Configuration de XDebug sur VsCode et Docker

Attention, cette configuration fonctionne pour Docker sous Windows. Cela devrait également fonctionner sous Mac, mais peut nécessiter quelques modifications (non-testé). Sous Linux, la gestion des hosts dans le compose.yaml sera sûrement à adapter.

Ressources utilisées pour réaliser ce tutoriel :

- https://stevegeorge.net/2022/02/10/setting-up-xdebug-3-on-a-docker-container-with-visual-studio-code/
- https://www.jetbrains.com/help/phpstorm/configuring-xdebug.html#configuring-xdebug-docker

1- Introduction

Ce guide va vous aider à installer XDebug : un debugger PHP.
L'utilisation d'un debugger est une grande aide, car elle vous permettra de faire du « pas-à-pas » dans votre code et ainsi, connaitre la valeur des variables de votre application tout au long de l'exécution de votre code.

2- Installation de XDebug 3 dans le container Docker

Ajouter les lignes suivantes dans le Dockerfile de votre container Apache pour installer et configurer XDebug :

```
RUN pecl install xdebug \
  && docker-php-ext-enable xdebug
RUN echo "xdebug.mode=debug" >> /usr/local/etc/php/conf.d/docker-php-ext-xdebug.ini \
  && echo "xdebug.client_host = host.docker.internal" >> /usr/local/etc/php/conf.d/docker-php-ext-xdebug.ini \
  && echo "xdebug.start_with_request = yes" >> /usr/local/etc/php/conf.d/docker-php-ext-xdebug.ini
```

3- Mettre à jour le fichier compose.yaml

Si vous utilisez la stack lamp que je fournis (https://github.com/guirod/docker-lamp.git), vous pouvez simplement mettre à jour le repo (dans le dossier docker-lamp, exécutez « git pull »).
Il vous faut ensuite mettre à jour le fichier compose.yaml pour qu'il ajoute les variables d'environnement propres à XDebug et la configuration des hosts.
Ajouter dans la définition de votre service Apache :

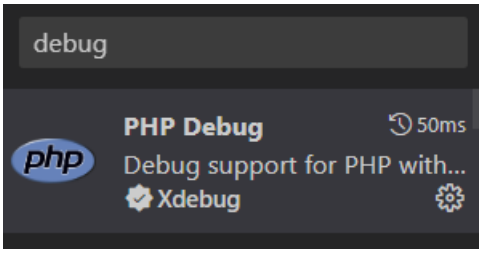
```
extra_hosts:
    - "host.docker.internal:host-gateway"
  environment:
    XDEBUG_MODE: develop,debug
    XDEBUG_CONFIG: client_host=host.docker.internal start_with_request=yes
```

4- Rebuild et relancer le container

```
#Si les containers sont en cours d'exécution
docker-compose down

#Rebuild le container
docker-compose build

#Relancer la stack
docker-compose up -d
```

5- Installation de l'extension VSCode

Installer l'extension « **PHP Debug** ».



6- Configuration de VS Code

Ouvrir Git Bash à la racine de votre Workspace et taper les commandes suivantes :

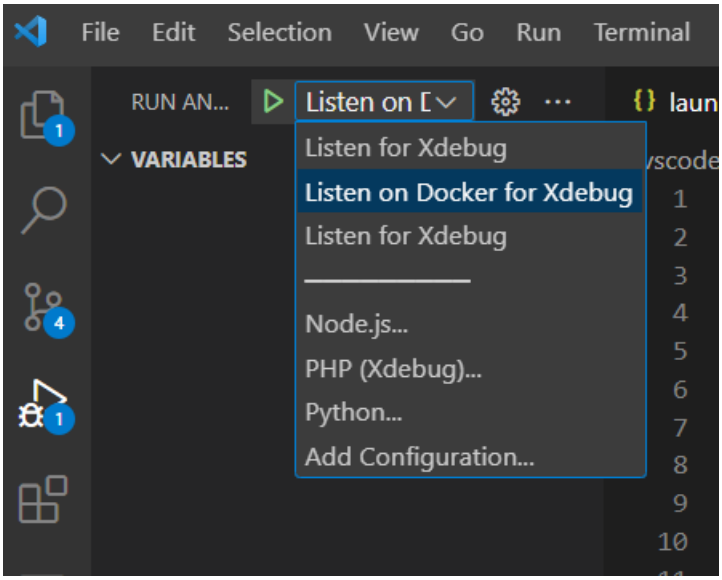
```
# création du répertoire .vscode (Inutile si le répertoire existe déjà)
mkdir .vscode

#Creation d'un fichier launch.json
touch .vscode/launch.json
```

