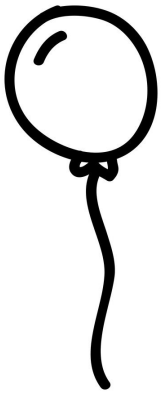
UNIVERSITÉ DE MONTPELLIER

I.U.T MONTPELLIER - SETE

RAPPORT FINAL DE PROJET



Jeu pour enfants

Réalisé par

Guilhem CROS
Daniel RAJAONARIVELO
Etienne TILLER
Yi YANG

Sous la direction de
Francis GARCIA

Pour l'obtention du DUT

Année universitaire: 2020 - 2021

Remerciements

Au terme de cette réalisation, nous tenons à exprimer notre profonde gratitude à notre professeur et encadrant M.Garcia Francis, pour son suivi et son soutien, qu'il n'a cessé de nous prodiguer tout au long de la période du projet.

Nous voulons également remercier Mme.Messaoui Anita pour le temps qu'elle a consacré et les précieuses informations qu'elle nous a prodiguées pour la rédaction de ce rapport.

Merci à Jérémy Macq pour son aide précieuse et ses conseils techniques.

Nous adressons nos remerciements aux membres des jurys pour avoir bien voulu examiner et juger notre travail.

Enfin, merci à toutes les personnes qui nous ont donné de leur temps pour tester l'application.

Sommaire

Table des figures

Glossaire

ARCEP : Autorité de régulation des communications électroniques et des postes

Programmation événementielle : Opposé à la programmation séquentielle, ce type de programmation repose principalement sur les événements ou changements d'états de variable (exemple : clic de souris).

Programmation orientée prototype : forme de programmation orientée objet sans classe. Elle est fondée sur la notion de prototype.

Prototype : Objet à partir duquel on crée de nouveaux objets. Tout objet Js contient un lien vers un autre objet de type prototype. Ce prototype possède aussi un prototype, et ainsi de suite, jusqu'à ce que le prototype d'un objet soit NULL.

Introduction

En France, 77% de la population de plus de douze ans possède un smartphone selon l'ARCEP* [1]. Ce chiffre ne cesse de croître depuis ces dernières années. Les smartphones, ainsi que les technologies tactiles en général, prennent de plus en plus de place dans nos vies, et continuent de se développer.

Aujourd'hui, dès le plus jeune âge, les enfants sont confrontés au numérique et parfois à ces technologies tactiles. Ces enfants ne sont pourtant pas sensibilisés à l'utilisation ni à la compréhension de ces technologies, et peuvent avoir des difficultés avec celles-ci.

Le projet qui nous a été confié consiste donc à développer une application, sur la demande d'écoles, sensibilisant de jeunes enfants de maternelle aux technologies tactiles. Cette application devra se présenter sous la forme d'un jeu sur tablette, dans lequel l'élève pourra se familiariser et s'entraîner à l'utilisation de la technologie tactile.

Nous allons dans un premier temps formuler toutes les exigences fonctionnelles et spécifications techniques dans le cahier des charges. Ensuite, après avoir détaillé les choix de conception effectués pour la création de notre application dans le rapport technique, nous présenterons les résultats du projet, vous trouverez également dans ce document un manuel d'utilisation et d'installation pour les enseignants. Enfin, un rapport d'activité décrivant les méthodes de travail que nous avons utilisées pour mener à bien le projet sera étudié.

1. Cahier des charges

Afin de mener notre projet à bien et de nous aider dans notre réalisation, nous avons réalisé un cahier des charges. Vous trouverez dans celui-ci la présentation du sujet de notre projet, les besoins fonctionnels ainsi que les besoins non-fonctionnels de l'application à réaliser.

1.1. Présentation du sujet et analyse du contexte

La réalisation d'une application ludique sur tablette, visant à sensibiliser des enfants, de trois ans environ, aux technologies tactiles, est une demande de plusieurs écoles primaires de la métropole de Montpellier.
Le logiciel sera utilisé dans des écoles, sur des tablettes fonctionnant sur le système d'exploitation Windows. Celui-ci devra être fonctionnel sans connexion internet, et toutes les données seront stockées sur la tablette utilisée par l'enseignant.
Il s'agit donc d'une réalisation visant à apporter des connaissances et des capacités à de très jeunes enfants dans le milieu scolaire. L'application se doit alors d'être simple d'utilisation et stimulante vis-à-vis de ces enfants. Mais elle doit aussi être simple et utile à l'enseignant, afin de lui apporter une plus-value dans son activité éducative.

1.2. Analyse des besoins fonctionnels

Comme expliqué précédemment, notre projet a pour but de permettre à un enseignant de pouvoir initier ses élèves de maternelle à l'utilisation d'appareil tactile.
Une partie dans ce jeu consiste, pour le joueur, à appuyer sur les ballons qui défilent sur l'écran pour les crever. Plusieurs niveaux de difficultés seront mis à disposition, les ballons se déplacent plus vite que dans les précédents niveaux.
Notre projet devra répondre à certains critères.Premièrement, l'enseignant doit pouvoir enregistrer ses élèves dans l'application afin de pouvoir par la suite les noter. Il doit également pouvoir modifier ou supprimer un élève si ce dernier ne fait plus partie de l'établissement par exemple.Ensuite, l'enseignant doit pouvoir lancer une partie du jeu qui exercera l'enfant à l'utilisation de la technologie tactile. C'est lui qui lancera la partie et qui ensuite, laissera l'enfant prendre le contrôle de l'appareil durant l'exercice, il interviendra ensuite pour mettre fin, manuellement, à cette dernière. Au moment où il met fin à la partie, il choisit une note qui sera représentée

par un dessin, ayant pour but d'encourager ou de féliciter l'enfant. Il pourra retrouver cette note dans les statistiques de l'application, où seront stockés tous les joueurs enregistrés ainsi que leurs différentes notes obtenues. Aussi, le temps d'éclater les ballons par chaque élève est inclus dans cette liste.

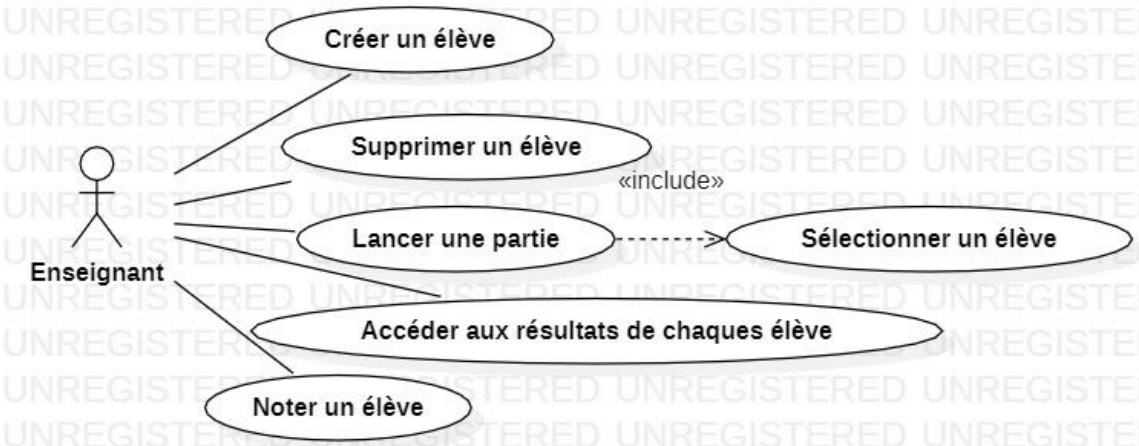


Figure 1 : Diagramme des cas d'utilisation

La figure 1 met en évidence les besoins fonctionnels de l'application pour un enseignant. Celui-ci souhaite pouvoir créer un élève, c'est-à-dire ajouter un joueur à la base de données de l'application, il devra également pouvoir le supprimer de cette base de données. Ensuite, il devra être en mesure d'accéder aux résultats déjà obtenus par les élèves. Enfin, il devra pouvoir lancer une partie en sélectionnant un joueur, qui correspondra à l'élève qui jouera cette partie, puis noter cet élève s'il le souhaite.

Afin de représenter plus concrètement les besoins et attentes du client, une maquette de l'application a été réalisée, sous forme d'esquisses (voir figure 2). Cette maquette a été réalisée de telle façon que l'application soit facile à prendre en main pour un adulte.

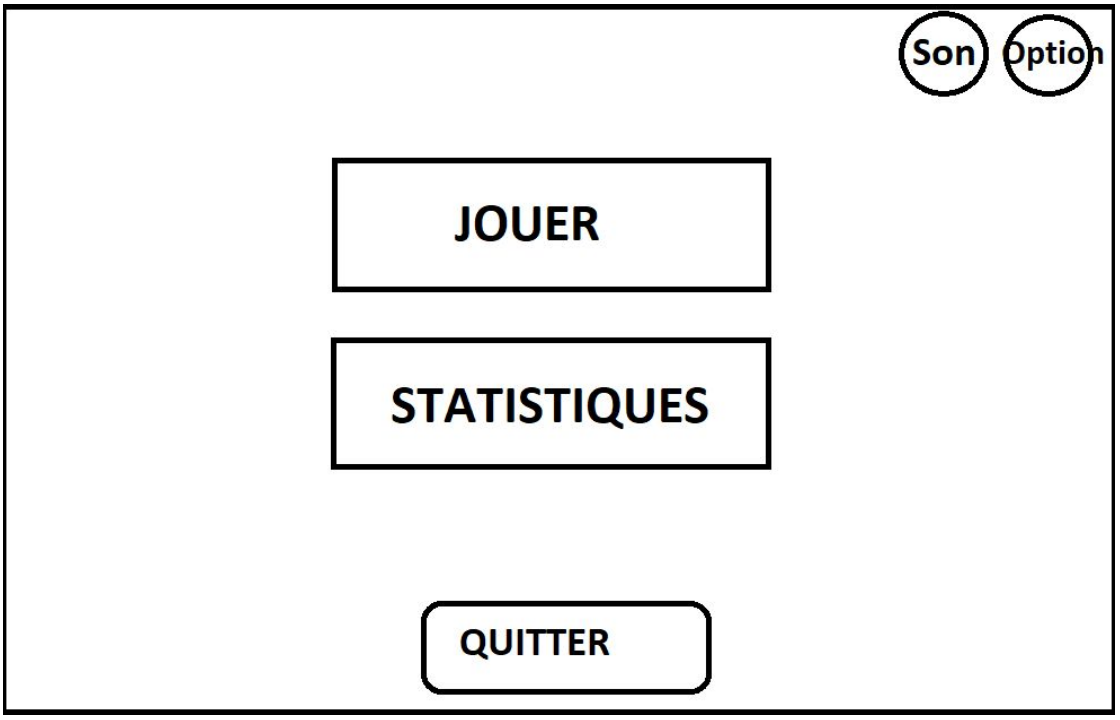


Figure 2 : Esquisse de la page d'accueil

Sur la figure 2, le bouton "jouer" permet à l'utilisateur de s'orienter vers le lancement d'une partie.
"Statistiques" correspond à la page administrateur, on y trouve les noms des joueurs ayant été créés, leurs statistiques et la possibilité de supprimer ces joueurs de la base de données.
"Quitter" ferme l'application.
Le bouton "son" permet de couper/activer le son du jeu.
Le bouton "option" ouvre une fenêtre d'options qui sont à établir.

