unity

unity

Certified

Programmer

Objectifs de l'examen

Programmeur
certifié Unity

Le rôle

Les professionnels en programmation Unity développent du contenu interactif dans Unity. Dans le cadre d'un travail d'équipe, le programmeur Unity donne vie à la vision de l'application en utilisant les fonctionnalités disponibles dans l'Éditeur Unity, ainsi que les ressources audio et visuelles créées par d'autres membres de l'équipe de développement logiciel (les experts en son et en graphisme, par exemple). Le programmeur Unity est un programmeur généraliste, qui peut résoudre les problèmes de codage difficiles et qui est chargé de contribuer à un large éventail de tâches techniques telles que l'intégration de ressources artistiques, le codage de l'interface utilisateur, la rédaction des scripts d'interaction utilisateur et des règles du système de jeu, l'implémentation de la logique d'état des applications, la simulation de la physique, le débogage du code et l'optimisation des performances.

La certification professionnelle Programmeur certifié Unity est destinée aux programmeurs de niveau débutant à intermédiaire et aux diplômés du postsecondaire qui recherchent un poste de programmeur dans différents secteurs. Cette certification vise à montrer aux employeurs potentiels que son titulaire :

- utilise une connaissance approfondie de la programmation dans le cadre de processus de développement logiciel professionnels pour créer et gérer des applications développées à l'aide du moteur Unity ;
- possède des compétences en matière de processus techniques, fait preuve d'un esprit logique et d'ingéniosité ;
- peut être chargé de gérer de manière autonome les tâches de programmation de routine de niveau intermédiaire, et de relever des défis techniques complexes avec des ingénieurs plus expérimentés.

Intitulés de poste pour ce rôle

- Programmeur Gameplay
- Ingénieur logiciel
- Développeur de logiciels
- Développeur Unity
- Développeur d'applications mobiles

Prérequis

Cette certification a été conçue pour les programmeurs récemment diplômés en programmation de jeux, en informatique ou dans des domaines apparentés ; les étudiants indépendants ayant à leur actif au moins deux ans d'études dans un programme universitaire ou équivalent, ou d'expérience professionnelle ; les professionnels en début-milieu de carrière ayant utilisé Unity dans un contexte professionnel. Les candidats doivent se présenter à l'examen en ayant déjà une expérience pratique de la programmation d'applications interactives avec Unity, qu'il s'agisse d'un travail individuel ou au sein d'une équipe polyvalente, ayant abouti à la création d'un prototype ou d'une démonstration technique.

Expérience préalable requise :

- Au moins 2 ans d'expérience pratique dans la programmation de jeux vidéo ou la programmation interactive en 3D avec Unity
- Au moins 2 ans d'expérience pratique dans la programmation informatique, y compris en C#
- Expérience du cycle complet du développement de logiciels, du concept à l'achèvement
- Compréhension des applications professionnelles du développement de logiciels avec Unity, y compris le développement de jeux vidéo, le divertissement interactif et la visualisation de conceptions
- Compréhension de base des ressources visuelles/3D et du pipeline d'animation Unity, y compris la configuration des personnages et de l'environnement
- Compréhension des pratiques de développement logiciel utilisées par les équipes professionnelles, y compris les tests unitaires et le contrôle de version
- Connaissance des services Unity utilisés pour la collaboration, la monétisation, les opérations en direct et le multijoueur
- Compréhension des mathématiques essentielles au développement interactif 3D, y compris l'algèbre linéaire et les opérations matricielles

Remarque : Cette certification a été développée pour la version 2017.3 de Unity.

Compétences de base

Dans ce domaine d'activité, les compétences de base consistent à contribuer à l'exécution technique d'un projet, depuis sa conception jusqu'à son lancement, et au-delà.

