

« Project Music »

Par Titouan LELLOUCHE, Marin TRAN et Adrien Planchon

*Sommaire :*

**I – Introduction**

**II - Présentation du projet**

**III – Réalisation du programme**

1. Répartition des tâches
2. Structure et fonctions
3. Mon rôle dans le projet
4. Problèmes rencontrés

**IV – Conclusion**

**V – Annexes (présentes sur Github)**

1. Project\_Music\_1\_0.py
2. clavier\_clavecin\_1\_0.py
3. clavier\_orgue\_1\_0.py
4. clavier\_piano\_1\_0.py
5. constantes\_1\_0.py

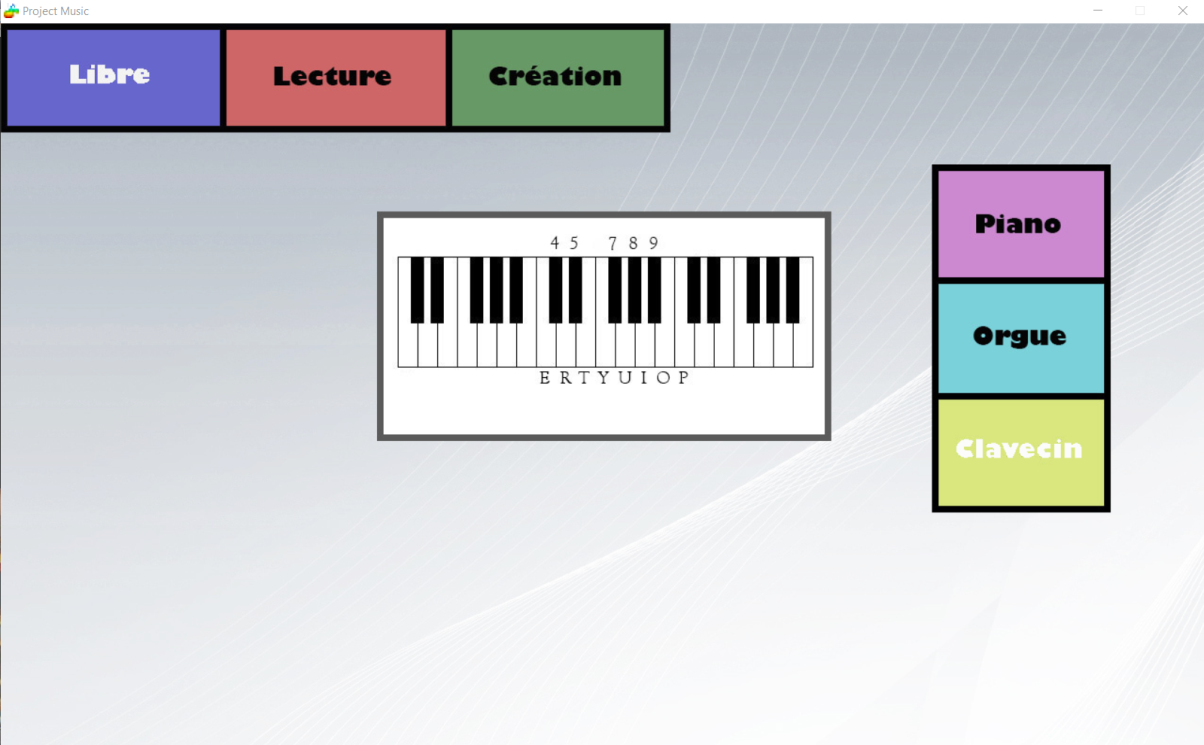
**I – Introduction**

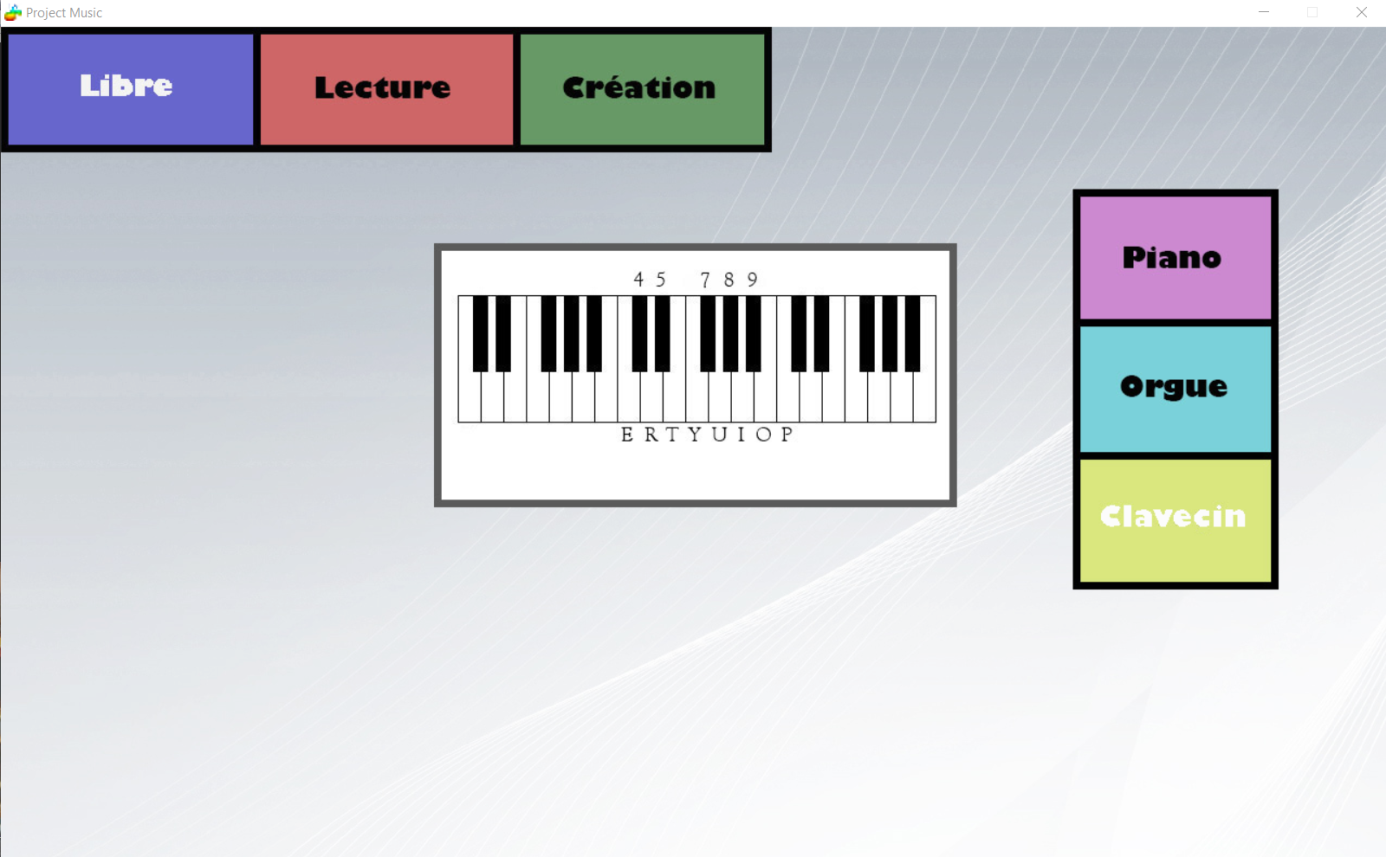
Cette année, nous avons eu l’occasion de travailler en groupe sur un projet commun dans le cadre de la spécialité ISN, Informatique et Sciences du Numérique. Un des premiers défis fut la composition du groupe qui se fit par hasard. En effet, Titouan LELLOUCHE, Marin TRAN et moi-même avons du, très vite, adopter une certaine cohésion afin de travailler efficacement et avec l’esprit de groupe. Il est intéressant de noter que cela se fit naturellement au fil des séances, en développant des automatismes.

Nous avons utilisé le langage Python et sa bibliothèque Pygame (expliqué plus tard) et nous avons réuni nos dossiers dans Github, ce qui a permis de rassembler nos codes et nos fichiers sur une plateforme commune à tous les membres du groupe, ainsi que de tenir un cahier des charges pour réaliser nos objectifs, se répartir les tâches et noter les problèmes auxquels nous étions confrontés.