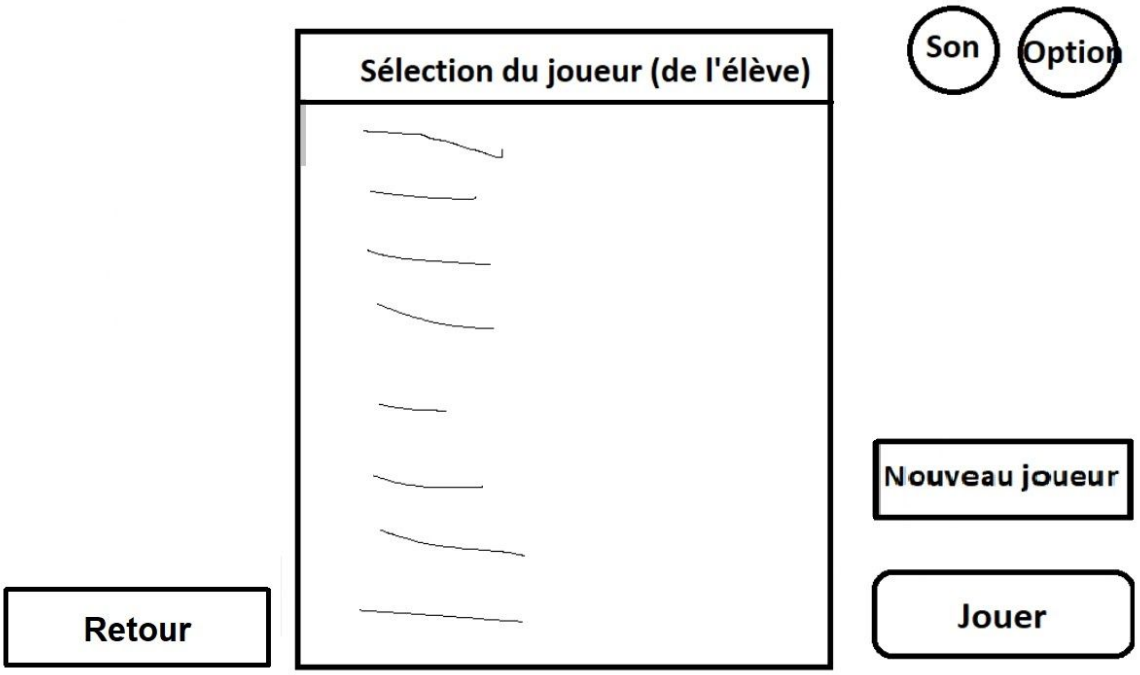


Figure 3 : Esquisse de l'onglet pré-jeu

La maquette (figure 3) correspond à la page qui s'ouvre après avoir cliqué sur "Jouer" dans la page d'accueil.
L'onglet retour ramène l'utilisateur à la page d'accueil.
Afin de lancer une partie, l'utilisateur sélectionne un élève dans la liste puis clique sur jouer. Si l'élève voulu n'existe pas, il peut le créer avec l'onglet "Nouveau joueur".

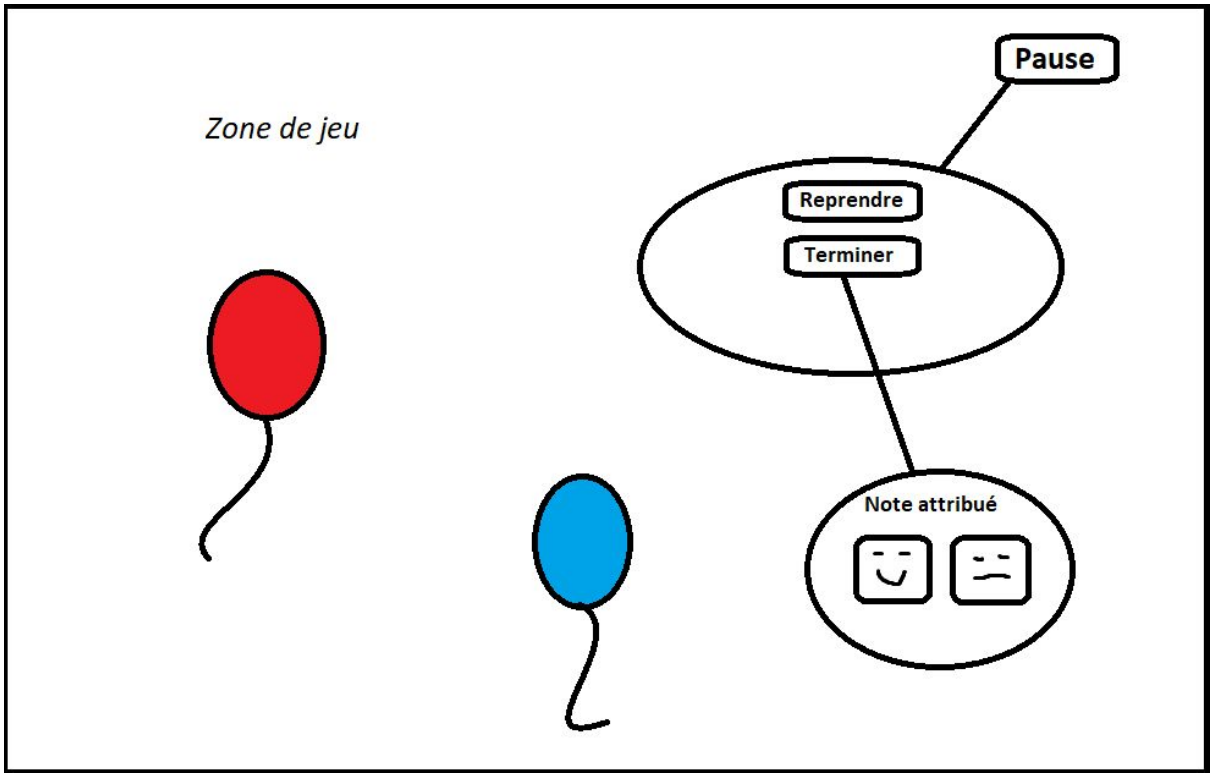


Figure 4 : Esquisse générale du jeu

La figure 4 correspond à la page de jeu qui s'affiche lorsque l'enseignant décide de stopper la partie de jeu en cliquant sur l'onglet "Pause". Il peut ensuite choisir de reprendre la partie, celle-ci reprendra alors à l'endroit exact où elle a été stoppée, ou il peut choisir de la terminer. S'il termine la partie, il doit alors mettre une note à l'élève, représentée par un smiley : souriant pour féliciter l'élève, neutre pour l'encourager. Cette note sera stockée et accessible dans les statistiques de l'application. S'affiche alors ensuite une page visant l'enfant.

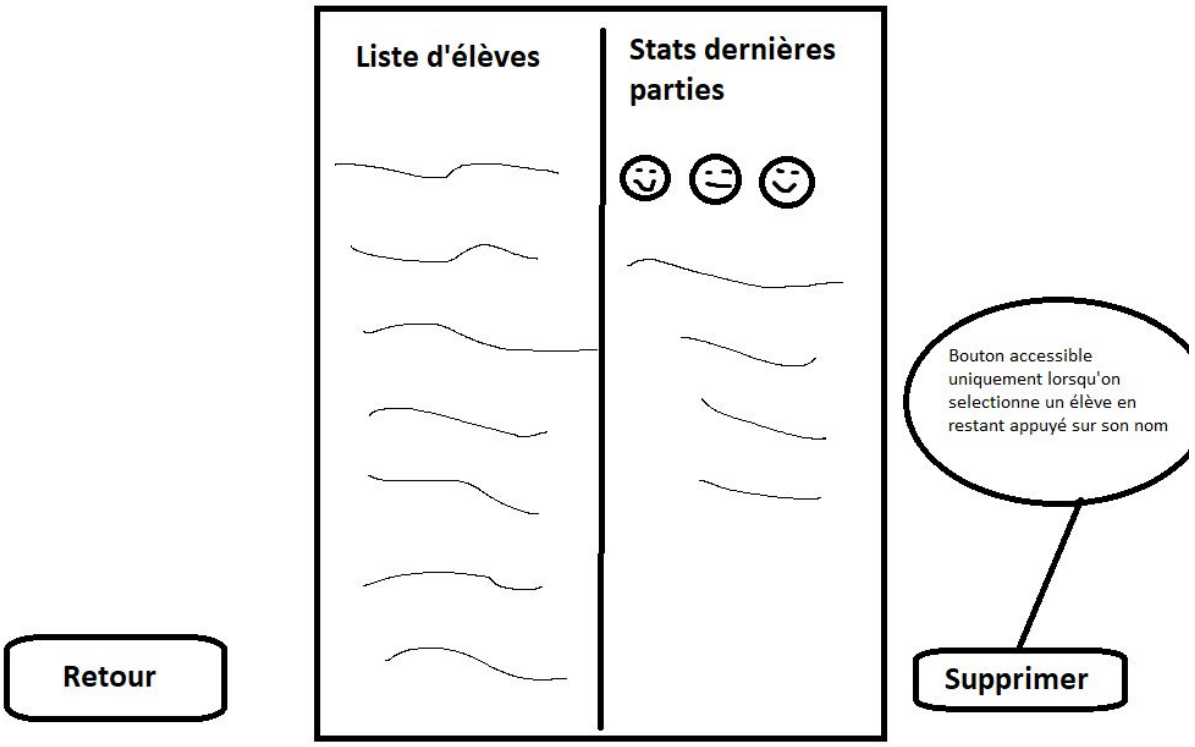


Figure 5 : Esquisse de l'onglet des statistiques

Sur la figure 5, l'onglet retour permet à l'utilisateur de revenir à la page d'accueil.
Le bouton supprimer n'est pas visible, il apparaît seulement si l'utilisateur reste appuyé quelques secondes sur un élève. Il peut alors le supprimer. Toutes les données liées à l'élève le sont alors également.
A l'inverse, lorsqu'un élève sera créé, c'est un onglet similaire à la figure 6 qui apparaîtra.

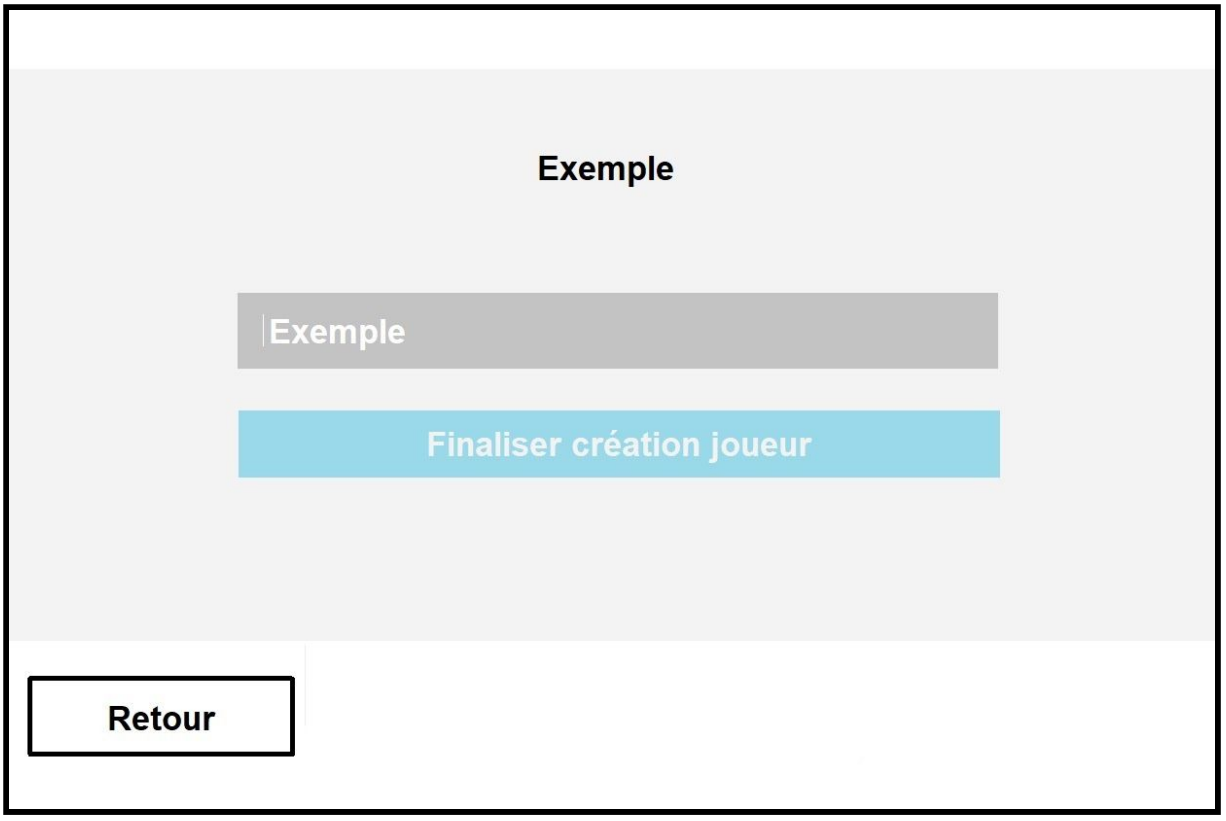


Figure 6 : Esquisse de l'onglet permettant la création d'un joueur

1.3. Analyse des besoins non-fonctionnels

Le rendu final du projet doit être adapté sur les tablettes distribuées par la mairie de Montpellier. Notre jeu doit donc être adapté sur Windows. Nous nous ne soucions donc pas de la compatibilité avec les appareils fonctionnant sous d'autres environnements. De plus, par souci de simplicité pour le client, nous devons générer un fichier exécutable qui lancera notre jeu. Nous n'avions aucune contrainte sur le langage de programmation mais notre rendu final doit être un fichier exécutable sur les tablettes déjà présentes dans les écoles. C'est pourquoi, la taille de notre programme ne doit pas excéder la capacité de stockage de la tablette.

