TD 2 - Prototype d'Expérimentation

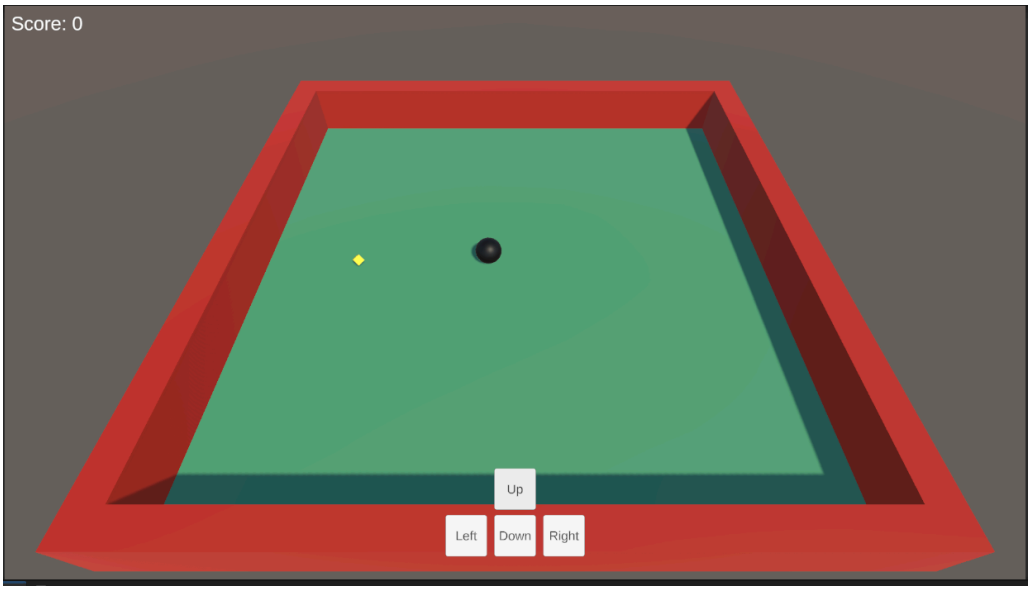
Dans ce TD, vous utiliserez le projet que vous avez implémenté dans le TD 1 (Roll-A-Ball) afin de le transformer en un prototype pour une expérimentation sur les interactions humain-machines.

La tâche de cette expérimentation sera de récupérer les 12 objets en déplaçant la bille.
Le principe de cette expérimentation est de tester deux types d'interactions pour déplacer la bille:

- Déplacement classique avec le clavier
- Déplacement via la souris et une interface.

Pour cela, le TD se découpera en plusieurs étapes pour l'implémentation:

1. La deuxième interaction
2. Déroulement de l'expérimentation
3. Récupération des mesures (temps pour effectuer la tâche)



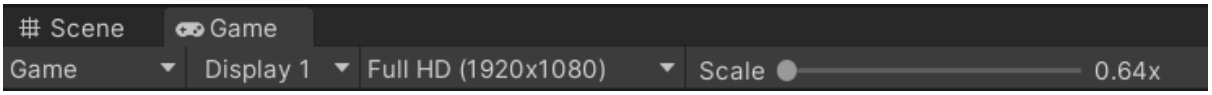
Partie 1 - Interaction via l'Interface

Pour cette partie, vous allez implémenter un mode de contrôle alternatif pour la balle.
Au lieu d'utiliser les flèches directionnelles du clavier, l'utilisateur devra utiliser 4 boutons sur l'interface pour appliquer une force à la balle.

Etape 1 - Créer l'interface

Dans le `Canvas` qui gérait le score précédemment, créez quatre `Buttons`. Comme pour le Texte, vous pouvez utiliser les `Buttons` du menu UI/Legacy.
Placez-les où vous le souhaitez sur la Scène, mais assurez vous qu'il soient quand même pratique à utiliser.

Afin de valider leur placement, passez sur la fenêtre **Game**.



Avec cette fenêtre (par défaut en **Free Aspect**), vous pouvez ajuster la résolution dans laquelle vous voulez lancer votre jeu. Choisissez une résolution (1920x1080 est plutôt standard) et ajustez votre interface en fonction.

