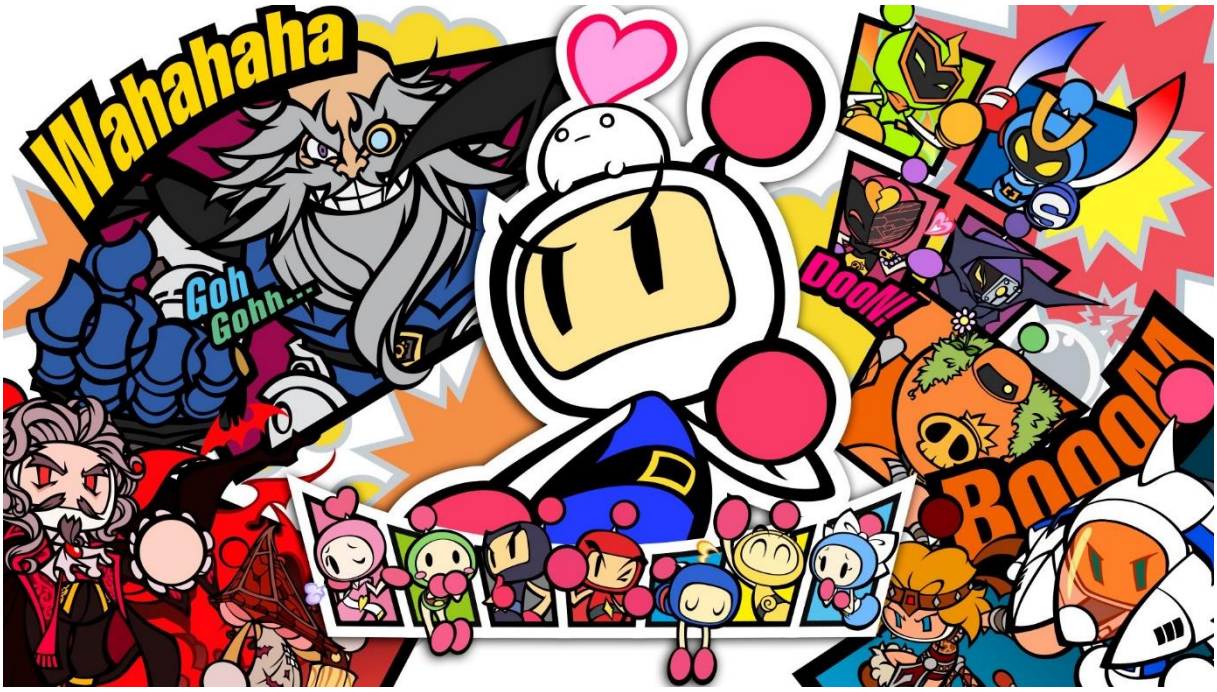
DOSSIER DE PROJET IN204



PROGRAMMATION EN C++ D'UN BOMBERMAN

Laetitia DEBESSE et Maxence NAUD

TABLE DES MATIERES

INTRODUCTION

Lors du cours d'IN204, nous avons appris les spécificités du C++ par rapport au C. Afin de mettre en application nos nouvelles connaissances, par groupe de deux, nous avons travaillé durant trois mois sur un projet.

Les sujets proposés étant des guides, nous avons choisi d'adapter celui s'intéressant à l'implémentation du jeu Tétris. Dès l'annonce des possibilités, Maxence a proposé l'idée de nous inspirer de Bomberman.

Figure 1 : Logo de notre jeu

LE PROJET

Présentation

Bomberman est un jeu datant de 1983 qui a su conserver son succès. En effet, depuis sa sortie, le jeu a été adapté sur de nombreuses consoles de jeu vidéo ainsi que sur ordinateur.

Figure 2 : Logo de Bomberman

Dans ce jeu, le joueur doit poser des bombes sur une carte afin de toucher ses adversaires lors de l'explosion tout en essayant de survivre. Le nombre de variations de cartes permet de créer de nouvelles ambiances et de mettre en place de nouvelles stratégies pour gagner.

