# Candide 2.0 ou l'optimisme en jeu vidéo

Nina Ionescu 3mg01

## 1 Introduction

*Il y avait en Westphalie...* Et si cet incipit mythique se retrouverait un jour pixelisé, cela donnerait quoi ? C'est ce à quoi j'ai essayé de répondre lors de ce travail de maturité, qui consistait à adapter Candide de Voltaire en un jeu vidéo.

sources : Phaser.js par Richard Davey Phaser-tilemap-plus par Colin Vella Bosca Ceoil par Terry Cavanagh Pyxel Edit par Danik ( pseudonyme )

## 2 Fonctionement du code

Ce projet a été réalisé en javascript, avec les additions de ECMAScript6 et de deux librairies conçues pour la réalisation de jeu-vidéo : Phaser.js ainsi que Phaser-tilemap-plus.js . La majorité du code comprend des objets et des classes créés afin de pouvoir les utiliser comme outils de création.

### 2.1 Système général

Avant toute-chose, il a fallu concevoir un système qui permette d'atteindre diverses variables au travers de tous les outils du jeu. C'est pourquoi l'objet "globals" (présent dans globals.js), une variable globale a été créée. Cet objet stocke toutes les variables afin d'y avoir accès facilement sans devoir se soucier des scopes des différentes classes et de leur méthodes.

Pour retenir les diverses actions effectuées par le joueur ainsi que des données importantes telles que son positionement ou le décor actuellement chargé à l'écran, un autre objet global (présent dans gameRef.js) est utilisé. Il sert de référence pour itialiser le personnage incarné par le joueur dans l'état où ce dernier l'avait laissé.

Afin d'étendre les possibilités futures de diffusion du projet, une option de traduction a été pensée dans le code de départ. Il s'agit d'un chiffre ( 0 pour le français, 1 pour l'anglais, ... ) stocké dans l'objet mentionné au paragraphe ci-dessus. Grâce à cette variable, la fonction de dialogList.js retourne les textes figurants dans le jeu dans leur bonne version.
"exemple ? "

## 2.2 Gestion des states du jeu

qu'est-ce qu'une state, comment ça marche etc...

### 2.2.1 Boot et Preload

boot fait démarrer le jeu, preload pour les assets etc...

### 2.2.2 Menu Principal

expliquer la fonction avec les boutons, dans un premier temps elle est avec une sprite sheet pour chaque langue, éventuelle amélioration.

### 2.2.3 Game

toutes les fonctions agissent ici, dans la partie create de phaser.

### 2.2.4 Battle

expliquer la dynamique de combat, les points de vie etc..

## 2.3 Joueur

expliquer les déplacements, les directions , les interactions etc...

## 2.4 Les pnjs

expliquer la classe, la bulle (classe) les interaction avec le joueur etc..

## 2.5 Gestion des dialogues

expliquer la police de caractères, la dynamique des choix, l'objet manager , la syntaxe (les arrays avec les callbacks etc...)

### 2.6 Gestion du terrain

expliquer les maps,collisions, tilesets,calques, warps , l'objet terrainmanager etc... expliquer le fail avec le tween manager pour le fade in et fade out.

### 2.7 Interaction avec les objets

à compléter idée: un objet possède ou non un callback

### 2.8 Gestion du menu et de l'inventaire

à compléter

### 2.9 Du code à l'application

parler d'éléctron, npm, des packages.json, du favicon, de la fenêtre et des sauvegardes de la partie. (fenêtres)

### 2.10 Publication

système de sauvegarde , gitpages

## 3 Scénario alternatif

expliquer les dialogues, conserver le contenu tout en l'adaptant etc... expliquer la complication, le rabotage etc..

## 4 Conception des assets

### 4.1 Visuel

logiciels utilisé , techniques(travailler avec des tiles,
couches de transparence...)
inspiration pour les personnages, dessins
originaux etc.. restrictions au niveau du pixel
art, base pour cohérence etc...

#### 4.1.1 Sprites de dialogues

techniques(couches, dessins - restrictions), exemple

### 4.1.2 Sprites d'overworld

techniques, exemples

### 4.1.3 Tilesets

techniques, exemples

## 4.2 Musical

logiciel utilisé , techniques (trouver une suite d'accord, les instruments etc..) , éventuellement les bruitages S