**II – Présentation du projet**

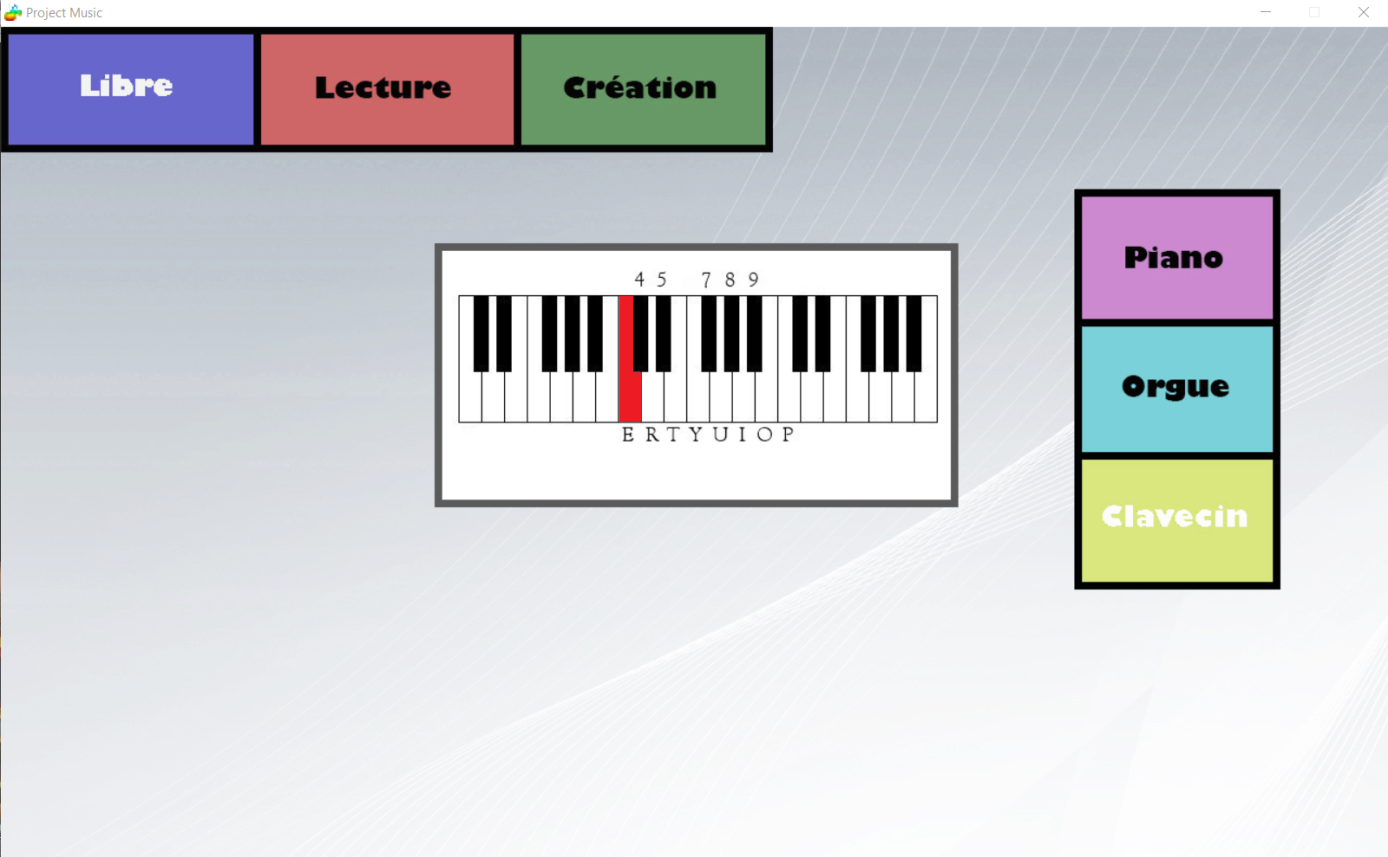
Nous voulions développer un programme capable de jouer des sons, s’inspirant d’un piano, et avec une interface visuelle interactive.