Plusieurs modes de jeu existent et les derniers opus se sont enrichis de nombreux mini-jeux. Parmi les variations, une partie peut se jouer en temps ou en nombre de vies ou bien des objets collectables donnant un avantage ou un malus peuvent être introduits.

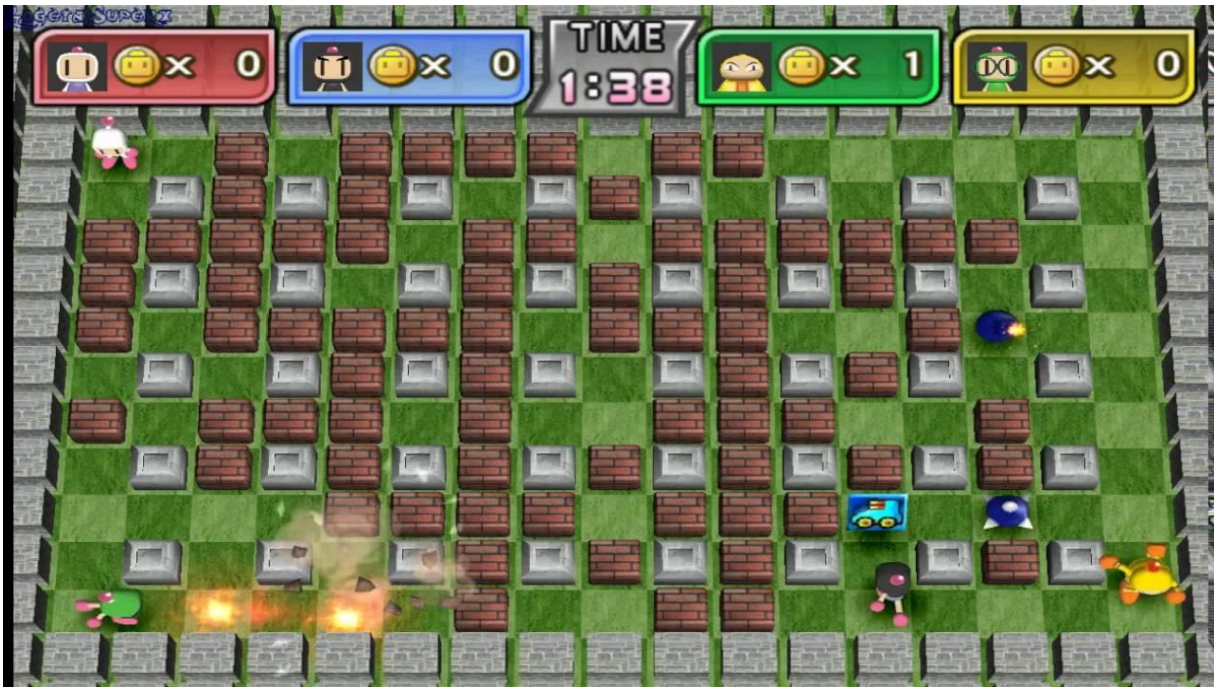


Figure 3 : Exemple de partie en temps (Wii)

Nos choix

Bomberman est donc un jeu offrant de nombreuses possibilités d'implémentation. Nous avons dû limiter le cadre du jeu pour respecter les délais imposés par le sujet.

Nous avons décidé de faire uniquement un mode deux joueurs local. Ils ont la possibilité de poser un nombre de bombes limités avec une amplitude de déflagration propre à chacun. Les explosions permettent à la fois de faire perdre une vie à un joueur, mais aussi de briser les murs cassables.

