

Envoi de données à partir d'un Arduino Yun sur une base de données

Sommaire :

I / Pré-requis réseau & PHP	1
II / Script PHP requête MySQL	2
III / Transfert du script vers Arduino	5
IV / Automatisation d'envoi de données	6

I / Pré-requis réseau & PHP

Afin de suivre convenablement cette documentation, il vous sera demandé quelques pré-requis matériels ainsi que logiciels.

Pour ce faire nous aurons besoin que :

- Votre Arduino soit connecté en Ethernet sur un HUB avec l'ordinateur **accueillant la base de données MySQL en local**. (Pour cela veuillez vous référer à la documentation concernant l'utilisation d'un Arduino Yun via connection Ethernet)
- Vous devez disposer de php-cli + l'extension MySQL & PDO pour PHP installé sur la partie Linux de votre Arduino. Pour cela voyons les étapes ci-dessous.

Installation de PHP et ses extensions sur la partie Linux de l'Arduino (manipulation à faire sur un point d'accès internet) :

- 1 . Se connecter en **SSH** sur votre Arduino
- 2 . Entrer la commande suivante afin de mettre à jour vos dépôts : *opkg update*
- 3 . Installer nano avec : *opkg install nano* (optionnel)
- 4 . Installation de php-cli : *opkg install php5-cli*
- 5 . Installation du php-mysqli : *opkg install php5-mod-mysqli*
- 6 . Installation de php-mod-pdo-mysql : *opkg install php5-mod-pdo-mysql*
- 7 . Dernière installation, le mod php-pdo : *opkg install php5-mod-pdo*

```
root@Arduino:~# opkg install php5-cli
Installing php5-cli (5.4.5-3) to root...
Downloading http://downloads.arduino.cc/openwrt-yun/1/packages/php5-cli_5.4.5-3_ar71xx.ipk.
Installing php5 (5.4.5-3) to root...
Downloading http://downloads.arduino.cc/openwrt-yun/1/packages/php5_5.4.5-3_ar71xx.ipk.
Installing libxml2 (2.7.8-2) to root...
Downloading http://downloads.arduino.cc/openwrt-yun/1/packages/libxml2_2.7.8-2_ar71xx.ipk.
Configuring libxml2.
Configuring php5.
Configuring php5-cli.
root@Arduino:~#
```

II / Script PHP requete MySQL

Afin d'envoyer toutes les données sur notre base de données nous devons passer par un script intermédiaire étant donné que Arduino n'est pas capable directement de communiquer avec une base de données. Pour ce faire nous allons exécuter un script PHP (ou autre langage suivant votre préférence), stocké sur la partie Linux de l'Arduino Yun, nous permettant d'envoyer les informations désirées. Ce script sera en réalité exécuté sur la partie Linux sous forme d'une commande exécutée. Pour exécuter cette commande via notre Arduino référez vous à la suite de la documentation ([IV](#)).

Pour l'instant voyons le script PHP en détail :

```
send.php
1  <?php
2
3      $line_id = $argv[1];
4      $stop_id = $argv[2];
5      $bus_id = $argv[3];
6
7      try {
8
9          $db = new PDO('mysql:host=arret;dbname=sin', 'hugo', '123');
10         $sql = $db->prepare("INSERT INTO schedule (date_time, line_id, stop_id, bus_id) VALUES (NOW(), :line_id, :stop_id, :bus_id)");
11         $sql->bindParam(':line_id', $line_id);
12         $sql->bindParam(':stop_id', $stop_id);
13         $sql->bindParam(':bus_id', $bus_id);
14         $sql->execute();
15
16         $sql->debugDumpParams();
17         var_dump($sql->errorInfo());
18
19     } catch (PDOException $e) {
20         echo 'Échec lors de la connexion : ' . $e->getMessage();
21     }
22
```

Le script complet

```

send.php
1  <?php
2
3      $line_id = $argv[1];
4      $stop_id = $argv[2];
5      $bus_id = $argv[3];
6

```

Récupération des arguments

Dans la partie ci-dessus du script nous récupérons les arguments de notre commande dans le tableau `$argv[]` respectivement avec les index souhaités, et nous les stockons dans des variables plus explicites.

