documentation_oculus

daniel.blanchard

May 2023

1 Initialisation

Documentation du Projet Unity - Build pour Android

Ce document fournit des instructions détaillées pour build votre projet Unity pour la plateforme Android. Assurez-vous de suivre chaque étape attentivement pour garantir un bon fonctionnement de votre application.

2 Prérequis

Avant de commencer, vérifiez les points suivants :

- Assurez-vous d'avoir Unity installé sur votre ordinateur.
- Activez le support pour Android dans Unity.
- Installez les packages requis pour votre projet : Oculus Integration, Animation Rigging, Oculus XR Plugin, Test Framework, TextMeshPro, Unity UI, Visual Scripting, Visual Studio Code Editor et XR Plugin Management.

3 Étapes de Build pour Android

- Vérifier les paramètres du projet
  1. Assurez-vous que votre projet Unity est configuré pour la plateforme Android.
  2. Sinon, suivez ces étapes pour effectuer la conversion :
     - Dans Unity, allez dans "File" (Fichier) ¿ "Build Settings" (Paramètres de Build).
     - Sélectionnez "Android" comme plateforme de Build.
     - Si la plateforme Android n'est pas installée, Unity vous proposera de la télécharger et de l'installer.
- Intégration des packages et des scripts
  1. Vérifiez si l'intégration automatique des packages, scripts et assets a été effectuée.
  2. Si ce n'est pas le cas, suivez les instructions ci-dessous : Ouvrez la fenêtre "Package Manager" (Gestionnaire de Packages) dans Unity.
  3. Recherchez les packages requis et installez-les : Oculus Integration, Animation Rigging, Oculus XR Plugin, Test Framework, TextMesh-Pro, Unity UI, Visual Scripting, Visual Studio Code Editor et XR Plugin Management.
  4. Assurez-vous que tous les scripts nécessaires sont correctement importés dans votre projet.
- (Optionnel) Fixer les lumières
- Si nécessaire, ajustez les lumières dans votre scène Unity. Vous pouvez utiliser le bouton Oculus situé en bas pour appliquer automatiquement les modifications à toutes les lumières.
- Build de l'application Android
  - Une fois que toutes les étapes précédentes ont été complétées, vous êtes prêt à build votre application pour Android.
  - Allez dans "File" (Fichier) ¿ "Build Settings" (Paramètres de Build) dans Unity.
  - Sélectionnez la plateforme Android.
  - Cliquez sur le bouton "Build" (Build) et choisissez un emplacement pour sauvegarder l'APK généré.
- Conditions d'utilisation et licences
  - Veuillez noter que tous les éléments constitutifs de votre projet doivent être libres de droits et accessibles, à condition qu'ils ne soient pas utilisés à des fins commerciales.

  - Si vous utilisez des packages ou des assets tiers, veuillez vous référer à leurs conditions d'utilisation et demander l'autorisation au propriétaire d'origine si nécessaire.

- Packages Requis Assurez-vous d'intégrer les packages suivants dans votre projet Unity :
  - Oculus Integration (disponible sur le Store d'Unity)
  - Animation Rigging
  - Oculus XR Plugin
  - Test Framework
  - TextMeshPro
  - Unity UI
  - Visual Scripting
  - Visual Studio Code Editor
  - XR Plugin Management

4 Détails des Scènes

Le projet comprend deux scènes :

1. **MainScene** : Cette scène représente l'écran de début du jeu. Elle contient un bouton pour lancer la scène principale du jeu ainsi qu'un bouton pour accéder à des informations pratiques dans une interface. La scène ne contient que ces éléments.
   Le script MenuCotroller, associe le chargement de la scène de jeu au boutton 'start' et la descript au bouton 'about.'

2. **OneRoom(Assets - Scène - MainScene - OneRoom - OneRoom (Cette scène est extraire de l'asset store qui a été modifié quand je jouai à vrchat expliquant sa configuration particulière)** : Cette scène est la scène principale du jeu. Les objets suivants sont intégrés dans cette scène :

