

CAHIER DES CHARGES

Formation Développeur de Jeux Vidéo Pro

David MEKERSA

(Apprentissage par Projet / © Active Learning by Gaming Campus)

--	--	--

NOM DU PROJET : Construire la partie Front de votre projet

Numéro: Project N°1

Date:

Formation / Promotion : Dev JV Durban

- **Discipline(s) / Enseignement(s) couvert(s) par le projet :**

Développement de Jeux Vidéo

- **Nom du professionnel / intervenant :**

David Mekersa

- **Profil LinkedIn :**

<https://www.linkedin.com/in/dmekersa/>

- **Modalités d'apprentissage :**

Cours théoriques et pratiques

Apprentissage par projets

Cours à distance, le soir

--	--	--

Intitulé, contexte et descriptif du projet (intitulé, contexte, périmètre) :

Contexte :

Projet :

Création d'un Jeu vidéo 2D avec un langage de Script. L'apprenant doit réaliser un jeu en vue de dessus, mettant en œuvre 5 concepts fondamentaux de la programmation de jeux vidéo :

- 1- *Les listes*
- 2- *Les angles et rotations*
- 3- *La modularité*
- 4- *L'injection de dépendances*
- 5- *Les machines à états*

Exemples de Gameplays :

- Hotline Miami
- Geometry War
- Guncar Arena
- Vindicators
- [Blasteroids](#)

Technologies :

Lua (langage), Love2D (framework), Visual Studio Code (IDE).

Contraintes :

Pas d'utilisation de librairies externes.

--	--	--

A l'issue du projet, l'étudiant sera capable de :

- *Implémenter des fonctionnalités de jeu en utilisant un langage de script (Le Lua).*
- *Programmer des déplacements et des rotations à 360° pour les personnages et les objets du jeu en utilisant les fonctions de trigonométries telles que sinus, cosinus, calcul d'angles et de distances.*
- *Définition de conditions de victoire et de défaite pour le jeu, en prenant en compte les objectifs du joueur et les événements qui se produisent dans le jeu.*
- *Mise en œuvre de la modularité dans le code du jeu, en utilisant les modules Lua pour structurer le code de manière logique et réutilisable.*
- *Découpage du jeu en différentes scènes (par exemple, menu principal, gameplay, écran de pause, victoire, défaite), en utilisant des techniques de gestion de l'état du jeu.*
- *Gestion d'une liste d'entités du jeu (par exemple, personnages, objets, ennemis, etc.), en utilisant des listes pour stocker et gérer ces données.*
- *Utilisation de l'injection de dépendance pour gérer les relations entre différentes parties du code du jeu et limiter la portée des variables (encapsulation).*
- *Utilisation d'une machine à état fini (MEF) pour gérer les états et les transitions du comportement des PNJ ou des ennemis dans le jeu. Cela peut inclure la définition de différents états (par exemple, "idle", "marche", "attaque", "fuite") et de transitions entre ces états, en fonction des actions du joueur ou de l'environnement.*

--	--	--

ECHÉANCES ET LIVRABLES

Dates échéances	Livrables attendus	Moyens / formats (comment on transmet le livrable)
Dernière semaine du module	Le code du projet	Dossier du projet sous forme d'archive, ou Github

Planning

Semaine 1

Kickoff
Coaching

Semaine 2

Coaching
Coaching

Semaine 3

Coaching
Coaching

Semaine 4

Accompagnement
Accompagnement

Semaine 5

Accompagnement
Accompagnement

Semaine 6

Soutenance sur RV

Ces dates peuvent être amenées à évoluer en fonction des contraintes
des intervenants.

--	--	--

Pré-Requis au projet :

Avoir suivi la formation de mise à niveau.