Exemple `$argv[x]` : `php-cli send.php argument1 argument2 argument3 ...`
`$argv[0]` `$argv[1]` `$argv[2]` `$argv[3]` ...

```

7  try {
8
9      $db = new PDO('mysql:host=arret;dbname=sin', 'hugo', '123');
10     $sql = $db->prepare("INSERT INTO schedule (date_time, line_id, stop_id, bus_id) VALUES (NOW(), :line_id, :stop_id, :bus_id)");
11     $sql->bindParam(':line_id', $line_id);
12     $sql->bindParam(':stop_id', $stop_id);
13     $sql->bindParam(':bus_id', $bus_id);
14     $sql->execute();
15

```

Connexion à la BDD et exécution de la requête

Concernant la connexion à la BDD et la gestion des requêtes nous utilisons l'extension PDO pour des questions de facilité (installée dans les pré-requis).

`$db = new PDO('mysql:host=hostname;dbname=nom_base', 'login', 'password');`

host : Comme précisé, l'hostname de la base, dans mon cas *arret* qui correspond à une IP locale dans le fichier *hosts* du système.

dbname : Le nom de votre base, ici *sin*.

Et les login et password pour la connexion.

```
$sql = $db->prepare("INSERT INTO schedule (date_time, line_id, stop_id, bus_id) VALUES (NOW(), :line_id, :stop_id, :bus_id);");
```

S'en suit une requête dite préparée, qui ici fait une insertion dans la base. Vous pouvez remarquer les différentes données qui vont varier dans notre requête avec les `:` devant. Ici `:line_id`, `:stop_id`, `:bus_id` ce ne sont que des alias MySQL pour l'instant que nous allons définir réellement ici :

```
$sql->bindParam(':line_id', $line_id);
$sql->bindParam(':stop_id', $stop_id);
$sql->bindParam(':bus_id', $bus_id);
```

Nous attribuons chaque alias MySQL à une variable PHP, vous pouvez créer autant d'alias que vous le souhaitez et les attribuer à des variables PHP, il vous suffit simplement de les ajouter dans votre requête.

```
$sql->execute();
```

Cette commande ne fait qu'exécuter la requête une fois sa préparation finie.

```
19 } catch (PDOException $e) {
20     echo 'Échec lors de la connexion : ' . $e->getMessage();
21 }
22
```

catch en cas d'erreur

Toute la partie *catch* n'est qu'une information en cas de soucis lors de l'envoi, toute la partie *try* (plus haut) sera ignorée et nous récupérerons simplement le message d'erreur afin de déboguer.

Finalement pour exécuter ce script manuellement dans votre terminal **Linux** entrez la commande suivante dans le répertoire où se situe votre script (n'oubliez pas l'extension `.php` de votre script) :

```
php-cli nom_du_script argument1 argument2 etc ...
```

Lien pour le script complet : <http://pastebin.com/yrN17Bad>

III / Transfert du script vers Arduino

Tout d’abord, la suite de cette documentation se déroulera sur un poste sous [Linux](#) pour des raisons de simplicité.

Nous allons voir comment envoyer le fichier “send.php” que l’on a créé précédemment sur la partie Linux de la carte Yùn.

Premièrement, brancher la Yùn, puis, ouvrez un terminal de Linux :

```
maxime@maxime-HP-EliteBook-8730w:~$
```

Allez chercher votre fichier à envoyer (dans notre cas dans le dossier “ex”) :

```
maxime@maxime-HP-EliteBook-8730w:~$ cd ex/
maxime@maxime-HP-EliteBook-8730w:~/ex$ ls
send.php
```

Pour envoyer ce fichier sur la partie Linux, faites “*scp send.php login@ip_arduino:/www*” :

```
maxime@maxime-HP-EliteBook-8730w:~/ex$ scp send.php root@192.168.0.3:/www
root@192.168.0.3's password:
send.php                                100% 580      0.6KB/s   00:00
```

On peut maintenant vérifier, le send.php est bien à sa place :

```
root@Arduino:~# cd ..
root@Arduino:/# ls www/
cgi-bin          keystore_manager_example  send.php
index.html       luci-static
```

