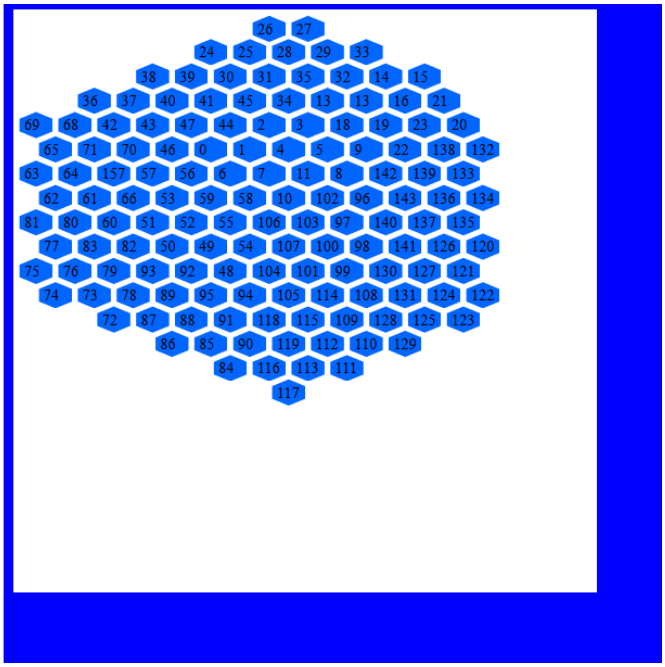
Interface web de Baby-Mind



Tables des matières

Résumé du projet

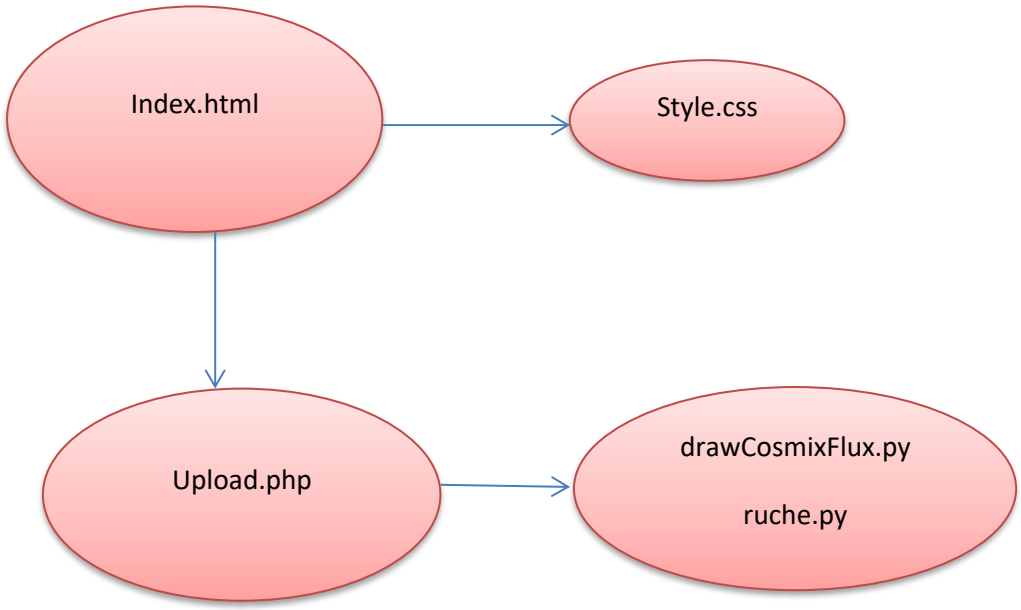
Le projet d'interface web baby mind a pour but de permettre aux utilisateurs d'obtenir des informations sur les observations du télescope et ceci directement depuis le web.

En effet, le télescope produit des fichier binaire DAQ qu'il est nécessaire de rendre lisible pour un être humain. La lecture et l'extraction d'information des fichiers (à l'aide d'un GUI python) existant déjà à la création de l'interface, ce projet s'appuie sur le script déjà existant.

