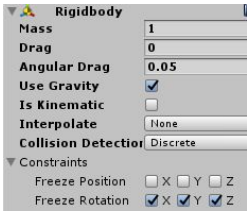
Orienter un personnage selon les axes de la caméra

(Déplacement avant/arrière et gauche/droite)



Nous avons vu comment tourner un personnage autour de lui même:

transform.Rotate(0, tourne , 0) ; // ici tourne est une variable de type float

Personnage qui s'oriente selon les axes de la caméra (déplacement avant/arrière et gauche/droite)

Pour orienter le personnage selon la vu de la caméra, il faut ;

- avoir une caméra qui est indépendante du personnage (non parentée), et
- la caméra doit garder une distance et ne pas tourner avec le personnage.

(la rotation de la caméra sera vue à la prochaine section).

Pour obtenir la direction vers l'avant de la caméra (son axe Z), il faut utiliser :

maCamera.transform.forward //maCamera est la référence à l'objet caméra

transform.forward = maCamera.transform.forward; //oriente le personnage vers le devant de la caméra

transform.forward = - maCamera.transform.forward; //oriente le personnage vers l'arrière (regarde la caméra)

Pour un déplacement sur les côtés de la caméra (son axe X), il faut utiliser :

maCamera.transform.right //maCamera est la référence à l'objet caméra

transform.forward = maCamera.transform.right; //oriente le personnage vers la droite de la caméra

transform.forward = - maCamera.transform.right; //oriente le personnage vers la gauche

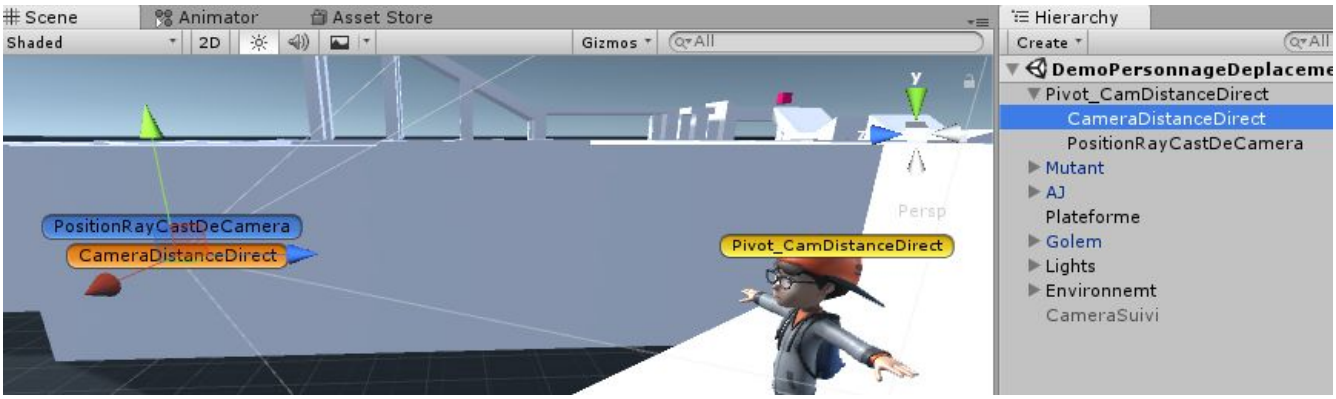
```
  // Les touches verticales,  déplace le personnage vers l'avant et arrière (selon l'axe Z de la caméra)
 // Les touches horizontales,  déplace le personnage vers la gauche et vers la droite (selon l'axe X de la caméra)
public float  vitesseDeplacement;   //la vitesse de déplacement du personnage
public GameObject camPivot;

void DeplaceDirectionCamera()
{
    // obtient les valeurs des touches horizontales et verticales
    float hDeplacement = Input.GetAxis("Horizontal");
    float vDeplacement = Input.GetAxis("Vertical");

    //obtient la nouvelle direction (     (avant/arrièrre)                    +                    (gauche/droite) )
    Vector3 directionDep = camPivot.transform.forward * vDeplacement + camPivot.transform.right  * hDeplacement;

    directionDep.y = 0;  //pas de valeur en y , le cas où la caméra regarde vers le bas ou vers le haut

    if (directionDep != Vector3.zero)      //change de direction s'il y a un changement
    {
        //Oriente le personnage vers la direction de déplacement, et applique la vélocité dans la même direction
       transform.forward = directionDep;
       rigidbodyPerso.velocity = (transform.forward * vitesseDeplacement) + new Vector3(0, rigidbodyPerso.velocity.y, 0);
    }
 }
```