Juridiquement, notre application est mise à disposition pour un usage pédagogique et non commercial.

Dans le but de mieux satisfaire notre client, on adapte l'ergonomie de notre programme. Pour que les enfants se sentent plus à l'aise, des İmages colorées et attrayantes sont utilisées dans notre programme. De plus, afin de diminuer le stress des enfants pendant qu'ils jouent, une musique joyeuse et relaxante est jouée en

fond de l'application. Pour faciliter le jeu aux enfants, des couleurs contrastées pour le fond et les ballons sont mises en place. Les enfants peuvent visualiser leur résultat dans l'écran directement pour se positionner.

Une fois le cahier des charges réalisé, et les besoins de l'application établis, nous avons pu commencer à réfléchir à la conception de notre programme.

2. Rapport technique

Notre projet se doit de respecter le cahier des charges défini, plusieurs choix se sont alors présentés. Vous trouverez dans ce rapport technique les choix de conception effectués ainsi qu'une description du fonctionnement du code permettant à l'application de remplir ses fonctions .

2.1. Conception

L'objectif de la conception du projet est de créer un système permettant de répondre aux besoins de celui-ci, tout en tenant compte des contraintes. Cette partie permet de mettre en évidence les choix pris lors de la conception de notre application, et qui ont guidé la réalisation de celle-ci.

2.1.1. Choix technologiques

La grande majorité des logiciels et applications compatibles avec Windows sont réalisés en C# ou C++. Cependant, ces langages de programmation nous sont presque inconnus. Ainsi, nous avons choisi de réaliser l'application autrement, afin de prendre le moins de temps possible dans l'apprentissage d'un nouveau langage.
Bien qu'il ne s'agisse pas d'une application ou d'un site WEB, l'application est entièrement construite à partir de langages WEB. C'est-à-dire que la partie visible et esthétique de l'application est principalement réalisée en HTML et CSS.
Les fonctions nécessitant une partie algorithmique sont quant à elles codées en JavaScript. De plus, certaines fonctionnalités apportées par la librairie Node Js sont utilisées, en particulier pour le stockage des données liés aux élèves.
Node.js est une plateforme logicielle en JavaScript orientée vers les applications réseau événementielles* [4], c'est-à-dire, des applications principalement définies par leurs événements, leurs changements d'état, comme un clic de souris ou un appuie sur une touche du clavier [5]. Ces évènements sont très importants dans notre contexte et facilitent le fonctionnement de notre application.
Enfin, la transformation de ce code, écrit langage WEB, en application, est effectuée à partir d'un module lié lui aussi à Node Js : Electron. Electron permet le développement d'applications de bureau, multi plateformes, en langages WEB. Ce module est la clé de notre application, celle-ci est en totalement dépendante et ne peut fonctionner sans. En conséquence, plusieurs fonctionnalités proposées par électron sont également utilisées pour le bon fonctionnement de l'application.

2.1.2. Conception de la base de données

L'un des besoins fonctionnels de l'utilisateur est le stockage de données liées aux joueurs. Cependant, bien que l'application soit codée en langage WEB, celle-ci ne doit pas avoir besoin de connexion Internet pour fonctionner ; les données sont donc stockées localement, sur la machine de l'utilisateur. Cela ne pose pas vraiment problème étant donné la faible quantité de données à stocker par l'utilisateur. Plusieurs frameworks ou modules liés à Node Js permettent un stockage de données local, mais la plupart sont difficiles à mettre en place ou bien trop complexes à utiliser pour le peu de données à stocker. Notre choix s'est alors porté sur l'utilisation de la fonctionnalité "File System" de Node Js. Celle-ci permet la lecture et l'écriture dans un fichier, son fonctionnement est sensiblement proche des fonctions d'écriture et de lecture de fichiers en C. Les données des joueurs sont finalement stockées dans un même fichier JSON, un type de fichier ayant la capacité de comprendre les objets JavaScript.

2.1.3. Conception des algorithmes

Comme énoncé précédemment, les algorithmes sont codés en JavaScript. Ceux-ci sont déclenchés par l'activation d'évènement, tels qu'un clic. Le JavaScript est un langage orienté objet à prototype* [2], c'est-à-dire qu'il s'agit d'un langage ne nécessitant pas la création de classe pour fonctionner, il utilise des prototypes*. Il s'agit d'un objet à partir duquel on va créer d'autres objets [3]. Un diagramme de classe pour un projet en JavaScript n'est donc pas forcément clair. La figure 7 a pour but de présenter une vision du fonctionnement de l'application.

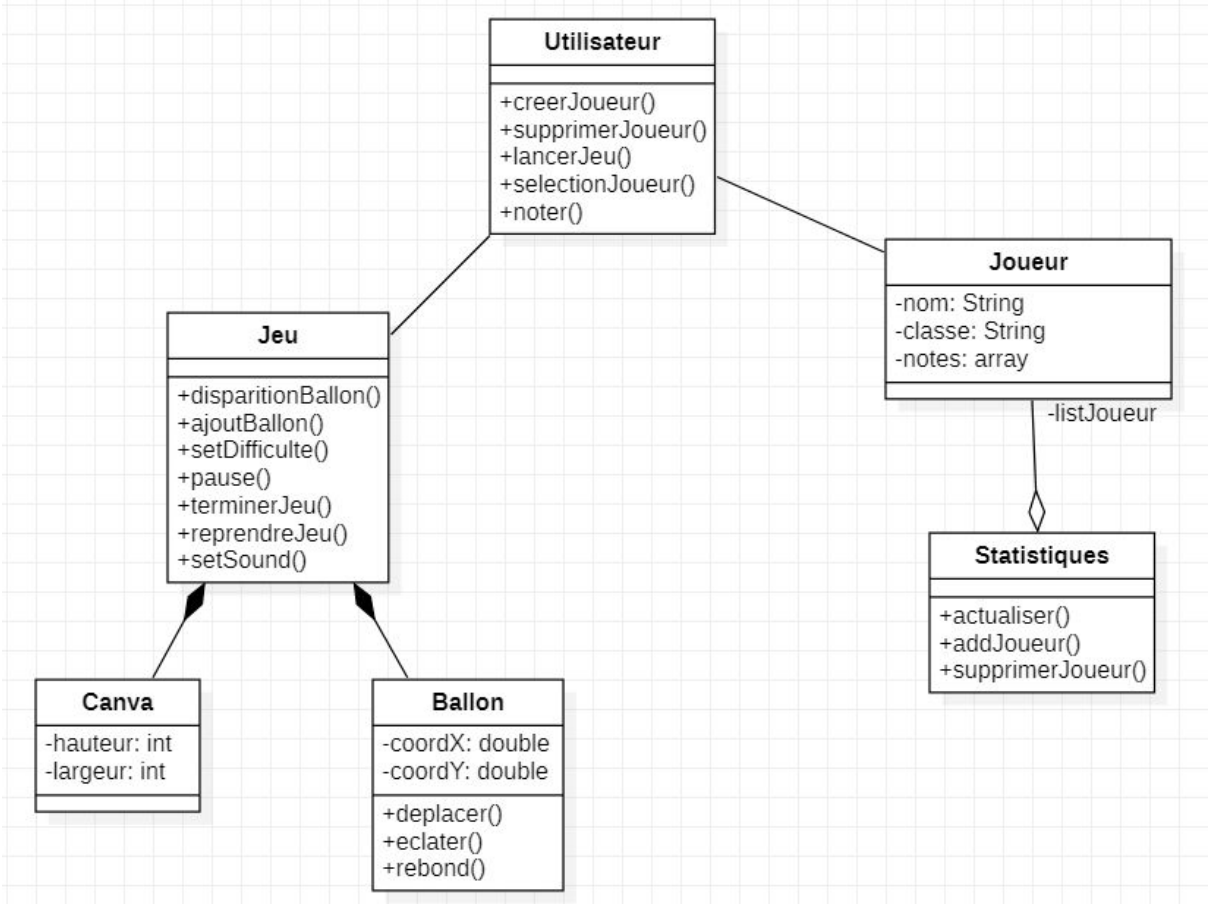


Figure 7 : Diagramme de classe de l'application lors de la conception

La classe "Ballon" de la figure 7 représente l'objet ballon, qui apparaît durant une partie de jeu et peut être éclaté lorsqu'on clique dessus, ce ballon possède une position, représentée pas sa coordX et sa coordY. Il peut alors se déplacer grâce à deplacer(). Ainsi, l'objet "Jeu" contient plusieurs ballons, mobiles. La fonction disparitionBallon() est liée au ballon pointé, la fonction ajoutBallon() permet quant à elle d'ajouter des ballons dans le jeu si celui-ci n'en contient plus assez. On retrouve également plusieurs fonctions liées aux fonctionnalités externes au jeu en lui-même, comme pause() que met la partie en pause, ou encore setSound(), permettant d'activer ou désactiver le son. L'objet "Jeu" contient aussi un "Canva", cela correspond à la taille que le jeu prend sur l'écran.
La classe "Utilisateur" regroupe les fonctionnalités nécessaires à l'utilisateur, c'est-à-dire les besoins fonctionnels de l'application.
Enfin, on retrouve l'objet "Joueur" qui désigne un élève et la classe "Statistiques" correspondant à la liste de tous les joueurs créés.

