `Host` on met : localhost
- Dans `Port` on laisse `80`
- Dans `Debugger` on met Xdebug
- On clique sur Apply puis sur OK.

- On peut vérifier dans `Debug pre-configuration` sur le lien `Validate` pour voir si le paramétrage est bon.
- On clique sur OK



Faire attention qu'à côté du petit Bug vert qui permet de lancer le debuggage, on a bien sélectionner dans la liste déroulante, la configuration qu'on vient de créer `local_dev`.



Comme pour les EDI codium et vscode, si on veut déboguer en utilisant le serveur interne de symfony au lieu d'apache pendant le développement, il faut alors modifier le port 80 pour 8000, comme dans l'image ci-dessous.



Utilisation avec NetBeans PHP IDE

On installe NetBeans PHP

- On télécharge :

Voir cette page : https://netbeans.org/downloads/

Ou pour un ordinateur 64bits :

```
cd ~/Téléchargements && wget
http://download.netbeans.org/netbeans/8.1/final/bundles/netbeans-8.1-php-lin
ux-x64.sh
```

- on installe NetBeans :

```
sh netbeans-8.1-php-linux-x64.sh
```



L'installation crée un dossier dans le répertoire courant de l'utilisateur.

- Pour déinstaller NetBeans :

```
cd /home/hypathie/netbeans-8.1/
```

```
sh uninstall.sh
```

Paramètrer NetBeans

l'onglet `Debugging` Dans la bare de menu choisir `Tools` puis `Options`, puis PHP, puis l'onglet `Debeugging` :

Dans le champ `Debeugger` port il faut mettre la même valeur que celle choisie dans le fichier php.ini, par exemple 1000.

Importer le projet du serveur dans NetBeans

On travaillera en local dans un environnement de développement et non dans le projet déployé sur le serveur.

On envoie sur l'ordinateur de développement le dossier de code source du projet qui est sur le serveur.

On se connecte en `ssh` sur son serveur. Depuis le serveur on envoie sur l'ordinateur son projet.

```
cd /var/www
```

Par exemple :

```
scp -r titi@75.239.10.80:essai/ toto@71.222.30.40: .
```

1. `-r` : signifie récursif, c'est pour envoyer un dossier plutôt qu'un fichier;
2. `titi` est le nom de l'utilisateur du serveur ;
3. `75.239.10.80` : ip du serveur ;(
4. `:` signifie le répertoire courant où il y a le dossier à copier ;
5. `essai/` : le dossier à envoyer ;
6. `toto` : est l'utilisateur de l'ordinateur de développement ;
7. `71.222.30.40` : est l'ip de l'ordinateur de développement si elle est de type 192.168.0.xx, il faut paramétrer le routeur de son FAI pour les BAUX permanents de l'ordinateur de développement sur son ip 192.168.0.xx afin de faire une redirection vers celle-ci);
8. `: .` : signifie que le dossier arrivera dans le répertoire courant de toto.

- Voir :
  - scp
  - scp envoi de clé ssh|Envoi de clés RSA pour SSH en scp

On charge dans NetBeans le dossier du projet

Pour travailler en local, sur l'ordinateur de développement, on a installer Apache et créé un virtualhost pour le site.

1. On créer un nouveau projet ;
2. On choisit PHP (catégories) et `PHP Application with Existing Sources` ;
3. On choisit le dossier qu'on a reçu du serveur (attention de l'avoir bien placé dans le même endroit que déclaré à Apache au niveau de `/etc/apache2/sites-available`);
4. Pour `Run as` on laisse `local web server`:
5. `url` : On met la même que celle qui permet d'accéder à son site depuis le navigateur de l'ordinateur de développement. (Voir comment paramétrer `/etc/hosts` ici : https://debian-facile.org/utilisateurs:hypathie:tutos:creer-son-site-web-en-php-_-prendre-en-main-apache2-sur-jessie#ajouter-son-domaine-dans-etc-hosts)

From:
http://debian-facile.org/ - **Documentation - Wiki**

Permanent link:
http://debian-facile.org/utilisateurs:hypathie:tutos:creer-son-site-web-en-php-_-installer-xdebug

Last update: **19/12/2022 19:12**