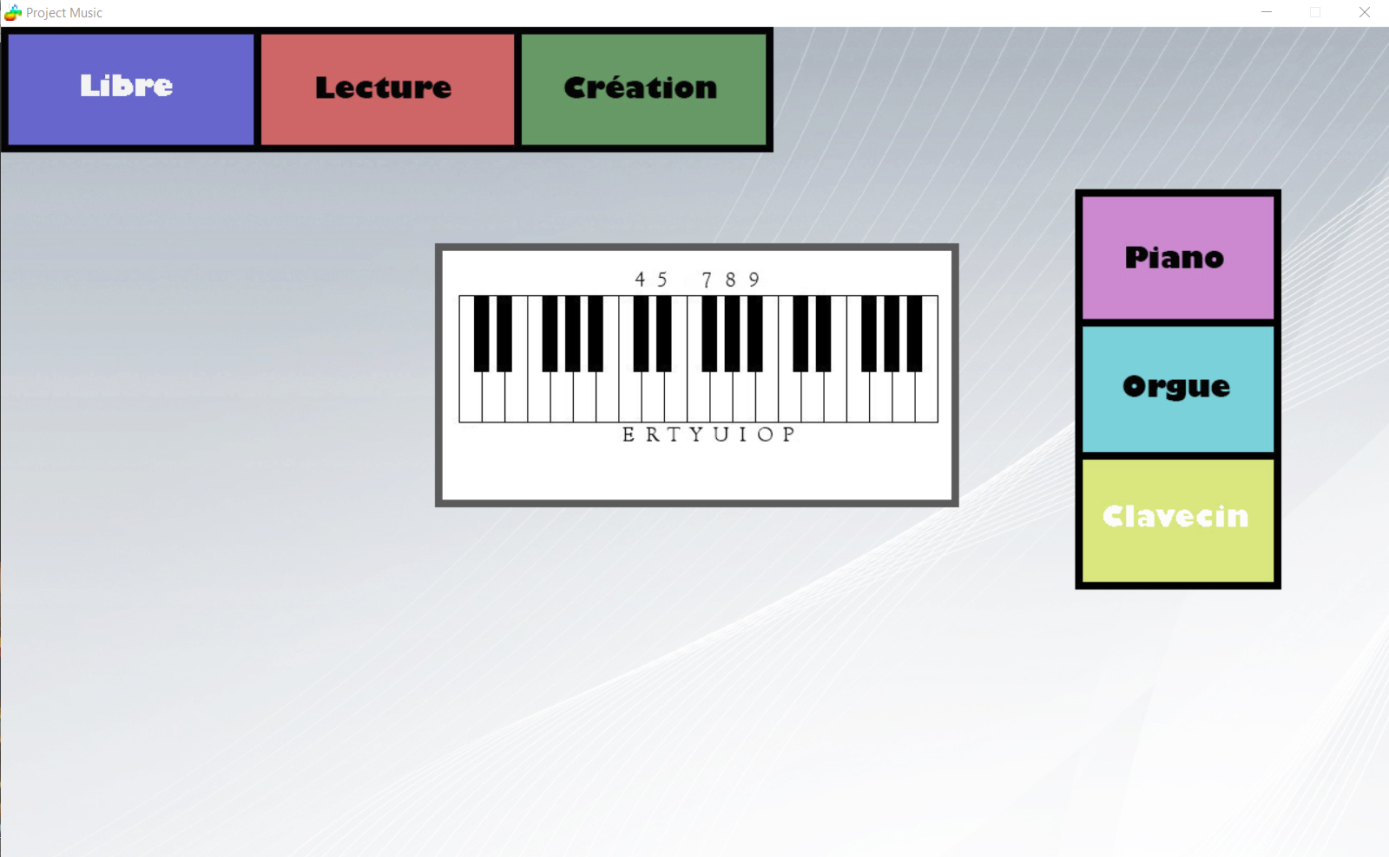


Nous avons décidé de rajouter le clavecin et l’orgue comme instrument pour diversifier les fonctions du programme.

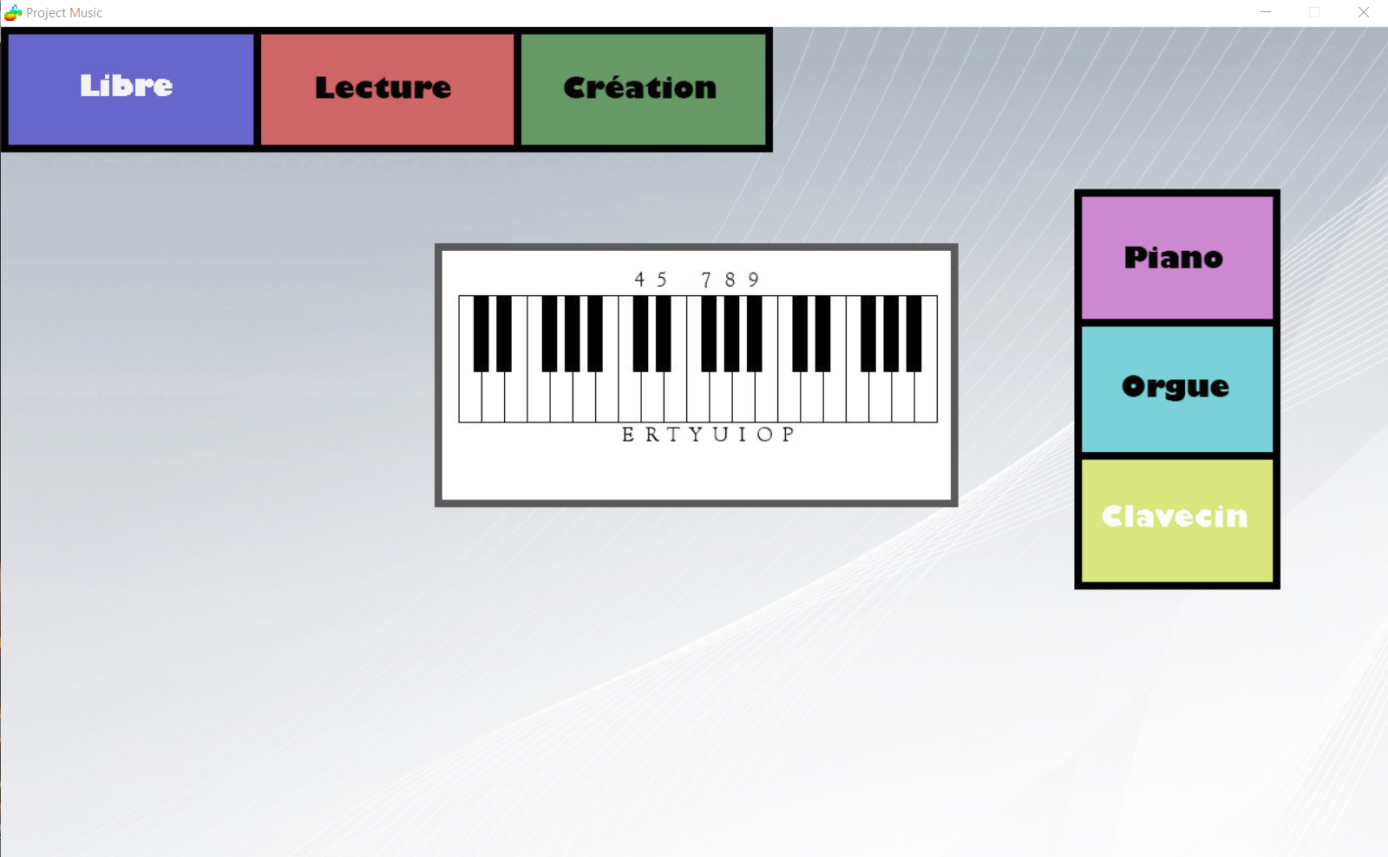
Connaissant le langage Python, nous avions besoin d’une bibliothèque capable de jouer des sons et associer des images. Nous avons donc choisi la bibliothèque Pygame pour Python que nous avons jugé la plus adaptée.