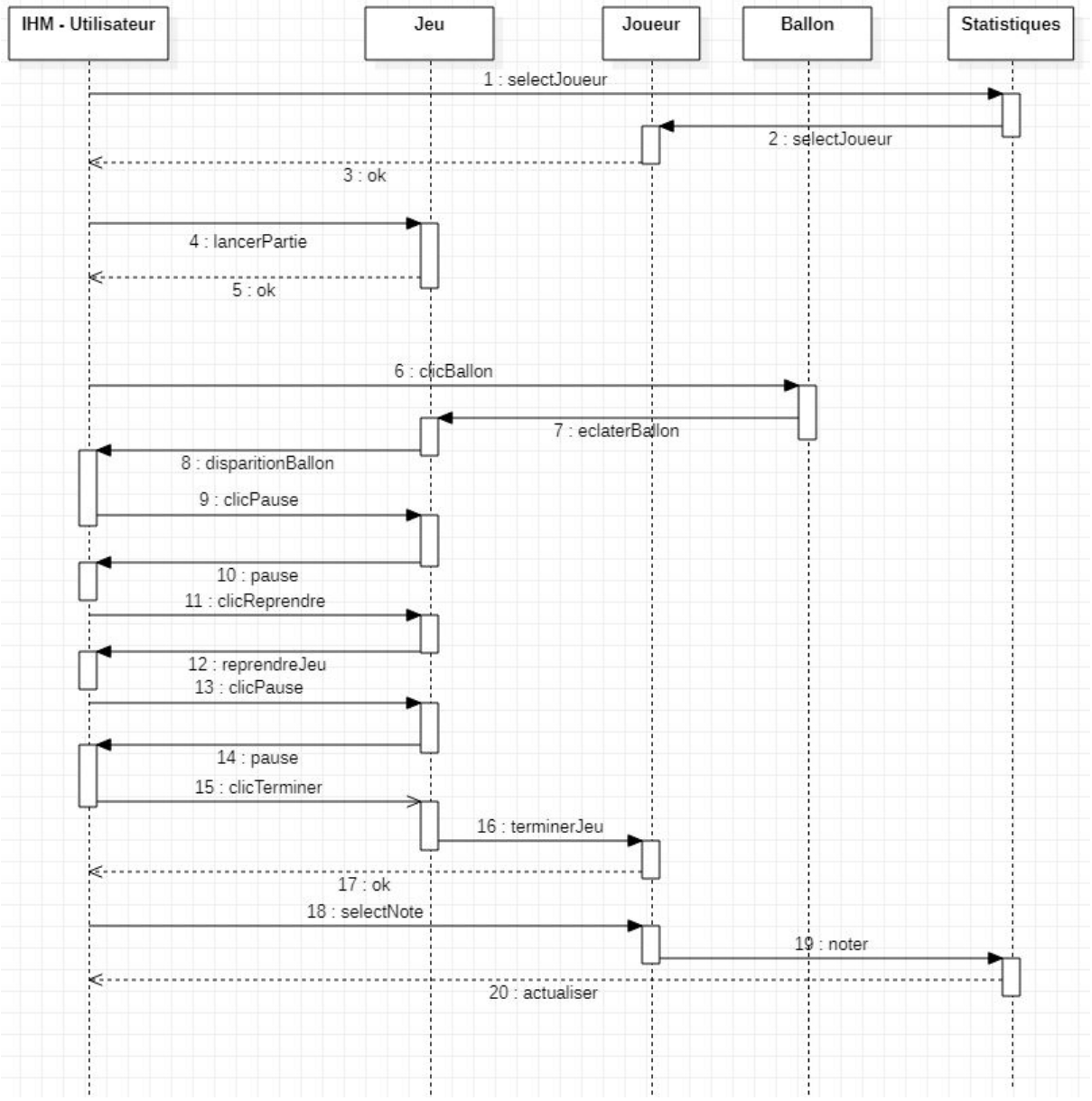


Figure 8 : Diagramme de séquence du déroulement d'une partie côté utilisateur

La figure 8 détaille les appels de fonctions et les connexions entre les objets, depuis le lancement à la fin d'une partie par l'utilisateur. Sur cette figure ne sont représentées que les fonctions activées du côté des actions de l'utilisateur, et non pas celles nécessaires aux fonctionnement du jeu.
Pour lancer une partie, l'utilisateur sélectionne un joueur dans les statistiques (1), ce joueur est enregistré par Statistiques (2) et l'utilisateur peut lancer la partie. Une fois la partie lancée, l'utilisateur crève un ballon via un clic (6), ceci active eclaterBallon (7) et la ballon disparaît de l'écran de jeu grâce à disparitionBallon (8).
Si l'utilisateur met en pause le jeu (10), tous les ballons sont stoppés et le menu pause s'ouvre, l'utilisateur peut alors reprendre le jeu en cliquant sur un bouton

reprendre (11), qui active la méthode reprendreJeu dans Jeu. Dans le menu pause, l'utilisateur peut aussi choisir de terminer la partie (15) en cliquant sur le bouton correspondant, "Jeu" va alors mettre fin à la partie (16). L'utilisateur doit ensuite choisir une note pour le joueur sélectionné précédemment (18) et cette note est enregistrée dans la liste de notes du joueur (19). Enfin, la base de données des joueurs est mise à jour avec cette nouvelle note (20).
Cependant, le cas où il n'existait aucun joueur dans les données de l'application, n'est ici pas présenté, l'utilisateur doit aussi pouvoir créer de nouveaux joueurs.

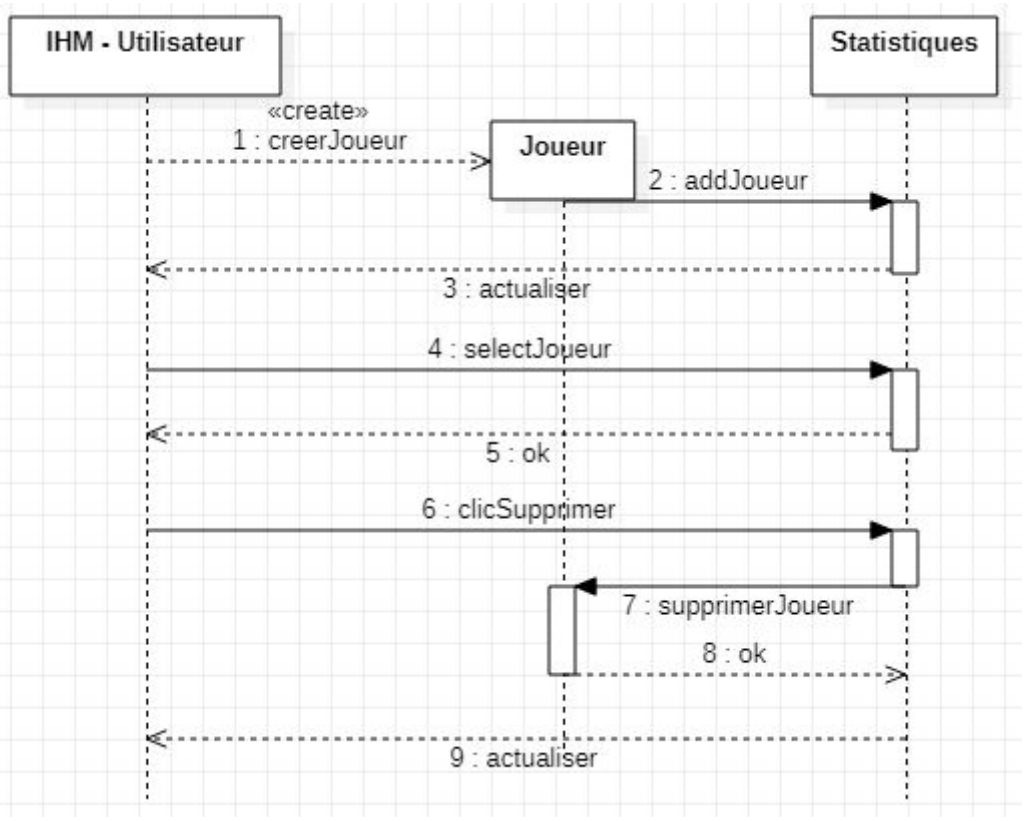


Figure 9 : Diagramme de séquence de la création et suppression d'un nouveau joueur sur l'application

La figure 9 met en évidence le déroulement de la création d'un nouveau joueur sur l'application, pour qu'il soit ensuite enregistrée dans les données de celle-ci.
Une fois le formulaire de création rempli, l'utilisateur clique sur un bouton activant la fonction creerJoueur (1) qui permet la création d'un nouvel objet de type Joueur avec les données entrées dans le formulaire. Ce joueur est alors ajouté à la liste des joueurs dans les statistiques (2), le dossier contenant les données des joueurs est par la suite mis à jour avec ce nouveau joueur (3).
D'un autre côté, on peut observer le déroulement de la suppression d'un joueur sur la figure 9. Le joueur visé est sélectionné par un clic de l'utilisateur sur son nom dans les statistiques (4). Un bouton "supprimer" apparaît, l'utilisateur appuie sur celui-ci

(6) et le joueur est retiré de la liste des joueurs de Statistiques (7). Le fichier de stockage des données de l'application est ensuite mis à jour (9), en y retirant le joueur précédemment supprimé.

Les figures 8 et 9 représentent la façon dont l'application répond aux besoins fonctionnels de l'utilisateur. Ces diagrammes ainsi que toute la conception réalisée sont la base sur laquelle est construite notre application, mais ces bases ont parfois été revues ou mises en place de manières quelque peu différentes.

2.2. Réalisation

Plusieurs fonctionnalités techniques sont nécessaires au bon fonctionnement de l'application, vous trouverez ici les principales modalités utilisés par l'application et une description détaillée de celles-ci.

2.2.1. Réalisation du menu principal

Au démarrage de l'application, le menu principal s'affiche. Il correspond à la page d'accueil du jeu. La phrase « Bienvenue sur Crev'Ballon » et la musique de fond attirent l'attention des enfants et a pour objectif d'attirer l'attention sur l'application.
Plusieurs actions sont réalisables à partir de ce menu :

- Lancer le jeu en cliquant sur le bouton « Jouer », qui fait un appel au fichier « selection.html ».
- Consulter les résultats des jours en cliquant sur le bouton « Statistique », qui fait un appel au fichier « stas.html »
- Consulter les crédits et les sources du jeu en cliquant sur « Crédits »
- Quitter le jeu via le bouton quitter
- Changer certains paramètres avec le bouton « Options »

Pour réaliser les fonctionnalités au-dessus. Un menu « menu.html » et son fichier de style css correspondant sont utilisés. Le fichier « menu.html » contient la structure de cette page et le répertoire « css » comporte le CSS nécessaires.
Dans « menu.html », les nom des éléments sont déclarés. Ils vont être utilisés lors d'appels dans cette page.

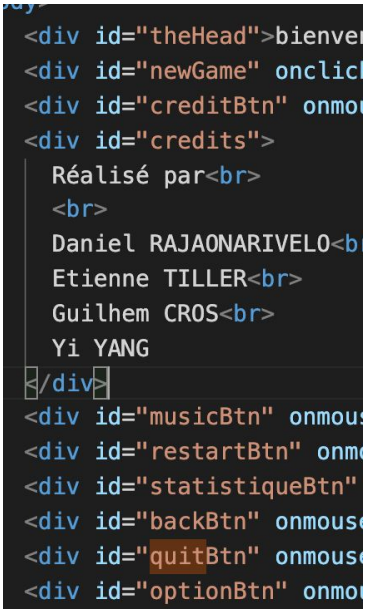


Figure 10: Les id pour les éléments de la page d'accueil

Lorsqu'un utilisateur touche un bouton, la couleur du bouton change en conséquence pour permettre à l'utilisateur de confirmer qu'il a touché le bouton. Par exemple, pour le bouton « Jouer », la couleur change au « goldenrod » quand l'utilisateur touche le bouton.

```
<div id="newGame" onclick="runGame()" onmouseover="this.style.backgroundColor = 'goldenrod'
```

Figure 11: Changement de la couleur

Afin de s'assurer que les utilisateurs voient les bons boutons sur la bonne interface, le « display » et l'id des éléments (voir dans la figure 10) sont adoptés. Pour mieux comprendre, on prend l'exemple du bouton « Statistiques ».

```
onclick="showStatistique()"
```

Figure 12: appel de la fonction showStatistique()

```
var showStatistique = function(){
 document.getElementById("theHead").style.display = "none";
  document.getElementById("creditBtn").style.display = "none";
  document.getElementById("statistiqueBtn").style.display = "none";
  document.getElementById("newGame").style.display = "none";
  document.getElementById("optionBtn").style.display = "none";
  document.getElementById("backBtn").style.display = "block";
  document.getElementById("quitBtn").style.display = "none";
  document.getElementById("musicBtn").style.display = "none";

}
```

Figure 13: La fonction showStatistique()

Sur la figure 12, il existe un onclick="showStatistique()", la fonctionnalité de ce code correspond à un appel de la fonction showStatistique() (voir dans la figure 13). La fonction showStatistique() adapte la visibilité de tous les boutons (par exemple le bouton « Quitter » -> quitBtn). Quand le .display = “none” ,le bouton ne s’affiche pas dans la page et il ne prend pas d’espace de la page. Au contraire, quand le .display = “block”, le bouton s'affiche normalement dans la page.

Afin de fournir aux utilisateurs une expérience agréable, la couleur bleu est adoptée pour le fond et la couleur orange et jaune sont utilisées pour les boutons.
Pour se faire, on définit ces paramètres dans le fichier css lié à la page.
Cette partie du code (figure 14) permet de générer l’esthétique des boutons, le texte qu’ils contiennent est ensuite lui aussi modifié par en CSS. La couleur #FF8A2F (orange) et le font-family « Courier New » sont utilisés.

```
text-align: center;
vertical-align: middle;
border: 2px solid goldenrod;
border-radius: 7px;
background-color: gold;
color: orangeRed;
font-size: 40px;
font-weight: bold;
font-family: "Courier New";
height: 1.2em;
width: 12em;
margin: 50px auto;
```

Figure 14: CSS pour les boutons

L'ensemble de l'application utilise de telles fonctionnalités HTML et CSS pour son affichage. Ces fonctionnalités sont complétées par les algorithmes JavaScript, permettant l'utilisation d'événements et le fonctionnement de l'application.