Etape 2 - Implémenter le mouvement

Dans le script qui gère le déplacement de la balle, vous avez simplement besoin d'ajouter deux méthodes qui gèrent respectivement, le déplacement horizontal et le déplacement en profondeur (sur l'axe z) de la balle.
Ces méthodes vont prendre en input, un float, que l'on va pouvoir ajuster directement dans l'Inspecteur du Button**.**

Vous pouvez implémenter ces méthodes exactement de la même manière que pour le mouvement précédent, en appliquant une force sur le RigidBody sur la balle.

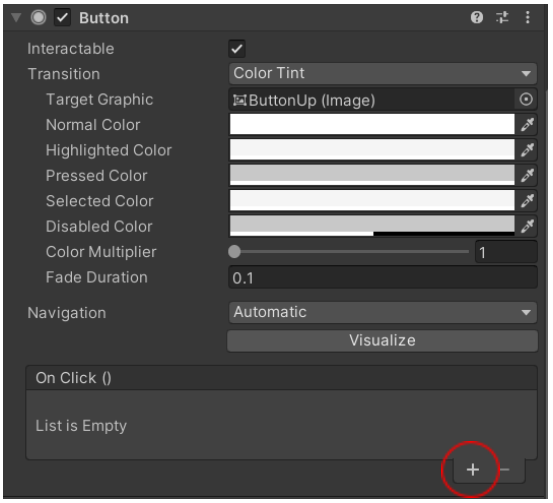
N'oubliez pas d'ajouter une condition pour activer/désactiver le mode de contrôle classique et d'ajouter une interface pour indiquer que le participant sache quel mode de contrôle est actuellement utilisé (via du texte ou autre)/

Etape 3 - Lier les méthodes et buttons

Pour lier les méthodes que vous avez créées au clic d'un bouton, vous n'avez pas besoin de réaliser un script.
En utilisant l'**Inspecteur** du GameObject qui contient le Button, vous pouvez cliquer sur le "+" de la fenêtre **"OnClick"** pour ajouter une action à effectuer quand le bouton sera cliqué.
Il vous faudra:

- Spécifier le GameObject (en le faisant cliquer-glisser) qui contient le Component avec la méthode à effectuer.
- Indiquer la méthode à effectuer.

Vous pouvez également définir un input pour cette méthode (si elle en possède un).



Partie 2 - Déroulement de l'Expérimentation

Pour notre étude, les participants devront **déplacer la bille pour atteindre chaque objectif.**
Cependant, si nous voulons évaluer uniquement la méthode de contrôle de la balle, et non la stratégie employée pour la récupération des collectibles, il va falloir **contrôler le déroulement de l'étude.**

Plutôt que de faire apparaître tous les cubes au lancement de l'expérimentation, il serait intéressant d'en f**aire apparaître un premier, puis de conditionner l'apparition du suivant au ramassage du précédent.**
On pourrait ainsi générer une séquence d'apparition qui serait la même pour chaque participant.

Encore une fois, cette partie peut se découper en plusieurs étapes:

1. Initialisation des variables de la séquence
2. Implémentation de la séquence
3. Ajout d'une phase de prise en main

Vous allez transformer le script `GameManager` qui gérait le score et la condition de victoire de Roll-A-Ball, pour qu'il gère maintenant le déroulement de l'expérimentation.

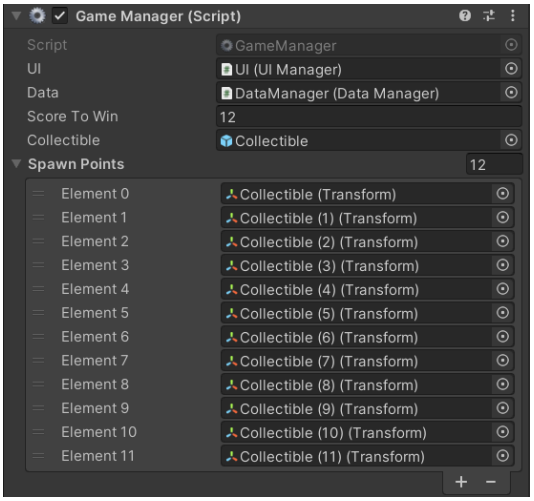
Etape 1 - Initialisation des variables

Vous aurez besoin des variables suivantes dans ce script:

- Une référence au `Prefab` du cube à collecter.
- Une liste des 12 positions auxquelles les cubes devront apparaître.

