Nom : …… **Prénom :** ……

Promotion : ……

Programmation C#/Unity3D

Évaluation des connaissances – Semestre 1

Partie 1 : QCM *(10 points – 20 min)*

Cochez la ou les bonne(s) réponse(s) parmi les affirmations suivant chaque question.

1. Unity3D est :
    - Un logiciel de création de jeux vidéo
    - Une marque d'imprimantes 3D
    - Un langage de programmation

2. Unity3D permet :
    - De créer des jeux mobiles
    - De créer des jeux pour PC
    - De créer des jeux consoles

3. Unity3D ne permet pas, **depuis le logiciel** :
    - De modifier du code
    - D'acheter des assets (images, modèles, code)
    - De modifier les données du jeu alors qu'il est lancé

4. L'interface est découpée en plusieurs onglets. L'onglet qui permet de placer les éléments dans l'espace, de les sélectionner, de les manipuler est :
    - Scene
    - Game
    - Inspector

5. L'onglet qui permet de voir les erreurs dans ses scripts ou quand le jeu est lancé est :
    - Inspector
    - Project
    - Console

6. Chaque élément dans la scène est obligatoirement :
    - Un Component (composant en français)
    - Un Shader
    - Un GameObject

7. Un GameObject…
    - A toujours au moins un composant Transform
    - A toujours un composant pour être affiché (SpriteRenderer en 2D)
    - Ne peut pas avoir plus de 20 composants

8. Un composant Transform sur un GameObject :
    - Définit la position, rotation et taille dans l'espace
    - Définit le nom de l'objet
    - Définit l'image à afficher

9. Si un GameObject **E** à un autre GameObject **P** comme parent dans la hiérarchie : (**E** est donc enfant de **P**)
    - Si **P** bouge, tourne ou change de taille alors **E** fera de même
    - Si **E** bouge, tourne ou change de taille alors **P** fera de même
    - **E** ne peut pas avoir un autre parent en plus de **P**
    - **P** ne peut pas avoir d'autres enfants que **E**

10. Une image 2D affichée à l'écran s'appelle :
    - Un sprite
    - Un mesh
    - Un shader

11. Pour gérer l'ordre d'affichage d'images à l'écran, on peut :
    - Modifier la position en Z des éléments
    - Modifier le Sorting Layer (calque) d'une image
    - Ordonner les objets dans la hiérarchie (le plus en haut est le premier affiché, et ainsi de suite)

12. Pour qu'un objet soit utilisé par le moteur physique **3D**, il faut :
    - Un rigidbody
    - Un rigidbody 2D
    - Un script

13. Pour qu'une collision ait lieu entre deux objets A et B, il faut **au minimum** :
    - Deux rigidbody (un sur A et un sur B) + un collider sur A
    - Deux colliders (un sur A et un sur B) + un rigidbody sur A
    - Un rigidbody sur A + un collider sur B

14. Un modèle sauvegardé et réutilisable de GameObject s'appelle :
    - Une frame
    - Un prefab
    - Une build

15. Au sujet de la caméra dans un jeu Unity3D :
    - Il peut y avoir plusieurs caméras dans la scène
    - La caméra est objet spécial qui n'apparaît pas dans la hiérarchie
    - La caméra ne peut pas être enfant d'un autre GameObject

16. Si la caméra est configurée en « orthographic », alors :
    - Plus un objet est loin plus il est petit
    - Les objets ont tous la même taille, peu importe la profondeur
    - Seuls les objets 2D sont affichés

17. Je suis une fraction de jeu totalement géré par Unity3D
    J'existe entre 30 et 60 par secondes dans un jeu (en général),
    Je regroupe l'exécution du code, la modification du jeu et l'affichage à l'écran
    Je suis-je suis :
    - Une asset
    - Une frame
    - Une texture

18. Un script C# pour Unity3D c'est :
    - Un fichier texte sur le disque
    - Un programme exécutable (.exe)
    - Un unique fichier qui contient tout le code de mon jeu

19. Un script C# :
    - N'est utilisé que s'il est ajouté comme composant d'un GameObject
    - Ne peut être utilisé que sur un seul objet à la fois
    - Peut avoir dans son code tout ce qu'il est possible de faire à la main dans le logiciel

20. Parmi les jeux suivants, lesquels sont fait avec Unity3D :
    - Overwatch (Blizzard)
    - Angry Birds (Rovio)
    - Steredenn (Pixelnest Studio)

Partie 2 : Projet Unity *(10 points – 60 min)*

Réalisation d'un petit jeu sous Unity3D.

Votre travail devra être rendu dans le format attendu, expliqué à la fin de ce document. Le non-respect de ce format entraînera une perte de points.