L’idée était de créer une interface colorée et assez dynamique. Nous avons donc pensé à allumer les touches du clavier de l’interface en fonction de la note que l’on jouait, de différentes couleurs.

Pour jouer les notes, nous avons essayé de respecter le plus la disposition des touches d’un piano, c'est-à-dire que les touches noires, en haut, se jouent avec les touches de la ligne des chiffres du clavier d’ordinateur, tandis que les touches blanches, en bas, se jouent avec la ligne « azerty » de ce clavier.

Nous avions également prévu d’ajouter une fonction lecture et enregistrement, mais cette tâche, assez longue sera finie après la présentation du projet devant le jury.



**III – Réalisation du programme**

1. Répartition des tâches

Après deux séances à travailler tous les trois sur la même chose, nous nous sommes vite rendu compte que se répartir les tâches était bien plus rapide pour avancer dans notre projet. Nous les avons répartit de cette façon :

* Titouan s’occupait de la structure générale du programme. C’est lui qui écrivait la plupart des fonctions nécessaires au bon fonctionnement du projet.
* Marin s’occupait de l’interface graphique et de son insertion dans le programme de sorte que les actions que l’on effectuait en utilisant notre programme se voient à l’écran. Le seul ayant Photoshop parmi nous, c’est grâce à lui que nous avons pu régler un de nos problèmes majeurs (expliqués dans le d)) : la transparence des images.
* Quant à moi, le seul faisant de la musique, il m’a été attribué la tâche de rechercher les sons, les travailler de sorte qu’ils puissent être lire par Python, et de les insérer dans le programme pour que chaque touche corresponde à une note. J’ai aussi, sous le format qui m’a été demandé par Titouan (fichier texte bloc-note), composé des musiques simples pouvant être jouées par le mode « lecture » qui n’a malheureusement pas pu être terminé.

1. Structure et fonctions

Le programme est divisé en plusieurs parties dont une principale. Celle-ci est primordiale puisqu’elle va piocher les constantes et les fonctions dans les programmes secondaires lorsque cela est nécessaire.

Une fois toutes les constantes importées, une fenêtre va s’ouvrir et la boucle centrale du programme va démarrer, et s’arrêtera seulement une fois que l’on quitte le programme.

Dans notre cas, la variable la plus importante est la variable « instrument » puisqu’elle permet de décider, selon sa valeur, si le programme va jouer en piano, clavecin, ou orgue selon le bouton de l’interface graphique sur lequel l’utilisateur clique.

* Les trois fonctions (celles du piano, clavecin et orgue) sont codées de la même façon, seule la valeur de leur variable change (pour les différencier), ainsi que les sons importés pour chaque instrument.

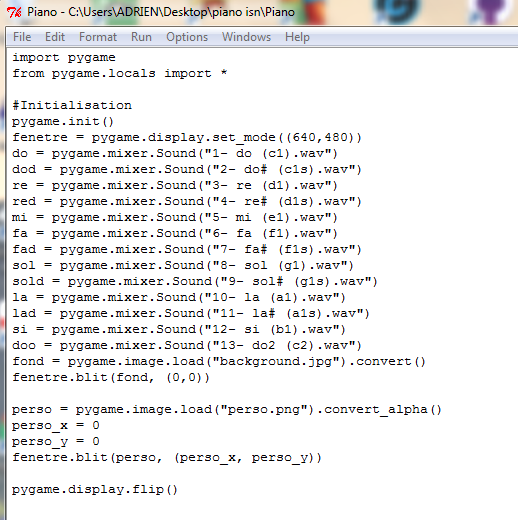
Lorsqu’une touche est pressée, la fonction va jouer et afficher la note correspondante à l’écran.

* Une autre fonction sert à détecter, selon la zone où l’utilisateur clique, quel instrument le programme doit jouer. En effet, nous avons pu délimiter des zones sur l’interface graphique et donner des instructions de sorte que chaque zone désigne un instrument différent.

Il faut savoir que les instructions, bien qu’étant identiques pour chaque fonction, ne peuvent pas être présentes dans le programme principal puisque quand le programme effectue des actions dues à des événements (comme le clic de l’utilisateur par exemple) il doit charger un répertoire qui répartit les différents événements et leurs effet. Or, si l’on place ce répertoire dans le programme, on remarque que le programme est largement ralenti. Cela est du au fait que le programme devrait charger le répertoire à la fois dans le programme et dans la fonction. Nous avons donc séparé le répertoire du programme pour plus de fluidité.

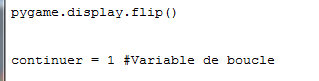
1. Mon rôle dans le projet

Comme énoncé précédemment, nous avons passé les premiers cours à travailler sur la même chose. Notre premier objectif était d’émettre des sons lorsque l’on appuyait sur une touche du clavier. J’ai été le responsable du premier squelette de notre programme, présent pour la première fois sur mon ordinateur.