IV / Automatisation d'envoi de données

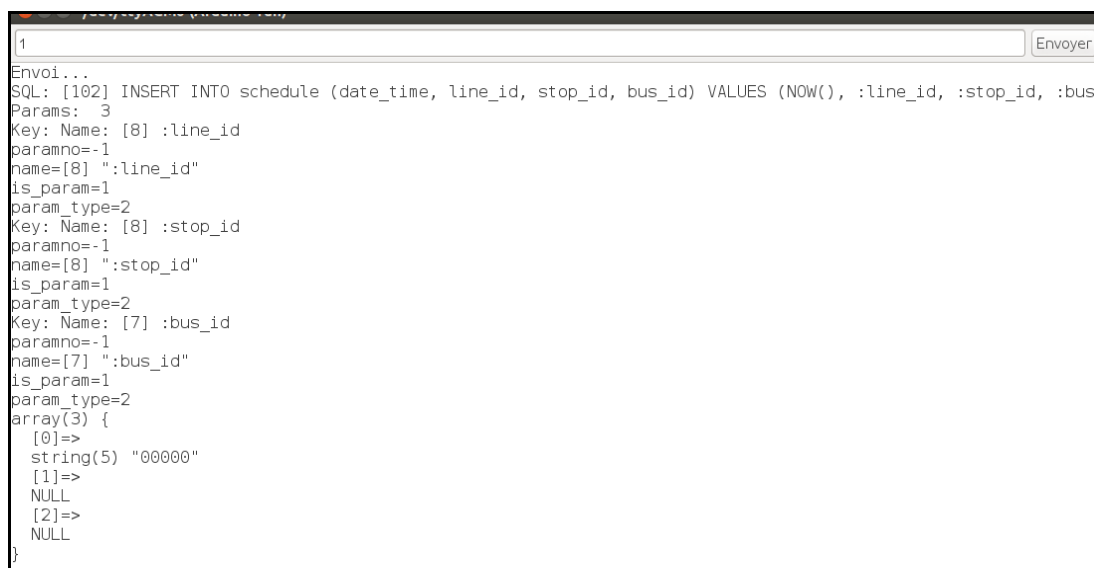
Toujours sur notre terminal Linux, voyons comment exécuter le script send.php.

Tout d'abord, pour exécuter "send.php", entrez la commande "*php-cli send.php line_id stop_id bus_id*" :

```
root@Arduino:~# php-cli /www/send.php 2 5 10
SQL: [102] INSERT INTO schedule (date_time, line_id, stop_id, bus_id) VALUES (NOW(), :line_id, :stop_id, :bus_id)
Params: 3
Key: Name: [8] :line_id
paramno=-1
name=[8] ":line_id"
is_param=1
param_type=2
Key: Name: [8] :stop_id
paramno=-1
name=[8] ":stop_id"
is_param=1
param_type=2
Key: Name: [7] :bus_id
paramno=-1
name=[7] ":bus_id"
is_param=1
param_type=2
array(3) {
  [0]=>
    string(5) "00000"
  [1]=>
    NULL
  [2]=>
    NULL
}
```

Maintenant que cela fonctionne, on peut passer à la partie Arduino. Pour cela, nous allons faire un programme qui va exécuter cette commande automatiquement lorsque l'on entre "1" dans le moniteur série.

Voici le lien du programme : <http://pastebin.com/uCWsi5ru> (programme commenté)



```
1 Envoyer
Envoi...
SQL: [102] INSERT INTO schedule (date_time, line_id, stop_id, bus_id) VALUES (NOW(), :line_id, :stop_id, :bus_id)
Params: 3
Key: Name: [8] :line_id
paramno=-1
name=[8] ":line_id"
is_param=1
param_type=2
Key: Name: [8] :stop_id
paramno=-1
name=[8] ":stop_id"
is_param=1
param_type=2
Key: Name: [7] :bus_id
paramno=-1
name=[7] ":bus_id"
is_param=1
param_type=2
array(3) {
  [0]=>
    string(5) "00000"
  [1]=>
    NULL
  [2]=>
    NULL
}
```