Raclétouille

version 1.0.0

DIVERSITY by EPITECH

I. Introduction

Sur les hautes montagnes des Alpinios, un célèbre restaurant de haute montagne Penguin-Grill prépare une raclette party. Cet évènement est le plus attendu de la région, Maurice le cuistot du restaurant était en train de préparer le festin fromager quand soudainement il surprend le clan des marmottes du 99 entrain de lui faire un mauvais tour.

Les marmottes ont jeté tous les ingrédients pour la raclette par-dessus le ravin et se retrouvèrent sur la montagne. Maurice voit sa raclette party s'envoler, mais il prend son courage à deux mains et dévale la piste pour essayer de rattraper les fromages et les pommes de terre tant convoités pour ce repas festif.

Aidez Maurice à sauver sa fête !

Maurice et ses amis à la poursuite des fromages

II. Consignes

- Lisez tout avant de commencer !
- Veuillez tout d'abord installer Unity. Vous retrouvez toutes les informations nécessaires dans les fichiers Unity – Installation et les premiers pas sur Unity que le Cobra vous fournira.
- Si tu bloques, rappelles-toi que tu es accompagné(e) ! Demande de l'aide à tes camarades ou à un Cobra. L'union fait la force !
- Internet est un outil formidable pour découvrir le fonctionnement des choses, Google est ton ami sers-t 'en régulièrement !
- Pour ce projet, il faut extraire l'archive fournie par les cobras.

Explication du code :

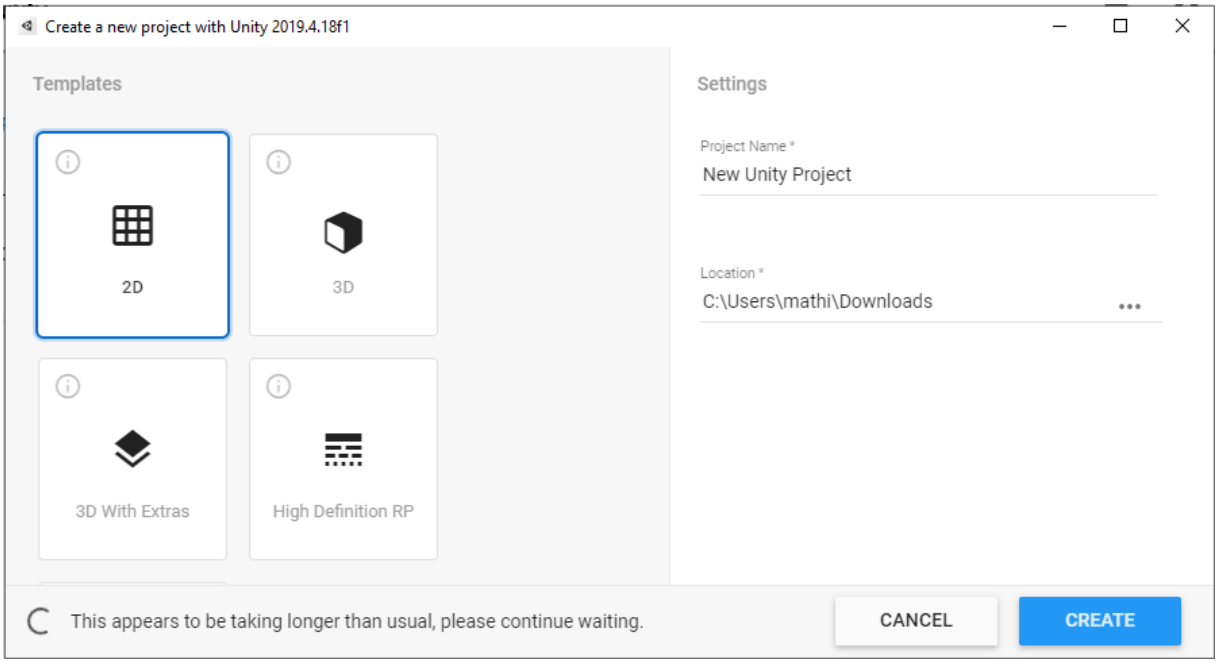
- Tout ce qui est écrit après un `//` est un commentaire pour t'aider à comprendre.

III. Mise en place de l'univers

a. Entrer dans un monde 2.0

Maurice dévale la pente mais rate tous les ingrédients et se prend tous les obstacles, dans un dernier effort il fait appel à Manigo, le grand maitre du temps. Manigo arrive à sa rescousse et lui dit : "Maurice ta raclette est partie en fumée, je vais te proposer une solution, tu dois utiliser Unity pour simuler ta descente afin de sauver ton festin, si cela fonctionne je te renvoie 10 minutes avant le drame et tu pourras reprendre le cours des choses... "

Manigo disparaît dans un épais brouillard, Maurice court chercher son ordinateur afin de sauver sa raclette party et ouvre un nouveau projet Unity. Maurice va devoir commencer par créer son nouveau projet. Pour cela, il ouvre Unity Hub et il appuie sur NEW. Dans la fenêtre qui s'est ouverte, il coche la case 2D et il donne un nom à son jeu : SOS Raclette.



Menu de sélection du type de projet

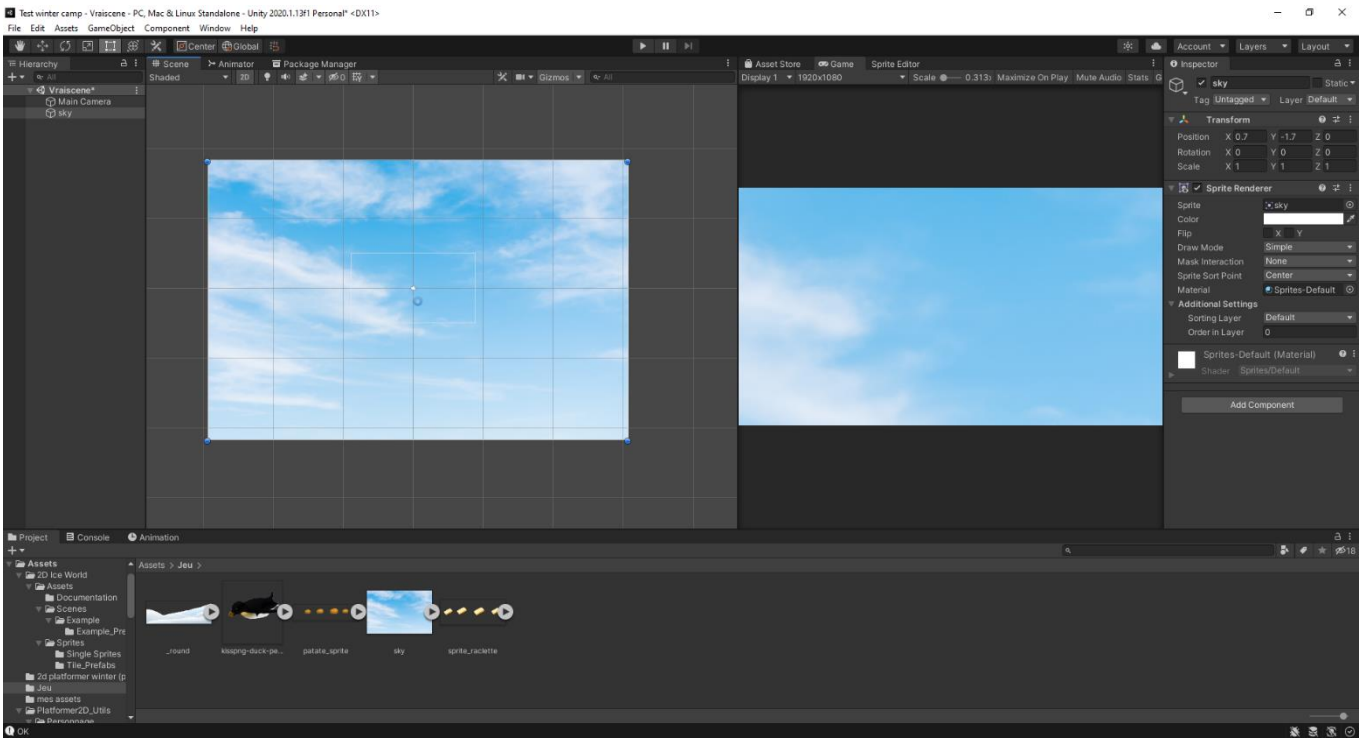
Une fois les changements effectués, il appuie sur CREATE. Unity va maintenant créer le projet, ce qui peut mettre un certain temps. Patience !

Poster qui se trouve dans la chambre de Maurice

b. L’Alpinos du monde virtuel

Dans un premier temps, insérez le contenu de votre fichier dézipé Assets_Raclétouille dans votre fenêtre Project->Assets.

