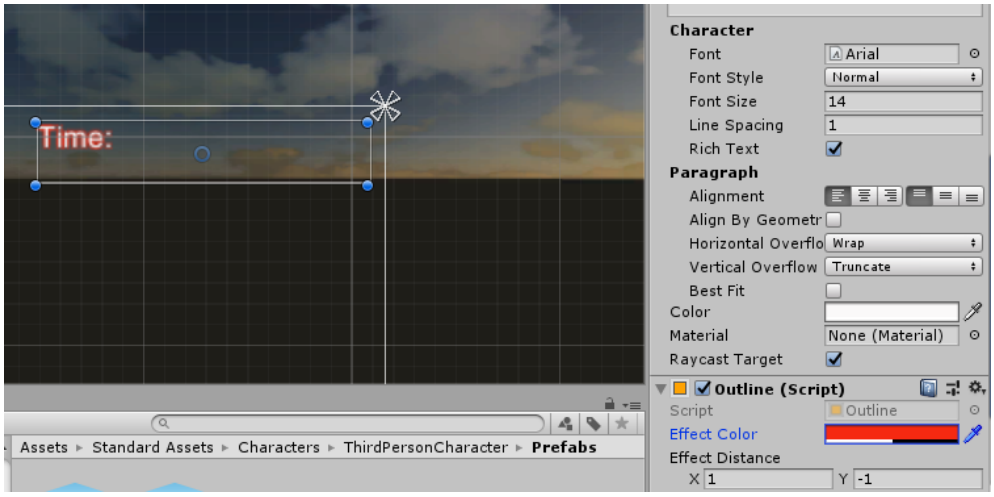
UNITY - TP3.1

TIMER

Vous allez ajouter le temps à votre jeu pour le rendre plus intéressant ! Créez une étiquette pour afficher le temps, lblTime. Pour améliorer son apparence ajoutez au label un composant ***UI > Effects > Outline***. Choisissez une couleur, rouge dans l'exemple, pour voir l'effet. Vous pouvez faire de même pour l'étiquette lblCoins.



Pour le compte à rebours, créez un nouveau script Countdown associé à l'objet World :

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

public class Countdown : MonoBehaviour {

  public float allowedTime = 90;
  Text timerText;
  float currentTime;
  GameObject canvasObj;
  Transform child;

  void Start() {
    canvasObj = GameObject.Find("Canvas");
    child = canvasObj.transform.Find("lblTime");
    timerText = child.GetComponent<Text>();
    StartCoroutine(TimerTick());
  }

  IEnumerator TimerTick() {
    currentTime = allowedTime;
    while (currentTime > 0) {
      //attendre 1 seconde
      yield return new WaitForSeconds(1.0f);
      currentTime--;
```

```
        timerText.text = "Time :" + currentTime;
      }
      // game over and restart
      SceneManager.LoadScene("Scene1Exterieur");
    }
}
```

Comme précedemment, nous récupérons l'étiquette grâce au canvas. Le IEnumerator TimerTick contient une boucle qui s'arrête lorsque le compteur arrive à 0. WaitForSeconds(1.0f) arrête l'exécution pendant une seconde, réduit le temps disponible et met à jour le texte à afficher. Les variables exposées sont des variables accessibles en dehors d'un script comme la variable allowedTime. Elles sont visibles dans l'Inspector et peuvent être connectées entre elles avec un simple glisser-déposer.

Testez votre jeu et faites une capture d'écran.

DE LA CONTINUITE DU TEMPS

Vous vous demandez surement comment faire pour que le Timer que nous avons implémenté fonctionne sur tout le jeu et non seulement pour un niveau. La réponse : les variables statiques !! (bon, c'est vrai, je ne vous apprends rien ;) pas besoin de deux points d'exclamation !!!)

Sur la scène terrain, créez un script définissant une classe avec des variables statiques :

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

public static class GameVariables {
   public static float allowedTime = 90;
   public static float currentTime = allowedTime;
   public static int coins = 0;
}
```

Modifiez maintenant les scripts Countdown.js et World.js pour utiliser les variables statiques : GameVariables.coins, GameVariables.currentTime et GameVariables.allowedTime.