Copiez le code suivant dans le fichier « launch.json » :

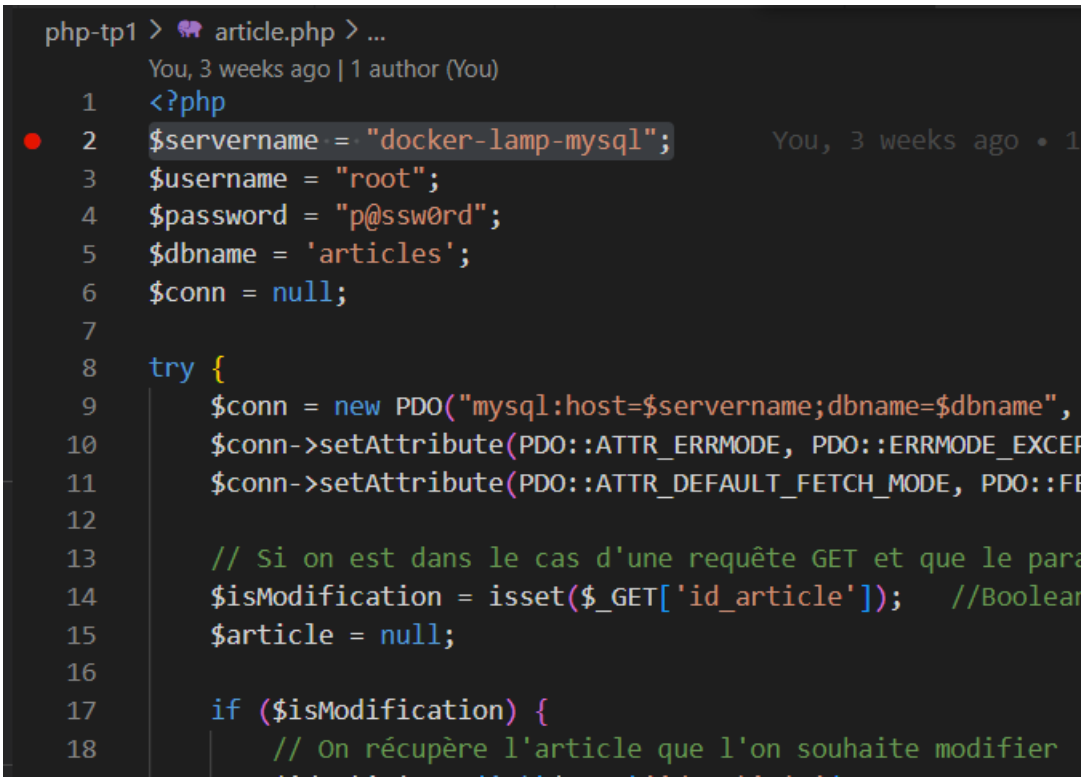
```
{
    // Use IntelliSense to learn about possible attributes.
    // Hover to view descriptions of existing attributes.
    // For more information, visit:
https://go.microsoft.com/fwlink/?linkid=830387
    "version": "0.2.0",
    "configurations": [
        {
            "name": "Listen for Xdebug",
            "type": "php",
            "request": "launch",
            "port": 9003
        },
        {
            "name": "Listen on Docker for Xdebug",
            "type": "php",
            "request": "launch",
            "port": 9003,
            "pathMappings": {
                "/var/www/html": "${workspaceFolder}"
            }
        }
    ]
}
```