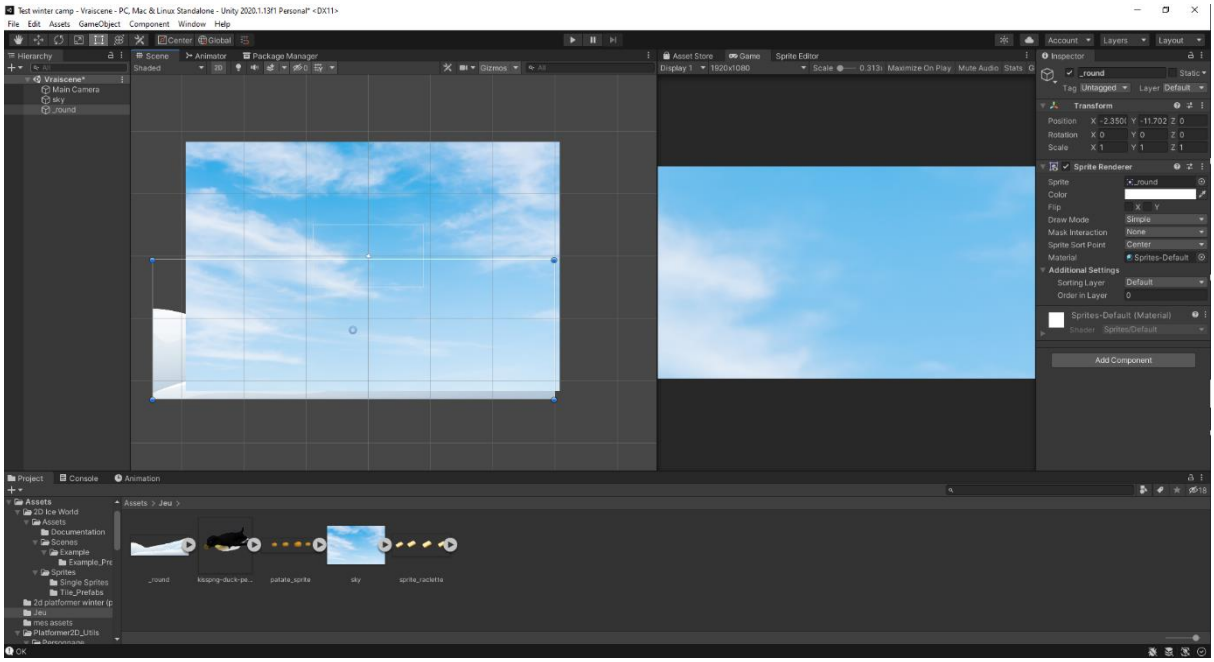
Maurice se retrouve sur Unity, et décide de chercher un fond qui ressemble à la couleur du ciel d’aujourd’hui, et il est bleu. Ça lui rappelle un tableau qu’il avait vu "Sky", ça représentait un ciel bleu avec quelques nuages, il retrouve une image ressemblant à cela dans son ordinateur. En se renseignant il comprend qu’il faut glisser déposer cette image dans la fenêtre "scène" créée précédemment.



Voilà à quoi ressemble l’écran de Maurice sur l’interface Unity

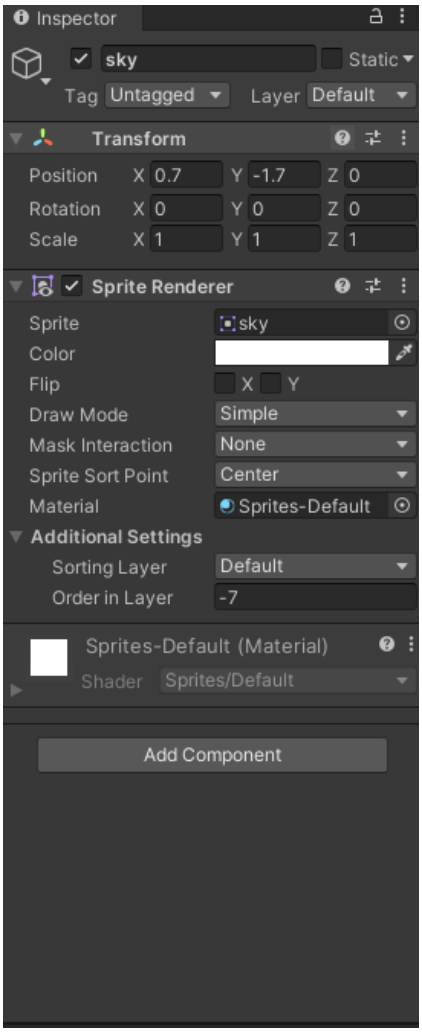
Maintenant que Maurice a ajouté le ciel, il faudrait peut-être redescendre sur Terre, mais comment atterrir en toute sécurité sans sol ?

Il se souvient d’une chanson qui lui disait "À la montagne gardes les pieds sur terre et pas dans le ciel car la neige est plus confortable qu’un nuage". Après avoir récupéré une photo du beau dénivelé "ground" de sa montagne, Maurice la glisse de la même manière que le ciel dans la fenêtre "scene" sur Unity.



Voilà à quoi ressemble l'écran de Maurice sur l'interface Unity

Mince ! Le ciel tombe sur la terre, il prend toute la place le sol s'est mis derrière le ciel. Vite, il faut réparer cela, il faut changer l'ordre d'apparition des images.

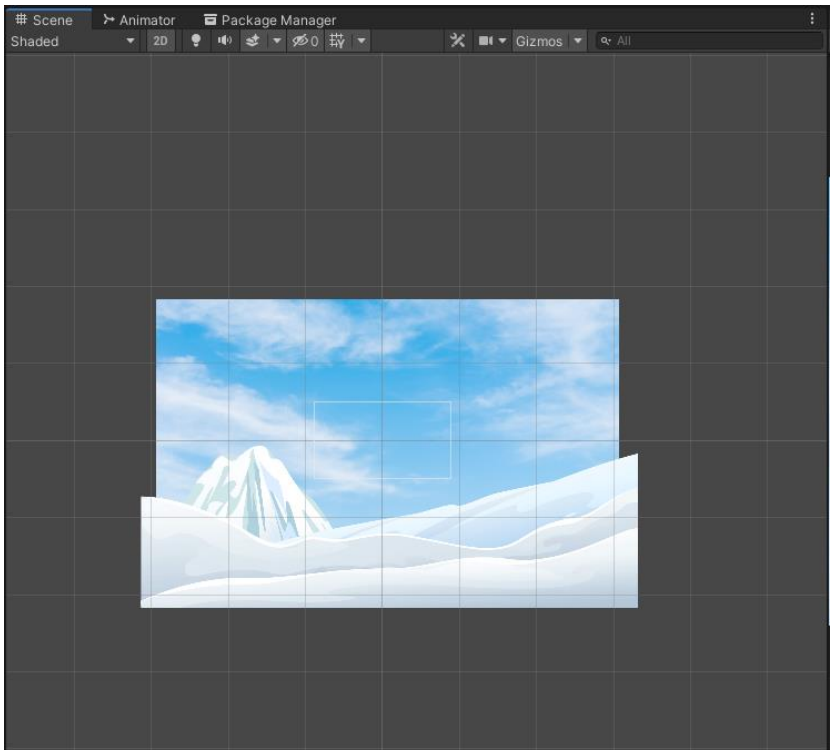


Interface Inspector de Unity

Pour cela on change le layer, c'est à dire l'ordre des couches. On passe le "Sky" à une valeur négative dans "l'Inspector" pour que ce soit toujours derrière les autres éléments qu'on ajoutera plus tard. Donc pour cela, on passe la valeur "Order Layer" à -7 et le ciel ne tombe plus sur le sol. Tout est à sa place maintenant !

On a le ciel, la terre, mais les Alpinos sont connues pour leurs montagnes non ? Alors maintenant, Maurice va en rajouter de belles couvertes de neige, celles de son enfance qui sont sur son ordinateur encore une fois ! On glisse donc l'image "montain" comme précédemment, dans la fenêtre "scene" entre "sky" et "ground".

Il n'a pas oublié ce qu'il a appris précédemment concernant l'ordre des couches dans l'inspecteur, pour ne pas avoir le même problème que tout à l'heure.



Le résultat de la scène créée par Maurice

Bravo vous avez réussi le niveau 1

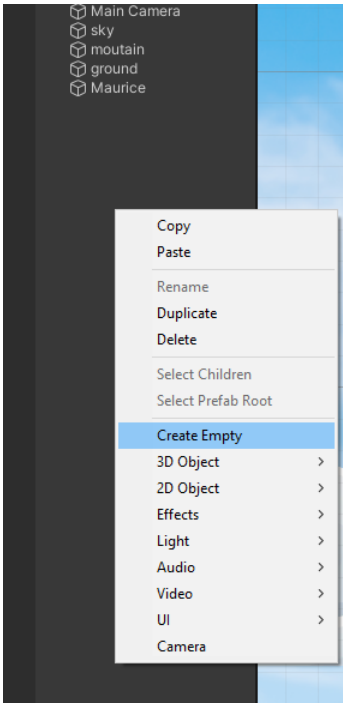
c. Clonage de Maurice

Maintenant que nous avons le décor, il faut que Maurice soit cloné dans le monde virtuel, il doit être présent dans ce jeu pour sauver le monde réel et les fans de raclette non ?

Il trouve une photo de lui dans un dossier "assets" et place celle-ci dans la scène qui a été créée. Maintenant que Maurice 2.0 se trouve sur l'écran, il faut rajouter la piste de ski permettant de dévaler la pente de la même manière que pour les montagnes.