[Atelier : Langage et programmation pour le jeu vidéo](#)

[Atelier : Pixels et origine](#)

[Atelier : Les bases d'un Lunar Lander en Love2D](#)

[Atelier Jeu de Pong : Apprenez à coder votre premier jeu vidéo rapidement et facilement](#)

[Atelier : Coder un inventaire de RPG avec une liste](#)

RESSOURCES

Ressources pour le travail préparatoire :

Ateliers :

[Guide : Rebonds et collisions](#)

[Atelier : Donner vie à une horde de zombies \(Les bases de l'Intelligence Artificielle\)](#)

[Atelier DLC 115 : Faire spawner des ennemis à intervalles réguliers](#)

[Atelier DLC 116 : Programmer le comportement des ennemis](#)

[DLC 142 – TANK WAR – Tirs orientés](#)

[Guide de programmation : Tanks](#)

[Atelier : Le secret des Tilemaps pour créer des niveaux](#)

[Atelier : Déplacements et collisions sur une Tilemap](#)

<https://www.gamecodeur.fr/dlc-76-attrape-moi-si-tu-peux-se-faire-poursuivre-par-un-ennemi/>

Et tous les cours qui pourraient vous aider sur :

<https://www.gamecodeur.fr/toutes-les-categories/>

Exercices :

Enoncés et solutions sur :

<https://www.gamecodeur.fr/programmation-lua-love2d-10-super-exercices-de-programmation-pour-sexercer-et-progresser/>

Semaine 1 :

- Ping Pong (pour les déplacements)
- Les cœurs (pour les boucles)

Semaine 2 :

--	--	--

- La sulfateuse (pour les tirs)

Semaine 3 :

- Cap Canaveral (pour les états)

Outils :

<https://code.visualstudio.com/>

<https://github.com/>

ACQUIS D'APPRENTISSAGE DÉTAILLÉS

CONNAISSANCES À ACQUÉRIR :

- *Connaissance du vocabulaire professionnel technique*
- *Connaissance d'un environnement de développement*
- *Connaissance des bonnes pratiques de la programmation*
- *Connaissance de l'architecture d'un programme de jeu vidéo*
- *Connaissance d'un langage de script*
- *Connaissance d'un Framework de programmation de jeux vidéo*
- *Connaissance des formats de fichiers appliqués au jeu vidéo*
- *Connaissance des concepts fondamentaux de programmation de jeux vidéo*

COMPÉTENCES À ACQUÉRIR :

- *Mettre en place un environnement de travail pour programmer un jeu vidéo*
- *Analyser et formaliser les fonctionnalités à programmer pour créer le jeu vidéo*
- *Programmer l'ensemble des algorithmes du jeu vidéo*
- *Déployer le jeu vidéo et contrôler son fonctionnement, corriger et faire évoluer ses fonctionnalités*

--	--	--

Grille d'évaluation du projet

Critères d'évaluation
Clarté et commentaires du code source : Le code est organisé et commenté de manière à faciliter la lecture et la compréhension
Fonctionnalité et stabilité : Le jeu fonctionne sans bugs ni plantages, permettant une expérience fluide
Découpage en modules : Le code est découpé en modules pour faciliter la compréhension, la réutilisation et la maintenance du code
Utilisation des machines à états : L'apprenant intègre une ou plusieurs machines à états pour gérer les comportements des éléments et/ou les états du jeu (Menus, gameplay, etc.)
Rotations et visées : L'apprenant maîtrise les fonctions trigonométriques de bases pour le déplacement et la rotation des éléments de gameplay (Sinus, cosinus, distance, calcul d'angles)
Utilisation des listes : Le jeu intègre l'utilisation des listes pour gérer les différents éléments du jeu (objets, ennemis, etc.) et pour faciliter la création de mécaniques de jeu complexes
Multiples scènes : Le jeu comporte plusieurs scènes (Menu, gameplay, pause, etc.)
Injection de dépendances : L'apprenant utilise l'injection de dépendances pour faciliter la communication et l'interaction entre les différents modules
Présentations claire du projet : L'apprenant peut faire une description structurée de son projet et expliquer les choix de conceptions
Maîtrise technique et conceptuelle : L'apprenant démontre, en répondant à des questions, qu'il maîtrise les concepts (fondamentaux : boucles, expressions, structures de contrôles, fonctions) et le langage utilisé dans son projet