Programmation d'interactions basiques

- Implémenter et configurer le comportement et la physique des objets du jeu
- Implémenter et configurer les entrées et contrôles
- Implémenter et configurer les vues et les mouvements de la caméra

Travail dans le flux artistique

- Comprendre les matériaux, les textures et les shaders, écrire des scripts qui interagissent avec l'API de rendu Unity
- Comprendre l'éclairage, écrire des scripts qui interagissent avec l'API d'éclairage Unity
- Comprendre les animations 2D et 3D, écrire des scripts qui interagissent avec l'API d'animation Unity
- Comprendre les systèmes de particules et les effets, rédiger des scripts qui interagissent avec l'API du système de particules Unity

Développement de systèmes d'application

- Interpréter des scripts pour le flux de l'interface, par exemple les systèmes de menus, la navigation dans l'IU et les paramètres de l'application
- Interpréter des scripts pour la personnalisation contrôlée par l'utilisateur, par exemple les créations de personnages, les inventaires, les pages d'accueil de boutiques et les achats dans l'application
- Analyser les scripts des fonctionnalités de progression des utilisateurs, par exemple le calcul du score, le passage des niveaux et les économies du jeu, en utilisant des technologies telles que Unity Analytics et PlayerPrefs
- Analyser les scripts des superpositions 2D telles que les affichages tête haute (HUD), les mini-cartes et les publicités
- Identifier les scripts permettant l'enregistrement et la récupération des données de l'application et de l'utilisateur
- Reconnaître et évaluer l'impact des fonctionnalités de jeu réseau et multijoueurs

Programmation pour la conception des scènes et des environnements

- Déterminer les scripts nécessaires pour l'implémentation de ressources audio
- Identifier les méthodes d'implémentation de l'instanciation, de la destruction et de la gestion des GameObjects
- Déterminer les scripts nécessaires pour l'orientation avec le système de navigation Unity

Optimisation des performances et des plateformes

- Évaluer les erreurs et les problèmes de performances à l'aide d'outils tels que Unity Profiler
- Identifier les optimisations permettant de répondre aux exigences liées aux plateformes de compilation et/ou aux configurations matérielles spécifiques
- Déterminer les possibilités et les optimisations d'interface utilisateur courantes pour les plateformes XR

Collaboration avec des équipes de développement logiciel professionnelles

- Reconnaître les concepts associés aux utilisations et aux impacts du contrôle de version à l'aide de technologies telles que Unity Collaborate
- Démontrer des connaissances en matière d'élaboration de tests (utilisation de Unity Profiler, mais également des techniques de débogage et de test conventionnelles) et leur impact sur le processus de développement logiciel
- Reconnaître les techniques de structuration des scripts pour assurer leur modularité, leur lisibilité et leur réutilisabilité

Sujets de l'examen de certification

Programmation d'interactions basiques

- Implémenter les comportements et interactions des Gameobjects et des environnements
- Identifier les méthodes permettant d'implémenter les entrées et les commandes
- Identifier les méthodes permettant d'implémenter les vues et les mouvements de la caméra

Travail dans le flux artistique

- Connaissance des matériaux, textures et shaders - API de rendu Unity
- Connaissances en matière d'éclairage - API d'éclairage Unity
- Connaissances en matière d'animation 2D et 3D - API d'animation Unity
- Connaissance des systèmes de particules - API de particules Unity

Développement de systèmes d'application

- Flux de l'interface tels que les systèmes de menus, la navigation dans l'IU et les paramètres de l'application
- Personnalisation contrôlée par l'utilisateur incluant les créateurs de personnages, les inventaires, les pages d'accueil de boutiques et les achats dans l'application
- Implémenter des fonctionnalités de progression des utilisateurs, telles que le calcul du score, le passage des niveaux et les économies du jeu, en utilisant des technologies comme Unity Analytics
- Implémenter des superpositions 2D telles que des affichages tête haute (HUD), des mini-cartes et des publicités
- Enregistrer et récupérer les données de l'application et de l'utilisateur
- Reconnaître l'importance et l'impact des fonctionnalités de jeu réseau et multijoueurs

Programmation pour la conception des scènes et des environnements

- Déterminer les scripts nécessaires pour l'implémentation de ressources audio
- Identifier les méthodes d'implémentation de l'instanciation, de la destruction et de la gestion des GameObjects
- Déterminer les scripts nécessaires pour l'orientation avec le système de navigation Unity

