GAMING CAMPUS

POUR LES PROS

CAHIER DES CHARGES

Formation Développeur de Jeux Vidéo Pro

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

NOM DU PROJET : Examen de Certification

Discipline(s) / Enseignement(s) couvert(s) par le projet :
Développement de Jeux Vidéo

Nom du professionnel / intervenant :

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 :

Le projet présenté pour l'examen de certification sera le projet réalisé en commun. Votre mission consiste à développer, sur la base du GDD fourni, un jeu de type Infinite Runner 3D sur le moteur de jeu Unity.
Vos livrables seront examinés par un jury dans le but de vous délivrer le Certificat de Compétences Professionnelles (CCP) "Programmer un jeu vidéo" issu du Titre professionnel RNCP 38294 Concepteur de Jeux Vidéo.

Rappel des éléments à prendre en compte pour la création de ce projet :

1. Structure du parcours :

- Le tracé est constitué de tronçons de terrains prédéfinis (Prefabs).
- Ces tronçons doivent pouvoir être assemblés par la génération procédurale en fonction du gameplay.
- Limitation : Le tracé est linéaire et ne comporte pas de virages.

2. Mouvements du personnage :

- Le personnage principal peut se déplacer horizontalement ou sur des lignes fixes.
- Le personnage peut être un humain, un véhicule ou toute autre entité, selon la thématique choisie.

3. Phases de jeu :

- Le gameplay doit évoluer au cours de la partie : prévoyez plusieurs phases qui augmentent la difficulté ou introduisent de nouvelles mécaniques.
- Exemple : Augmentation progressive de la vitesse, apparition d'obstacles ou d'ennemis supplémentaires.

4. Éléments interactifs :

- Le joueur peut looter des éléments (pièces, diamants, etc.) disséminés le long du parcours.

- Les objets à looter peuvent par exemple être disposés de manière à encourager le joueur à se déplacer latéralement ou à prendre des risques.

5. Obstacles et ennemis :

- Le parcours doit comporter des obstacles (fixes ou mobiles) à éviter.
- Types d'obstacles : Murs, barrières, trous, véhicules, etc.
- Certains obstacles peuvent être en mouvements : plateformes qui se déplacent, éléments qui tombent sur le joueur, etc.
- Des ennemis peuvent également être introduits dans les tronçons de terrain pour varier les défis.

6. Persistance des données :

- Certaines données (par exemple, score, objets collectés) devront être persistantes entre les scènes via un fichier JSON.
- Les éléments de gameplay qui nécessitent cette persistance doivent être identifiés et communiqués aux programmeurs.

Exemple de références :
● Subway Surfer

LIVRABLES ATTENDUS

- **Documentation technique** :
  elle doit inclure

  - la description du moteur de jeu
  - les modalités de programmation du moteur et du gameplay (technologies, langages, outils, structuration du code, nomenclature, bonnes pratiques)
  - les procédures de test et de débogage
  - un planning de production démontrant les interactions avec les autres membres du projet

  La documentation doit être complète, lisible, compréhensible. Elle répond au besoin d'information des développeurs et des testeurs. Les solutions techniques sont adaptées aux contraintes du projet (temps, budget, plateformes) et aux attentes en termes de performances.

- **Présentation des développements et améliorations des fonctionnalités du moteur de jeu :**

  - l'historique des évolutions et des modifications apportées depuis la première version du jeu
  - les évolutions doivent se refléter dans la documentation technique

- **Code source et prototype jouable :**

  - l'ensemble du code source, facile à lire (commentaires, indentation, nomenclature, …), modulaire, documenté et permettant des évolutions ultérieures
  - prototype stable (pas de crash, pas de bugs) et performant
  - le gameplay respecte le cahier des charges

- **Rapport de tests :**

    - la procédure de tests est précise et exhaustive. Elle comprend des tests dans lesquels le joueur effectue des actions inhabituelles ou non prévues.
    - un outil de débogage a été utilisé au cours des tests (fournir des screenshots, logs, … prouvant ce point)
    - un rapport présentant un diagnostic précis des problèmes ainsi que les corrections apportées (réécriture de code, ajout de conditions, modification de variables, …)
    - des tests complémentaires ont été menés pour s'assurer que les corrections n'ont pas entraîné d'autres problèmes

- **Documents complémentaires**

    - le cahier des charges et le GDD
    - en complément du planning, joindre des comptes-rendus de réunions et tout document démontrant un travail en équipe.