En voilà, un bien joli pingouin !

Petite chose à savoir, Maurice est une personne qui adore le rangement et est un peu maniaque sur les bords, du coup quand il voit que son plan de travail sur Unity n'est pas rangé correctement il devient tout fou de ménage ! Par chance il existe "empty object" sur Unity qui permet de créer des objets vides mais également de ranger des éléments.



Avec cet outil il va trier un peu ce qui traine dans sa fenêtre, pour ce faire il déplace son curseur dans la fenêtre hiérarchie et utilise le clic droit de sa souris pour appuyer sur "Create Empty".

Une fois l'outil utilisé on se retrouve avec un objet vide, il faut donc le renommer "decor", puis on en va glisser "sky", "montain", et "ground" dans cet objet.

Menu de rangement des assets

d. On met ses skis et on dévale la piste !

Il est temps de comprendre pourquoi Maurice n'a pas pu dévaler la piste à temps, alors il lance une simulation cependant il n'y a pas encore de piste ! Mais comment faire... ? On va l'ajouter ce qui permettra de comprendre pourquoi le festin est partie en fumée ! Il faut donc l'ajouter directement de nos assets, attrapez l'image nommée "piste" et glissez la dans votre scène !

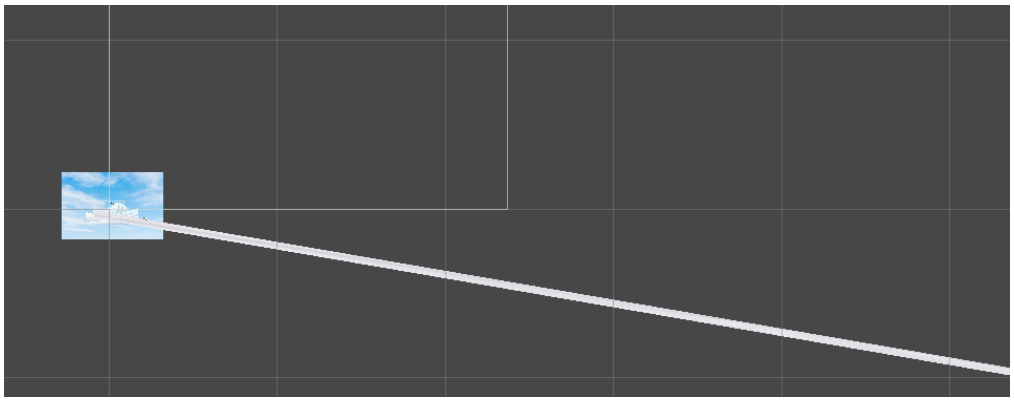
Une fois votre descente mise en place, n'oubliez pas de modifier le paramètre "order in layer" de votre piste, car elle aussi peut se retrouver derrière votre décor !

Menu pour choisir la couche d'apparition de la piste

Inclinez là afin qu'elle possède un vrai aspect de descente !

Rendu de la fenêtre Game

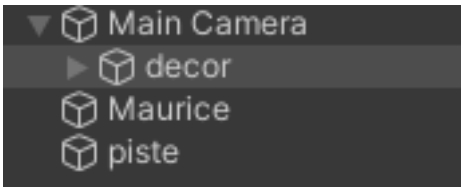
Comme vous vous en doutez, étirez là au maximum afin qu'elle soit bien plus fun à dévaler, il faut bien évidemment rester cohérant avec le fait que la piste soit descendante. Ce qui donne ceci :



Rendu final de la piste

Une vraie piste est faite pour glisser, maintenant il faut relier le joli décor en fond à la caméra, sinon le pingouin dévalera la piste sans fond, en effet, il restera statique et ne suivra pas le pingouin dans sa folle descente.

Pour ce faire, allez dans la fenêtre "hiérarchie" et glissez le décor sur la caméra mère, ce qui nous permettra d'avoir un suivi du décor avec la caméra. Le décor sera directement placé en dessous de "Main Camera".



Résultat du raccord du décor à la caméra

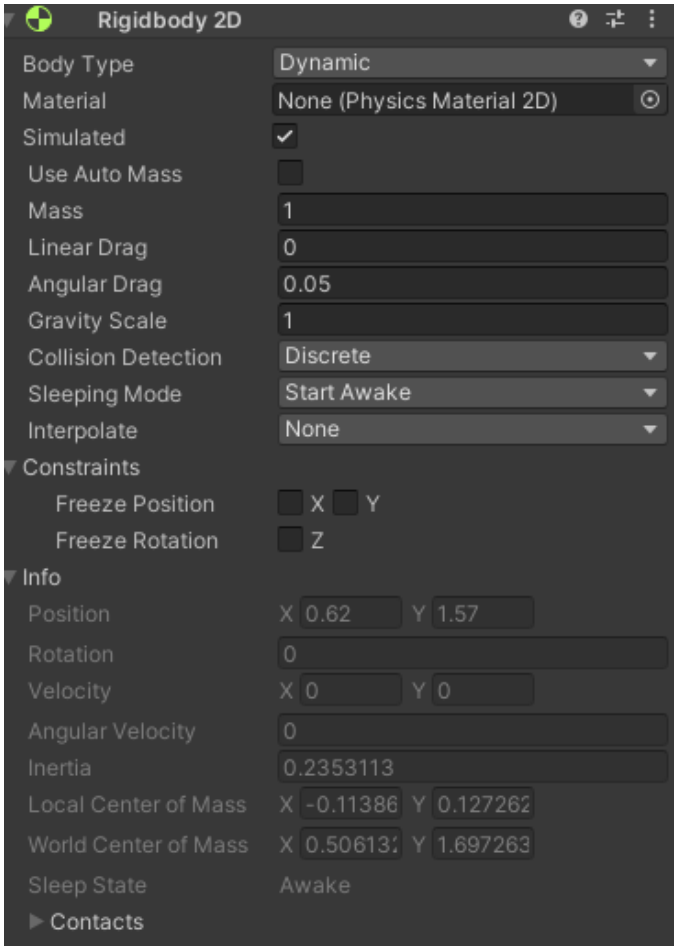
Bravo ! Continuez ainsi !

Image drôle

e. Gravité, ici et ailleurs : SOS Newtong !

Maurice se demande s'il est possible de démarrer la simulation ! Et si on commençait par appuyer sur "play" ? Mais que se passe-t-il, pourquoi il ne passe rien ? Comment va-t-on sauver la raclette party de Maurice …

Il ne se passe rien car il n'y pas de gravité sur son clone ! Il faut que le clone ait un nouveau composant. Ce composant est le RigidBody, il faut qu'il soit lié à notre clone pingouin. Appuyez sur "Add Component" tout en bas, et ajoutez un "RigidBody 2D".

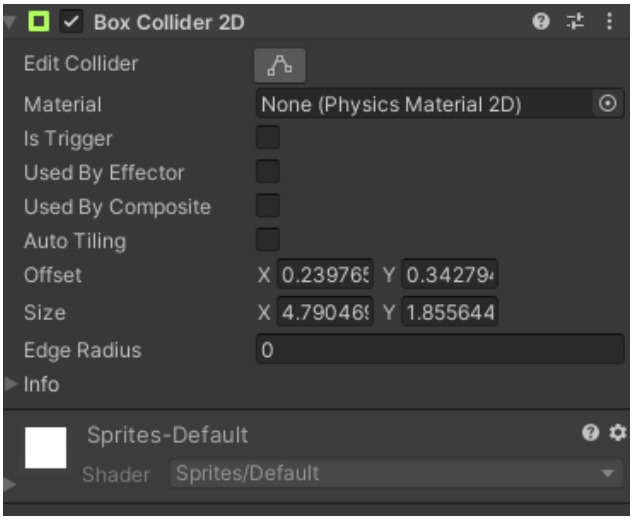


Détails du composant RigidBody de notre personnage.

Maurice s'empresse d'appuyer sur "play". Mais il semblerait qu'il tombe dans le vide avec le décor et la caméra, c'est incroyable, le clone ne peut se mouvoir correctement, mais pas de panique, il y a toujours une solution !

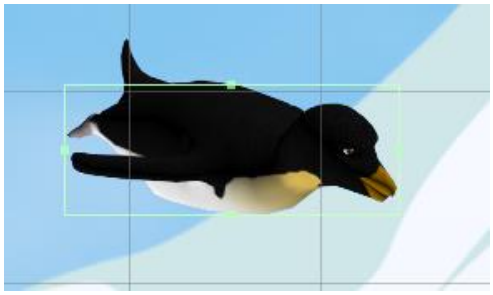
Et cette solution, il la trouve ici. Il va falloir rajouter une collision ou "collider" au pingouin et la piste. Le but ici est de délimiter la partie sur laquelle le personnage peut aller. Le "collider" du sol et celui du personnage vont se rencontrer ce qui l'empêchera de chuter car ils ne peuvent pas s'entrecroiser. Grâce à son "collider", il va donc se poser sur le sol.

Il faut donc dans un premier temps commencer par Maurice 2.0. Pour cela, dans la fenêtre "Inspector" du pingouin, ajouter un composant se nommant Box Collider 2D.