Pour le premier, un `Prefab` peut simplement être référencé comme un `GameObject`, puis cliquer-glisser dans l'**Inspecteur** depuis la fenêtre des `Assets`**.**

Pour le second, il existe plusieurs méthodes pour réaliser cette liste. Vous pourriez tout simplement récupérer les coordonnées de chaque position pour les input dans cette liste.
Une autre méthode consisterait à créer des `GameObjects` vides et les placer à chaque position.
Un `GameObject` vide ne l'est, en réalité, pas vraiment. Chaque `GameObject` possède un `Transform` qui contient des informations comme sa position, sa rotation et sa taille. (Doc)
En créant une liste de `Transforms`, puis en référençant chaque `Transform` des `GameObjects` vides que l'on a créé, on peut ainsi contrôler la position de chaque point d'apparition simplement en les déplaçant.



Etape 2 - Implémentation de la séquence

La première méthode à implémenter, va être celle pour faire **apparaître un collectible.**
Cette méthode va utiliser la méthode ***Instantiate*** de MonoBehaviour **(Doc).** Vous devez simplement lui indiquer le GameObject a instancier (ici, le Prefab du collectible). Vous pouvez également définir sa position dans le ***Instantiate***, ou juste après.
Pour sa position, vous devez **trouvez le moyen d'instancier les collectibles en séquence**, et donc de parcourir la liste des positions définies précédemment.

Cette méthode va devoir ensuite être appelée au lancement de votre étude (pour faire apparaître le premier cube) et à chaque fois que le participant ramasse un cube jusqu'au dernier.

Etape 3 - Implémentation de prise en main

Il est important dans une étude, de ne pas lancer le participant directement dans cette dernière pour ne pas biaiser les résultats par rapport à la prise en main des commandes.

Vous devez donc **implémenter une phase similaire à votre étude.** Pour cela vous pouvez reprendre les étapes précédentes, mais au lieu de faire ramasser les cubes dans les 12 positions. Instanciez-les dans les 4 coins de l'arène.

C'est également dans cette phase qu'**un mode de contrôle sera choisi automatiquement par le programme.**

Il est important de ne pas transitionner automatiquement entre cette phase de prise en main et l'étude, mais plutôt quand le participant sera prêt.
A la fin de la séquence de prise en main, ajoutez un bouton qui lancera la séquence de l'étude

Partie 3 - Récupération des mesures

Nous souhaitons obtenir **le temps passé à récupérer chaque cube** pour déterminer quelle méthode est la plus rapide.
Il va nous falloir un nouveau script, qui va gérer l'acquisition des données.

Créez donc un script DataManager, et placez-le sur un objet vide de votre **Scène**. Ne placez pas tous vos scripts "Manager" sur un seul objet vide pour des raisons de lisibilité et de bonne pratique.

Ce script contiendra une liste vide de **float** qui va nous servir à stocker les valeurs de notre étude.

Ensuite, il faudra créer une méthode qui va **calculer le temps pour ramasser chaque collectible et ensuite l'ajouter dans la liste**. L'idée derrière cette méthode est de faire attention de calculer le temps, soit depuis le début de l'étude (pour avoir le cumul), ou depuis le dernier ramassage. L'utilisation de Time.time devrait vous être utile (Doc).

Vous pourrez ensuite appeler cette fonction au moment où un objet sera ramassé.

Bonus - Votre propre étude

Maintenant que vous avez réalisé le prototype pour cette étude, à votre tour de proposer une étude!

Il existe beaucoup d'autres interactions que vous pouvez étudier dans ce contexte et beaucoup d'autres types de métriques que vous pouvez extraire de ce type d'expérimentation.

Pour cette partie:

- Proposez et implémentez un autre type d'interaction.
- Proposez et implémentez une autre mesure à extraire.