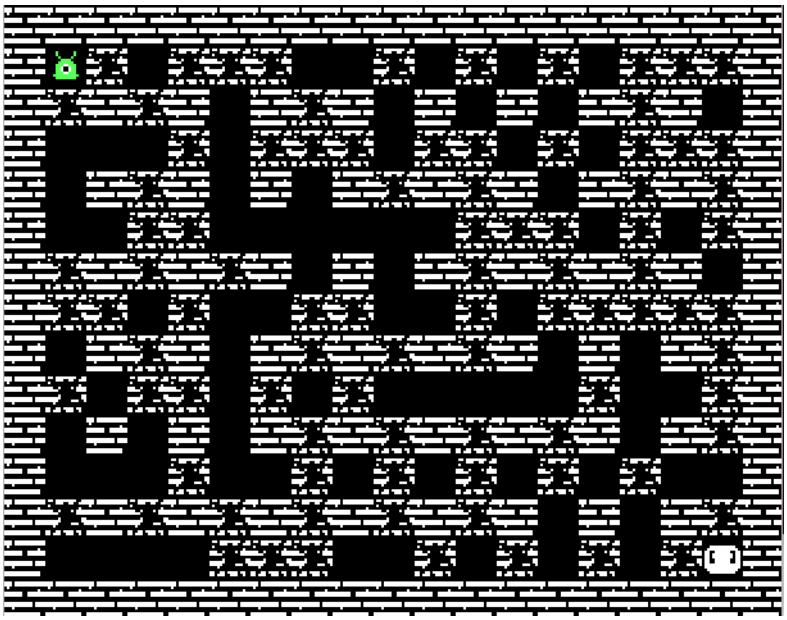


Figure 4 : Début de partie

Un menu permet au joueur de choisir son personnage ainsi que la taille de la carte. Les blocs cassables sont disposés aléatoirement sur celle-ci, tandis que les joueurs commencent dans les coins opposés.

