

Juliette HEUANGTHEP & Axel CANTE (TD 04)

ECE PARIS | ING2 – PROMO 2019

*Rapport de projet : SNOOPY’S REVENGE*



PRESENTATION DU SUJET

Dans le cadre de l’enseignement *C++ : Programmation orientée objet* de la 2nde année du cycle préparatoire aux études d’ingénieur, il nous a été proposé de réaliser le jeu console « La Revanche de Snoopy ».

Il s’agit d’un jeu de réflexion qui consiste à contrôler Snoopy de sorte qu’il récupère quatre oiseaux dans un temps limité tout en évitant les pièges (ex : balle rebondissante, blocs piégés, …) afin de passer au niveau suivant et augmenter son score.

Le cahier des charges nous indique qu’il faut un personnage : Snoopy, quatre oiseaux à récupérer, une balle qui tue Snoopy lorsqu’il le touche et des blocs (poussables, cassables et piégés, ces derniers tuent Snoopy).



Figure Snoopy's Magic Show, jeu vidéo créé en 1990

SOMMAIRE

PARTIE CONCEPTION

I. Organisation de l’équipe

II. Planning

III. Diagramme de classes général

IV. Scénarios

V. Modèle et contrôleur des parties clefs du projet et découpage modulaire

PARTIE REALISATION

I. Graphe d’appels et organigrammes de sous-programmes essentiels

II. Astuces et originalités

III. Protocole de tests

IV. Bilan objectif collectif et individuel

V. Sources

PARTIE CONCEPTION

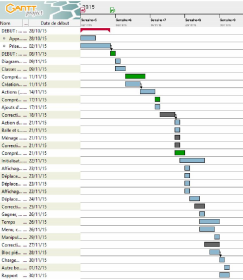
1. Présentation de l’équipe

Nous sommes une équipe de deux ex-Prépac : CANTE Axel et HEUANGTHEP Juliette su TD04 ING2.

Le fait que nous soyons issus de la même classe d’ING1 accélérée a facilité notre entente même si nous ne nous connaissions pas. Ceci nous a permis d’être efficaces en termes d’organisation et de cohésion de groupe.

Nous nous sommes naturellement organisés comme suit :  
Nous avons débuté par la théorie : le C++, en nous exerçant sur les TP et cours de C++ donnés par M. Diedler ainsi que OpenClassroom, et le versionning grâce aux cours de M. Cros et M. Pinto. Nous avons de même étudié le cahier des charges et établi des diagrammes de classe.  
Puis, nous sommes passés à la pratique en nous entraidant lors de nos séances de travail.

1. Planning : organisation de l’équipe



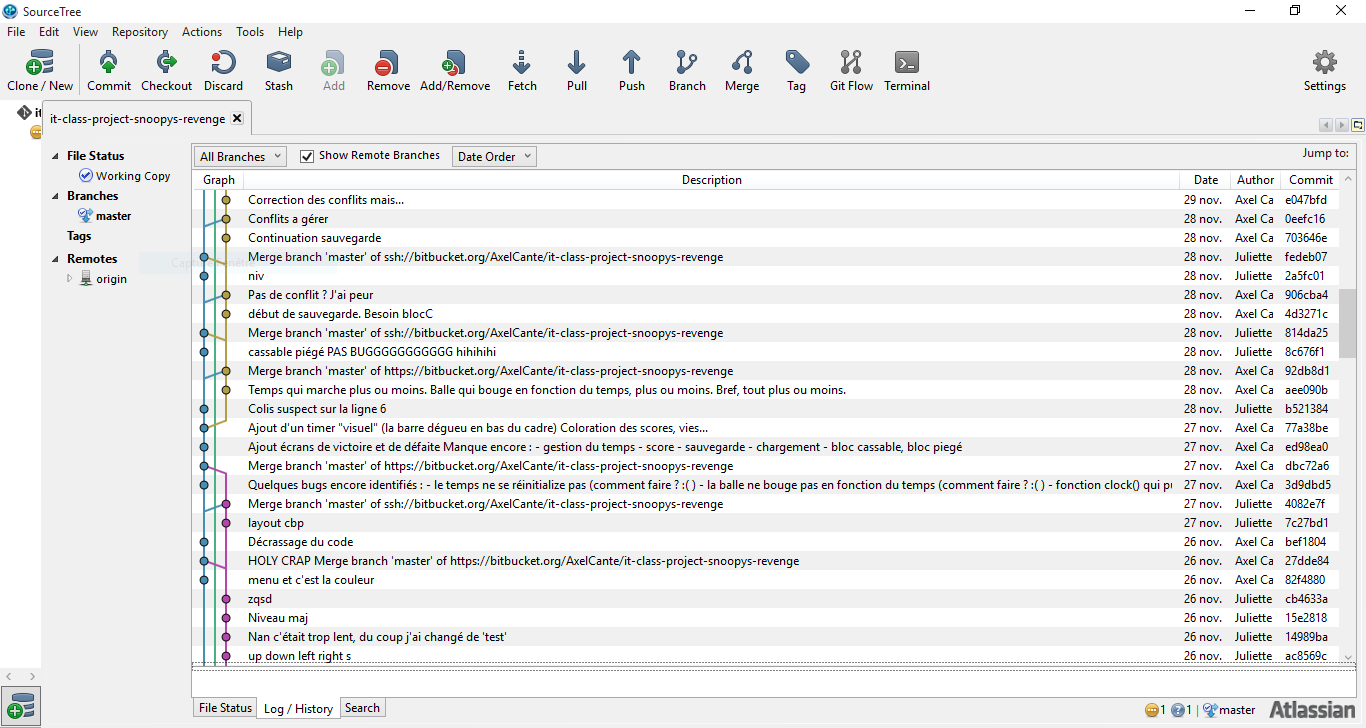
C’est le diagramme de Gantt (logiciel utilisé pour planifier notre projet).