L'interface met en relation un utilisateur distant avec un script, extrait de GUI. En effet, l'utilisateur rentre les données voulues, l'interface va alors exécuter le script et va ensuite s'occuper de transmettre les images d'évènement ainsi que des graphiques à l'utilisateur.

Structure du projet

Ce projet est constitué de 5 fichiers distincts :



Fichier index. html

Le fichier html est deux parties :

1. La première partie comprend l'interface utilisée pour rentrer les différentes données. Aujourd'hui, seulement quelques-uns des champs proposés sont utiles. Ces champs sont compris dans un formulaire html Ces données sont ensuite gérées par le fichier php.
2. La seconde partie consiste en une représentation visuelle de la caméra. Elle est composée d'une balise « pixelsDIV » qui contient de nombreux hexagones. Chaque hexagone représente un pixel de la caméra. Ces hexagones sont des images svg, (*Scalable Vector Graphics*). Chaque image est contenue dans sa propre div, qui est identifiée par le numéro de pixel correspondant à l'hexagone.

Ce fichier utilise le fichier style.css pour la mise en forme.

Fichier style.css

Ce fichier est utilisé pour la mise en page du html. Il met les deux principaux conteneurs à la même hauteur et positionne les pixels selon leur emplacement sur la caméra. Chaque hexagone est positionné 50 pixels en dessous de l'élément supérieur, et 40 pixels sur la gauche ou droite de son voisin.

Fichier upload.php

Il est possible de diviser le fichier php en 4 parties distinctes :

Récupération des données.

Les données sont envoyées par le formulaire html via la méthode POST.

La partie de code ci-dessous présente la gestion de l'upload de fichier. On va parcourir tous les fichiers uploader, vérifier que ce sont bien des fichiers daq avec la ligne **2** On copie ensuite les fichiers dans un répertoire définis pour que le script python puisse les utiliser. Si ce ne sont pas les bons des types de fichiers on les supprime simplement.

```
1.  for ($i=0;$i < $total; $i++){
2.      if ($_FILES["upload"]['type'][$i]=='application/octet-stream'){
3.              $tempPath=$_FILES["upload"]['tmp_name'][$i];
4.              $path=$dir."/".$_FILES["upload"]['name'][$i];
5.              if(copy($tempPath, $path)){
6.                      unlink($tempPath);
7.              }
8.      }else{
9.              echo 'file deleted: '.$_FILES["upload"]["name"];
10.             $tempPath=$_FILES["upload"]['tmp_name'][$i];
11.             unlink($tempPath);
12.     }
13. }
```

Préparation et exécution du script python

On va ensuite créer une commande shell à exécuter, en concaténant "python drawCosmicFlux.py " ainsi que la suite d'argument :

i) Dossier de reception
ii) Depuis
iii) Jusqu'à
iv) Par étape de
v) Valeur DAC
vi) Le choix (LG, HG ou TOT)
vii) Le temps alloué
viii) 1 ou 0 pour savoir si ce sont des fichiers uploadés ou depuis une date rentrée
ix) La date où trouver des les fichiers si on les prend comme ça.

On exécute ensuite cette commande et on vérifie la sortie.

Récupération et affichage des résultats

Pour afficher les résultats, on fait une boucle sur le dossier d'image que le script python crée. Pour afficher ces résultats, on utilise avec affichage :

```
if ($fichier != '.' && $fichier != '..'){
        $source=$images."/".$fichier;
        $counter++;
        //echo "source: .".$source;
        echo
                "<a href=\"".$source."\" download> Download ".$fichier."</a></br>"; //Download
        file
        echo

                "<img
                src=\"".$source."\"
                alt=\"".$fichier."\"
                height=\"500px\"
                width=\"500px\" />";
        echo
                "</br>";
```

Nettoyage

Afin de ne pas trop utiliser de place sur le serveur, les fichiers et les images sont supprimés à la déconnexion du client.

