COUSTON Emma
GRUET Lucie

BE Programmation Orientée Objet :

Introduction :

Dans le cadre du BE Programmation Orientée Objet, nous avons décidé de réaliser un petit **jeu « Question pour un champion » de 5 questions** prises aléatoirement dans une base de données de 15 questions. Le joueur, peut donc lancer le jeu, répondre aux 5 questions et avoir un score final avec une petite appréciation.

Diagramme de classe :

Après analyse de notre système, nous avons pu construire le diagramme de classe suivant :

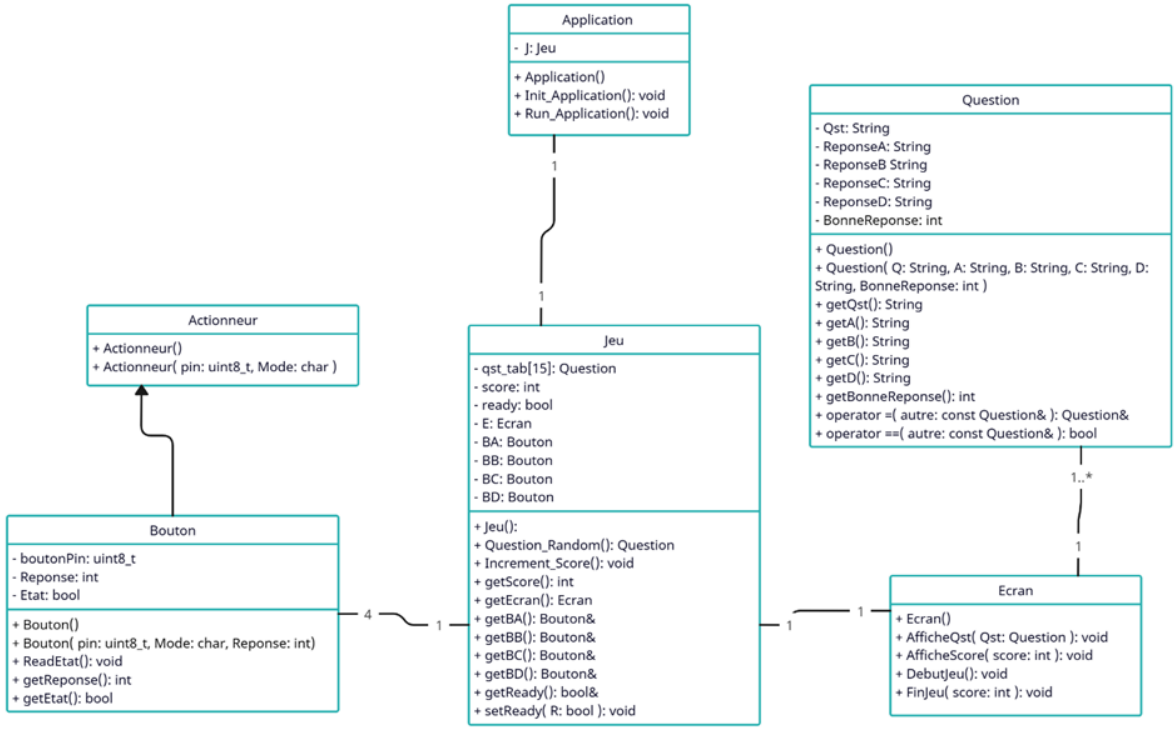
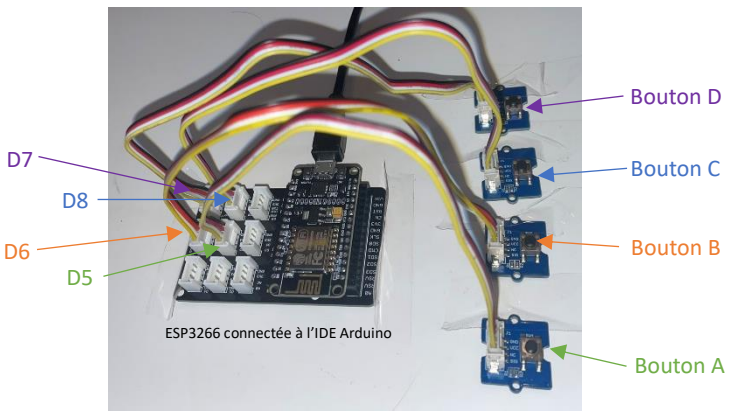


Schéma de fonctionnement :

Pour réaliser notre projet, nous avons utilisé une **carte ESP3266** ainsi que **4 boutons poussoirs**. L'objectif initial était d'afficher les questions sur un écran OLED (et non pas LCD car nous désirions avoir plus de 32 caractères). Cependant, par manque de temps et car nous n'avons pas trouvé de librairies fonctionnelles pour faire fonctionner notre écran OLED, nous affichons les questions dans le terminal d'Arduino.