1. Créez un nouveau projet Unity pour un jeu 2D

2. Téléchargez le pack d'assets à l'URL **https://goo.gl/Eapd8r**
   Vous devez avoir 5 fichiers :
   - Un décor d'arrière-plan **background-far**
   - Un décor pour le premier plan **background-close**
   - Un joueur tyrannosaure **player**
   - Un morceau de viande **meat**
   - Un son **eat**

3. Importez tous ces fichiers dans Unity

4. Construisez un décor avec les images fournies *(0.5 pt)*

5. Ajoutez le joueur, un Ptérodactyle. *(0.5 pt)*

6. Faites en sorte : *(1 pt)*
    - que le joueur soit toujours affiché par-dessus l'eau et les montages
    - que l'eau soit affichée par-dessus les montages

7. Créez un nouveau script C# «PterodactylScript» et ajoutez le au Ptérodactyle. *(0.5 pt)*

8. Ajoutez un déplacement **horizontal** (gauche/droite) et **vertical** (haut/bas) au joueur dans votre script C#. *(1 pt)*
   (indices : Input.GetAxis(« Horizontal ») et Input.GetAxis(« Vertical ») dans Update)

9. Ajoutez un **Rigidbody2D** et un **Box Collider 2D** au Ptérodactyle.
   Désactivez la gravité *(0.5 pt)*

10. Ajoutez un morceau de viande à l'écran *(0.5 pt)*

11. Ajoutez un **Box Collider 2D** au morceau de viande en mode **Trigger**. *(0.5 pt)*

12. Sauvegardez le morceau de viande comme **prefab.** *(0.5 pt)*

13. Ajoutez plusieurs morceaux de viande **utilisant le prefab** précédent *(0.5 pt)*

14. Modifiez **PterodactylScript** pour que les morceaux de viande soient **détruits** quand le joueur les touche. *(1 pt)*

    Indices :

    1. *OnTriggerEnter2D(Collider2D c)*
    2. *Destroy(c.gameObject)*

15. Modifiez **PterodactylScript** pour ajouter un paramètre de **vitesse** pour le déplacement du joueur. *(1 pt)*
    - Ce paramètre doit pouvoir être **modifié depuis Unity3D**.
    - La vitesse doit pouvoir être différente en **x** et en **y**

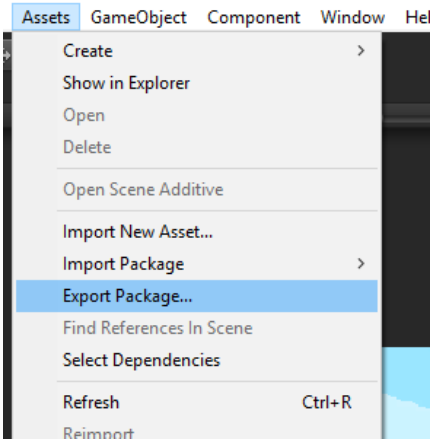
16. Jouez un son quand la viande est mangée *(2 pts)*
    - Ajouter une variable publique de type **AudioClip** au **PterodactylScript**
    - Dans l'inspecteur, remplir la nouvelle case qui s'affiche sur le script du joueur avec le son **eat** importés dans les assets.
    - Dans le script, jouer le son après la destruction de la viande
      Indice : AudioSource.PlayClipAtPoint(son, transform.position);

Comment rendre le projet Unity *(10 min)*

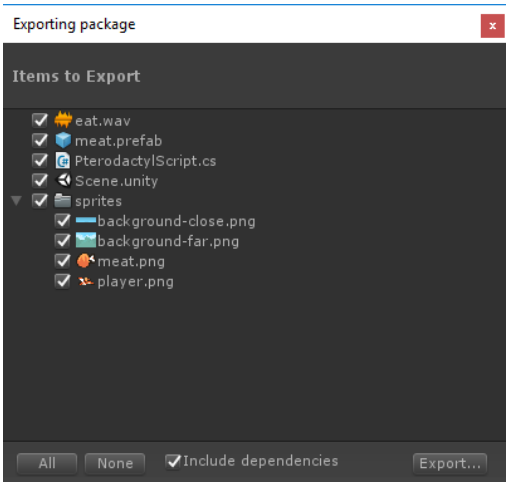
1. **ENREGISTREZ LA SCENE**

2. Assurez-vous de ne pas sélectionner d'éléments dans l'onglet Project (cliquez sur du vide si besoin)

3. Utilisez le menu Assets -> Export Package…



4. Vérifiez bien que tous vos fichiers sont sélectionnés pour l'export



5. Sauvegardez le fichier dans un endroit que vous pouvez retrouver.
   Le nom du fichier doit être
   <nom>_<prenom>_<promo>.unitypackage

   Exemple : mayance_damien_l1g4.unitypackage

6. Envoyez-moi le fichier par e-mail à
   ens.mayance@ism-laval.net

7. N'oubliez pas de me rendre le QCM papier !