La technique utilisée pour le nettoyage est de demander au serveur d'ignorer la déconnexion du client. De cette manière on peut continuer à exécuter des instructions. Une fois que les images sont affichées, on rentre dans une boucle infinie qui va tester si le client est encore connecté ou pas. Pour cela on envoie de simple balise. Une fois que la connexion avec le client est perdue, le script php va alors supprimés les fichiers et les figures.

L'inconvénient de cette méthode est que la page php ne sera jamais totalement chargée et donc que rien ne s'affiche. C'est pour cela qu'à divers endroits dans le code j'utilise les commandes **ob_flush**(); **flush**() qui permettent de vider le buffer d'éléments à afficher, sans attendre la fin du chargement de la page.

Fichiers python

Il y a deux fichiers python, le premier qui s'appel drawCosmixFluy.py est le fichier principal. Ce fichier s'occupe de créer le répertoire de figures, où il stockera toutes images créées.

Il s'occupe également de l'analyse des fichiers daq « et de demander au deuxième fichier » ruche.py, de créer une image de la ruche de pixel selon certains critères, rentrés en paramètre.

drawCosmixFluy.py crée également 4 images de graphique.

Toutes les images sont enregistrées au format svg. Ce format a été privilégié de manière à permettre un très grand zoom sans altérer la qualité de l'image, ce qui peut être particulièrement bénéfique pour les graphiques.

Les principales fonctionnalités de ces scripts proviennent directement du GUI, même si un filtrage ainsi qu'une partie de réécriture a été nécessaire.

Historique de développement

1. Semaine du 25 février : attribution des projets et premier contact avec le client. Première réunion le vendredi. Cette première réunion est consacrée à la présentation du télescope ainsi qu'à la définition des objectifs. Le bilan de cette réunion est que le projet s'orienterait plutôt vers la création d'une API pour utiliser les résultats du télescope.
2. 1er mars au 18 avril : développement d'une interface de programmation. Durant cette période, quelques réunions techniques ont eues lieux afin de mieux comprendre le code de GUI.
3. 18 avril : changement de direction. Le client souhaiterait finalement une interface web.
4. 18 avril au 26 avril : recherche d'une solution automatisée pour porter le GUI sur le web, cependant à cause du développement du GUI avec la libraire *tkinter*, aucune solution ne fonctionne.
5. 26 avril au 23 mai : développement de l'interface finale, en passant par des technologies classiques (html, css et php), comme présentée dans ce document. Création également de la documentation reliée à cette interface.

Difficultés rencontrées

Compréhension et relecture du GUI

Le GUI écrit n'étant pas documenté, cela m'a demandé beaucoup de temps afin de le relire et de le comprendre. En effet, de nombreuses fonctions ainsi que la programmation orientée objet sur python m'étaient inconnues. C'est pour cela que la plus grande partie du temps passé sur ce projet a été utilisé de manière à comprendre cette partie qui est essentielle tant pour le début d'API que l'interface web.

Apprentissage des technologies du web

Pour créer l'interface web, il m'a fallu apprendre plusieurs langages : html, css, php et javascript qui n'a finalement pas été utilisé. En effet, il me semblait plus pratique que les calculs se fassent côté serveur de manière à pouvoir exécuter le script python que j'ai réécrit. De cette manière, le cœur des calculs n'a pas eu besoin d'être réécrit dans un autre langage. Cependant, l'extraction des éléments nécessaire à la création des images a quand même demandé une masse de qunantité de travail non négligeable dans ce projet.

Cependant, le php étant un langage très utilisé et ayant une importante documentation, l'apprentissage a été relativement rapide, de même que pour le html et le css.

Conclusion

Pour conclure ce projet de semestre m'a apporté différentes choses. Tout d'abord j'ai pu expérimenter le fait de devoir travailler avec un code qui est peu et mal commenté, ce qui m'a fait comprendre l'importance de commenter et documenter son code. Il m'a également permis de découvrir de nombreuses facettes du langage python qui m'étaient inconnues tant lors de la relecture du GUI que du développement de l'API. De plus, j'ai également pu apprendre la base de la programmation web.