JIN – Human-Systems Interactions

Projet : 2D Character Controller

Objectifs : Le but de ce projet est de
- Programmer, tester et rendre paramétrables des interactions manette
- Concevoir des algorithmes de mouvements et de détections de collisions

Pré-requis : Cours IHM, Bonne connaissance Unity

Temps : du mardi 3/10 au dimanche 29/10/2017
- 3 séances de 1h45 encadrées + travail non-encadré

1. Prise en main (noté, individuel)

0. Connecter le gamepad Xbox (360 ou One) en usb.

1. Lire la doc de l'Input Manager (http://docs.unity3d.com/Manual/class-InputManager.html). Grâce au mapping figure 1, y ajouter ou modifier 8 axes pour représenter les 4 boutons A, B, X, Y puis le stick gauche et la croix directionnelle (donner les mêmes noms).

3. Créer une scène contenant une sphère blanche au centre. Créer un script qui la colore selon le bouton en cours d'appui (blanc quand relâché), et qui la translate dans le plan de l'écran avec le stick gauche ET la croix.

4. Générer puis tester un build (webgl ou exe windows). Archiver les fichiers générés dans un fichier "[JIN-IHM-TP1] Nom.zip" et le déposer sur le moodle (heure de dépôt prise en compte dans la notation)

5. Se servir de la scène pour tester les effets de différents paramètres de l'Input Manager.

Figure 1 :
Mapping Unity-Xbox 360 controller (http://wiki.unity3d.com/index.php?title=Xbox360Controller)

NB : Attention au mapping avec le contrôleur Xbox One potentiellement différent selon la version de windows utilisée…

2. Mini-Projet : 2D Character Controller (noté, binôme)

L'objectif de ce projet est de programmer le contrôle d'un game object basique dans un environnement type jeu de plateforme 2D, sans utiliser les packages unity (pas de character controller) et le moins possible le moteur physique (tests de collision et lancers de rayons uniquement).
L'interface visée principalement sera la manette mais le projet devra être jouable facilement au clavier (via l'abstraction des axes/boutons de l'Input Manager).
L'environnement visuel sera volontairement simple. L'accent sera mis sur les fonctionnalités de contrôle/mouvement et les réglages disponibles pour un game designer dans l'inspector unity.

2.1. Scène

Le level design doit mettre en valeur vos algos et vos réglages de contrôle, pas votre qualité de designer ; attention à ne pas faire preuve de trop d'imagination/difficulté qui nuiront aux tests (ex. figure 2)…

Type	2D ou 3D
Caméra	fixe et orthographique
Objet contrôlé	un carré
Autres objets	a) fixes, à géométrie simple alignée sur les axes : sols infranchissables, murs, plateformes traversables… b) pentes et plateformes mobiles seront autorisées dans un 2nd temps
Aucun ajout	pas de son, pas d'effets visuels (objets rigides, pas d'animation, matériaux simples)

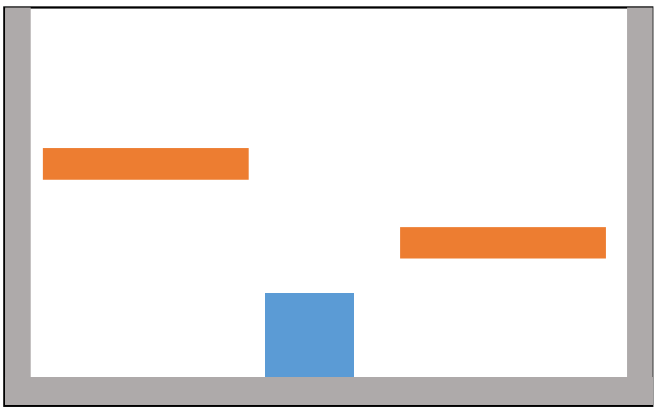

Figure 2 : exemple de 1ère scène de travail.

2.2. Contrôles de base

Déplacement horizontal

Contrôles	Stick gauche + croix + flèches du clavier
Principe de base	Contrôler la vitesse avec les axes, en déduire la position
Variations et réglages possibles	Valeur de la vitesse maximale* Vitesse maximale atteinte immédiatement ou au bout d'un temps d'appui ? Inertie lors de l'arrêt ? Boost de demi-tour ? Sprint (autre bouton) ? Dash (autre bouton) ? Input Manager Unity

Saut (et déplacement en saut)

Contrôles	Bouton A + touche espace
Principe de base	Impulsion (= vitesse) unique vers le haut puis la gravité s'applique jusqu'à ce que l'objet retrouve un sol Possible uniquement si l'objet est sur un sol
Variations et réglages possibles	Valeur de l'impulsion* Valeur de la gravité* Une seule valeur d'impulsion/hauteur de saut ou bien dépendante du temps d'appui ? Contrôle de la direction en cours de saut ("air control") ? Vitesse horizontale différente au sol et en l'air ? Inertie horizontale : retombe droit ou en parabole ? Double saut Wall Jump (2 murs uniquement) ? Input Manager Unity

Collisions verticales et horizontales

Principe de base	Déplacement/saut bloqués vers le haut et sur les côtés s'il y a un obstacle Vérifier s'il y aura collision dans la prochaine frame (ne pas attendre d'être en collision) en connaissant la position, la vitesse et l'environnement Si oui positionner au bon endroit
Variations et réglages possibles	Type particulier des plateformes qui bloquent uniquement vers le bas Tolérance de distance au sol pour pouvoir ressauter ? Tolérance de temps depuis le dernier contact pour pouvoir sauter ?

Autres

Variations et réglages possibles	Sol rebondissant Déplacement visqueux, venteux… Frottement de descente contre un mur

2.3. Réglages

Vous fournirez dans l'inspector les outils nécessaires à un game designer pour paramétrer vos contrôles. Les réglages notés * dans la partie précédente sont obligatoires.

2.4. Notation

Rendu pour le **dimanche 29 octobre 2017 20h00** = dépôt sur moodle d'un dossier zippé à vos 2 noms comportant :

- L'ensemble du projet Unity, nettoyé des ressources et dossiers inutiles (library, obj, temp, *.pdb…)
- Un build webgl/html5 fonctionnel (testé avec des manettes !) que je mettrai directement en ligne
- Un court document pdf récapitulant les fonctionnalités disponibles et les valeurs exactes de vos réglages (ex : vitesses, gravité, tolérances, inertie…)

NB : La version principale du programme que vous fournirez ne comportera aucun artifice visuel ou sonore : le seul critère sera la jouabilité manette/clavier et les fonctionnalités fournies.
Vous pourrez fournir une 2ème version comportant des améliorations de toutes natures (level, visuel, audio…) qui vous servira pour votre cv.

La **notation** portera principalement sur :

- les contrôles de base
- les fonctionnalités plus avancées
- les réglages possibles pour un game designer dans l'inspector

Elle sera complétée par un **bonus de jouabilité** donné par un panel d'utilisateurs (intervenants JIN…) à partir de la version html5 en ligne
La notation ne portera pas sur

- la beauté du code
- la version plus esthétique

3. Autres exercices pour s'entrainer sur d'autres problématiques

Voir JIN-IECb 2015-2016, TP manette.