IMERIR
une école
CCI PERPIGNAN

Promotion 2018-2020 | Imérir

Documentation Du Projet

SOCIAL_DEVICES

BOUCHIER, LOREK

Date : 22/01/2018
Version 1.0

SOMMAIRE

1. Introduction

2. Mise en place du système sur Raspberry PI

On considère que votre Raspberry Pi est déjà fonctionnelle avec un linux graphique installé.
Si ce n'est pas le votre cas, il existe déjà de nombreux tutoriel à ce sujet.

Nota Bene: Chaques lignes de commande données est à rentrer dans le terminal de la Raspberry.

Avant toutes installations de modules, il est important de vérifier que le système est à jours:

```
sudo apt-get update && sudo apt-get upgrade
```

Notre système d'affichage utilise un serveur apache2:

```
sudo apt install apache2
```

Sachant que nos codes utilisent du php et du xml, il faut installer les modules qui permettent de rendre le code fonctionnel:

```
sudo aptitude install php
sudo aptitude install php-xml
```

Il faut redémarrer apache pour la prise en compte des nouveaux modules:

```
sudo service apache2 restart
```

Il est nécessaire d'avoir un navigateur afin d'afficher une page avec du code html:
Vous êtes dispensé de cette commande si vous avez déjà un navigateur

```
sudo apt-get install iceweasel
```

On passe maintenant à l'installation de la partie du système en python.
Pour installer des modules pour du python, on doit par avance installer pip:

```
sudo easy_install python-pip
```

Pour récupérer des informations sur une page html:

```
sudo pip install BeautifulSoup4
```

Pour communiquer avec l'arduino:

```
sudo pip install pyserial
```

Pour récupérer les tweets, on a besoin de twython:

```
sudo pip install twython
```

Dernière étape: installation de notre code sur le serveur et mise en route du système.

Telecharger nos fichiers à l'adresse du github :
https://github.com/squaregolab/SocialDevices

Dézippez le contenu du fichier "SocialDevices-master.zip" et copiez le contenu dans le dossier du serveur apache :

```
sudo cp -r   .../SocialDevices-master/*   /var/www/html
```

remplacer "..." par le chemin du dossier contenant le code

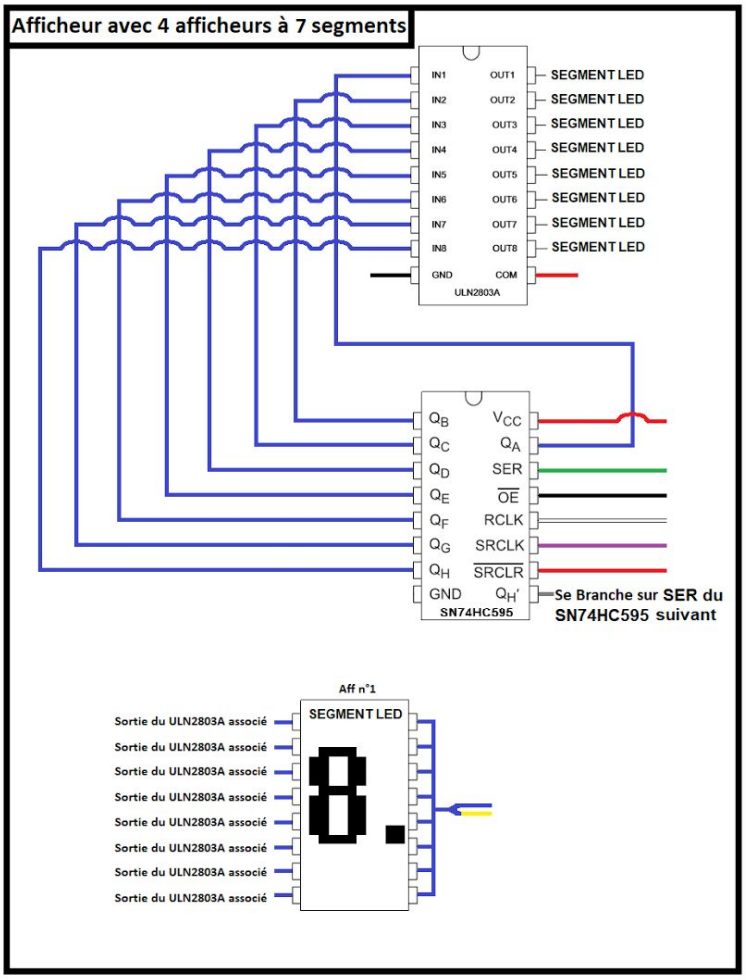
exemple avec une clé USB:
sudo cp -r   media/pi/USB/SocialDevices-master/   /var/www/html*