Ci-dessous un schéma de fonctionnement :



ESP3266 connectée à l'IDE Arduino

Comment faire fonctionner notre système :

Point de départ :

- La carte ESP3266 est alimentée par un port de l'ordinateur et le programme est uploadé sur la carte (Attention, ne rien brancher sur les pin D3 et D8 lors de l'upload)
- Le logiciel Arduino IDE est ouvert sur le serial monitor (Tools → Serial Monitor)
- Les boutons poussoirs sont branchés de telle façon :
  - Bouton A sur la pin D5
  - Bouton B sur la pin D6
  - Bouton C sur la pin D8
  - Bouton D sur la pin D7

Lancement du jeu :

- Reset la carte ESP66 pour lancer le programme (bouton RST sur la carte)
- L'écran de bienvenue s'affiche, Appuyer sur le bouton A pour démarrer les questions
- Répondre aux 5 questions (qui s'affiche une par une) à l'aide des boutons poussoirs en appuyant sur le bouton correspondant à votre réponse
- L'écran de fin de jeu s'affiche (avec votre score)
- Pour recommencer une partie, Reset la carte

Le déroulement du projet :

Pour réaliser ce projet nous avions 3 séances de TP de 3h.

1ère séance :

Lors de cette séance, nous avons principalement **installé et pris en main les logiciels** que nous allions utiliser (Arduino IDE, GitHub + GitBash). Cela a pris beaucoup de temps car nous avons rencontré des problèmes logiciels pour connecter notre carte ESP3266. Nous avons d'ailleurs seulement réussi sur une de nos deux machines, c'est donc avec celle-ci que nous avons effectué tous nos tests, bien que nous programmions avec les deux.
Par ailleurs, c'est dans cette séance que nous avons **réalisé notre diagramme de classe**.

2ème séance :

Dans cette séance, nous avons commencé à **coder les premières classes** : **Actionneur + Bouton**, et **Question + Ecran**. Nous avons perdu beaucoup de temps sur cette séance car nous avions correctement codé les classes Bouton et Actionneur, mais les tests ne fonctionnaient pas. Nous sommes donc remontées pas à pas pour comprendre notre erreur, jusqu'à essayer de flasher un exemple de bouton poussoir avant de comprendre que nous avions un bouton branché sur la pin D3, ce qui empêchait le programme de s'uploader sur la carte…
De plus, nous avons commencé à nous confronter aux problèmes des librairies pour faire fonctionner un écran OLED en I2C.

3ème séance :

Dans cette séance, nous avons **codé la classe Jeu**, pour laquelle nous n'avons pas rencontré de problème particulier mais qui fut un peu longue à coder.
Nous avons également essayé, pendant toute la séance et avec l'aide des professeurs, de faire fonctionner l'écran OLED, avant d'abandonner et de décider de faire les affichages dans le terminal. Nous avons là aussi perdu beaucoup de temps, et étions très frustrées de ne pas pouvoir aller au bout du projet en affichant sur notre écran OLED

Temps en dehors :

Le projet n'étant pas du tout fini en seulement 3 séances, nous avons dû y consacrer pas mal de temps en dehors. Après avoir pris la décision de réaliser les affichages dans le terminal, nous avons **fini de coder la classe Ecran** et avons entamé le **code de la classe Application**.

Lors de la réalisation de ce code, nous avons rencontré un **problème majeur :** l'appui sur un bouton n'avait aucun effet dans notre programme. Ce problème a remis en question tout notre code car nous n'avions pas de raison de penser que cette commande ne marcherait pas. Les classes Actionneur, Bouton, Ecran et Jeu avaient toutes été testées et fonctionnaient et notre code dans Application était bon.

Après de très nombreux tests, nous avons réussi à remonter à **l'origine du problème** : la classe Application possède un attribut privé qui est une instance de la classe Jeu, qui elle possède un attribut privé qui est une instance de la classe Bouton, qui elle a un attribut privé qui est Etat. Ainsi, lors de l'appel J.getBA().ReadEtat(), par souci d'autorisation d'accès, ReadEtat() lisait l'Etat de BA mais le stockait dans une copie du Bouton BA et ne mettait donc jamais à jour le vrai attribut privé Etat de l'instance BA.

Nous avons donc réussi à **résoudre ce problème** en faisant retourner une référence de BA par getBA().
Par la suite, tout le reste du code d'Application fut réalisé sans encombre.
Enfin, nous avons dû désactiver les watchdog reset software et hardware afin d'assurer que le programme ne se reset pas durant l'attente d'appui sur un bouton. Tout reset s'effectue simplement par le bouton reset de la carte ESP3266.

Conclusion :

Pour conclure, nous avons apprécié ce projet car il fut très instructif et une bonne application pour bien intégrer le C++ et la programmation orientée objet en général à nos connaissances. Cependant nous avons été souvent bloquées par un manque d'information (comment faire lorsque la carte n'est pas détectée sur notre PC, pin à ne pas utiliser lors de l'upload, limitations d'accès aux attributs, quelles librairies utiliser pour de l'I2C…).
Nous avons également été très frustrées du manque de temps. Avec un peu plus de temps (et d'accès à l'aide des professeurs), nous aurions sûrement pu réaliser l'affichage sur écran OLED, rajouter un timer pour le temps de réponse aux questions, améliorer l'optimisation de notre code, ou encore rajouter des fonctionnalités de jeu comme « indice », choisir le nombre de réponses proposées etc…