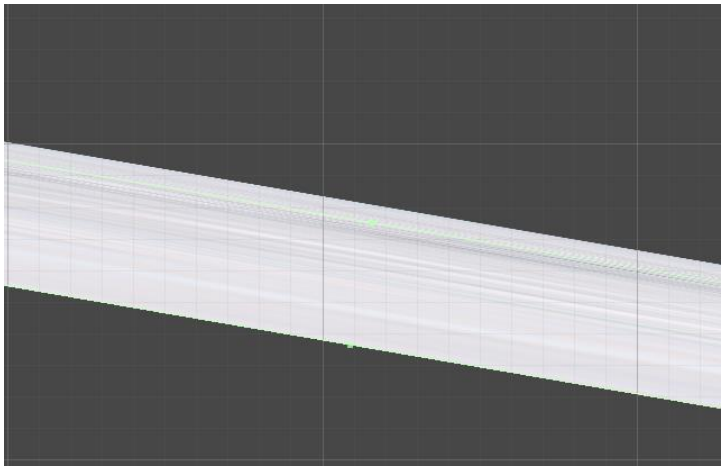
Menu Box Collider 2D

Une fois sur la fenêtre il faut éditer les collisions afin de l'adapter à notre personnage, pour cela il faut appuyer sur "Edit Collider". Le but est d'adapter le rectangle vert pour qu'il soit à la taille du personnage. Une fois que la collision est bien réglée rappuyez sur "Edit Collider" pour valider.



Maurico et son rectangle vert

Maintenant que cela est réalisé, le clone sera affecté par la gravité, cependant pour qu'il puisse s'arrêter ou interagir avec un sol par exemple, il faut ajouter un autre "Collider". Dans le cas de la simulation de Maurice, c'est la piste a également besoin de collisions, pour ce faire il faut cliquer sur la piste et rajouter aussi un composant "Box Collider 2D ".



La ligne verte est bien alignée sur la piste

Maurice appuie sur "Play", et ça fonctionne, son clone est bien positionné sur la piste !

Bravo vous avez réussi le niveau 2

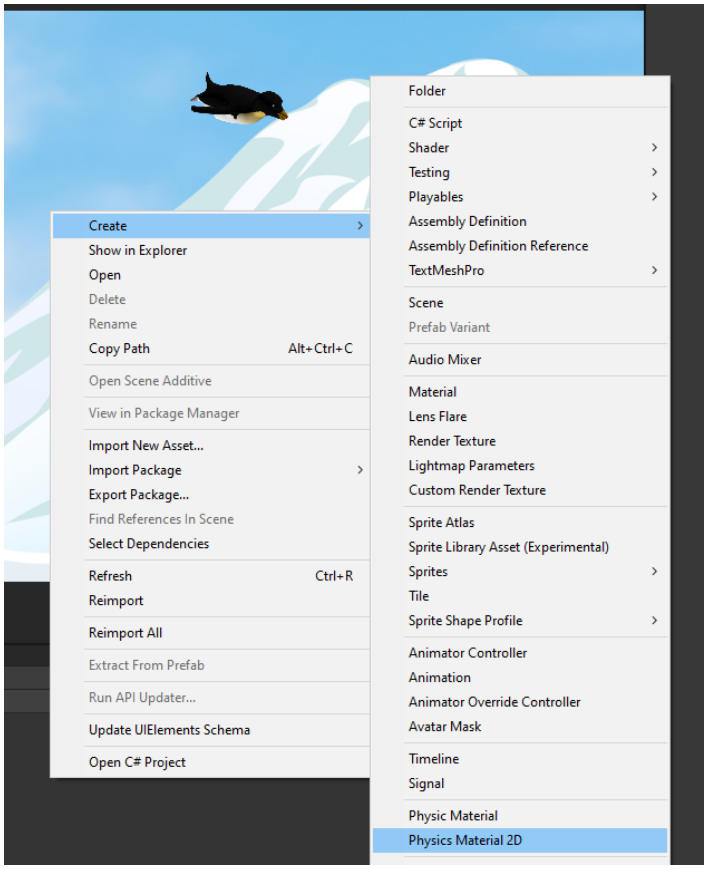
IV. En avant la glisse !

a. Qui a mangé mes mister freeze ?

Le clone suit une trajectoire mais il ne glisse pas sur le sol... Alors qu'en réalité, si on se tient sur un sol givré, on patine.

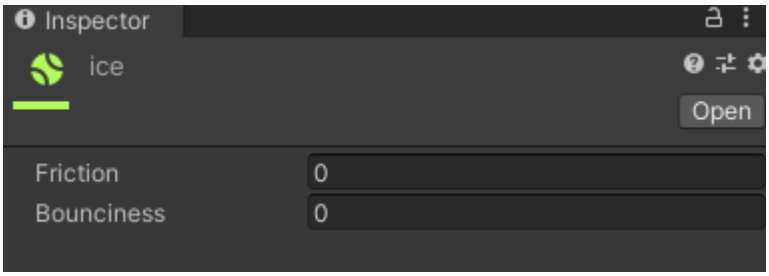
Pour la simulation soit la plus précise possible, il va falloir créer un "Physics Material 2D" qui va simuler le glissement des sols gelés.

Pour ajouter ce composant, il faut aller dans la fenêtre "project", cliquer droit dans les assets et créer à l'aide de la section "Create", le "Phyics Material 2D" en le sélectionnant dans la liste.



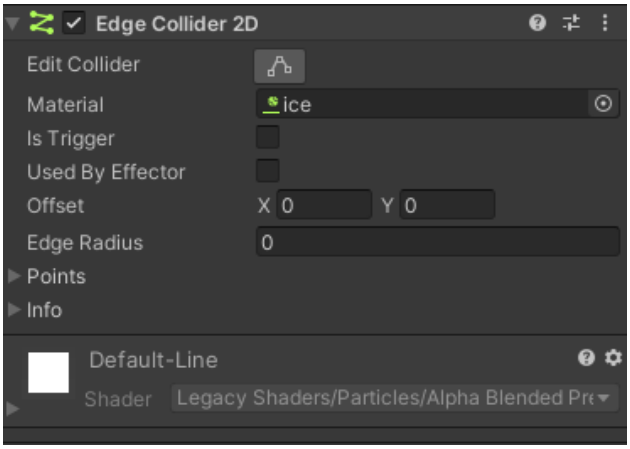
Chemin à parcourir pour créer un « Physics Material 2D »

Une fois le "Physics Material 2D" créé il faudra renommer l'élément généré en "ice", et mettre ses paramètres à zéro.



Menu « Physics Material 2D » nommé « ice » avec ses paramètres à zéro

Une fois cette modification réalisée, il faudra rajouter l'élément "ice" au "Collider" de notre piste de glisse. Pour ce faire, il faut aller dans "l'inspector" de la piste et ajouter ce dernier dans "Material". On peut soit l'ajouter en cliquant sur l'icône en forme de rond en bout de ligne, ou soit en le glissant des assets directement dans la case.



Menu du « Collider » avec le « Material » ice

Si on rappuie sur "Play", le pingouin glisse enfin sur la belle piste.

b. On a perdu le caméraman !

Maintenant que le Maurice 2.0 glisse, on se rend compte que la caméra ne suit pas, pour changer ça il faut écrire un petit script.

C'est quoi un script ? Un script est une succession de lignes de codes qui permettent d'automatiser une action. Ici dans ce cas, c'est Maurice qui doit être suivi par la caméra de façon automatique.

Il y a quelques temps, Maurice a utilisé des "empty", désormais on va avoir une nouvelle utilisation de ces derniers ! Il faut maintenant faire un clic droit dans le menu hiérarchie afin de créer un "Empty", grâce à "Create Empty". Une fois cela fait, il faut le renommer en "Camera Controller".

Menu hiérarchique

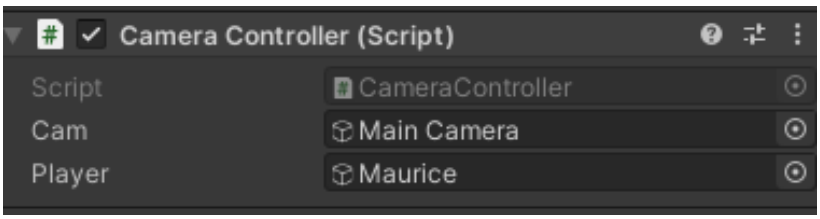
Ce nouvel objet va nous permettre d'y mettre le script, qu'on va créer dès maintenant ! Pour ce faire, allez dans "l'Inspector" de la "Camera Controller" et ajoutez un composant. Ce sera un "New Script", ce qui ouvrira un éditeur de code afin de coder tout ça !

Pour commencer, rajouter des variables publiques, cela permettra d'ajouter l'objet qui nous intéresse directement dans "l'inspector" afin qu'il puisse être utiliser dans le code par la suite. Il va falloir ajouter une variable pour la caméra qui va suivre le véritable Maurice et une autre pour le clone qui sera l'objet qui sera suivi.

A la suite des variables, insérer la fonction `private void FixedUpdate()`. Supprimez ce qui a été automatiquement généré et remplacez cela par l'exemple ci-dessous, il faudra également donner à la caméra, un vector3. Il faut veiller à ce que les majuscules et l'orthographe soient respectées, à la moindre erreur le script ne fonctionnera pas !