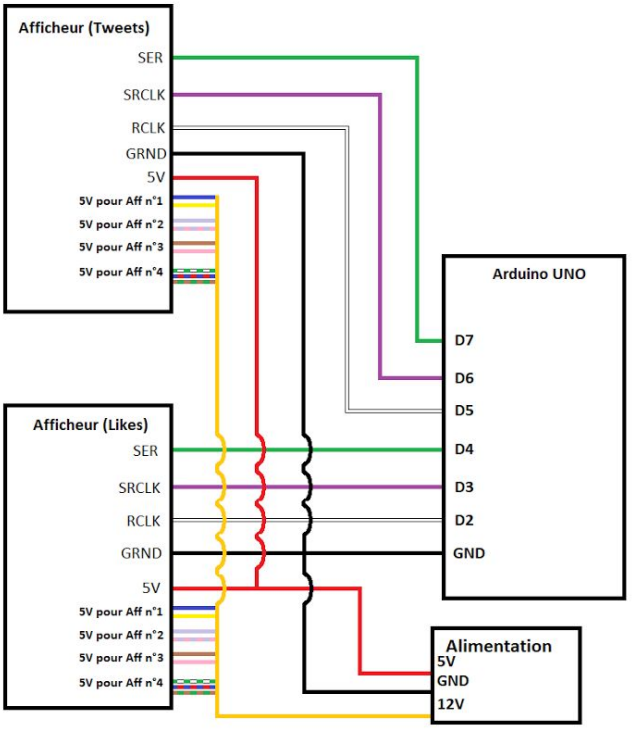
On autorise apache d'avoir les droits sur tous les fichiers du dossier html :

```
cd /var/www/html
sudo chmod 777 *
```

3. Installation du système électrique

Schéma de branchement des afficheurs à l'Arduino UNO:





L'Arduino est branché au Raspberry Pi en USB.

4. Mise en route du système

a. Obtention d'une clé d'authentification

Envoyer des requêtes au serveur de de twitter nécessite l'obtention de clés d'authentification, car les requêtes utilisent un système de tokens.

Page permettant la demande de clés:
https://apps.twitter.com/

Explication en vidéo de l'obtention des clés:
https://www.youtube.com/watch?v=x_p9_047h-s

Les 4 clés sont à ajouter dans les fichiers de code python: SocialDevice.py et Recup_20.py

Par défaut à la ligne 150 dans les deux fichiers:

```
#Remplacez **** par vos clé
APP_KEY = '****'
APP_SECRET = '****'
OAUTH_TOKEN = '****'
OAUTH_TOKEN_SECRET = '****'
```

Exemple de rendu avec vos clés:

```
#Remplacez **** par vos clé
APP_KEY = 'RaNW7WIg0k3E8zEQ0sOr0NrlF'
APP_SECRET = 'bhkcLkAd2WrecH8ItupJUliqgQAShH940r2PnhgNVSN5Bt2LtJ'
OAUTH_TOKEN = '8912679127-FJJYV1jD5mfceSyOaif2WVO9MgFO63duNwjhksA'
OAUTH_TOKEN_SECRET = 'nTn27oP20gtiCCE3aqxL4GMfWe105My2Q2MfJYP6v5xDL'
```

A savoir les ces clés sont personnelles et reliées à votre compte tweeter, les partager serait créer une faille de sécurité.