Nous avons utilisé GanttProject afin de réaliser un planning de ce qui a été établi durant la mise en place du projet.

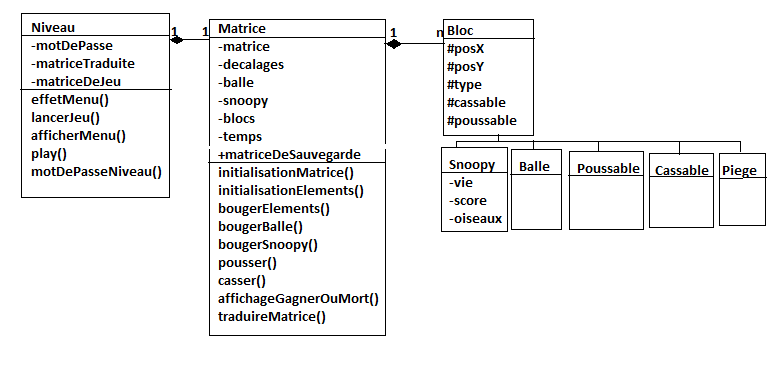
Pour plus de précision, le code couleur est le suivant :

* rouge : les étapes importantes de début ;
* bleu : l’avancement dans le code ;
* vert : la compréhension approfondie de SourceTree (\* voir page suiante);
* gris : ménage et correction de bugs.



Ceci est le logiciel SourceTree qui permet d’utiliser gitbucket afin de versionner le projet en équipe.

1. Diagramme de classes général



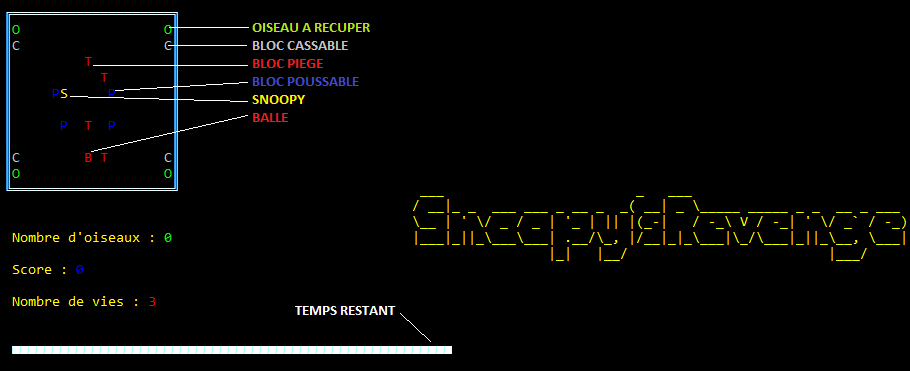
Voici le diagramme des classes de notre jeu. Il permet de montrer la visibilité des membres des classes, leur cardinalité et leur relation entre elles

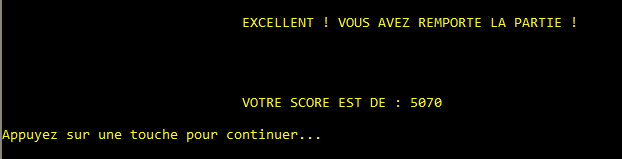
Il y a une matrice pour un niveau et il y a n blocs pour une matrice. Le niveau et la matrice sont liés fortement (composition), les blocs et la matrice le sont aussi (relation de composition).