Rotation d'une caméra autour du personnage à l'aide de la souris

La technique utilisée ici est de placer un objet vide (le pivot) au niveau de la tête du personnage et de parenter la caméra au pivot. Le pivot peut tourner à l'aide de la souris et suit la position du personnage, la caméra va mimer ces mouvements.

Notes importantes

- La caméra ne doit pas avoir de script qui la maintient à distance constante puisqu'elle suit le pivot et c'est le pivot qu'on déplace en suivant le personnage.
- Pour ajuster la distance (avant/arrière, haut/bas, gauche/droite) entre la caméra et le personnage, il suffit de modifier la position de la caméra par rapport au pivot.




```
//Ce script permet de suivre la position du personnage et de tourner le pivot d'une caméra selon les mouvements de la
// souris (G/D et haut/bas).
// le pivot est un objet vide, la caméra est parentée au pivot et suit ces mouvements,

public class cameraDistanceDirect : MonoBehaviour {
 public GameObject cible;                          // le personnage
 public GameObject camPivot;                       // la caméra
 public GameObject positionRayCastCamera;          // mémorise la position de lancement du RayCast de la caméra
 public float hauteurPivot;                        // Ajustement de la position du pivot en hauteur
 public float distanceCameraLoin=4;
 public float distanceCameraPret=0.5f;

 // Position le pivot de la caméra au même endroit que le personnage + une certaine hauteur (prêt de la tête)
 // le pivot tourne par le déplacement de la souris en X et Y
   void Update ()
  {
            transform.position = cible.transform.position + new Vector3(0, hauteurPivot,0);
            transform.Rotate(Input.GetAxis("Mouse Y"), Input.GetAxis("Mouse X"),0);  // ou (-Input.GetAxis("Mouse Y"))
            // pour inverser la direction
```

```
        //Annuler la rotation en Z
            transform.localEulerAngles  =  new Vector3(transform.localEulerAngles.x,transform.localEulerAngles.y, 0 );
           //Caméra regarde le pivot
            camPivot.transform.LookAt(transform);
    }
```

Zoomer la caméra lorsque la vue est obstruée

Plusieurs techniques sont possibles. Dans cet exemple, on utilise une technique qui détecte s'il y a un objet entre la caméra et le personnage à l'aide d'un RayCast. Si un objet est détecté, on approche la caméra du personnage. S'il n'y a plus d'objet, on s'assure de replacer la caméra à sa distance d'origine.

Pour tracer le rayon, on utilise un deuxième objet-vide (*positionRayCastCamera*) que l'on place comme enfant du point de pivot (comme la caméra). On le place à la même position que la caméra et on oriente son axe de Z vers le personnage. C'est à partir de cette position que le RayCast sera tracé. L'avantage, c'est qu'il restera toujours à la même distance de la caméra, contrairement à cette dernière qui pourra s'approcher du personnage si un objet se trouve dans le champ de vision.

```
//S'il y a un objet entre la caméra et le personnage, avance la caméra vers la tête du personnage sinon la caméra retrouve sa
//position de départ.
//Le Raycast est lancé à partir de la position éloignée de la caméra, fournie par le objet vide de référence
if(Physics.Raycast(positionRayCastCamera.transform.position, positionRayCastCamera.transform.forward, distanceCameraLoin))
{
    camPivot.transform.localPosition = new Vector3(0,0,distanceCameraPret);
}
else
{
    camPivot.transform.localPosition = new Vector3(0,0,distanceCameraLoin);
}
```

Remarque: Avec ce type de déplacement lorsqu'il faut activer l'animation de course alors il faut utiliser la vélocité de l'objet au lieu de sa variable vDeplacement (qui peut être négative ou 0 mais le personnage se déplace sur les côtés):

~~animPerso.SetFloat("vitesseDep", vDeplacement);~~

```
animPerso.SetFloat("vitesseDep", rigidbodyPerso.velocity.magnitude);
```