Nota Bene: N'oubliez pas de mettre vos clés sur les deux fichier .py

b. Partie Python

Les codes en python sont exécutés par un script en shell:

```
#!/bin/sh

cd /var/www/html

#Recuperation des 20 derniers tweets
sudo python Recup_20.py
```

```
#Passage en mode stream: il capture les nouveaux tweets
sudo python SocialDevice.py
```

Je laisse la possibilité de créer le script à n'importe quel endroit, sachant qu'un script peut être paramétré pour se lancer directement au démarage.

Manuellement le script se fonctionne à la commande:

```
sudo ./script.sh
```

Remplacer "script" par le nom que vous avez donner à votre script
N'oubliez pas d'aller, avec la commande cd, *à l'emplacement de votre script*

Nota Bene: Attention le code ne pourra pas fonctionner si vous n'avez pas branché votre Arduino et si vous n'êtes pas branché en réseau.

c. Lancer « SocialDevices »

Les fichiers, pour l'affichage, comportant pas mal de liens entre eux, il est important de s'assurer qu'ils sont tous au même endroit, dans le localhost : sur votre raspberry */var/www/html/localhost*. Si vous ne le trouvez pas, vérifiez avoir bien installé ce qui a été demandé précédemment.

Fichiers nécessaires pour la partie affichage :

- ❏ data_posts.xml → fichiers data généré et mis-à-jour par le code de la partie « Python » ;
- ❏ Counter.txt & Filter.txt
- ❏ fractal.js & modernizr-2.js → framework ;
- ❏ page.php → page principale d'affichage (les autres seront appelées par elle) et la gestion du rafraîchissement ;
- ❏ page_adress.php → page d'option pour choisir l'adresse de facebook pour le compte de like ;
- ❏ page_counter.php → page d'option pour réinitialiser le compteur de post ;
- ❏ page_filter.php → page d'option pour changer le filtre ;
- ❏ iframe_posts.php → page affichants les posts/tweet ;
- ❏ style_popup.css
- ❏ style_screen.css
- ❏ function.js
- ❏ sponsor.pnj

Pour rappel, chacun de ces fichiers est téléchargeable en open source sur le GitHub : https://github.com/squaregolab/SocialDevices

Pour lancer le « SocialDevice » lancer un navigateur et entrez dans la barre de recherche l'adresse de notre page de démarrage :

```
localhost/page.php
```

A savoir que le code est optimisé pour le navigateur Mozilla Firefox.

Si vous utiliser plusieur raspberry pour afficher sur plusieurs écrans. Pour les raspberry « esclaves », il faudra entrer, non pas le localhost mais l'adresse du raspberry maître :

```
adresseIP/page.php
```

Partie Affichage et interface utilisateur

Cette partie se concentre sur l'affichage, la manipulation de données à partir de fichiers. Elle se base sur du Apache2 pour pouvoir réagir, avec l'utilisateur, et rafraîchir l'affichage, en temps réel.

a. Langages de programmation :

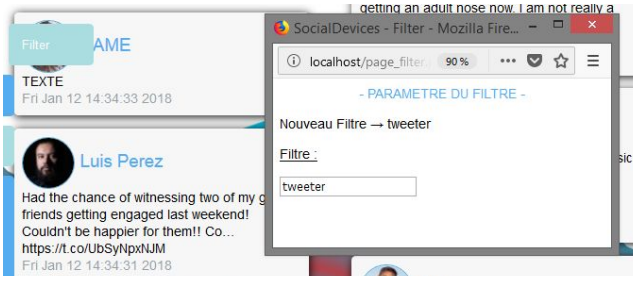
Le code utilise plusieurs langages :

- ❏ Html ;
- ❏ Xml : Pour le fichier data, contenant les informations des posts ;
- ❏ Javascript : Pour certaines fonctions. *Nous aurons aussi une partie intégrant du JQuery pour le rafraîchissement* ;
- ❏ Php : Pour la manipulation des données automatique en générale (le php est ce qui fait la « dynamique » d'un site) ;
- ❏ CSS : Pour la mise en page, le design.