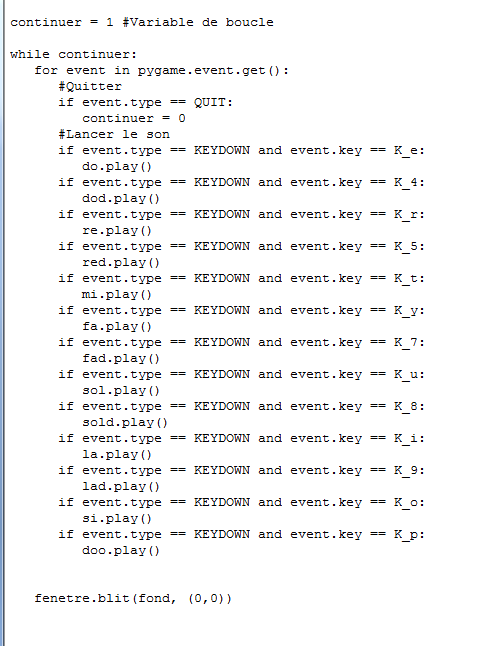
Ainsi :

On commence par importer la bibliothèque Pygame de Python pour pouvoir jouer des sons. S’en suis l’initialisation de toutes les variables, dans notre cas, les 13 notes d’une section de piano (do, do#, ré, ré#, mi, fa, fa#, sol, sol#, la, la#, si, do) ainsi que l’ouverture d’une fenêtre au démarrage de notre programme (dont on choisi la dimension) et d’une image de fond appelée ici « background ».

Ensuite, on a une fonction de pygame permettant d’afficher l’image, puis une variable définissant notre boucle :



En effet, tant que la variable « continuer » est égale à 1, le programme continue de tourner :



Si la variable « continuer » est égale à 0, le programme s’arrête, quand on clique sur la croix par exemple.

Sinon, on associe chaque touche du clavier à un son, par la fonction « if event.type == KEYDOWN » qui signifie le moment où l’on appuie, allié avec « and » à la fonction « event.key == K\_LETTRE » où « LETTRE » est la touche sur laquelle on appuie pour jouer le son.

La fonction « fenetre.blit » permet de choisir notre image de fond et de la placer où l’on veut dans la fenêtre, en l’occurrence depuis le haut à gauche dans notre cas.

Enfin, nous avons remarqué qu’il est impératif de finir la rédaction de notre programme par une fonction bien précise :



Sans elle, au moment où l’on ferme la fenêtre, pygame ne réponds plus et à des difficultés à se fermer.