2.2.2. Réalisation du jeu

Lorsque le jeu se lance, c'est-à-dire après avoir cliqué sur le bouton jouer dans la page de sélection de joueur, le fichier « game.html » est appelé. Deux balises seulement sont contenues dans le corps du fichier « game.html » : un canvas et un fichier JavaScript nommé « renderer.js ». L'essentiel du jeu est conçu dans le fichier « renderer.js ».
Dans ce fichier « renderer.js » le canvas du fichier html est récupéré dans une variable/objet pour lui attribuer une dimension de la taille de la fenêtre pour qu'il prennent toute la surface de l'écran. Le jeu a ainsi un plan et un repère comme on le voit sur la figure 15.

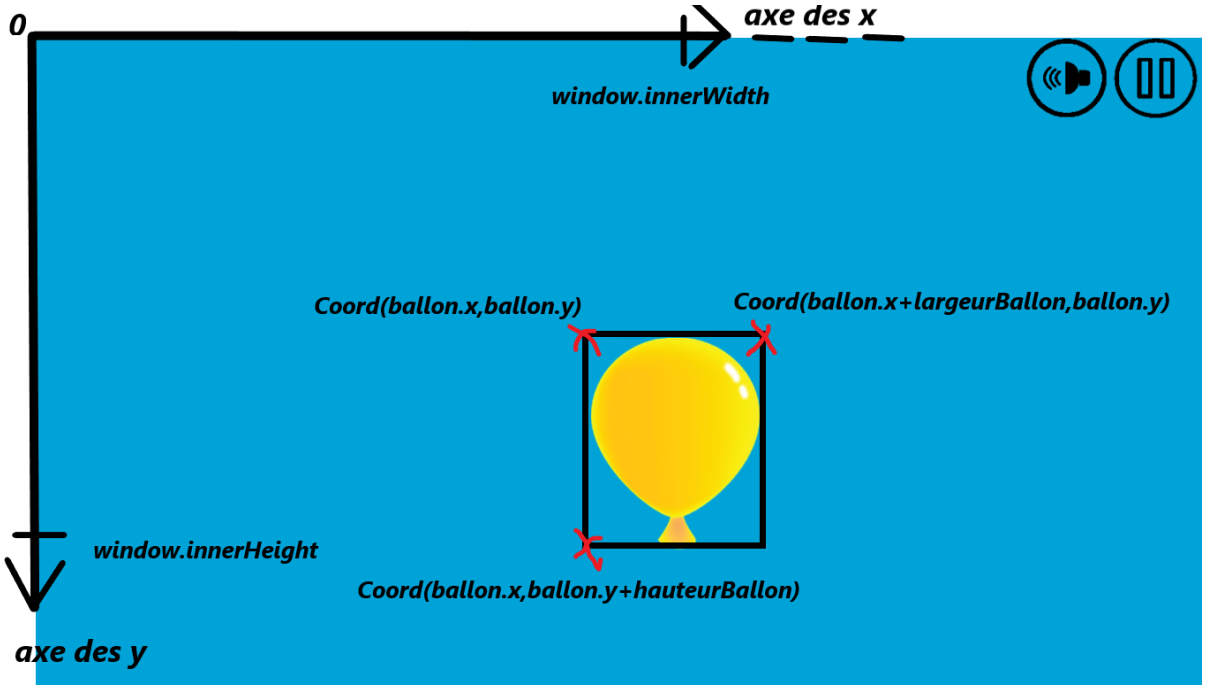


Figure 15 : coordonnées d'un ballon dans un plan

Un contexte de dessin « 2D »nommé « ctx » est ensuite associé pour pouvoir dessiner dans le canvas en utilisant l'interface CanvasRenderingContext2D. Grâce à cette interface, des ballons peuvent être dessinés dans le canvas. Un ballon est une classe ayant comme paramètre une image, des coordonnées et une direction.

```
class ballon{
    constructor(){
        this.image = new Image();
        this.image.src = listeImage[getRandomInt(6)];
        this.x = getRandomInt(canvas.width-largeurBallon);
        this.y = canvas.height*(1+Math.random()/2);
        do {
            this.pasy = Math.random();
            this.pasx = Math.random();
          } while (this.pasy<0.75||this.pasx<0.4);
        this.pasx = this.pasx * sens[getRandomInt(2)];
        this.pasy = this.pasy * sens[1];
        this.click = false;
        this.sortie = false;
    }
    update(){
        this.avance();
        this.setDirection();
        this.setSortie();
        this.eclaterBallon();
    }
```

Figure 16: Constructeur de la classe ballon

À chaque fois qu'un objet de type ballon est instancié (sans paramètre), le constructeur de ballon va initialiser aléatoirement les variables de l'objet nouvellement créé.

- Une image lui est attribuée avec l'instruction suivante : « listeImage[getRandomInt(6)] ». La variable listeImage est un tableau contenant les liens URL des 6 ballons présents dans le dossier média. Ainsi, avec la fonction « getRandomInt(6) », qui renvoie un chiffre aléatoire entre 0 et 6, l'attribut image contient un lien d'une image de ballon aléatoire.
- L'abscisse du point est aussi générée aléatoirement entre 0 et la largeur de la fenêtre en prenant en compte la largeur du ballon.
- L'ordonnée du point est générée au-delà du plan (de 1 à 1,5 fois la hauteur de la fenêtre). Les coordonnées du ballon sont donc en dehors du plan lors de l'instanciation, elle n'est donc pas immédiatement visible lorsqu'il est dessiné.
-la direction du ballon est un attribut du ballon (pasx et pasy). La direction verticale est toujours négative en multipliant pasy par sens[1] (qui contient « -1 »), pour que la ballon se déplace de bas en haut obligatoirement. Les ballons n'ont pas la même vitesse (aléatoire) mais ils ont une vitesse minimum. (car 0,4<pasX<0,99 et 0,75<pasY<0,99).
-les attributs click et sortie, permettent de connaître l'état du ballon.

Tout ceci permet à l'application de dessiner un ballon dans le canva avec la méthode « drawImage » de l'interface "CanvasRenderingContext2D" : «ctx.drawImage(element['image'],element['x'],element['y'],largeurBallon,hauteurBallo

n);». La méthode ci-dessus colle l'image du ballon dans le canvas à partir des coordonnées x, y et avec pour dimension, les variables globales largeur ballon et longueur ballon. Ce sont des variables globales car leurs tailles peuvent varier en fonction des difficultés du jeu.

La fonction update() s'occupe de calculer la prochaine coordonné du ballon. Après avoir calculé les prochaines coordonnés, la fonction vérifie s'il faut changer la direction du ballon, si le ballon est sorti ou s'il doit s'éclater. Il contient les 4 fonctions suivantes :

- avance() : modifie l'attribut « x » et « y » en incrémentant x et y de pasX et pasY respectivement.
- setDirection() : la direction du ballon peut changer en changeant le signe de pasX et pasY. Ils changent dans deux cas : quand le ballon touche d'autres ballons et quand le ballon touche l'extrémité latérale de l'écran. Cela va permettre de faire rebondir les ballons contre le mur ou contre d'autres ballon.
- setSortie() : la fonction va permettre de vérifier si le ballon est sorti du canva, c'est-à-dire si l'ordonnée du ballon est inférieur à 0. Si le ballon est sorti, alors la fonction disparitionBallon() est exécutée.
- eclaterBallon() : vérifie si le ballon a été touché. Pour savoir qu'un ballon est touché, les coordonnées du toucher courant sont comparées avec les coordonnées du ballon. S'il a était cliquer alors la fonction disparitionBallon() est exécutée.

La gestion des touchers d'écran utilise les événements tactiles (touch events).

```
let mx=0;//coordonne x du clic
let my=0;//coordonne y du clic

let sens =['1','-1'];

function finClick(){
    mx = 0;
    my = 0;
    isClicking = false;
  }

  //--------touch et mouse event-----------

document.addEventListener('touchstart', e => {
    isClicking = true;
    mx = e.touches[0].clientX;
    my = e.touches[0].clientY;
  });

document.addEventListener('touchend', e => {
    if (isClicking == true) {
        setTimeout(finClick,10);
    }
  });
```

Figure 17 : événements tactiles du fichier renderer.js

Quand un utilisateur touche l'écran, l'événement touchstart s'enclenche. Il met dans les variables globales « mx » et « my » respectivement les fonctions e.touches[0].clientX et e.touches[0].clientY, qui renvoient les coordonnées du clic. À la fin du toucher (touchend), les variables mx et my sont réinitialisées à 0 pour que les coordonnées mx et my stockent uniquement les coordonnées du clic au moment du toucher. Ceci évite que ballons qui passent dans ces coordonnées après le clic s'éclatent.

La fonction qui dessine en permanence se nomme « render ». Elle s'exécute en permanence, et fait appel à elle même. C'est cette fonction qui affiche les ballons sur le canvas. Pour dessiner tous les ballons en même temps, un tableau de ballon est créé au préalable. Ainsi, avec une boucle « foreach », les ballons sont dessinés un à un et leurs coordonnées suivantes sont calculées avec la fonction « update() ».

```
const render = () => {
    jouerMusiqueJeu();                                          1
    setBackground();
        if (speed<speedMax){
            speed =speed*1.0001;
        }
        if(nbBallon<nbBallonMax){                               2
            ajoutBallon();
            nbBallon = nbBallon+1;
        }
        tableauBallon.forEach(element => {                                                          3
            ctx.drawImage(element['image'],element['x'],element['y'],largeurBallon,hauteurBallon);
            if (element['click']==true||element['sortie']==true){
                var pos = tableauBallon.indexOf(element);
                disparitionBallon(pos);
            }
            jeuEnCours = enCours(jeuEnCours);

            if (jeuEnCours){                                    4
                element.update();
                setSound();
                ctx.drawImage(boutonPause,canvas.width*0.92,0,canvas.width*0.08,canvas.width*0.07);
            }else{
                actionMenue();
            }

        });

    window.requestAnimationFrame(render);
}
window.onload = render;
```

Figure 18 : fonction "render()" du fichier renderer.js

Au niveau de la figure 18 :
1) Un fond coloré est dessiné à chaque fois que la fonction est appelée pour effacer les anciens dessins de ballons. Une piste audio est aussi jouée pour accompagner le jeu.
2) Cette condition permet de toujours avoir un nombre de ballons constant (nbBallonMax) dans le plan. NbBallon est une variable initialisé à 0 représentant le

nombre de ballons actuel dans le plan. NbBallonMax est une variable globale représentant le nombre de ballons maximum pouvant être dans le plan. La fonction ajoutBallon va permettre d'ajouter un ballon dans le tableau et ainsi être dessiné dans le plan.

3) Cette étape permet pour chaque ballon (élément) du tableau de ballons d'être dessiné (afficher) dans le canvas. Le ballon disparaît si le ballon a été cliqué ou est sorti, avec la fonction « disparitionBallon » qui supprime du tableau le ballon qui a pour indice l'entier passé en paramètre. Comme le ballon n'est plus dans le tableau, le tableau n'est plus dessiné et est effacé lors du prochain appel de la fonction « render » et donc de « setBackground » qui remplit le canva d'un arrière-plan par-dessus les ballons précédents.

4) le booleen jeuEnCours permet de savoir si le jeu est en pause ou non. Si le bouton représentant le jeu pause est cliqué alors le boolean jeuEnCours retournera "false". La fonction actionMenue() affiche ensuite le menu pause.