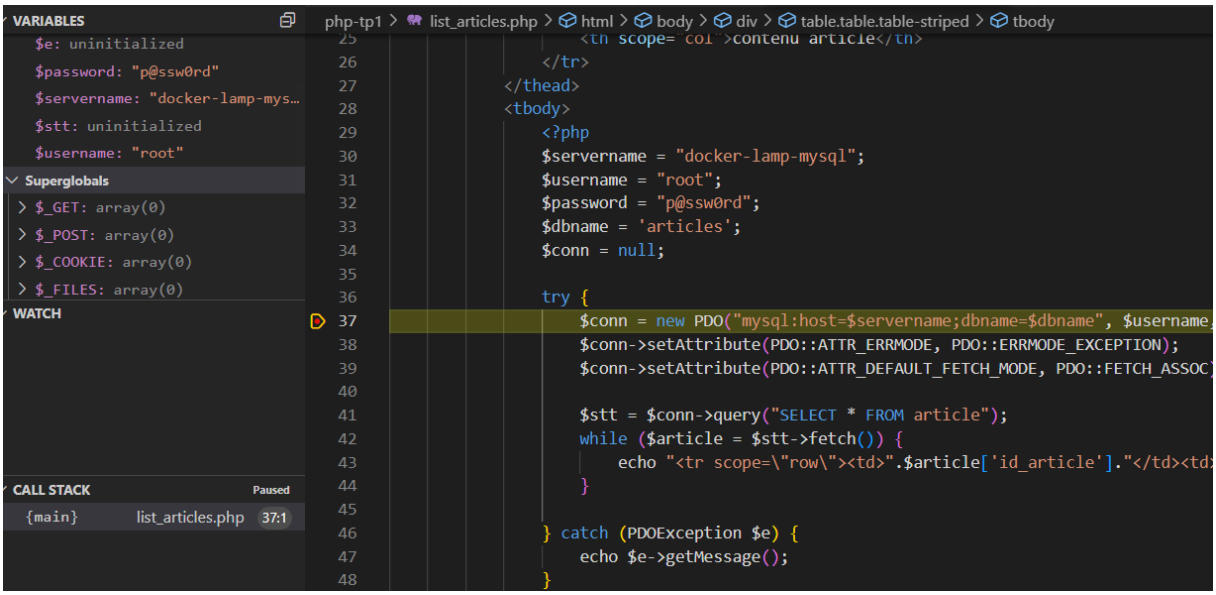
7- Enjoy

Vous pouvez maintenant lancer le debugger via vscode en utilisant « lister on docker for xdebug » dans l'onglet de debug de vscode.



Vous n'avez plus qu'à mettre des points d'arrêt en cliquant à gauche du bout de code sur lequel vous souhaitez vous arrêter puis vous pourrez faire du pas-à-pas à partir d'ici.




VARIABLES
$e: uninitialized
$password: "p@ssw0rd"
$servername: "docker-lamp-mys…
$stt: uninitialized
$username: "root"
Superglobals
$_GET: array(0)
$_POST: array(0)
$_COOKIE: array(0)
$_FILES: array(0)
WATCH
CALL STACK
Paused
{main}
list_articles.php
37:1
php-tp1 > list_articles.php > html > body > div > table.table.table-striped > tbody
<th scope="col">contenu article</th>
</tr>
</thead>
<tbody>
<?php
$servername = "docker-lamp-mysql";
$username = "root";
$password = "p@ssw0rd";
$dbname = 'articles';
$conn = null;
try {
$conn = new PDO("mysql:host=$servername;dbname=$dbname", $username,
$conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
$conn->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_ASSOC)
$stt = $conn->query("SELECT * FROM article");
while ($article = $stt->fetch()) {
echo "<tr scope=\"row\"><td>".$article['id_article']."</td><td
}
} catch (PDOException $e) {
echo $e->getMessage();
}