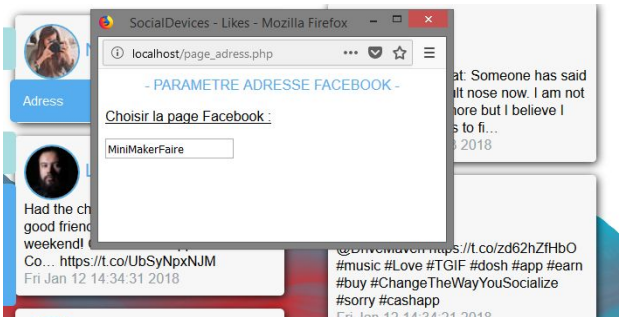
b. Mode d'emploi :

Vous trouvez sur cette page un menu qui vous permet de :

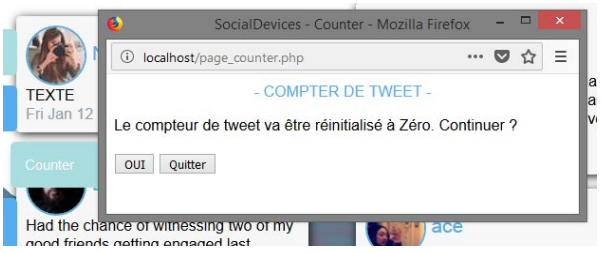
- ❏ Modifier le filtre. Celui-ci influence directement le type de post qui s'affichera ;



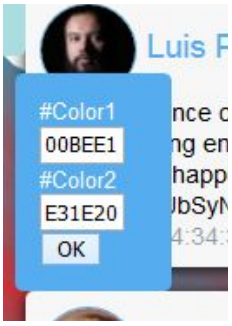
- ❏ Modifier l'adresse de la page Facebook où est récupéré le nombre de « Like », affichés sur « l'afficheur externe »;



- ❏ Réinitialiser le compteur de post/tweet, également sur « afficheur externe » ;



- Changer les couleurs de la fractal en arrière fond, deux couleurs, à choisir en hexadécimal (exemple : 97D46F).



c. Explication en vue de futures modifications :

Tout d'abord, sachez que vous pouvez très bien travailler l'affichage, en local sur votre ordinateur grâce à un serveur tel que « Wampserver ».
Disponible ici : http://www.wampserver.com

Ensuite, si vous souhaitez modifier les codes directement sur la raspberry, il vous faut aller dans un terminal, vous positionnez dans le fichier :

```
cd /var/www/html
```

Puis faire un nano pour modifier le fichier choisi :

```
nano page.php
```

Pour quitter faire CTRL+X.

PAGE PRINCIPALE

Affichage avec le filtre « love ».

Sur la page principale, page.php, on utilise de « l'overlay » pour avoir en fond la fractal (« overlay_down ») et en premier plan les sponsors et le menu (« overlay_up ») et au milieu les posts (« overlay_middle »).

La fractal nécessite les fichiers modernizr-2.js et fractal.js. Et pour fonctionner, les paramètres d'initialisation de la fractal doivent se trouver sur la même page où la fractal est appelée (dans page.php donc). Car nous utilisons « document.getElementById » pour les couleurs et « position.innerHTML » pour l'affichage de la fractal.
Les couleurs sont traitées dans fractal.js qui utilise lui-même modernizr-2.js. Dans le menu, nous demandons simplement le code hexadécimal et nous rajoutons le « # » devant, avec une concaténation, avant d'envoyer les couleurs aux fonctions de fractal.js.
Vous pouvez très bien faire un autre type de formulaire pour les couleurs, mais il faudra les transformer par la suite, avant envoi. Pour plus de documentation je vous invite à vous rendre sur les sites référent de ces fichiers :
http://rectangleworld.com/blog/archives/538
http://www.modernizr.com