Une fois ce menu pause ouvert, on y retrouve 4 boutons distincts. Le bouton reprendre permet de reprendre le jeu. En appuyant sur difficulté, la difficulté change. Il y a trois difficultés : Facile, difficile et très difficile. Plus le niveau est difficile, plus la vitesse des ballons est rapide, et la dimension des ballons diminue. Le changement de niveau ne coupe pas la partie, et le redimensionnement des ballons est visible depuis le menu pause. L'utilisateur est redirigé vers le menu principal ou sur la page de notation lorsqu'il clique sur les boutons « retour au menu » et « noter ».

2.2.3. Communication entre processus

La librairie d'Electron donne accès à des IPC (Inter Process Communication) qui ont pour objectifs de faciliter la mise en relation des processus de l'application.

```
const electron = require("electron");
const ipcRenderer = electron.ipcRenderer;
```

Figure 19 : Importation d'Electron et création d'un ipcRenderer

```
const {app, BrowserWindow, ipcMain} = require('electron')
```

Figure 20 : Création d'un ipcMain

Il existe deux sortes d'IPC :

- L'ipcMain : Il est disponible dans le processus main de l'application, c'est lui qui fait la liaison entre les processus des fenêtres ouvertes et le processus main par le biais des ipcRenderer.
- L'ipcRenderer : Cet IPC est disponible dans les processus des fenêtres Electron, il envoie des messages à l'ipcMain et reçoit également les messages provenant de ce dernier.

En effet, quand une application Electron s'exécute, le processus main démarre et peut exécuter différentes fenêtres qui possèdent chacune un ipcRenderer. Il est impossible de communiquer directement entre les différentes fenêtres sans passer par le processus main.

Pour communiquer avec l'ipcMain, les ipcRenderer ont une méthode "send", elle leur permet d'envoyer un message à l'ipcMain.

```
ipcRenderer.send("ask-id");
```

Figure 21 : Utilisation de la méthode "send"

Pour assurer la communication entre les deux ipc, il est important d'ajouter auparavant un écouteur d'événement dans le processus main.

```
ipcMain.on("ask-id",function(event){
  event.sender.send("id",idEleve);
})
```

Figure 22 : Création d'un écouteur d'évènement

Dès qu'il reçoit le message en question d'un ipcRenderer, alors il exécute une fonction, on peut y retrouver plusieurs paramètres.

Le premier paramètre "event", représente l'événement il est souvent utilisé comme dans la figure 22 afin de retrouver la source de l'événement. Les autres paramètres s'il y en a, sont ceux qui peuvent être transmis par l'ipcRenderer dans la méthode "send".

```
boutonJouer.addEventListener("click",function(event){
    ipcRenderer.send("idEleve",idEleve);
    ipcRenderer.send("jouer");
})
```

Figure 23 : Envoi d'une donnée par la méthode "send"

Ainsi, les ipc peuvent communiquer entre eux et donc faire passer des instructions ou des données de processus en processus.

Dans notre application, le menu et le jeu en lui-même ne sont pas sur le même processus, cependant la sélection du joueur se fait dans le menu et la notation se fait donc sur un autre processus. L'application doit donc transmettre le joueur sélectionné afin que ce soit lui qui soit noté à la fin dans la base de données. Pour cela, le programme utilise les ipc de telle sorte que lors de la sélection du joueur, son id est transmis au processus main.

Le processus main reçoit donc l'idEleve qu'il stocke dans une variable grâce à son écouteur d'événement.

```
ipcMain.on("idEleve",function(event,args){
  idEleve = args[0];
})
```

Figure 24 : Réception et stockage dans le processus main de "l'idEleve"

Ensuite, l'application instaure un nouvel écouteur d'événement dans le processus main "ask-id", comme sur la figure 22.

Quand un processus a besoin de l'id d'un joueur, il appelle le processus main avec le message "ask-id" (voir figure 21).

Comme l'ipcMain renvoie un message au processus demandeur de l'id, alors il faut que ce dernier ait aussi un écouteur d'événement afin de recevoir l'id.

```
ipcRenderer.on("id",function(event,args){
    idEleve = args[0];
})
```

Figure 25 : Réception et stockage de l'idEleve par l'ipcRenderer

Ainsi, l'id du joueur a été transmise d'un processus à un autre et dans ce cas-ci, le joueur sélectionné va pouvoir être noté à partir d'un autre processus.

L'application utilise ces ipc pour la notation, comme vu précédemment, mais aussi pour quitter l'application ou encore gérer le son de celle-ci.

2.2.4. Réalisation de la base de données

Comme expliqué précédemment, les données de l'application sont stockées dans un fichier JSON. Les fichiers JSON sont des fichiers qui comprennent les objets JavaScript. Il est donc facile d'écrire des objets JavaScript dans un fichier JSON et de les lire par la suite.

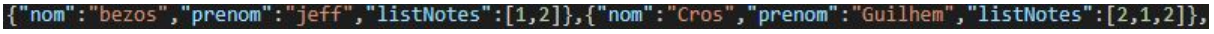


Figure 26 : Exemple de fichier JSON contenant la liste des joueurs

La figure met en évidence que les fichiers JSON sont indexés comme en JavaScript, facile à la lecture et donc facile à stocker en interne pour une base de données non complexe.

Pour écrire ou lire dans un fichier en JavaScript, il est nécessaire d'importer la fonctionnalité "File System". Ensuite, pour lire dans un fichier à partir de "File System", le programme utilise "readFileSync".

```
var data = fs.readFileSync('stockage/listeEtudiant.json');
```

Figure 27 : Utilisation de la méthode "readFileSync"

Les données du fichier JSON sont stockées dans une variable, cependant, même si pour l'application il s'agit d'une liste de joueurs, lorsque cette liste est lue, elle n'est pas directement comprise comme telle. Pour y remédier, il est nécessaire d'utiliser la méthode "JSON.parse".

```
listEtudiant = JSON.parse(data);
```

Figure 28 : Utilisation de "JSON.parse"

A ce moment-là, pour la figure 28, la variable "listEtudiant", contient réellement une liste d'objet "Etudiant" qui provient du fichier JSON. Il est donc possible de modifier cette liste en y ajoutant un étudiant ou en le notant par exemple.

```
if (listEtudiant[idEleve].listNotes.length < 3){
    listEtudiant[idEleve].listNotes.push(note);
}
else {
    listEtudiant[idEleve].listNotes.shift();
    listEtudiant[idEleve].listNotes.push(note);
}
```

Figure 29 : Utilisation de l'objet de la BDD comme un objet en JavaScript

Enfin, pour sauvegarder les modifications faites sur cette liste dans la base de donnée, le programme fait appel à la méthode "writeFile", mais avant cela traduit l'objet en question en une chaîne de caractère, pour pouvoir le stocker correctement et sans problèmes dans le fichier JSON.

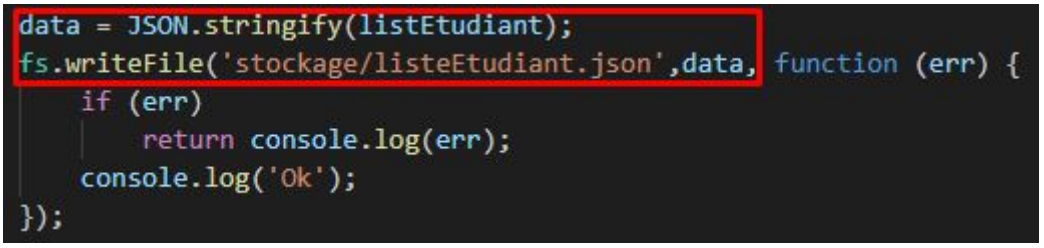


Figure 30 : Conversion de la liste en String puis stockage dans le fichier JSON à l'aide de la méthode "writeFile"

Ces opérations sont répétées pour toutes les fonctionnalités liées à la base de données locale, par exemple pour la création, la suppression ou encore l'ajout d'une note à un joueur. Elles sont nécessaires au bon fonctionnement de l'application et à la gestion des données de l'utilisateur.

2.2.5. Gestion des ressources

Beaucoup de fonctionnalités de l'application sont similaires pour plusieurs pages. Par exemple, une musique de fond sur le menu. La musique sur l'application continue normalement sur tout le menu, pourtant ce dernier est fait de plusieurs pages. Pour cela, une certaine gestion des fichiers est instaurée. En effet, il existe un fichier JavaScript (app.js) et CSS (app.css) commun pour la majorité des pages de l'application. Afin que les fichiers JavaScript et CSS restent chargés même après un changement de page, la méthode "loadPage" est appelée.

Cette méthode prend trois paramètres, "page" correspond au nom de la page qui doit être affichée, "jsLoad" et "cssLoad" permettent au programme de savoir si la page en question contient un fichier JavaScript et un fichier CSS associé.

```
function loadPage(page, jsLoad = false, cssLoad = false) {
    var jsFile = (typeof jsLoad === 'string') ? jsLoad : page;
    var cssFile = (typeof cssLoad === 'string') ? cssLoad : page;

    fetch('./src/html/' + page + '.html')
    .then(data => data.text())
    .then(html => app.innerHTML = html)
    .then(function() {
        if (jsLoad){
            var pageScript = document.createElement("script");
            pageScript.setAttribute("type","text/javascript");
            pageScript.setAttribute("src", './src/js/' + jsFile + '.js');
            app.appendChild(pageScript);
        }
        if (cssLoad){
            var pageStyle = document.createElement("link");
            pageStyle.setAttribute("rel", "stylesheet");
            pageStyle.setAttribute("type", "text/css");
            pageStyle.setAttribute("href", './src/css/' + cssFile + '.css');
            app.appendChild(pageStyle);
        }
    });
}
```

Figure 31 : Définition de la méthode loadPage

Les fichiers JavaScript et CSS associés à une page possèdent le même nom à l'exception de l'extension évidemment. Cependant il est possible qu'un fichier CSS ou JavaScript soit utilisé par plusieurs pages sans pourtant qu'il soit indispensable dans toutes les pages non plus. C'est pour cela que les variables contenant les noms des fichier JavaScript et CSS dans la figure 31 peuvent avoir le nom de la page, ou, si un nom a été indiqué, alors il sera pris en compte.

```
loadPage('home', true, 'index');
```

Figure 32 : Exemple d'utilisation de la méthode "loadPage"

Comme présenté dans la figure 32, la page index est chargé, elle possède un fichier JavaScript qui se nomme "home.js" et un fichier CSS qui se nomme "index.css".

```html
<!DOCTYPE html>
<html>
  <head id='head'>
    <meta charset="UTF-8">
    <title>Jeu_enfant</title>
    <link rel="stylesheet" href="src/css/app.css" />
  </head>
  <body>
    <div id="app">
    </div>
    <script type="text/javascript" src="./src/js/jquery-3.5.1.min.js"></script>
    <script type="text/javascript" src="./src/js/app.js"></script>
  </body>
</html>
```

Figure 33 : Index.html

Lorsque l'application démarre, c'est la page représentée dans la figure 33 qui est chargée avec le CSS et le JavaScript commun à toutes les pages. Lors de l'appel à la méthode "loadPage", le contenu de la div qui contient l'id "app" va être modifié avec la méthode innerText.

La méthode fetch est une méthode Javascript qui permet de "copier" le contenu d'un fichier, ici le nom de la page que l'on souhaite est saisi, on le stocke dans la variable data. Cette dernière est ensuite traduite en texte et placée dans la variable html. Enfin, la variable "app" qui correspond à la div dans index.html, utilise sa méthode "innerHTML" afin de modifier son contenu en y insérant le contenu du fetch. Ainsi on reste sur la même page mais on change le contenu de celle-ci.

Une fois la page chargée dans l'index.html, si elle possède un fichier JavaScript ou css associé, alors il est inséré lui aussi dans la div "app" de l'application .Pour le menu, un seul fichier html est chargé avec son JavaScript et son CSS commun pour toutes les pages.De cette manière, la musique ne varie pas de page en page et cela permet aussi de ne pas recopier du code plusieurs fois.

4. Résultats

En plus de la programmation de l'application, une vérification de ses fonctionnalités et la réalisation des documents facilitant son utilisation sont nécessaires. Vous trouverez ici les détails des tests effectués avant la validation du projet et les documents visant à aider l'utilisateur dans l'installation et l'utilisation du logiciel.