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CameraController : MonoBehaviour
{
    public GameObject Cam;
    public GameObject Player;

    private void FixedUpdate()
    {
        Cam.transform.position = new Vector3(Player.transform.position.x,
            Player.transform.position.y, Cam.transform.position.z);
    }
}
```

Maintenant que tout est prêt, il faut attribuer des objets dans "l'Inspector" de notre "Camera Controller". Puis dans l'option "Cam" on y met la "Main Camera" et sur "Player", on y met "Maurice".



Menu du script dans "L'inspector" de la caméra

C'est l'heure de tester ! Parfait la caméra suit bien notre Maurice National.

Félicitations ! Maurice est suivi par des paparazzis !

Photo typique d'un paparazzi !

c. L’activité physique régulière de Maurice

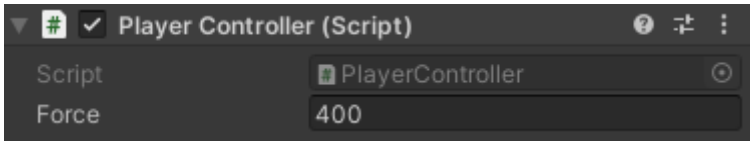
Lors de sa descente, il va devoir attraper le fromage à raclette et les pommes de terre perdues lors du hold-up de son restaurant, pour cela il va devoir effectuer un saut ! Pour sauter, il faudra créer un script rien que pour cet exploit !

Pour ce que Maurice 2.0 saute, il faudra appuyer la touche espace de votre clavier, ainsi il pourra s’élancer vers le haut.

Dans la fonction `Update()`, pour que le saut se fasse correctement il faut ajouter une condition ainsi lorsque la touche espace est enfoncée et que le clone touche bien le sol, il saute. Si ces conditions ne sont pas vérifiées, il pourrait sauter à l’infini…

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;
using UnityEngine.UI;

public class PlayerController : MonoBehaviour
{
    private Rigidbody2D _rigidbody2D; //utilisation du RigidBody de notre Maurice
    public float force = 400; // la force qu’on donne au saut
    private bool grounded; // variable si on touche le sol ou non

    private void Start()
    {
        _rigidbody2D = GetComponent<Rigidbody2D>();
    }
    private void Update()
    {
        if (Input.GetKeyDown(KeyCode.Space) && grounded)
        {
            Debug.Log(“SPACE”); //message qui apparait dans la console pour vérifier
qu’on a bien sauté
            _rigidbody2D.AddForce(Vector2.up * force); //action de sauter

        }
    }
    private void OnCollisionExit2D(Collision2D other)
    {
        grounded = false; //si ça ne touche pas le sol
    }
    private void OnCollisionEnter2D(Collision2D other)
    {
        grounded = true; //si ça touche le sol
    }
}
```

À présent il suffit de tester la nouvelle fonctionnalité, si Maurice saute trop haut ou s’envole comme un certain Super-Pinguin, on peut régler la force de saut directement dans "l’Inspector".



Détails du script de Maurice dans " l’inspector "

V. Il est temps de tout reprendre en main !

a. Une raclétouille sans fromage c'en est pas une !

Quel prouesse, Maurice est un rider maintenant, il est trop chaud, mais maintenant il va devoir ramasser le fromage et les pommes de terre provenant de son restaurant ! Pour cela, on va commencer par animer un sprite.

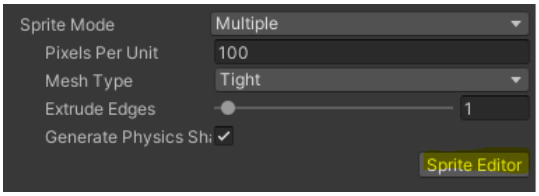
Mais qu'est-ce qu'un "sprite" ? C'est une image qui contient plusieurs fois le même élément dans des positions différentes qui, misent les unes après les autres, donnent une impression de mouvement.

Pour cela il faut cliquer sur l'image du fromage de raclette qui se trouve dans votre fenêtre Project dans les assets, qui se présente comme ceci :

Sprite de fromage à raclettouille

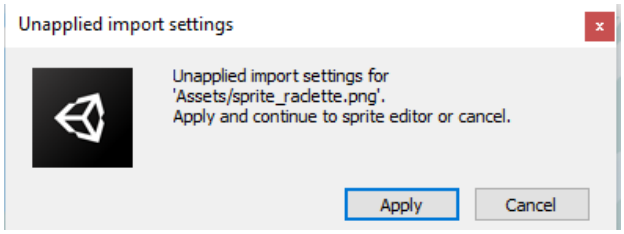
Il faut de nouveau se rendre dans "l'Inspector" pour modifier les paramètres du fromage et faire comprendre à Unity que l'on veut que l'image puisse subir des mouvements, et non l'afficher comme on a pu le faire avec d'autres éléments.

Pour faire cela, il faut dans un premier temps modifier le "Sprite Mode" en le passant en mode "Multiple" pour indiquer qu'il n'y a pas qu'une seule image dans cet élément. Le procédé est le suivant : on clique sur Sprite Editor, pour découper l'image.



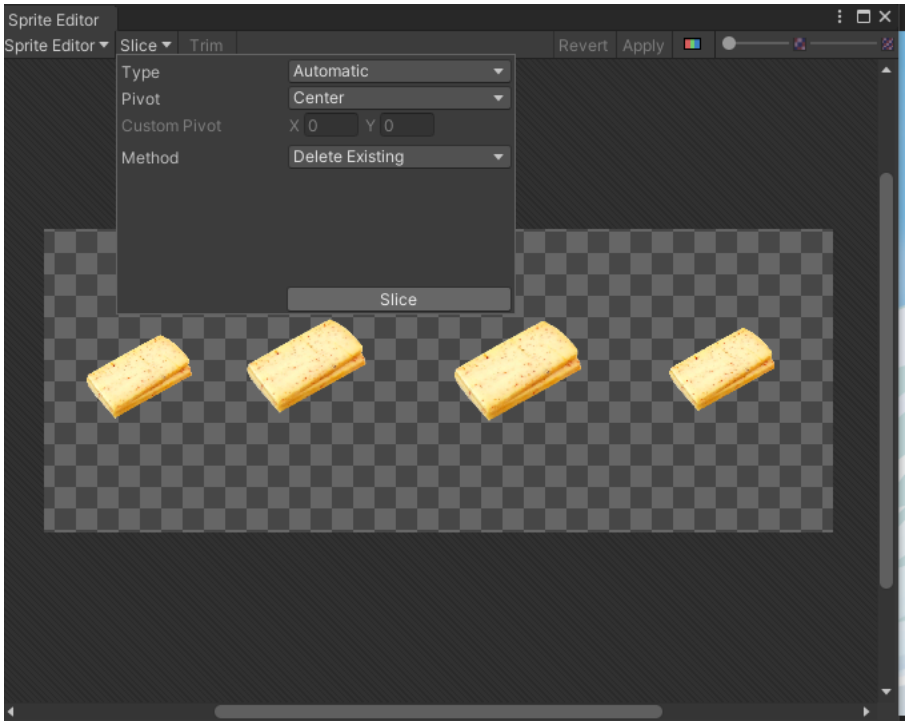
Menu « Sprite Editor »

Une fenêtre s'ouvre, il faut cliquer sur "Apply".



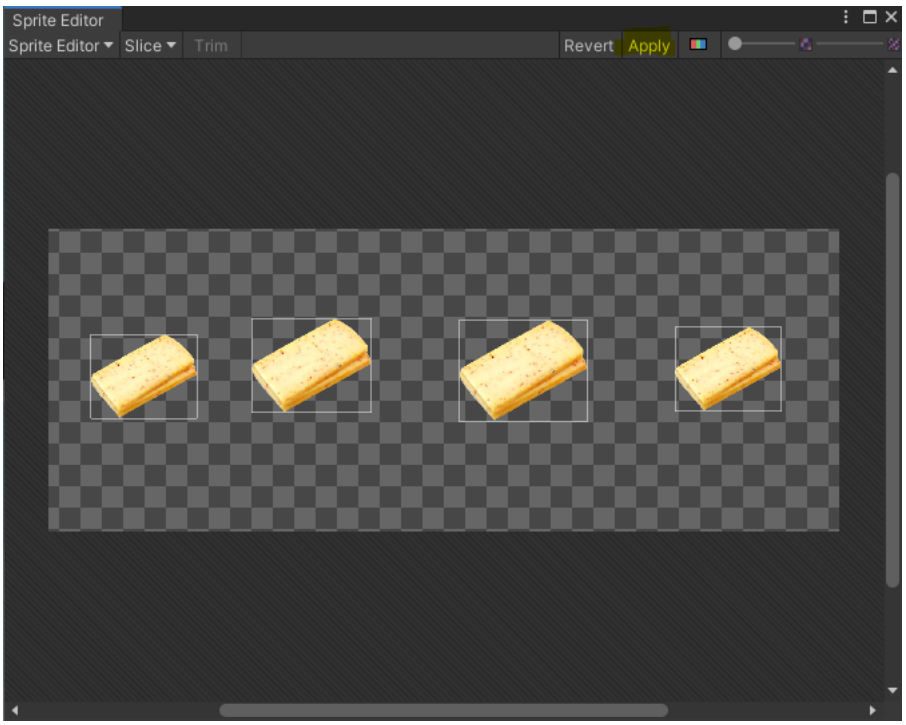
Fenêtre de validation des modifications

Une nouvelle fenêtre va s'ouvrir, il faudra cliquer sur l'option "Slice" en haut à gauche de celle-ci, puis confirmer en rappuyant sur "Slice" du menu récemment ouvert.