Maintenant vous pouvez enrichir votre jeu, par exemple :

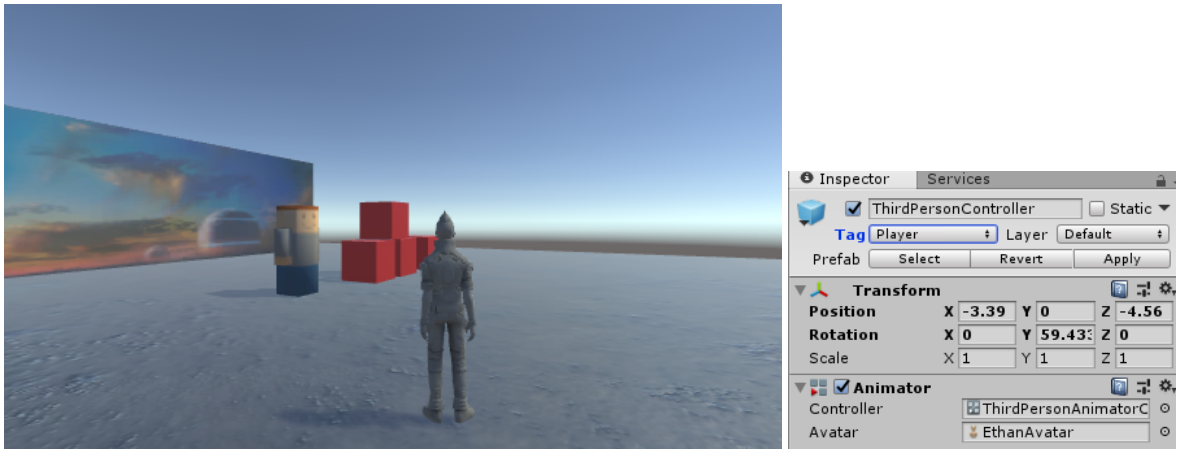
- affichez un message différent lorsque le joueur a trouvé toutes les pièces.
- ne permettez pas au joueur de changer de niveau que lorsqu'il aura trouvé un nombre prédéfini de pièces
- créez des pièces différentes qui donnent plus de points ou qui en enlèvent !

Indiquez dans votre compte rendu le(s) mécanisme(s) que vous avez choisi.

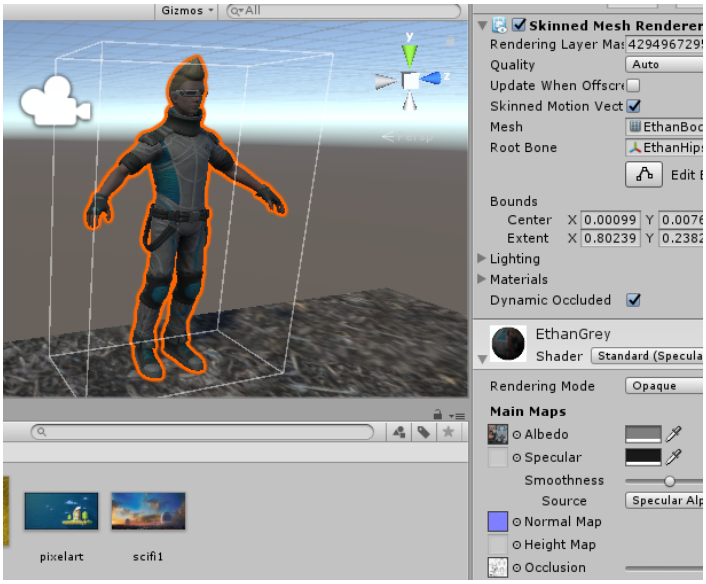
Sauvegardez votre scène et ouvrez la scène BoxBoy. Créez les éléments nécessaires à la mise à jour et à l'affichage du Timer. Faites une capture d'écran.

LE THIRD PERSON CHARACTER

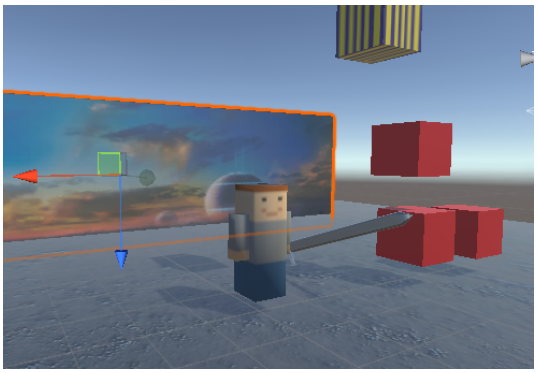
Enlevez le script de contrôle par les flèches de BoxBoy et créez un ThirdPersonCharacter (***Standard assets > Characters > ThirdPersonCharacter> Prefabs > ThirdPersonController***). Placez la caméra en tant qu'enfant du ThirdPersonController et modifiez ses paramètres. J'ai utilisé comme position -0,5, 1.5, -3, rotation 0,0,0. Ce dernier point est important, une rotation peut faire tourner le personnage lorsqu'il avance. Modifiez le « tag » du character à Player.