Référentiel d'évaluation :

Définition des solutions techniques liées au développement du jeu Dans le cadre d'un projet fictif ou réel portant sur la programmation d'un jeu vidéo, le candidat doi**t rédiger la documentation technique** (ou bible technique) cadrant la programmation du moteur de jeu et des fonctionnalités du jeu. Compétences : Définir les solutions techniques liées au développement du jeu en collaboration avec l'ensemble des métiers de la production - en s'appuyant sur le cahier des charges du jeu - en définissant les outils, les technologies, les bonnes pratiques de codage, et les procédures de test et de débogage - en produisant la documentation technique associée afin de cadrer la programmation du moteur et des fonctionnalités du jeu.	. La documentation technique est claire et structurée de façon logique et le vocabulaire utilisé est professionnel (normes d'usage, traduction technique dans un langage accessible et compréhensible.). La documentation technique est complète, elle comprend : la description du moteur de jeu, les modalités de programmation du moteur et du gameplay (outils, technologies, langage de programmation, structuration du code, la nomenclature, bonnes pratiques), les procédures de test et de débogage. . Le contenu répond aux besoins d'information des développeurs et des testeurs. . Les solutions techniques envisagées sont adaptées au regard des contraintes du projet (temps, budget, plateforme cibles) et des attentes en termes de performance.
Développement des fonctionnalités du moteur de jeu Dans le cadre d'un projet fictif ou réel portant sur la programmation d'un jeu vidéo, le candidat doit présenter les **développements et améliorations des fonctionnalités du moteur de jeu.** Compétences : Développer les fonctionnalités du moteur de jeu - en identifiant au préalable les besoins en collaboration avec le directeur technique et les métiers de la production - en utilisant les outils et les langages de programmation adaptés afin d'optimiser le moteur de jeu pour les spécificités du projet	. Les évolutions proposées sont réalistes et répondent aux besoins identifiés. . La programmation du moteur de jeu est compréhensible : elle est facile à lire et à modifier pour les autres développeurs. . La programmation du moteur de jeu est documentée et maintenable, elle permet des modifications ultérieures dans le cadre des changements de conception ou des mises à jour du jeu. . Le moteur de jeu est stable et fiable (absence de bogues). . Les fonctionnalités développées sont performantes.

Programmation du gameplay Dans le cadre d'un projet fictif ou réel portant sur la programmation d'un jeu vidéo, le candidat doit **présenter les scripts de programmation ainsi que le prototype de jeu.** <u>Compétences :</u> Programmer les mécaniques de jeu, les niveaux, ainsi que les interactions entre le joueur et l'environnement - en coordination avec l'ensemble des corps de métiers (game design, tech art, game art, level design…) - en utilisant les outils et les langages de programmation adaptés afin de développer le prototype du jeu vidéo	. La programmation du gameplay est compréhensible : elle est facile à lire et à modifier pour les autres développeurs. . La programmation du gameplay documentée et maintenable, elle permet des modifications ultérieures dans le cadre des changements de conception ou des mises à jour du jeu. . La programmation du gameplay est fiable : elle fonctionne correctement dans toutes les situations, même lorsque le joueur essaie de faire quelque chose d'inhabituel ou de non prévu. . La programmation est modulaire : elle est facile à modifier, à réutiliser et à adapter à différents types de jeux ou de scénarios. . Les fonctionnalités du jeu et les interactions fonctionnent comme prévu dans le cahier des charges.
Diagnostic et résolution des problèmes techniques Dans le cadre d'un projet fictif ou réel portant sur la programmation d'un jeu vidéo, le candidat doit présenter la **méthodologie et les actions mises en oeuvre pour diagnostiquer et résoudre un problème technique**. <u>Compétences :</u> Diagnostiquer et résoudre les problèmes techniques - en identifiant et en analysant les parties de codes à l'origine des dysfonctionnements - en implémentant des solutions de techniques de génération de fichiers d'évènements - en corrigeant les erreurs, au moyen d'outils de débogage afin d'améliorer les performances du jeu vidéo.	. Le diagnostic est précis : la cause racine du problème a été identifiée. . Le code concerné a été corrigé selon la méthode la plus appropriée en fonction de la nature du bogue : (réécriture du code, ajout de conditions, modification des variables, l'utilisation d'un outil de débogage…) . Des tests approfondis ont été réalisés pour s'assurer que le bogue a été complètement éliminé, et que les corrections effectuées n'ont pas générées de nouvelles erreurs.