4.1. Installation

Installation :
Pour installer l'application Crev'Ballon, téléchargez le logiciel d'installation disponible ici : https://mega.nz/file/LlVVya5A#FpEJAO5f1W0mL5AIcoKO0tYkxXZU7WwaN5cZMX62qNM
Une fois le téléchargement terminé, exécutez ce logiciel et acceptez l'apport de modification sur votre machine.
Une fenêtre va alors s'ouvrir. Acceptez les conditions d'utilisation en cliquant sur le bouton "accepter", puis continuez, lancez l'installation du logiciel en cliquant sur le bouton "extraire".
L'installation de l'application va alors se lancer automatiquement. Une fois celle-ci terminée, vous pourrez fermer la fenêtre en cliquant sur "terminer".
Un dossier "Crev'Ballon" devrait avoir été créé dans le dossier où se trouve le logiciel d'installation (probablement dans l'onglet "Téléchargements" de votre machine). Ne supprimez pas ce dossier, il est nécessaire au fonctionnement de l'application.
Nous vous conseillons de le déplacer dans un répertoire plus sûr, comme par exemple le dossier "Documents", afin d'éviter une suppression par mégarde.
Un raccourci vers l'application a été créé sur votre bureau. L'application est prête à être utilisée.

4.2. Manuel d'utilisation

Pour exécuter l'application, cliquer sur l'icône correspondant, ayant été créé sur votre bureau. L'application va alors s'ouvrir sur le menu principal.
Pour quitter et fermer l'application, appuyez sur "Quitter".

1/ Lancer une partie

Afin de lancer une partie, depuis le menu principal, cliquez sur le bouton "Jouer". Vous allez être redirigé vers la page de sélection du joueur. Si aucun joueur n'est

déjà enregistré ou si vous voulez en créer un nouveau, cliquez sur “Nouveau joueur” et remplissez les champs correspondant, puis enregistrez. Cliquez ensuite sur “Jouer”. Sinon, sélectionnez le joueur que vous voulez utiliser et cliquez sur “Jouer”.
Une partie est alors lancée, le joueur peut désormais crever les ballons !

2/ Terminer une partie et noter un joueur

Lorsque vous êtes dans une partie et que vous voulez stopper celle-ci, cliquez sur l'icône pause. La partie va alors être mise en pause et un menu avec plusieurs options va s’ouvrir. Pour terminer la partie, cliquez sur l’option “Noter”, vous serez alors redirigé vers l’onglet de notation. Deux notes peuvent être attribuées au joueur : “Super !” et “Encourageant”. Ces notes sont représentées par deux images distinctes. Cliquez sur l’image correspondant à la note que vous voulez donner au joueur pour finaliser la notation. La note sera enregistrée dans les statistiques et vous serez redirigé vers un onglet correspondant à la note, à l’attention du joueur.
Cliquez sur “Menu” pour revenir au menu principal lorsque vous souhaitez quitter cet onglet.

3/ Gestion du son de l’application

Les thèmes musicaux et les sons de l’application peuvent être désactivés si nécessaire. Deux manipulations distinctes sont possibles.
Depuis le menu principal, cliquez sur le bouton “Options”, la page d’options de l’application va alors s’ouvrir. Cliquez ensuite sur “Musique”, tous les sons de l’application sont alors désactivés.
Depuis une partie en cours, cliquez sur l'icône de son, en haut à droite de l'écran. Les sons sont alors désactivés.
Pour réactiver les sons, effectuez à nouveau l’une des deux manœuvres précédentes.

4/ Le menu pause

Lorsque une partie de jeu est lancée, celle-ci peut être mise en pause depuis l'icône correspondant, en haut à droite de l’écran. Vous accédez ainsi au menu pause du jeu. Plusieurs options sont disponibles depuis ce menu.

- L’option “Reprendre” permet de reprendre la partie là où elle a été stoppée.
- L’option “Difficulté” permet de modifier le niveau de difficulté du jeu. Trois niveaux de difficultés sont disponibles : facile, normal, difficile, facile étant la difficulté standard. Vous pouvez modifier cette difficulté sur l’option. Une fois la difficulté choisie, cliquez sur “Reprendre” pour reprendre la partie.
- L’option “Retour au menu” permet de stopper la partie, sans noter le joueur. Vous serez redirigé vers le menu principal de l’application.

- L'option "Noter" permet de noter le joueur(voir *2/ Terminer une partie et noter un joueur*)

5/ Les statistiques

Depuis le menu principal de l'utilisation, vous pouvez accéder aux statistiques de tous les joueurs via le bouton "Statistiques".
Ces statistiques correspondent aux dernières notes obtenues par le joueur.
Vous pouvez depuis ce menu, supprimer un joueur en cliquant sur le joueur que vous souhaitez supprimer, puis en appuyant sur "supprimer".
Vous pouvez également supprimer tous les joueurs enregistrés sur l'application en sélectionnant "Tout supprimer", validez ensuite la suppression en cliquant sur "Supprimer" ou annulez en cliquant sur "Annuler".

4.3. Validation

Tout au long de la réalisation du projet, des tests ont été réalisés, pour vérifier les fonctionnalités développées, sur PC et tablette.
Avant la validation finale de l'application, plusieurs gammes de tests plus globaux ont été effectuées.
En premier lieu, nous avons nous même testé toutes les fonctionnalités de l'application sur PC. Pour effectuer ces tests, l'application et sa console était lancées afin de s'assurer qu'aucunes erreurs n'étaient renvoyées. Chaque fonctionnalité a été testé une part une puis le jeu a été utilisé normalement dans sa globalité. Cela a permis de révéler quelques dysfonctionnement, mais aucun problème majeur n'a été révélé. Nous avons alors fait au mieux pour régler ces bugs mineurs.
Une fois ces tests validés, une vérification du bon fonctionnement du programme sur tablette a été effectuée, étant donné qu'il a été conçu pour l'utilisation sur ce type de machine.
En plus de nos tests personnels, nous avons partagé le lien de téléchargement de l'application avec nos connaissances, afin d'avoir plus de tests effectués mais aussi un avis extérieur sur notre production. Nous avons ainsi pu vérifier à nouveau les fonctionnalités pour lesquelles des problèmes avaient été repérés, et avons fait en sorte de les réparer.
Nous nous sommes également préoccupé des avis extérieurs quant à l'esthétique et l'ergonomie de l'application. Etant donné que nous avons créé l'application, certaines fonctionnalités nous paraissaient triviales mais ne l'étaient pas pour tous. Par exemple, la méthode de suppression d'un joueur ne semblait pas très compréhensible pour l'une des personnes qui a pu tester l'application. Nous avons essayé de comprendre pourquoi et sommes ensuite revenus sur l'ergonomie de cet onglet, pour qu'il soit plus clair pour les utilisateurs.

Une fois tous ces tests passés et les problèmes de fonctionnalités réglés, nous avons décidé de valider notre production.

5. Rapport d'activité

Dans l'objectif de réaliser ce projet dans les meilleures conditions et de la meilleure façon possible, nous avons mis en place un suivi régulier de notre progression. Ce suivi était d'autant plus important dû au fait que nous ne pouvions plus nous réunir en présentiel à l'IUT à cause de la pandémie actuelle. Un choix d'une méthode de travail particulière a été pris ainsi que le choix des outils nous permettant de rester en relation régulière.

5.1. Méthode de développement et outils

Au niveau du travail au sein de l'équipe elle-même, tout s'est globalement bien déroulé au cours du projet.
En premier lieu, pour ce qui est de la partie analyse, nous avons travaillé en groupe, de façon à ce que chacun puisse donner son avis. L'analyse a donc été simple et claire pour tous.
Ensuite, lors de la conception de l'application, chaque membre du groupe a pu faire ses propres recherches de son côté, et proposer des solutions, plus ou moins adaptées à la réalisation du programme. Nous nous sommes finalement mis d'accord tous ensemble sur les choix technologiques et la façon dont nous voulions réaliser l'application.
Enfin, tout au long du développement du code du projet, nous avons principalement travaillé en autonomie sur la fonctionnalité que nous avions chacun décidé de réaliser. Nous avons séparé ces fonctionnalités en différentes tâches à réaliser. Cela a permis à tout le monde de travailler sur le projet en réalisant ce que chacun avait envie de faire. Bien évidemment, si l'un membre de l'équipe avait des difficultés à réaliser sa tâche, les autres membres lui apportaient leur aide.
Afin de rester en communication nous avons utilisé le logiciel Discord, sur lequel nous communiquions nos avancées et nos problèmes comme le montre la figure 34.
Ce logiciel a aussi été utilisé pour des réunions à l'oral, afin de partager des idées, ou donner notre avis sur certaines tâches réalisées.

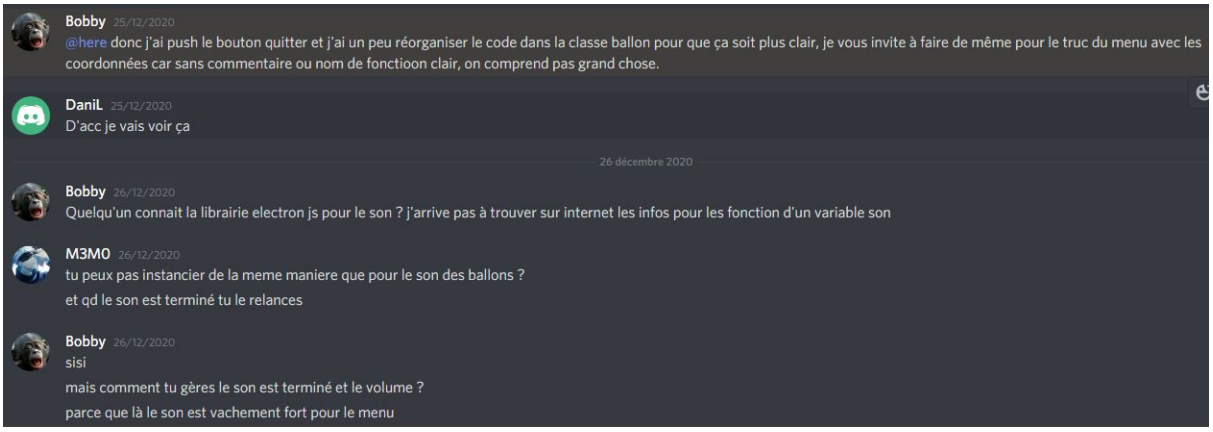


Figure 34 : Un de nos échanges pour le projet sur discord

Du côté du contact avec notre tuteur, M.Garcia, avant le confinement, des réunions étaient réalisées toutes les semaines. Nous communiquons aussi par mail avec l'objectif de partager notre avancement, nos idées et demander des conseils.
Durant le confinement, nous avons eu plus de mal à organiser des réunions régulièrement, mais sommes restés en contact régulier par email.Cependant, durant le mois de Décembre, nous avons très peu avancé sur le projet et donc avons beaucoup moins contacté M.Garcia. Ceci était dû à une mauvaise gestion du projet et de temps de notre part, nous avons suspendu le projet durant deux semaines et ainsi retardé notre progression.

5.2. Planification des tâches

Afin de réaliser notre projet dans les temps, nous avons choisi une gestion de projet en cascade. En effet, le projet sur lequel nous devions travailler était simple à comprendre et à visualiser, les attentes du client étaient plutôt claires. Nous avons alors pensé qu'une gestion en cascade était adaptée à ce projet, et nous paraissait , dans ce contexte, plus logique à réaliser. Ainsi nous avons réalisé un planning des tâches à effectuer en suivant la méthode de travail en cascade. Nous nous sommes mis d'accord sur le temps que nous pensions mettre pour chaque étape et avons essayé de suivre ce calendrier.