Optimisation des performances et des plateformes

- Évaluer les erreurs et les problèmes de performances à l'aide d'outils tels que Unity Profiler
- Identifier les optimisations permettant de répondre aux exigences liées aux plateformes de compilation et/ou aux configurations matérielles spécifiques
- Déterminer les possibilités et les optimisations d'interface utilisateur courantes pour les plateformes XR

Collaboration avec des équipes de développement logiciel

- Contrôle de version : Impacts et utilisations d'outils comme Unity Collaborate
- Le test et son impact sur le processus de développement logiciel
- Reconnaître les techniques de structuration des scripts pour assurer leur modularité, leur lisibilité et leur réutilisabilité

Exemples de questions

Question 1

Un programmeur doit implémenter un système de menus d'IU. Chaque menu comprend un panneau et un ou plusieurs boutons, tous apparentés à un objet UI Canvas. L'intégralité de ce système sera créé dans une scène distincte, chargée de façon additive.

Le style graphique des panneaux et des boutons doit être cohérent (couleur, texture, type de transition des boutons, etc.), mais la directrice artistique n'a pris aucune décision définitive. Elle aimerait travailler sur ces paramètres en même temps que le programmeur élabore l'IU. Ses modifications auront une incidence sur tous les objets de la scène, qu'ils soient nouveaux ou existants.

Pour le programmeur, quelle est la meilleure façon d'utiliser les fonctionnalités Unity pour créer facilement un système de menus fonctionnel tout en laissant la directrice artistique travailler simultanément (et indépendamment) sur le style visuel et l'ambiance ?

A Créer des sous-classes pour `UI.Button` et `UI.Panel`, puis définir les valeurs liées au style visuel et à l'ambiance en les programmant

B Créer de nouveaux matériaux pour les boutons et les panneaux, puis les attribuer à tous les boutons et panneaux de la scène

C Utiliser des prefabs pour un bouton et un panneau, puis demander à la directrice artistique de les modifier

D Écrire un script pour rechercher/remplacer les valeurs du fichier Scene en fonction des entrées de la directrice artistique

Question 2

Un jeu de course infinie en 3D se déroule dans une gare de triage, le long de plusieurs voies ferrées parallèles. Le joueur court en permanence sur les voies et doit éviter les trains arrivant en sens inverse en sautant par-dessus ou en se positionnant sur une voie adjacente.
Chaque nouveau train ajouté à un voie est placé derrière tous les autres trains de cette voie. Toutefois, les trains étant immobiles ou se dirigeant vers le joueur ont des vitesses variables, ils se chevauchent parfois, ce qui doit être corrigé.

Quelle est la façon la plus performante d'empêcher les nouveaux trains de chevaucher les trains existants sur la même voie ?

A Lorsqu'un train arrive sur la voie, déterminer un point d'apparition qui évitera ce problème en utilisant les vitesses du nouveau et du dernier train placés sur cette voie, ainsi que le point où le dernier train placé sur la voie disparaitra lorsqu'il croisera le joueur.

B Lorsqu'un train se déplace, créer un raycast vers l'avant depuis l'avant du train, et pousser vers l'avant tous les trains touchés par le raycast à la vitesse du train le plus rapide.

C Lorsqu'un train arrive sur la voie, y ajouter un Rigidbody, puis utiliser les forces pour déplacer les trains.

D Lorsqu'un train arrive sur la voie, utiliser un BoxCast d'une longueur proportionnelle à la vitesse du train pour garantir qu'il n'entrera pas en collision avec un autre train avant de se trouver derrière la caméra.