PS : Pour améliorer l'apparence du TPC, importez la texture EthanAlbedo.png. Ajoutez ensuite cette texture à « l'Albedo » du matériau EthanGrey.



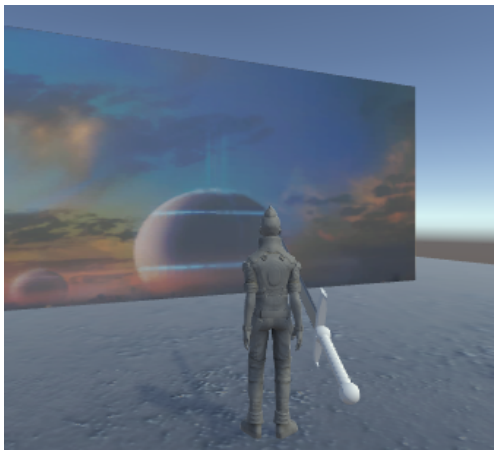
Le but du « jeu » dans ce niveau est d'arriver près de BoxBoy pour récupérer une épée (ou un autre objet de votre choix). Importez l'objet sword.fbx et glissez-le comme enfant de BoxBoy. Modifiez ses paramètres pour l'adapter à la scène.



Notre personnage doit récupérer l'épée et pour cela, il doit s'en rapprocher suffisamment. Créez le script ci-dessous et associez-le à l'épée. Il nous permet de détecter la présence de l'avatar, sa distance, et de lui transférer l'épée s'il est assez proche :

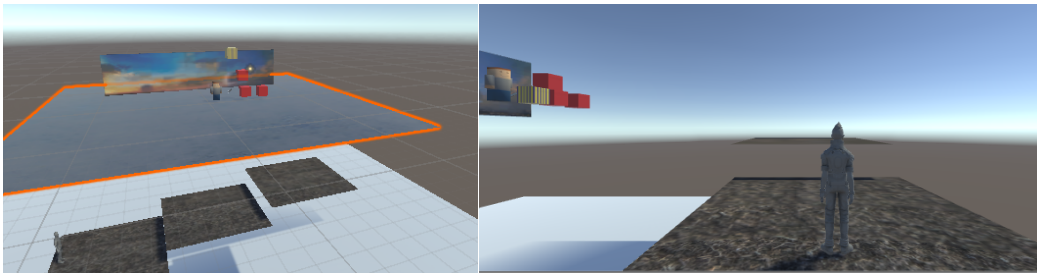
```
float distance = 10;
GameObject player;

// Use this for initialization
void Start () {
   player = GameObject.FindWithTag("Player");  // pour trouver le personnage
}

        // Update is called once per frame
        void Update () {
   distance = Vector3.Distance(this.transform.position, player.transform.position);
   if (distance < 1)
   {
      this.transform.parent = player.transform;
      this.transform.localPosition = new Vector3(0.2f, 1.0f, 0.0f);
      this.transform.localRotation = Quaternion.identity;
      this.transform.localRotation = Quaternion.Euler(110, 180, 90);
   }
}
```

Modifiez les valeurs des transformations pour qu'elles correspondent à l'échelle et position de vos modèles.



LES PLATEFORMES

Nous allons donner du fil à retordre à notre personnage principal. Créez plusieurs plans séparés et modifiez leurs positions de manière à créer un escalier. Placez le personnage sur la première marche. Le personnage doit maintenant sauter pour arriver jusqu'à la plateforme où se trouve BoxBoy.



Si le personnage rate une marche, il tombe à l'infini. Pour pouvoir continuer le jeu en relançant le niveau automatiquement, nous allons créer un collider parallèle à la plateforme avec des dimensions suffisantes pour contenir toutes les marches (et la plateforme elle-même). Créez un nouvel objet vide et ajoutez-lui un BoxCollider. Activez l'option isTrigger pour appeler une fonction lorsque le collider de l'objet est activé. En associant à l'objet le script suivant, à chaque fois que le personnage tombera, il sera renvoyé au début du niveau :

```
void OnTriggerEnter(Collider other)
{
   SceneManager.LoadScene("BoxBoy"); // le nom de votre scène
}
```

Trouvez un mécanisme pour rendre plus difficile l'accès à l'épée : plus de cubes, un labyrinthe, etc. Décrivez ce(s) mécanisme(s) dans votre compte rendu. Une fois le TP fini, continuez avec le TP3.2 qui met en place des animations.

À TRÈS BIENTÔT!