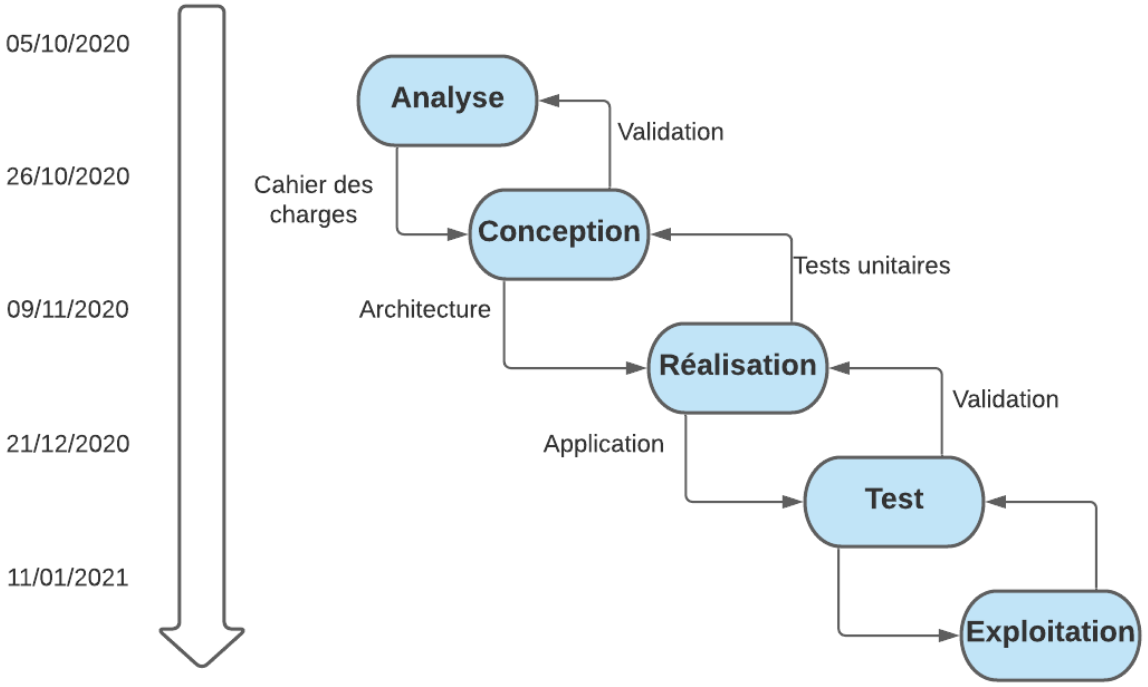


Figure 35 : Diagramme de prévision du projet selon le modèle en cascade

La figure 35 présente nos prévisions et le planning que nous avions prévu lors du lancement du projet. La période la plus longue correspondait à la réalisation du programme, c'est-à-dire le code de et la mise en place de l'application. Nous pensions globalement pouvoir terminer ce projet avant les vacances de Noël.

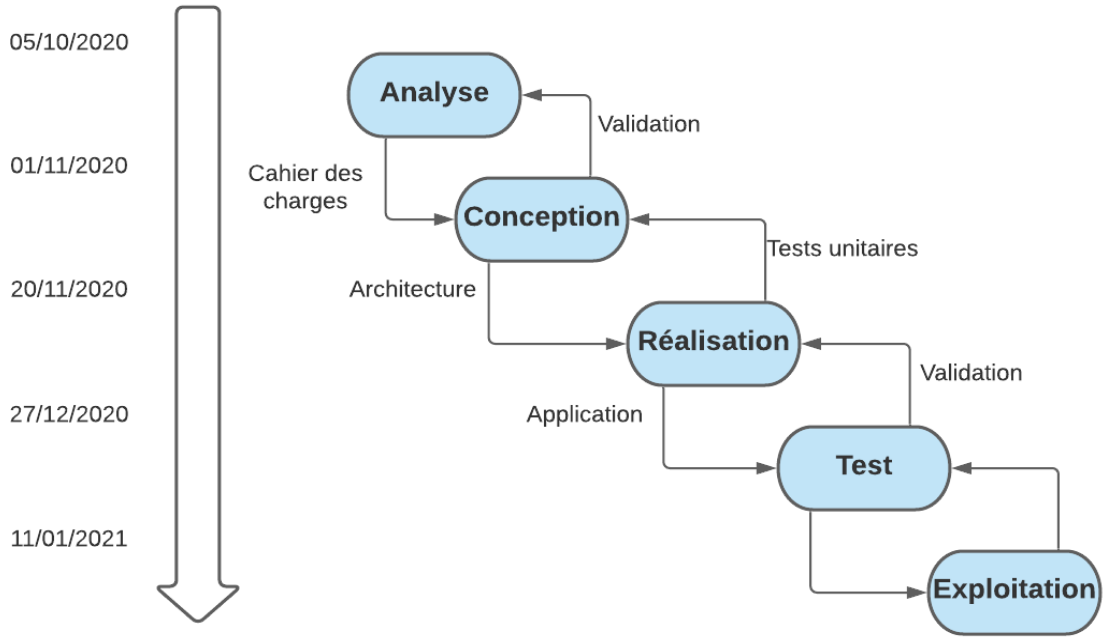


Figure 36 : Diagramme du déroulement réel du projet selon le modèle en cascade

La figure 36 correspond au déroulement réel du projet. En comparaison avec la figure 35 , nous avons pris du retard dès les premières semaines par rapport à nos prévisions.
Environ une semaine de retard a été prise, nous n'avons ainsi pas pu finir le projet avec les vacances de Noël comme nous l'aurions voulu. De plus, nous n'avons pas essayé de rattraper notre retard sur le planning, et aurions même pu augmenter ce retard, étant donné que nous avons mis le projet en suspens durant deux semaines en Décembre. Cette pause dans le projet était due à la forte demande de travail en plus du projet durant cette période. Nous n'avons pas géré notre temps comme nous aurions dû et avons préféré repousser les tâches que nous avions à faire . Finalement, cela ne nous a heureusement pas porté préjudice, et nous avons pu finir dans les temps. Cependant nous aurions pu gérer de meilleure façon notre projet en suivant sur le planning et en gérant notre temps de travail de meilleure façon.

5.3. Bilan critique par rapport au cahier des charges

Par rapport au cahier des charges, l'ensemble des besoins fonctionnels et non-fonctionnels est respecté. En plus de ce qui était demandé, nous avons rajouté une fonctionnalité permettant le changement de difficulté du jeu, dans l'objectif de ne pas lasser les enfants.
Nous avons également essayé de faire en sorte que le jeu soit le plus agréable possible esthétiquement et ergonomiquement parlant. De plus, nous avons essayé de rapprocher au mieux notre application des jeux pour enfant que l'on peut trouver en ligne, en créant une application colorée, possédant des musiques et sons.

Nous aurions aimé gérer de meilleure façon notre projet, en particulier au niveau du temps. Cela aurait pu nous permettre d'améliorer l'interface et l'univers proposé par l'application.
Un autre ajout que nous aurions apprécié mettre en place est la présence de marques sonores s'activant lorsque l'utilisateur clique sur certains boutons. Cela pourrait aider certains utilisateurs dans l'utilisation de l'application, mais aussi, ajouterait un peu de vie et contribuerait à créer l'univers de l'application.

En sommes, si nous pouvions le faire, nous voudrions que notre application soit aussi attrayante voire plus, que certaines applications pour enfants que l'on trouve sur le WEB, en revoyant l'esthétique, l'ergonomie et l'univers de celle-ci.

Conclusion

La mission qui nous a été donnée était la réalisation d'un jeu pour enfant, sur tablette, selon un modèle et les envies du clients qui étaient un jeu de ballons.
L'application devait être adaptée à une utilisation dans des écoles, par de jeunes enfants.
Après avoir construit le cahier des charges et établi les besoins fonctionnels et non-fonctionnels du programme, nous avons réalisé l'application de telle sorte qu'elle réponde à tous ces besoins.
Nous sommes fiers du projet que nous rendons et heureux d'avoir pu travailler dessus. Nous aurions cependant aimé, avec plus de temps, adapter le jeu à une utilisation plus vaste, comme sa mise en ligne sur des plateformes de jeux gratuits par exemple.
Ce projet pourrait également évoluer, en lui apportant d'autres modes de jeu que celui déjà proposé, ou d'autres niveaux de difficulté.
Tout au long de ce projet, nous avons été amenés à apprendre de nous même de nouvelles connaissances techniques. Le JavaScript et sa librairie Node Js ont été une partie importante de cet apprentissage, mais aussi le module permettant à notre application de fonctionner, Electron.
Nous avons aussi pu mettre à l'épreuve nos méthodes de travail. Ainsi nous avons remarqué que celles-ci pouvaient grandement être améliorées, mais ont été suffisantes pour le développement du programme.
Finalement, ce projet nous a permis d'avoir une représentation concrète du déroulement d'un projet dans une entreprise. Nous avons, grâce à celui-ci, acquis de nouvelles compétences techniques et développé nos méthodes de travail personnelles et en équipe.

Bibliographie

[1]
Pôle «Evaluation et Société», «Baromètre du numérique 2018», dec. 03, 2018. https://www.arcep.fr/uploads/tx_gspublication/barometre-du-numerique-2018_031218.pdf (consulté en nov. 2020).
[2]
Piku, «JavaScript», dec. 15, 2020.
https://fr.wikipedia.org/wiki/JavaScript (consulté en dec. 2020).
[3]
Tieno, «Programmation orientée prototype», mai. 18, 2008.
https://fr.wikipedia.org/wiki/Programmation_orient%C3%A9e_prototype (consulté en nov. 2020).
[4]
JackPotte, «Node.js», nov. 18, 2018.
https://fr.wikipedia.org/wiki/Node.js (consulté en dec. 2020).
[5]
Thomas.Nunes, «Programmation événementielle», nov. 14, 2013.
https://fr.wikipedia.org/wiki/Programmation_%C3%A9v%C3%A9nementielle
(consulté en dec. 2020).
[6]
Electron, «Documentation d'electron».
https://www.electronjs.org/docs (consulté tout au long du projet).
[7]
Codevolution, «Electron js Tutorial - 6 - IPC», aou. 24, 2017.
https://www.youtube.com/watch?v=rX3axskesDw&t=452s (consulté en dec. 2020).
[8]
The Coding Train, «8.5: Saving Data to JSON File with Node.js - Programming with Text», nov. 18, 2016.
https://www.youtube.com/watch?v=6iZiqQZBQJY (consulté en dec. 2020).
[9]
"Auteur externe", «Vos applications avec Electron», nov. 19, 2016.
https://zestedesavoir.com/tutoriels/996/vos-applications-avec-electron/ (consulté en dec. 2020).
[10]
Will Alexander, «Apprenez à programmer avec JavaScript», jan. 04, 2021 (dernière mise à jour).
https://openclassrooms.com/fr/courses/6175841-apprenez-a-programmer-avec-javascript (consulté en nov. et dec. 2020).

Résumé en français

Ce rapport concerne le déroulement et la réalisation d'un projet de troisième semestre effectué pendant l'année universitaire 2020 / 2021, à l'IUT Informatique de Montpellier. Plus précisément, il détaille la création du projet "Crev'Ballon", une application ludique pour enfants, de sa conception à son développement en HTML, CSS et JavaScript via l'utilisation de la librairie Node Js, mais aussi la gestion du projet en elle-même. Il contient des documents tels que le cahier des charges, le rapport technique et le manuel d'utilisation de l'application ainsi que des documents plus généraux, comme le rapport d'activité.

L'application a pour but d'être utilisée dans des écoles, sur tablette, pour sensibiliser de jeunes enfants aux technologies tactiles, et se présente sous la forme d'un jeu. Elle doit permettre aux enseignants d'évaluer les compétences des élèves et d'avoir accès à ces évaluations.

Mots clés : Node Js, langages WEB, JavaScript, jeu, tablette, application ludique, enfants, éducation.

Résumé en anglais

This report concerns the development and the implementation of the third semester project, carried out during the 2020/ 2021 academic year, at the Montpellier's Computer Science IUT.
More precisely, it details the creation of the project "Crev'Ballon", a fun application for children, from its design to its development in HTML, CSS and JavaScript, using the Node Js library. It also contains the description of the management of the project.
You will find documents such as the specifications, the technical description and the user manual of the application as well more general documents, such as the activity report.

The application is made to be used in schools, on tablets, to sensitize young children to tactile technologies, and is presented in the form of a game.
It must enable teachers to assess the skills of students and to have access to these assessments.

Key words : Node Js, WEB languages, JavaScript, game, tablet, fun application, children, education