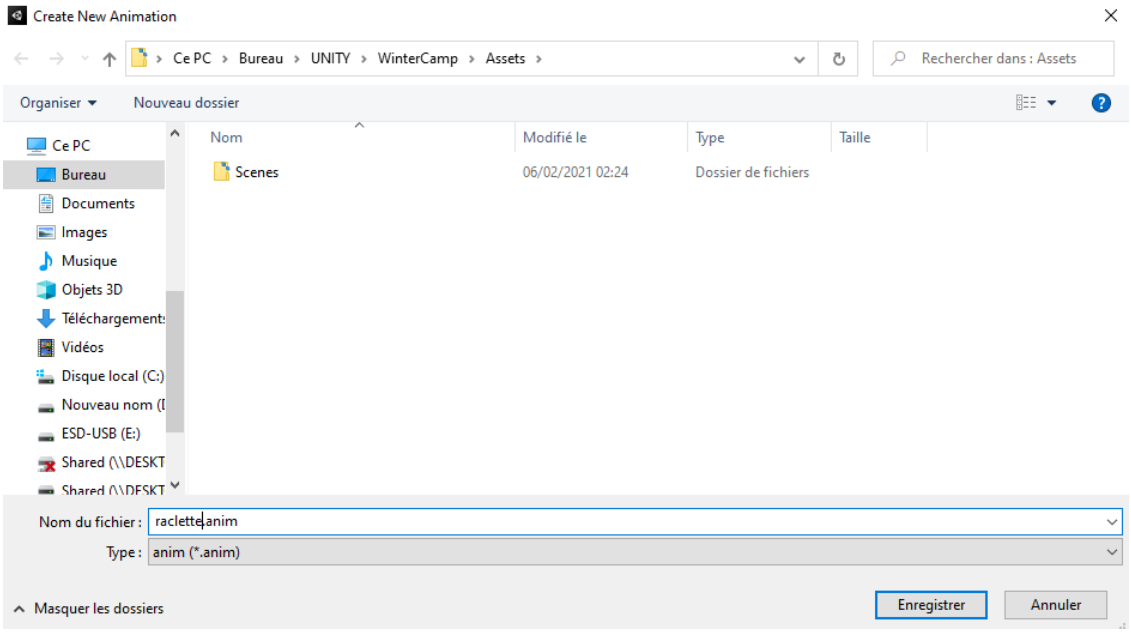
Editeur de Sprite, dans le menu « Slice »

A la suite de cette action, des rectangles vont apparaître autour de chacun des fromages, cela veut dire qu'ils ont été bien découpés ! Pour valider les modifications, il faut absolument cliquer sur "Apply", ensuite il n'y a plus qu'à quitter la fenêtre.



Editeur de Sprite

Maintenant que tout est modifié, il faut glisser notre image de fromages de "assets" jusqu'à la fenêtre "Scene" pour la rajouter dans le jeu. Une fois fait, une fenêtre s'ouvre pour enregistrer l'animation réalisée à l'aide du sprite, on la renomme et enregistre.



Interface Windows pour enregistrer l'animation

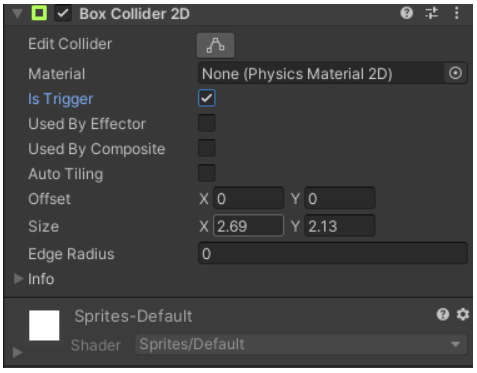
Une fois enregistrée, elle apparaît dans le jeu ! Il ne faut pas hésiter à régler la position si elle est derrière le décor et la redimensionner si elle est trop grande.

Position du fromage le long de la piste

Il est l'heure de tester, on appuie sur "Play" et hop ça bouge !

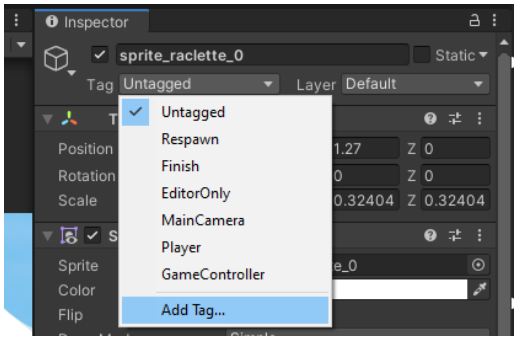
Il faut maintenant les ramasser, pour cela il faut que le fromage utilise l'outil ultime... Les "Collider", sinon le pingouin ne saura pas qu'il est rentré en contact avec l'élément !

Pour faire cela, on se rend dans "l'inspector" du fromage et ajoute un composant "Box Collider 2D", ce qui permettra au fromage de signaler sa présence. Cette fois on coche "is Trigger", ce qui permet d'indiquer qu'il n'est pas obligé de rester bloqué par cet objet, Maurice 2.0 pourra le traverser.



Menu « Box Collider 2D » du fromage

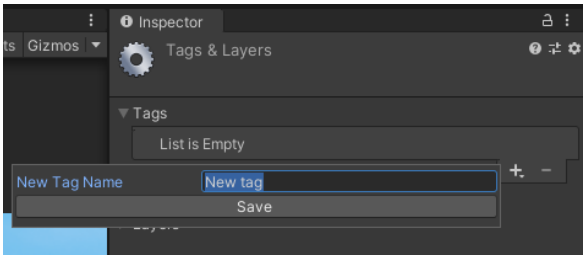
Afin que le personnage puisse interagir avec l'objet, on crée un tag, ce qui permettra de savoir à quelle catégorie il appartient. Mais comment faire cela ?



Ajout d'un Tag pour le Fromage

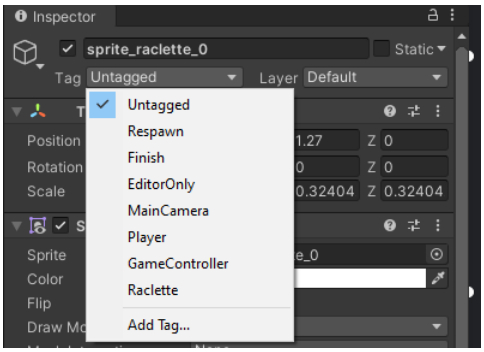
En allant dans "l'Inspector" du fromage, et cliquant sur l'option tag pour ajouter un Tag, on sélectionne l'option "Add Tag... " pour en créer un nouveau.

Une fenêtre apparaît permettant la création d'un nouveau Tag, on appuie sur "+", pour le nommer en "Raclette", par exemple.



Création du Tag

Quand on retourne dans la catégorie Tag du petit fromage, un nouveau nom est apparu, il suffira de le sélectionner.



Le nouveau Tag est dans la liste

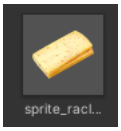
Mais bon, Maurice doit pouvoir ramasser ces fromages lors de son passage, alors il faut modifier le script pour qu'il puisse les faire disparaitre dès qu'il va les toucher. La manipulation demande qu'on retourne dans "l'Inspector" du personnage et d'ouvrir le script en double cliquant sur ce dernier afin d'ajouter une nouvelle fonctionnalité. On ajoute donc la fonction ci-dessous, à la suite des précédentes :

```
private void OnTriggerEnter2D(Collider2D collision)
{
    if(collision.CompareTag("Raclette")) // si je rencontre un objet qui a le tag
    {
        Debug.Log("hop un ingrédient de plus"); // hop dans ma
console pour vérifier si j'ai bien touché
        Destroy(collision.gameObject); // je détruis l'objet que j'ai touché
    }
}
```

Et si on testait ? Paf le fromage est ramassé ! Bien évidemment, on en attrape qu'un seul !

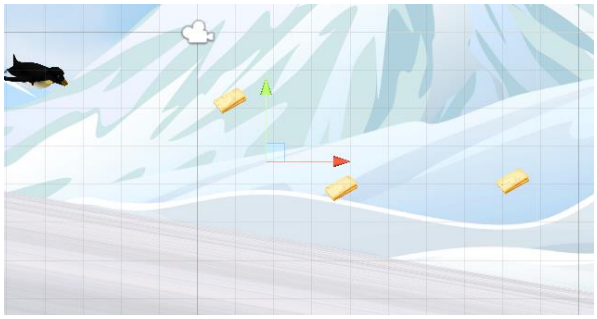
Cependant un fromage c'est bien mais plusieurs c'est mieux ! Pour aller plus vite et ne pas faire tout de zéro pour le prochain fromage, on crée des préfabriqués, c'est-à-dire des fromages qui auront les mêmes caractéristiques que celui que nous venons de faire, qui reprendra des informations comme son "Tag", son "Collider" et ses dimensions pour en citer quelques-uns.