Figure 5 : Choix du personnage et choix de la taille de la carte

NOTRE TRAVAIL

Architecture

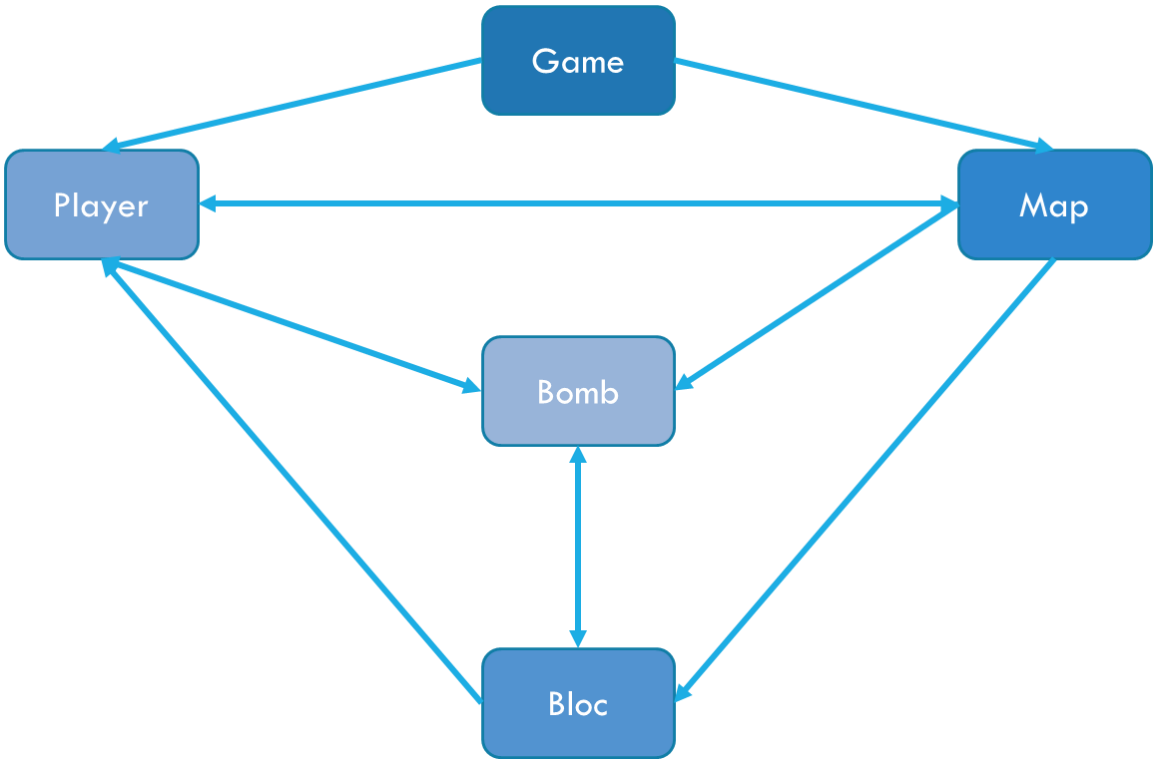


Figure 6 : Liens entre les différentes classes

Le cœur du jeu se situe dans la classe Game. C'est elle qui gère l'affichage des différents menus d'options et qui ensuite instancie les deux joueurs (Player) ainsi que la carte (Map).

La carte (Map) est constituée principalement d'un vecteur 2D contenant ses différentes cases (Bloc). C'est la classe Map qui s'occupe de traiter les entrées claviers des joueurs et d'appeler les fonctions correspondantes : déplacement d'un joueur, pose d'une bombe (Bomb).

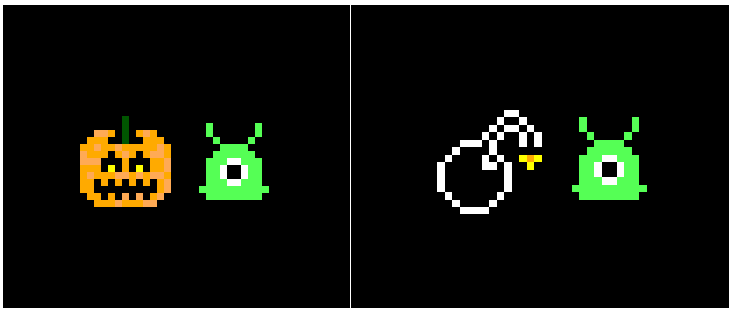


Figure 7 : Les deux modèles de bombe

Etapes

Nous avons commencé par discuter à deux de l'architecture du code, des classes à implémenter et de leurs attributs et méthodes. Suite à cela, nous avons fait le choix de programmer séparément. Maxence s'est dédié à l'affichage du jeu et Laetitia au « back ».

L'AFFICHAGE

Figure 8 : Les différents personnages du jeu

Nous avons choisi la bibliothèque Qt qui est très riche. Cependant, cet avantage s'est révélé être un obstacle en début de projet. En effet, la prise en main s'est faite en tâtonnant à cause du nombre de possibilités offertes.

Il a fallu comprendre et implémenter un moyen de créer une fenêtre et de l'afficher. Ensuite, la création d'un objet cercle et son affichage dans la fenêtre a été ajouté. Pour se rapprocher de l'utilisation des personnages plus tard, le lien avec les entrées clavier a été fait ce qui a permis de faire se déplacer le cercle dans la fenêtre. L'étape suivante a été de permettre à ce cercle de créer d'autres cercles qu'il laissait sur place à la manière des bombes.

Un blocage a eu lieu à ce moment-là. En effet, nous voulions utiliser nos propres images et non simplement des formes géométriques. Un long temps de documentation a été nécessaire pour trouver cet ajout. C'est la découverte du fichier .qrc qui a rendu cette volonté possible. Il permet de donner à Qt l'accès à des images, musiques stockées sur l'ordinateur.

Pour rendre le jeu plus convivial, des menus ont été ajouté. La classe Button de Qt a été modifiée afin d'afficher les images avec les textes nécessaires et avec le bon comportement au clic. Plusieurs fenêtres se succèdent ainsi du menu de démarrage, au choix des personnages jusqu'aux options.

L'esthétisme de l'affichage a été amélioré. L'apparence des personnages change selon leur sens de déplacement, le fond de la fenêtre a été changé et l'animation détaillée de la déflagration d'une bombe a été dessinée puis intégrée grâce à QTimer.