Les posts sont affichés sur page.php via la fonction get(), ce qui récupère le code généré par iframe_post.php. Nous affichons ensuite où ce code dans la div « RefreshDiv ».

```
$.get("iframe_post.php",function(data){
        $("#RefreshDiv").html(data);
});
```

Nous englobons ensuite cela dans un setInterval() pour le rafraichissement ce cette div contenant les post, seulement. Cela évite des confrontations avec l'interface utilisateur (menu/fractal). Le reste de page.php ne sera rechargée que si les couleurs de la fractal sont modifiées.

PAGE AFFICHANT LES POSTS

C'est sur iframe_post.php, que les posts sont générés. Nous manipulons les données du fichier xml, data_posts.xml, généré par la « partie python ».
Un fichier de donnée xml se compose tel que l'exemple ci-dessous :

```
<baliseglobale>
        <baliseconteneur>
                <balise1>Informations</balise1>
                < balise2>Informations</balise2>
                <balise3>Informations</balise3>
                < balise4>Informations</balise4>
        </baliseconteneur >
        <baliseconteneur>
                <balise1>Informations</balise1>
                <balise2>Informations</balise2>
                <balise3>Informations</balise3>
                <balise4>Informations</balise4>
        </baliseconteneur >
</baliseglobale>
```

Nous chargeons les données avec :

```
        $xml=simplexml_load_file('data_posts.xml') ;
```

Lorsque le fichier xml contient, comme pour notre projet « SocialDevices », plusieurs posts (« baliseconteneur » sur l'exemple type) il faut appeler la balise, contenant l'information désirée, comme un tableau. Il suffit alors de faire un « echo » tel que par exemple :

```
        echo $xml->post[3]->user_name //affichera le pseudo du 3ème post.
```

Dans iframe_post.php, un « for » est utilisé pour afficher de manière automatique les 20 premiers posts.

PAGES D'OPTIONS

Le menu utilise les fonctions « window.open() » et « window.close() » en javascript (définies dans function.js) et ouvrent dans des « pop-up » les pages d'optionS :

- page_filter.php
- page_adress.php
- page_counter.php

On utilise des lectures écritures dans des fichier.txt :

- Counter.txt

- Facebook.txt
- Filter.txt

Ces fichiers sont utilisés par la « partie Python » pour générer les nombres de « Like » et de « tweet » sur les afficheurs à segments externe, ainsi que pour la récupération des tweets.

PAGES DE STYLE

Le css à une très grande importance dans l'affichage. En plus, des basiques apparences, c'est le css qui gère la transparence, les positions et les mouvements des différents composants. Nos css sont adaptés pour une dimension d'écran de 1920 par 1080. La plupart des paramètres ont une adaptation automatique à l'écran sur la hauteur. Mais il faudra vérifier la largeur maximum des images, par rapport à votre écran ou vous aurez un dépassement.
Donc dans style_screen.css (rattaché à page.php)

```
#pic_sponsor{
        max-width:19vh;
}
```

Et

```
#picture_media{
        max-width: 30vh ;
}
```

Complément: Installation du serveur local test WAMP :

Le PHP à pour rôle de générer des pages web. Cependant, seul un serveur peut lire du PHP et notre ordinateur tout seule, n'est pas un serveur. Nous allons donc mettre en place un serveur local pour pouvoir tester et continuer notre projet. Pour ce faire on installe WAMP (sous Windows).

Lien de téléchargement : http://www.wampserver.com

Une fois l'installation terminée, mettez vos fichier web dans le dossier www de wamp, couramment C:\wamp\www. C'est de là que le « serveur local » lancera votre page web.

Ensuite, lancez Wamp, une icône s'affiche dans la barre de raccourcie Windows

Faites clic-gauche → Localhost pour lancer le serveur.

Votre navigateur se lancera et vous montrera les fichiers présents dans ce dossier www. Choisissez votre page de démarrage et voyez par vous-même votre site.