Afin de créer ces derniers, on glisse le fromage de la hiérarchie jusqu'aux "Assets" dans la fenêtre "Project". Pour savoir si cela a bien fonctionné, il suffit de voir si le préfabriqué possède un fond gris !



Fromage préfabriqué

Désormais, on peut ajouter des fromages tout le long de la piste en glissant directement le préfabriqué dans la scène.



Les fromages sur la piste

🌟 Félicitations ! Vous avez réussi le niveau 4 ! 🌟

b. Une passion, compter le nombre de fromage à raclette

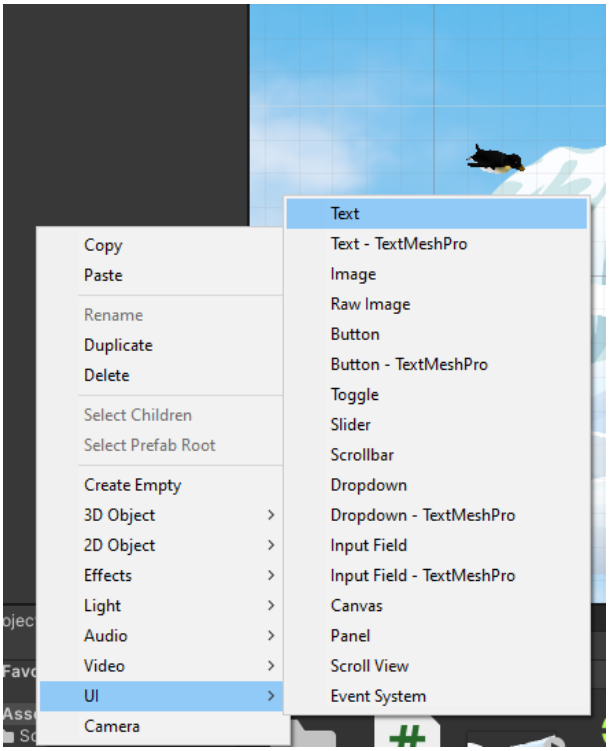
Désormais Maurice peut ramasser les fromages mais il faut les compter maintenant ! Pour ce faire il faut rajouter un score dans le script et créer un texte d'affichage. Rajoutez ces variables en dessous de celle écrite précédemment dans le script.

```
public Text score_txt; // on y ajoutera l'objet contenant le texte
public int score = 0; // pour compter les fromages
```

À la suite, ajouter des lignes dans la fonction "OnTriggerEnter2D(Collider2D collision)".

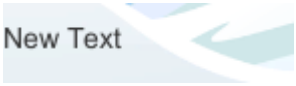
```
if (collision.CompareTag("Raclette")) // si je rencontre un objet qui a le tag là
{
    Debug.Log("hop un ingredient de plus"); //hop dans ma console pour verifier si j'ai bien touché
    score++; // on ajoute le nouveau fromage à notre compteur
    PlayerPrefs.SetInt("Score", score); // permet de transporter son score de scene en scene
    score_txt.text = "Score : " + PlayerPrefs.GetInt("Score").ToString(); // On affiche notre score

    Destroy(collision.gameObject); // je détruis l'objet que j'ai touché
}
```

Maintenant nous allons créer l'affichage du texte. Pour cela vous allez devoir faire un clic droit dans votre hiérarchie, cliquer sur "UI" puis "Text".



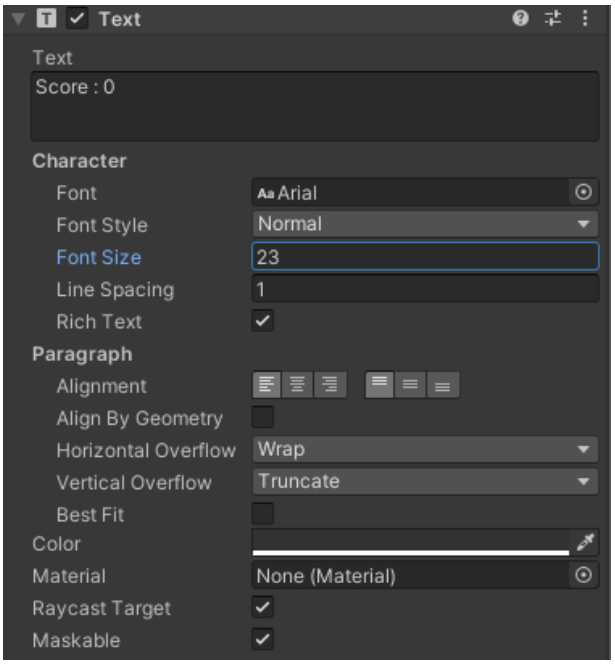
Chemin pour créer un texte

Par la suite, cela créera un "Canva" et une sous-catégorie "Text". Dans la fenêtre "Game", un aperçu indiquera où se situera l'élément dans le jeu. En jetant un coup d'œil en bas à gauche de la fenêtre on trouve donc l'élément.



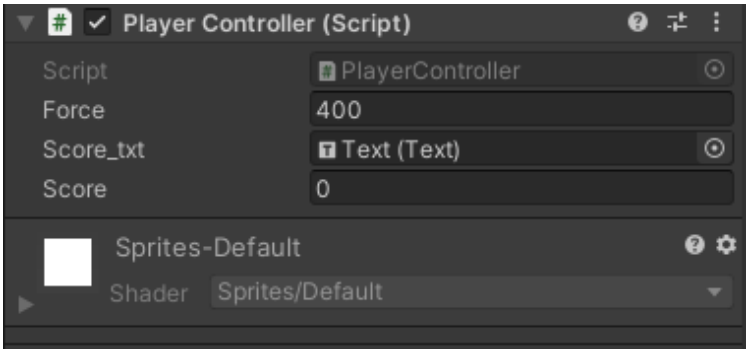
Le nouveau texte sur la piste

Afin de modifier ce texte, on va se rendre dans "l'Inspector" du "Text" contenu dans le canva pour modifier le composant correspondant. On peut donc changer le contenu, la police d'écriture, la taille et pleins d'autres.



Interface de modification du texte

Il ne nous reste plus qu'à ajouter l'objet "Text" dans la variable "Score_txt" du composant provenant du script.



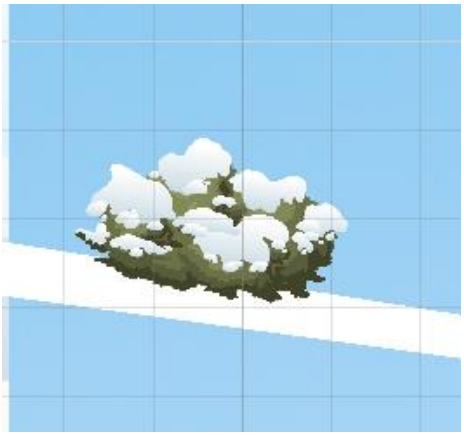
Ajout de l'objet contenant le texte

Il est temps de tester le score en appuyant sur "Play" !

c. Attention aux buissons !

Maurice se souvient pourquoi il avait été envoyé dans le passé par Manigo, il s'était pris des buissons lors de sa descente. Ça aurait été trop facile s'il devait juste attraper le fromage ! Pour que la simulation soit la plus réaliste possible, il faut ajouter des obstacles qui fera qu'on doit recommencer le jeu.

Pour ce faire il faut ajouter un buisson dans la scène, puis réduire la taille, et ensuite l'adapter pour le sol. On va y ajouter un "Box Collider 2D" afin que Maurice 2.0 détecte sa collision, si besoin on peut l'éditer. Il faudra cocher "is Trigger" pour que le clone ne soit pas directement arrêté.



Un buisson bien gênant

Ensuite comme précédemment avec les fromages, on crée un Tag "Buisson" pour qu'on puisse ajouter une interaction dans le script du Clone.

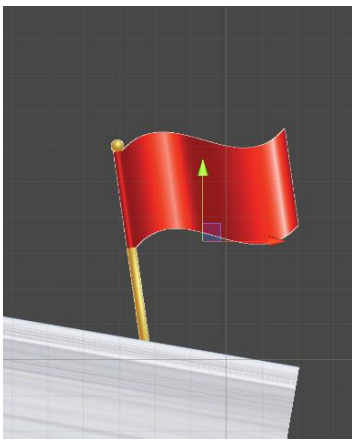
```
if (collision.CompareTag("Buisson")) // si je rencontre un Buisson
    SceneManager.LoadScene("SampleScene"); // je relance la scene "SampleScene"
```

Super ! Désormais on peut perdre et recommencer !