1. Scénarios

Dans cette sous-partie nous observons une partie classique de « La revanche de Snoopy ».



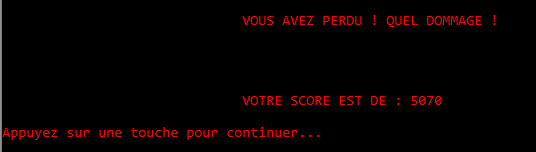
1 Le menu s’affiche à l’cran et l’utilisateur choisit une nouvelle partie  
  
  
  
2 On entre dans le niveau 1 après ue petite aide  
  
  
3 Schéma explicatif : La matrice contenant les différents blocs (poussables, cassables, piégés), Snoopy et la Balle.



4 Gagner la partie



5 Passer au niveau suivant



6 Ou bien perdre (quand on manque de temps ou bien qu’on a plus de vie)

On retourne au menu ensuite. On pouvait également choisir un bonus ou bien un niveau spécial en entrant un mot de passe correct.

1. Modèle et contrôleur des parties clefs du projet et découpage modulaire
2. MVC

MVC non exhaustif :

|  |  |
| --- | --- |
| Données | Actions |
| Vies, score, oiseaux : entiers | Calculer, Afficher |
| Matrice : vecteur de vecteur de classe | Calculer, Afficher |
| Menu : chaînes de caractères | Afficher |
| Temps : double | Calculer, Afficher |
| Directions : caractère | Calculer |
| Nombre de lignes et de colonnes de la matrice : entiers | Définies par défaut |
| Différents blocs, le personnage : blocs | Créer, Modifier, Afficher |
| Choix : caractère | Saisie |

1. Analyse chronologique descendante
2. Afficher la page d’accueil du menu : avec les choix possibles (Nouvelle partie, charger partie, high scores, entrer mot de passe, quitter le jeu).
3. Obtenir le choix du joueur concernant le traitement à faire
4. Entrer dans le sous-programme sélectionné

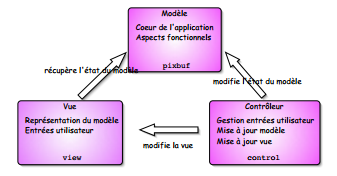
Figure 2 Organigramme : ACD général

1. Découpage modulaire du projet

L’organisation de notre code permet une compréhension lisible du code :

* Main.cpp : Lance le programme.
* Niveau.cpp : gère l’interaction avec le joueur (comme le menu par exemple) et met à jour le jeu selon le niveau.
* Console.cpp : gère la vue.
* Matrice.cpp : gère les fonctions qui permettent la création de matrices, elle les initialise, les manipule.
* D’autres fichiers.cpp correspondant à des classes qui manipulent, calculent via des méthodes.

L’architecture MVC se présente sous cette forme :



http://pageperso.lif.univ-mrs.fr/~liva.ralaivola/lib/exe/fetch.php?id=teaching%3A20062007%3Aprojet&cache=cache&media=teaching:20062007:modules.pdf.

PARTIE REALISATION

1. Graphe d’appels et organigrammes de sous-programmes essentiels
2. Astuces et originalités

* *Barre de temps visuelle*
* *Prévenir ‘a’ pour casser quand cassable*
* *Balle rebondissante sur les blocs*
* *Mot de passe pour bonus*
* *Niveau caché ave possibilité d’améliorer le score en jouant depuis le niveau 1*
* *Action bonus (bombe)*
* *Modifier les points de vie en passant par bonus*
* *Défiler dans le menu*
* *On ne voit plus le curseur dégueulasse*
* *Couleurs, cadre affichage de caractères bizarres :P*

1. Protocole de tests
2. Bilan objectif collectif et individuel
3. Sources