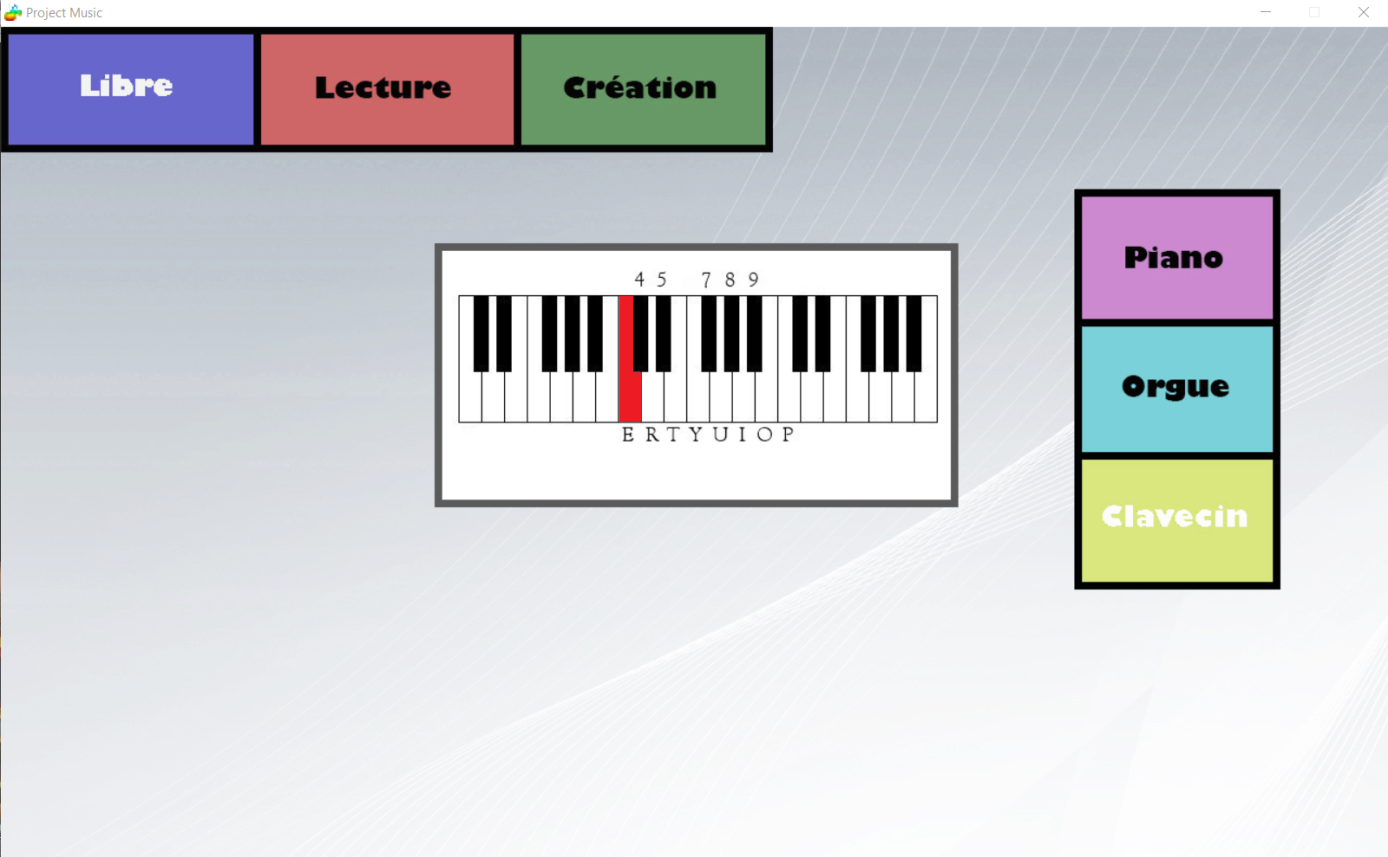
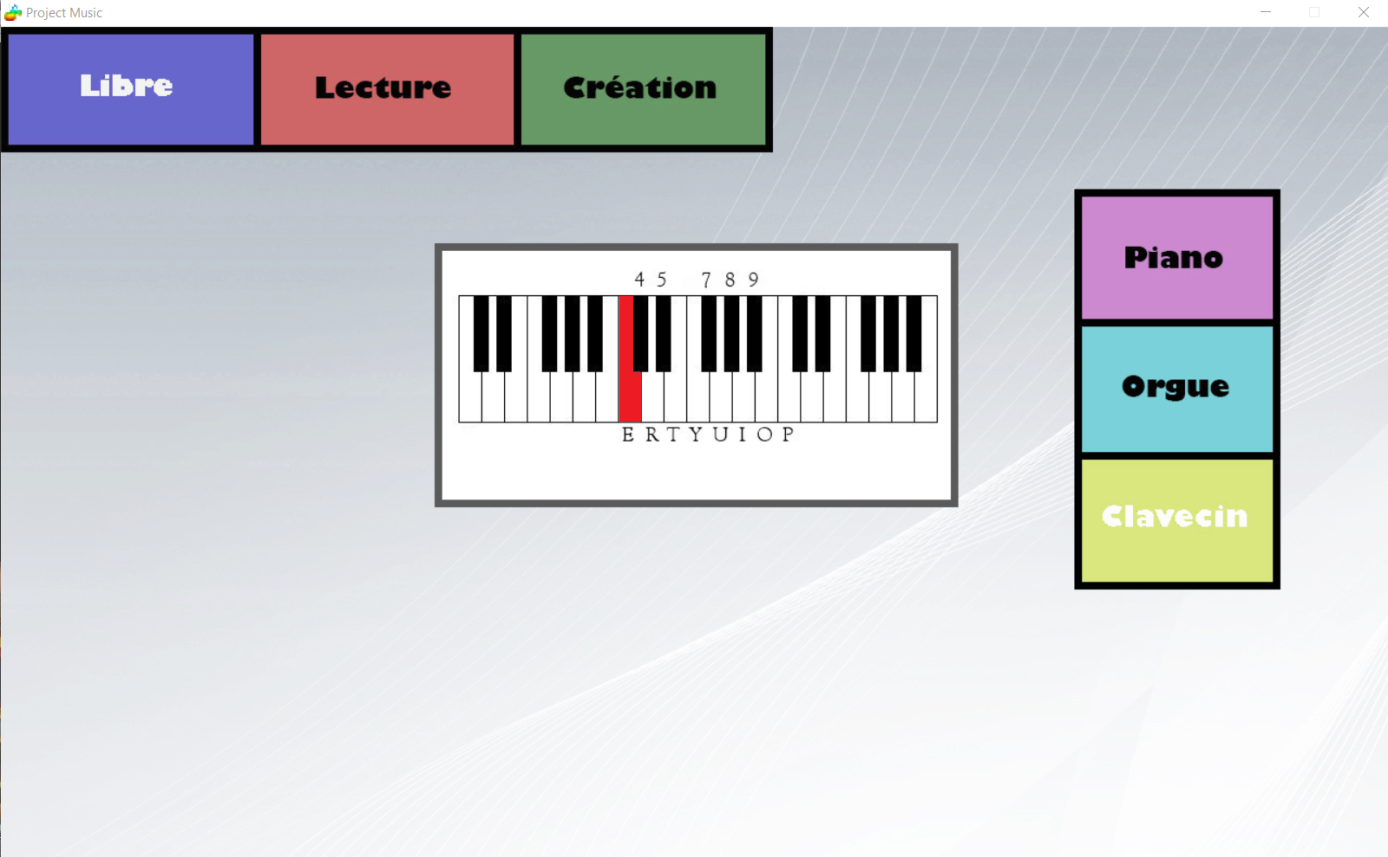
Une autre tâche très importante dont je me suis chargé fut la recherche et la conversion des sons. En effet, nous avions besoin d’autres instruments que le piano pour un programme plus complet. Au début à la recherche de sons de guitare et de trompette, l’idée fut abandonnée puisque il était très difficile de trouver les dièses et bémols (notes noires indispensables sur les pianos, mais moins évidentes sur d’autres instruments comme ceux recherchés). On a donc décidé de prendre des instruments plus proches du piano pour obtenir ses notes plus facilement : le clavecin et l’orgue. Après avoir enregistré moi-même les notes depuis mon piano électrique en format mp3, il fallait les convertir dans le seul format que Pygame peut lire : le format WAV.

