Compte rendu séance 4 :

Il n'y a eu aucun changement notable dans les fichiers *.h* déjà existants (corrections mineures), cependant un nouveau a été créé !

c.f. rapport n°3: *left_piano_checking()* et *right_piano_checking()* ne sont pas faites pour fonctionner simultanément, une fonction sera créée ultérieurement pour cela.

Ce nouveau fichier nommé *general_play.h* contient la (fameuse) fonction mentionnée ci-dessus qui se nomme *pianos_checking()*. Voici donc l'actuel forme du fichier *.ino* :

```
#include <Tone.h>   //Aka "the library that saved us"

#include "notes.h"
#include "fonctions_basiques.h"

#include "principal_right_piano.h"
#include "other_left_piano.h"
#include "general_play.h"

#include "melodies.h"

void setup() {
  first_initializing();
  left_piano_initializing();
  right_piano_initializing();
  cch_short(left_tone, right_tone);
}

void loop() {
  commands(Serial.read());
  pianos_checking();
}
```

Je rappelle que l'ordre des importations (si je puis dire) a son importance : le contenu de *principal_right_piano.h* et de *other_left_piano.h* est primordial pour que celui de *general_play.h* ait du sens et fonctionne.

Pas de changement pour le *setup()*, mise à part la procédure *cch_short()* qui prend maintenant en compte deux paramètres obligatoires : avec les deux objets de type *Tone*, chacun représentant un buzzer, on peut faire un son stéréo au démarrage de la carte (avec les sons sortant de *left_tone*, le buzzer gauche, qui sont un octave plus bas).

Passons maintenant au *loop()* !

Voici le code de la (fabuleuse) fonction *pianos_checking()* :

```
void pianos_checking() {
  basic_updater();
  left_t_update();
  right_t_update();
  while (tl_played || tr_played) {
    basic_updater();
    if (correct_left_playing() || correct_right_playing()) {

      super_3_times_200_updater();

      basic_updater();
      if (right_all_ok()) {right_play(200);}
      else {right_tone.stop();}
    }
    left_t_update();
    right_t_update();
    commands(Serial.read());
  }
  left_tone.stop();
  right_tone.stop();
};
```

Je ne vais pas tout détailler, c'était juste pour montrer que le programme final se met constamment à jour (si je puis dire), le nombre de fois que l'on retrouve *«update»* en témoigne. Il y a aussi les fonctions *left_all_right()* et *right_all_ok()* qui participent à la mise à jour constante:

```
bool left_all_ok() {
  left_t_update();
  return (tl_played && correct_left_playing());
};
```

```
bool right_all_ok() {
  right_t_update();
  return (tr_played && correct_right_playing());
};
```

Terminons par la plus importante procédure appelée dans *pianos_checking()*. Son (petit) nom est *super_3_times_200_updater()*.

C'est une boucle *for* à 3 itérations dans laquelle on retrouve une boucle *for* à 4 itérations, celle-ci contenant un *delay(50)*. Ce n'est donc techniquement pas correct quand je dis «mise à jour constante», le «checking» est donc fait toutes les 50 millisecondes, délai tellement court qu'il est humainement imperceptible (si je puis dire).

Justification (inutile) du nom de cette fonction :

```
void super_3_times_200_updater() {
  for (int i = 0; i < 3; i++) {
    ...
    for (int j = 0; j < 4; j++) {
      delay(50);
      ...
    }
  }
};
```

3 × 4 × 50 millisecondes --> « 3 times 200 milliseconds »