LE BACK

La première étape a été de créer la carte du jeu. Cependant, avant d'implémenter la classe Map, il a fallu implémenter la classe Bloc. Cette dernière hérite de la classe mutex. En effet, si deux joueurs essaient au même instant de se déplacer sur le même bloc, une concurrence d'accès aurait lieu. La classe mutex permet d'avoir accès aux fonctions *lock()* et *unlock()* et donc de protéger le bloc. Les blocs ont un type qui identifie leur nature parmi mur, cassable et vide. Quant à la classe Map, elle a pour attribut principal un vecteur deux dimensions de Bloc.

Suite à cela, la classe Player a été ajoutée. Il a fallu permettre à un joueur d'avoir une position initiale sur la carte, et de bouger dans les quatre directions tout en s'assurant qu'il ne puisse pas sortir des limites.

Les bombes ont alors été intégrées avec la classe Bomb. Elles sont initialisées à partir des attributs de Player comme notamment l'amplitude de la déflagration et le temps avant explosion. Pour commencer, poser correctement une bombe sur la case voulue a été implémenté, puis provoquer l'explosion avec la destruction de l'objet.

Pour réaliser les tests à chaque étape permettant de s'assurer du bon fonctionnement des ajouts, un fichier *main* a été édité. Dès l'introduction de la classe Player, les threads C++ ont été employés afin de simuler plusieurs joueurs en même temps.

Figure 9 : Les déflagrations des bombes

LA FUSION

Notre travail ayant été très indépendants durant deux mois, la fusion a été complexe. Nous avons choisi d'ajouter le « back » à l'intérieur du code gérant l'affichage. Les classes déjà présentes dans l'affichage ont été enrichies avec les attributs et les méthodes de leur homonyme du « back ». Les autres ont été copiées et collées.

L'ajout du « back » à l'affichage a permis d'implémenter des fonctions supplémentaires. Les bombes sont surveillées par Qt qui provoque leur explosion et les conséquences sur les objets atteints comme par exemple, la destruction des blocs ou la perte de vie des joueurs.

L'affichage ne présentant aucune carte de jeu, nous avons dû intégrer complètement la classe Map à la classe Game, puis changer la gestion des joueurs (Player). Nous avons ajouté un second joueur local, les blocs cassables et une initialisation aléatoire de la carte.

La tâche la plus compliquée a été l'implémentation des bombes (Bomb). L'affichage des bombes était très complet mais il fallait réussir à ajouter toutes les fonctions du « back ». De nombreux « segmentation fault » nous ont presque découragés mais à force de persévérance nous sommes arrivés au bout de chaque bug.