La majeure différence entre le fichier MP3 et WAV est la qualité. En effet, un fichier MP3 est un fichier compressé échantillonné à entre 128 et 320 kilobits par seconde alors qu’un fichier WAV est un format non compressé, donc sans perte. La qualité de ce dernier est donc plus élevée, mais son poids aussi.

Pour convertir des fichiers sons, il existe beaucoup de logiciels. Le plus connu est Audacity, celui que j’ai utilisé.

1. Problèmes rencontrés

Notre premier problème fut l’installation de Pygame. En effet, il fallait que la version que l’on installe corresponde bien avec une certaine version de Python. Nous avions tous la dernière version de Python (3.5). Or, la dernière version de Pygame ne fonctionne qu’avec la version 3.2 de Python. Cette information est surtout présente sur les forums où des personnes rencontrent ce même problème. Donc nous avons du télécharger et travailler sur une version plus ancienne de Python. De plus, il fallait aussi faire attention à la version de Pygame que l’on installait selon le processeur de notre ordinateur (32 ou 64 bits).

Notre deuxième problème majeur fut la transparence des images. En effet, nous voulions, pour rendre le programme plus agréable, faire s’allumer les touches sur lesquelles on appuyait. Pour cela, nous avons donc créé 13 images du piano avec une touche coloriée différente à chaque fois, et lorsque l’on appuyait sur une touche, une fonction faisait appel à l’image correspondante. Or, avec cette technique, on a observé que lorsque l’on appuyait sur deux touches en même temps, une seule des deux touches s’allumait à l’écran. Nous avons donc tout de suite pensée à l’idée de transparence… Oui mais comment faire ? Nous ne connaissions aucun logiciel qui donnait des images que l’on pouvait superposer, ayant un format compatible avec Pygame.

On a ensuite pensé à Photoshop.

En effet, avec ce logiciel, il est possible de découper les images dans une certaine forme, dans notre cas les découper selon la forme de son placement sur le clavier de piano à l’écran, et sous

un format png (donc compatible avec Pygame). Marin s’en est chargé.

Notre dernier gros problème est toujours d’actualité. Faute de temps, nous n’avons pas pu le régler.

Il s’agit du moment où l’on change d’instrument. En effet, lorsque l’on clique sur un autre instrument que celui effectif, il arrive que la variable du mode instrument soit partagée entre son ancienne et sa nouvelle valeur.

Par exemple, quand on est sur le mode piano, et que l’on clique sur l’orgue, on peut souvent entendre les notes d’orgue et de piano qui se jouent en même temps, ce qui est fort désagréable.

Il faut savoir que nous connaissons l’origine du problème. En effet, lorsque le programme débute, les variables « instrument » et « mode » ont des valeurs par défaut (voir programme en annexe), on n’observe donc aucun problème lorsque les fonctions importent les constantes. Cependant, quand les fonctions doivent importer une nouvelle fois les constantes (lors d’un changement d’instrument), ces deux variables sont remises à leurs valeurs par défaut.

Pour contourner ce problème, l’utilisateur n’a qu’à appuyer une fois de plus sur l’instrument qu’il veut utiliser, même si cela rend notre programme bien moins charmant.

**IV – Conclusion**

Pour conclure, notre programme est complet et possède les caractéristiques que l’on lui voulait au début. Mais notre ambition ne s’arrête pas là. En effet, d’autres fonctionnalités (mode lecture et mode enregistrement), importantes à nos yeux, sont en cours d’élaboration car nous avons envie d’aller au bout de nos objectifs, une des valeurs principales que l’ISN nous transmet.

Je suis satisfait d’avoir choisi la spécialité ISN cette année. En effet, il nous faut faire preuve d’un tout autre mode de pensée, de travail, et d’organisation qui, je pense, est primordial pour cette matière d’avenir. J’ai remarqué que l’on apprend presque tout par soit même ou en échangeant avec ses amis. Il faut donc savoir être autonome.

Je suis également satisfait d’avoir été dans un groupe que je n’ai pas choisis. Cela apprend à former une cohésion d’équipe, essentielle dans cette matière, et c’est l’occasion de rassurer ceux qui ne sont généralement pas à l’aise avec le travail de groupe.

Si on m’avait dit au début de l’année que je serai capable de créer un programme comme celui-ci, avec seulement deux personnes du même niveau que moi, je ne l’aurai surement pas cru. C’est comme cela que je réalise les progrès que j’ai fait.

**V – Annexes**

(Voir programme sur Github)