Un Gif qui fait mal aux yeux

🌟 Bravo vous pouvez perdre 😳 🌟

d. Ce n'est pas la fin, ce n'est que le début …

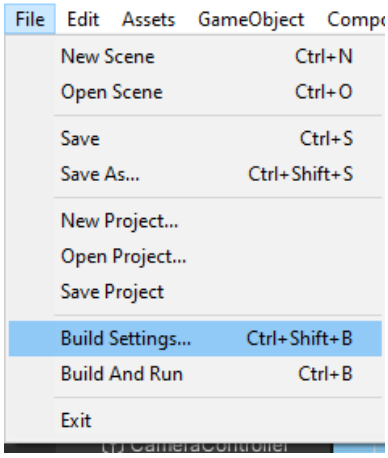


Drapeau de fin

Pour le moment le temps est suspendu, mais il faut revenir dans le présent. Afin de réaliser cette tâche, rajouter un élément qui va indiquer que c'est la fin du parcours. Insérer l'image "Flag" à l'endroit où la descente se termine.

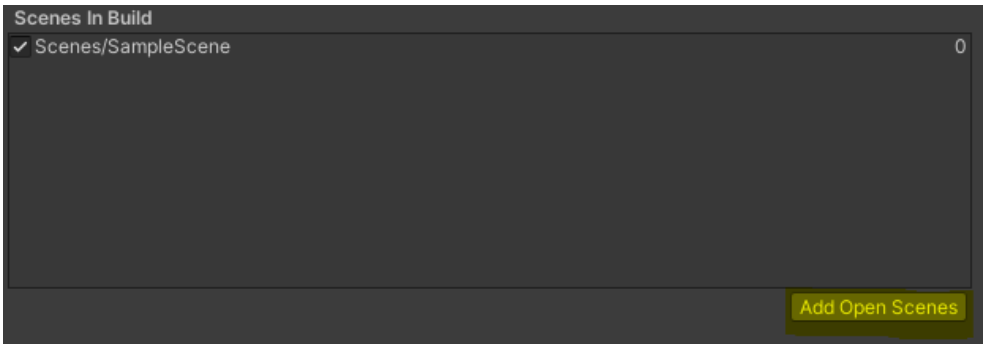
Ensuite rajoutez un "box Collider 2D" à cet endroit afin que l'on puisse savoir quand on rentre en contact avec l'objet et cochez "is Triger". Créez-lui un tag " End " et attribuez-lui dans son "Inspector". Par la suite, il faut créer une nouvelle "scene" pour avoir un écran de fin, afin de retourner dans le temps présent.

Mais avant tout ça, il faut qu'on ajoute la scène actuelle dans le "build" du jeu. Qu'est-ce que le "build" ? C'est ce qui permet d'indiquer à Unity que cette scène fait partie de la fin du jeu. Pour faire cela, il faut aller dans l'onglet "File" tout en haut à gauche, cliquer dessus et aller dans "Build Settings...".



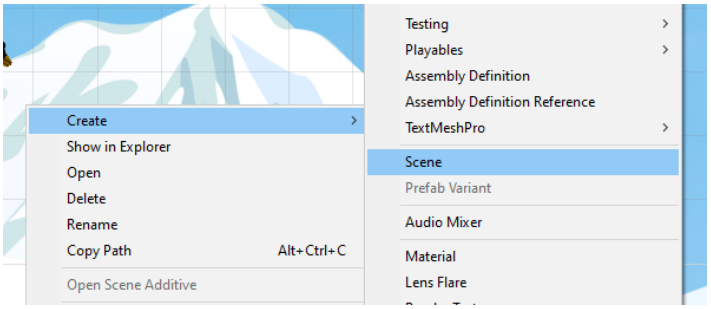
Menu « File »

A la suite de cela, une nouvelle fenêtre va s'ouvrir regroupant l'ensemble des scènes ajoutées, pour le moment aucunes d'entre elles ne seront présentes. Il faudra alors appuyer sur "Add Open Scenes" pour ajouter nos scènes ouvertes. La première sera forcément indiquée en 0 car ce sera sur celle-ci que votre jeu s'ouvrira. Une fois cela fait, fermez cette fenêtre.



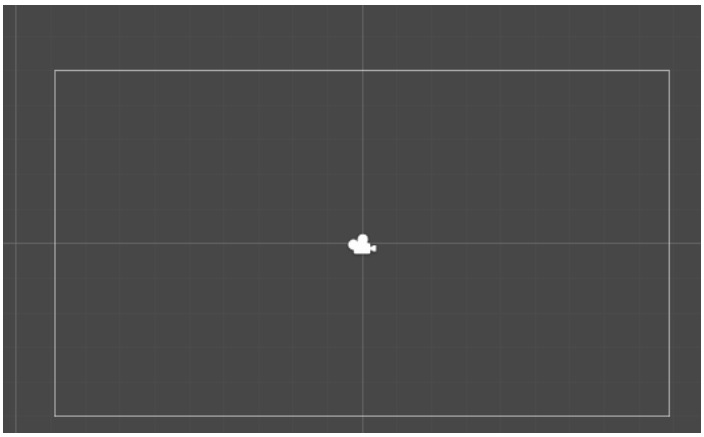
Menu des scènes contenues dans le Build

Il faut donc commencer par créer la nouvelle scène, aller dans la fenêtre "project", puis se rendre dans le dossier "Scenes" et faire un clic droit afin de la réalisée.



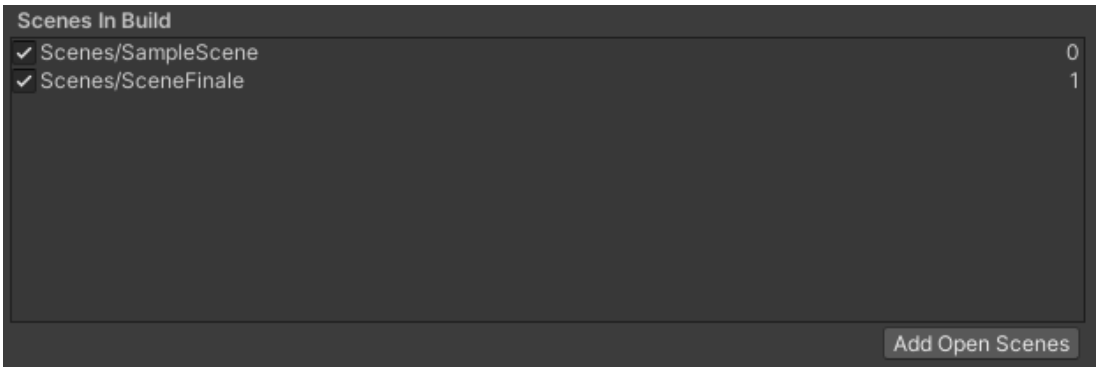
Chemin pour créer une scène

Une fois cela créée, on la renomme "SceneFinale", par la suite on se rend dans cette nouvelle scène en double cliquant dessus, sans oublier de sauvegarder la première scène avant de la quitter ! Comme on peut le constater, cette scène est pour le moment complètement vide.



Scène « SceneFinale » vide

Dans un premier temps, on l'ajoute au "build" comme on a pu le faire ci-dessus.



Menu des scènes contenues dans le Build

Grâce à cela, il sera possible de créer son propre écran de fin personnalisé ! Par exemple s'il vous manque l'inspiration, les images "pinguouin_fete" et "raclette_party" sont disponibles pour cela, on peut aussi ajouter un texte !

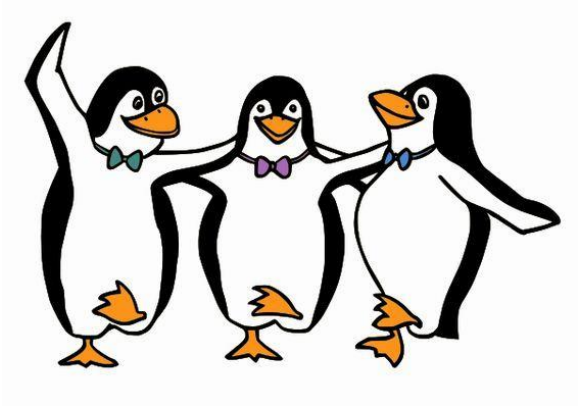
Raclétouille pour tout le monde

Il faudra alors sauvegarder la scène, pour retourner dans la première. Maintenant il ne reste plus qu'à ajouter du code dans le script afin de relier ces deux scènes.

Sans oublier la façon dont a été réalisée le relancement de la scène quand Maurice se cogne dans un buisson, vous allez vous en servir pour en pour faire une condition de rencontre avec le tag « End » et cette fois au lieu de relancer la scène actuelle, vous demandez seulement à lancer la dernière scène.

VI. Conclusion

Incroyable ! Grâce à la simulation, Maurice a pu récupérer un maximum d'ingrédients possibles pour sa raclette party, Manigo réapparait et indique à Maurice qu'il sera renvoyé dans le temps présent afin qu'il puisse réellement sauver sa soirée avec ses meilleurs amis.



La dance de Maurice et ses amis

Quelques idées de bonus :

- Rajoutez de la musique et des bruitages lors des sauts et quand vous ramassez les fromages
- Rajoutez d'autres éléments à ramasser qui donnent des points différents, vous avez d'ailleurs un sprite de patate à votre disposition
- Faire des tremplins pour encore plus de fun
- Faire un menu pour commencer le jeu
- Faire un bouton pour relancer le jeu et tenter d'améliorer son score à la fin