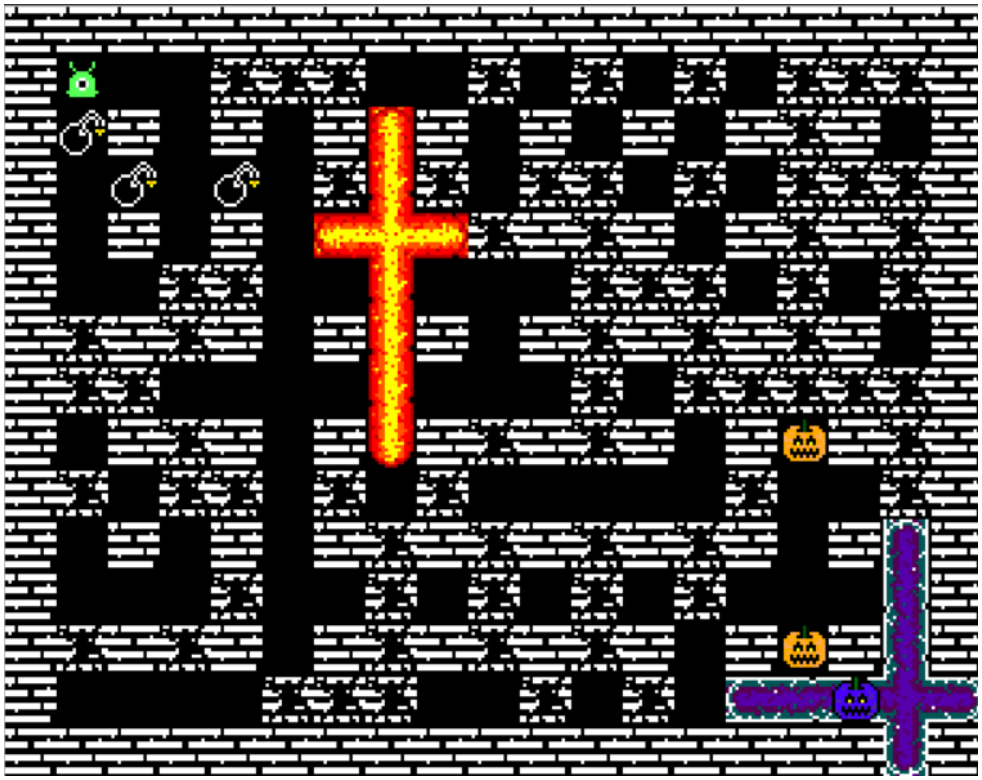


Figure 10 : Partie en cours opposant Dipsy et Tinky Winky

PISTES D'AMELIORATIONS

Ce que nous aurions aimé faire

Par contrainte de temps et également de soucis de place sur un seul clavier, nous n'avons pas pu implémenter jusque 4 joueurs locaux. Nous aurions aimé pouvoir rendre possible un mode multijoueur via internet. Cependant, il aurait fallu en plus créer un serveur ce qui nécessite d'autant plus de temps.

Afin d'enrichir l'expérience utilisateur nous voulions ajouter des objets qui apparaitraient aléatoirement sur la carte. Ceux-ci permettraient d'augmenter la portée des bombes du joueur ou bien de diminuer leur temps avant explosion. Cela ajouterait un autre enjeu en dehors d'attaquer son adversaire qui serait améliorer au mieux et au plus vite son personnage.

Ajouter différents modes de jeu comme par exemple des éléments qui peuvent se modifier sur la carte permettrait d'ajouter des éléments de gameplay interactifs. Avoir la possibilité de jouer contre une intelligence artificielle permettrait aux gens seuls de pouvoir jouer.

Ce que nous pourrions améliorer

Dès que nous cliquons à l'intérieur de la fenêtre en jeu, le focus se perd et nous ne pouvons plus ni déplacer les joueurs ni poser de bombes ni quitter la fenêtre avec la touche « échap ». Cependant, il existe une solution ! Appuyer sur « tab » permet de resélectionner la carte correctement.

Ajouter une fenêtre annonçant la fin de partie et le gagnant permettrait de mieux comprendre lorsque le jeu se termine. Pour le moment, le joueur mort s'affiche en grand au centre de la fenêtre. Permettre aux joueurs d'inscrire leur nom permettrait de garder une trace des scores de chacun.

Figure 11 : Mort de Laaa-la

CONCLUSION

Ce projet nous a permis de mener un projet d'un bout à l'autre sur plusieurs mois. Dorénavant, nous sommes capables d'utiliser C++ pour coder. Ce travail à deux nous a appris à nous organiser mais également à valoriser les compétences de chacun selon le moment.

NOTRE CODE

Compilation

La compilation sur Mac et Linux se fait de manière similaire. (Les instructions précédées d'un « $ » sont à faire dans le terminal à la racine du projet)

$ qmake -project

Ouvrir le fichier affichage.pro et y ajouter à la toute fin la ligne :

QT += gui core multimedia widgets

$ qmake

$ make

Attention : la compilation prend beaucoup de temps lors de la génération de

Lien github

https://github.com/piplouille/bomberman

Manuel du jeu

FONCTIONS PRINCIPALES

Fonction	Touche
Quitter	Echap
Revenir au jeu (focus)	Tab

JOUEUR 1

Fonction	Touche
Déplacement en haut	Flèche du haut
Déplacement à droite	Flèche de droite
Déplacement en bas	Flèche du bas
Déplacement à gauche	Flèche de gauche
Poser une bombe	Shift

JOUEUR 2

Fonction	Touche
Déplacement en haut	Z
Déplacement à droite	D
Déplacement en bas	S
Déplacement à gauche	Q
Poser une bombe	E