   - **Camera** : La caméra utilisée pour la visualisation du jeu.
   - **Lights** : Les lumières utilisées dans la pièce pour l'éclairage.
   - **OverPlayerController** (du package Oculus) : Ce script contient les mains du personnage 3D. Les scripts nécessaires sont gérés par le système intégré et ne nécessitent pas de modifications supplémentaires.
   - **RoomController** : Ce script sert simplement à stocker tous les objets de la scène de type décoration.
   - **FormulaireController** : Ce script gère le formulaire dans la scène. Il comprend les éléments suivants :
     - **ButtonVR**, **Keyboard**, **KeyboardButton** et **TypingArea** : Ces scripts permettent d'utiliser le clavier du jeu. Ils référencent le **KeyboardController** qui contient le script et tous les assets du clavier tels que les boutons.
     - **KeyboardController** : Ce script contient le formulaire et tous les éléments qui le constituent.
     - **FormulaireScript** : Ce script référence chacun des éléments du formulaire et envoie les données au Google Doc à l'aide de ses identifiants spécifiques définis dans le script.
   - **ItemController** : Ce script gère les éléments du jeu, tels que la pierre, le papier, les ciseaux et la main de l'IA. Il comprend les fonctionnalités suivantes :
     - Il contient les éléments du jeu (pierre, papier, ciseaux, main de l'IA).
     - Il inclut un menu de contrôle du jeu pour les tests futurs (version 5.0+).
     - Les scripts portent le même nom que l'objet auquel ils sont attachés et envoient le signal "pierre" au **ItemController**, qui se chargera d'envoyer les données au **GameController**.

  - Dans cette version, des déclencheurs (triggers) qui modifient la texture ou les sons ont été directement intégrés avec les déclencheurs par défaut des objets, sans nécessiter de script supplémentaire.
  - Seule la musique est jouée par l'**ItemController**.
  - Ces éléments supplémentaires concernent des tests supplémentaires pour cette version qui n'a pas encore été déployée.

- Les contrôleurs mentionnés précédemment ont pour rôle principal de gérer les changements scéniques simples tels que les changements de texture. Les codes associés sont assez simples et englobent les scripts suivants : AISettings, ActionsSettings, Impressions (formulaire) et MenuController (scène de début de jeu).
- GameController :
  - Il contient le script du jeu.
    (a) Il inclut le code de l'IA :
      * Ce code utilise une structure de dictionnaire pour stocker les données.
      * Il les stocke dans la mémoire cache du casque jusqu'à ce que les données soient réutilisées. La documentation fournie avec l'IA fournira plus de précisions sur l'algorithme, mais le changement est similaire, avec comme seule différence l'utilisation de la mémoire du casque.
  - Le code de gestion du jeu est comprise.
    (a) Les fonctions mentionnées sont assez transparentes, mais pour résumer : - Il contient des fonctions de transformation de format (STRUCT vers String) pour les fonctions de jeu.
    (b) D'autres fonctions ont pour but de formater les données pour le formulaire ou de fournir des détails, comme la fonction GetAdversaireType() qui renvoie 'aléatoire', 'probabiliste' ou 'impossible' en fonction du mode de l'IA.
    (c) Les fonctions de jeu sont transparentes, tout comme les fonctions de contrôle. Par exemple, ai_impossible permet de définir l'IA en mode 'impossible', où elle jouera le coup opposé au joueur.
    (d) L'ensemble des données est stocké dans les variables suivantes :
      i. private Dictionary¡string, int¿ Moves : contient l'ensemble des coups joués et leur nombre pour la méthode probabiliste 1 (celle fournie par l'interface). Dans la version actuelle, les deux méthodes (celle de l'interface et une autre) sont utilisées en alternance, avec une plus grande probabilité accordée à celle de l'interface.
      ii. private List¡string¿ Games_resultats : contient les résultats des matchs.

iii. private int Games_resultats_log_win : affiche le nombre de victoires.

iv. private int Games_resultats_log_loose : affiche le nombre de défaites.

v. private int number_of_matches : indique le nombre de tours de jeu.

(e) La fonction Play() a les responsabilités suivantes :

* Mettre à jour la base de données (BDD) des coups enregistrés.
* Utiliser une fonction qui fait référence à la main (hand) dans le GameManager pour jouer l'animation et le coup calculé par l'IA.

Cette fonction est chargée de gérer la logique du jeu, de mettre à jour les données nécessaires et de coordonner les actions de l'IA avec les animations et les mouvements de la main dans le GameManager.

5 Compléments

La version actuelle est basée sur les versions précédentes, mais elle ne contient pas de sauvegardes. Cela a entraîné des problèmes de plantage et de nombreux bugs. De plus, nous avons rencontré des difficultés avec Unity et Git, et il est important de noter que nous sommes des débutants sur Unity. Nous n'avons consacré qu'une heure maximum à VRChat, principalement pour la création d'objets, ce qui est un temps limité pour maîtriser tous les aspects.

Certains éléments, tels que le script "pierre", ne sont pas détaillés ici, mais leur fonction est d'envoyer "player_move = pierre" au GameManager. Ils ne sont pas inclus dans ce document pour réduire sa complexité.

Nous vous remercions de votre attention et restons disponibles pour toute question ou demande de démonstration.