Grille d’évaluation

1 Définir les solutions techniques liées au développement du jeu	
Définir les outils, technologies et bonnes pratiques pour concevoir un jeu vidéo à partir d'un cahier des charges	La **documentation technique** est claire et structurée de façon logique et le vocabulaire utilisé est professionnel (normes d’usage, traduction technique dans un langage accessible et compréhensible.).
Définir les procédures de tests et de debug pour optimiser le rendu d'un jeu vidéo	Les solutions techniques envisagées sont adaptées au regard des contraintes du projet (temps, budget, plateforme cibles) et des attentes en termes de performance.
Produire des documents techniques, compréhensibles par l'ensemble des métiers de la production d'un jeu vidéo	Le contenu répond aux besoins d’information des développeurs et des testeurs. La documentation technique est complète, elle comprend : la description du moteur de jeu, les modalités de programmation du moteur et du gameplay (outils, technologies, langage de programmation, structuration du code, la nomenclature, bonnes pratiques), les procédures de test et de débogage.
2 Développer les fonctionnalités du moteur de jeu	
Identifier les ressources humaines et matérielles nécessaires pour la création d'un jeu vidéo	Les ressources sont clairement identifiées et listées dans la documentation.
Utiliser les outils et langages de programmation adaptés au développement d'un jeu vidéo Programmer avec le langage C# Utiliser le moteur de jeu Unity	Les évolutions proposées sont réalistes et répondent aux besoins identifiés. La programmation du moteur de jeu est compréhensible : elle est facile à lire et à modifier pour les autres développeurs. La programmation du moteur de jeu est documentée et maintenable, elle permet des modifications ultérieures dans le cadre des changements de conception ou des mises à jour du jeu Le moteur de jeu est stable et fiable (absence de bogues). Les fonctionnalités développées sont performantes.

3 Programmer les mécaniques de jeu, les niveaux, ainsi que les intéractions entre le joueur et l'environnement	
Coordonner son travail avec l'ensemble des corps de métiers du jeu vidéo	La programmation du gameplay est compréhensible : elle est facile à lire et à modifier pour les autres développeurs.
Construire des environnements grâce aux Tilemaps	La programmation du gameplay est fiable : elle fonctionne correctement dans toutes les situations, même lorsque le joueur essaie de faire quelque chose d'inhabituel ou de non prévu
Développer des comportements programmés	La programmation du gameplay est fiable : elle fonctionne correctement dans toutes les situations, même lorsque le joueur essaie de faire quelque chose d'inhabituel ou de non prévu
Utiliser la POO (Programmation Orientée Objet)	La programmation du gameplay est fiable : elle fonctionne correctement dans toutes les situations, même lorsque le joueur essaie de faire quelque chose d'inhabituel ou de non prévu
Afficher et manipuler des images dans un framework	La programmation est modulaire : elle est facile à modifier, à réutiliser et à adapter à différents types de jeux ou de scénarios.
Concevoir l'architecture d'un design pattern	La programmation du gameplay est fiable : elle fonctionne correctement dans toutes les situations, même lorsque le joueur essaie de faire quelque chose d'inhabituel ou de non prévu
Utiliser les principales fonctionnalités de Unity : game objects, components, physics, etc.	La programmation du gameplay est fiable : elle fonctionne correctement dans toutes les situations, même lorsque le joueur essaie de faire quelque chose d'inhabituel ou de non prévu
Intégrer des éléments et assets graphiques dans un moteur de jeu	La programmation est modulaire : elle est facile à modifier, à réutiliser et à adapter à différents types de jeux ou de scénarios.

4 Diagnostiquer et résoudre les problèmes techniques	
Analyser des dysfonctionnements, identifier les parties responsables dans le code	Le diagnostic est précis : la cause racine du problème a été identifiée.
Proposer des solutions aux dysfonctionnements	Le code concerné a été corrigé selon la méthode la plus appropriée en fonction de la nature du bogue : (réécriture du code, ajout de conditions, modification des variables, l'utilisation d'un outil de débogage...)
Corriger les dysfonctionnements grâce aux outils de debug	Des tests approfondis ont été réalisés pour s'assurer que le bogue a été complètement éliminé, et que les corrections effectuées n'ont pas générées de nouvelles erreurs.