Question 3

Un programmeur travaille sur une pièce à l'aspect sombre et maussade, et doit créer une torche vacillante qui projette une ombre mystérieuse et dansante sur les murs, le sol et le plafond. Il écrit ces fonctions dans un script `MonoBehaviour` associé à la torche :

```
void Start()
{
    Light light = GetComponent<Light>();
    light.lightMapBakeType = LightMapBakeType.Mixed;
    light.type = LightType.Area;
    light.shadows = LightShadows.Soft;
    light.range = 5f;
}
void Update()
{
    GetComponent<Light>().intensity = Mathf.PerlinNoise(Time.time, 0);
}
```

La torche ne projette ni lumière ni ombre durant l'exécution. Dans l'Éditeur Unity, l'éclairage est configuré aux valeurs par défaut.

Que doit faire le programmeur pour que ce code fonctionne correctement ?

A Définir `light.lightBakeType` à `LightmapBakeType.Realtime`

B Définir `light.range` à `10`

C Définir `light.shadows` à `LightShadows.Hard`

D Définir `light.type` à `LightType.Point`

Question 4

Un programmeur développe un jeu de simulation d'exploitation minière dans lequel le joueur peut creuser le sol à la recherche de minéraux. Sur l'un des sites, le joueur peut créer un tunnel qui traverse un réseau de grottes existant. Le document de conception précise que n'importe quel son produit dans le réseau de grottes actuel et les nouveaux tunnels doit comporter une réverbération. Le programmeur doit s’assurer que l’utilisateur se trouve constamment dans la ReverbZone de la grotte la plus proche.

Comment doit-il manipuler les propriétés de la classe AudioReverbZone pour répondre à ces exigences ?

A Augmenter les réflexions pour qu'elles correspondent à la nouvelle zone

B Augmenter la valeur maxDistance des deux ReverbZones afin que ces dernières se touchent dans la nouvelle zone de connexion

C Augmenter la réverbération afin qu'elle soit adaptée à la nouvelle zone

D Augmenter la valeur decayTime de la nouvelle zone

Question 5

Lors de l'écriture d'une fonction de chargement, un programmeur reçoit une erreur de compilation :

```
error CS1624: The body of `CustomAnalytics.LevelLoading()' cannot be
an iterator block because `void' is not an iterator interface type
```

```
void LevelLoading(){
    AsyncOperation async = SceneManager.LoadSceneAsync("Level_01");
    while (!async.isDone)
    {
        yield return null;
    }
}
```

Que doit-il faire pour corriger cette erreur ?

A Remplacer `yield return null` par `yield return WaitForSeconds(0)`

B Remplacer `void LevelLoading()` par `IEnumerator LevelLoading`

C Remplacer `SceneManager.LoadSceneAsync("Level_01")` par `Application.LoadLevelAdditiveAsync("Level_01")`

D Remplacer `while (!async.isDone)` par `while (!async.allowSceneActivation)`

Question 6

Le système d'entrée d’un jeu de conduite est mappé de sorte que l’axe d'entrée horizontal contrôle la direction. Durant les tests, on s'aperçoit que certains joysticks enregistrent des informations concernant la direction même lorsque le manche est centré.

Pour résoudre ce problème, quelle modification doit être apportée à l'axe dans le système d'entrée ?

A Augmenter la valeur Gravity

B Définir l'option Snap à true

C Augmenter la valeur Deadzone

D Réduire la valeur Sensitivity

Question 7

Dans un jeu d'aventure se déroulant sur une autre planète, le joueur doit exterminer plusieurs formes de vie. Son score augmente à chaque mort. Le document de conception précise que le score doit être associé au compte du joueur afin que ce dernier puisse le retrouver par la suite, que ce soit dans autre session ou sur un autre appareil.

Quelle est la méthode la plus fiable pour stocker les données de score ?

A Utiliser `DontDestroyOnLoad()` sur le GameObject contenant les données de score, puis télécharger ces dernières sur un serveur juste avant la fermeture de l'application

B Enregistrer le score dans `PlayerPrefs` chaque fois qu'il est mis à jour et téléchargé sur un serveur juste avant la fermeture de l'application

C Utiliser une valeur statique pour stocker les données de score afin qu'elles soient disponibles dans la prochaine session de jeu

D Utiliser la sérialisation pour stocker en permanence les données de score et les télécharger sur un serveur

---

Bonnes réponses : C, A, D, B, B, C, D