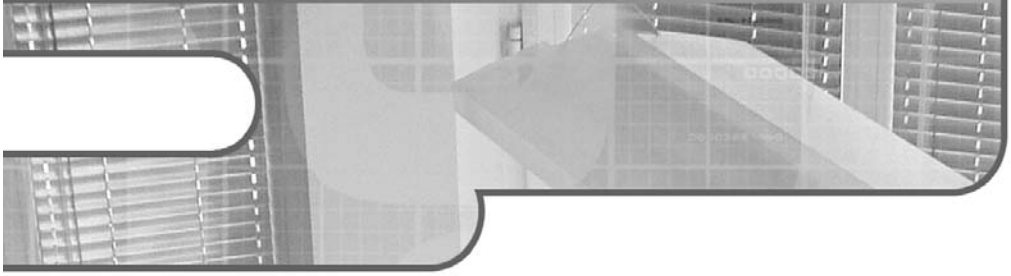


Chapitre 3
Animation 2D

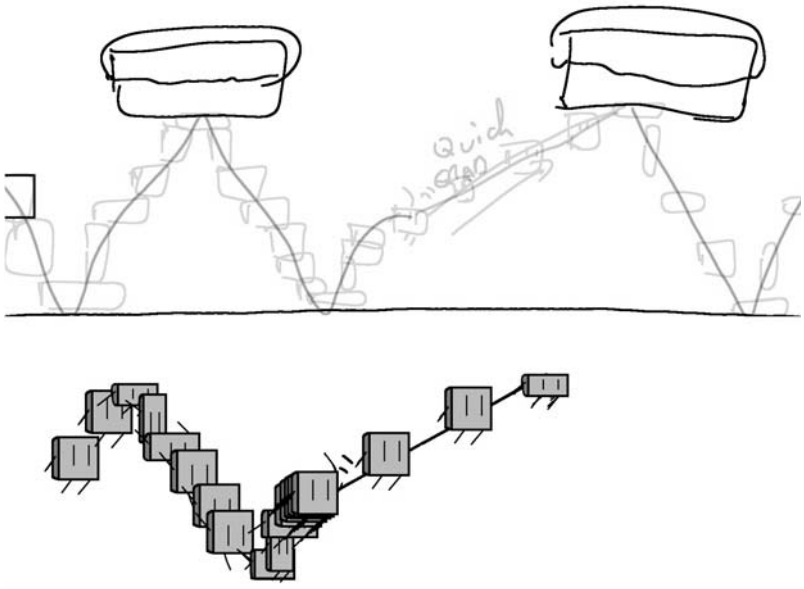
1. Création de sprites animés

1.1 Introduction à l'animation 2D

L'animation dans les jeux vidéo est ce qui les rend vivants et attrayants. Son principe est simple : enchaîner une suite d'images pour donner l'impression de mouvement, de vie. La popularisation de l'animation a commencé au début des années 1900 avec des longs métrages d'animation qui sont les pionniers de la méthode de séquences d'images pour l'animation. C'est ensuite vers les années 1950 avec Disney que l'animation a été grandement popularisée aux yeux du monde avec des courts et longs métrages bourrés d'animations détaillées. Cette période est même considérée comme l'âge d'or de l'animation tant elle a marqué son époque. Depuis cette époque, on a continué à innover dans la méthode de travail pour l'animation et les techniques utilisées grâce aux nouvelles technologies qui réduisent grandement la charge de travail tout en gardant une grande qualité.

Même si, depuis les dernières années, l'animation 3D est la plus utilisée, cela ne veut pas dire que l'animation 2D est remplacée, et c'est dans les jeux vidéo que l'on retrouve le plus cette cohabitation entre 2D et 3D.

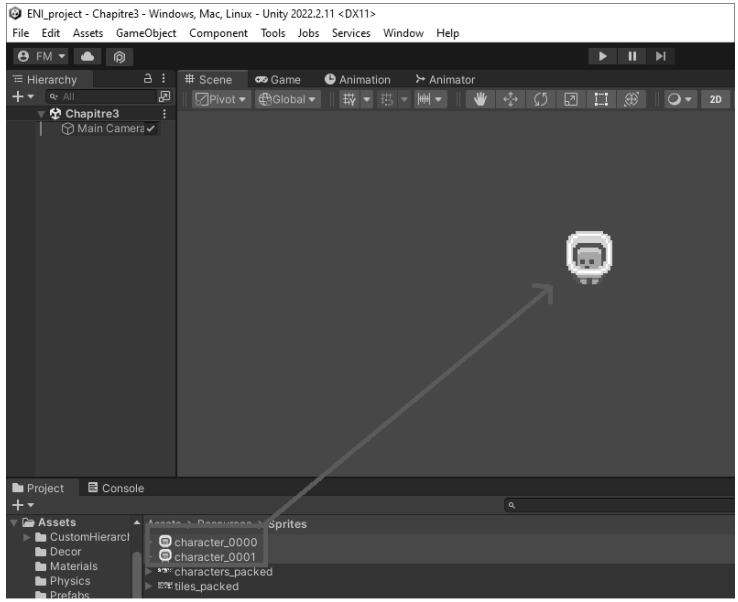
L'animation 2D offre une grande liberté dans l'esthétique de l'animation d'un personnage comme elle n'est pas liée à un « squelette », et peut donc avoir des déformations exagérées pour accentuer des émotions ou des situations. Cependant, cela demande énormément de ressources, aussi bien humaines que fonctionnelles, car plus une animation sera détaillée, plus elle demandera de temps à être réalisée. En effet, chaque séquence d'une animation doit être dessinée une à une et plus une animation doit être fluide, plus elle a besoin d'image de transition.

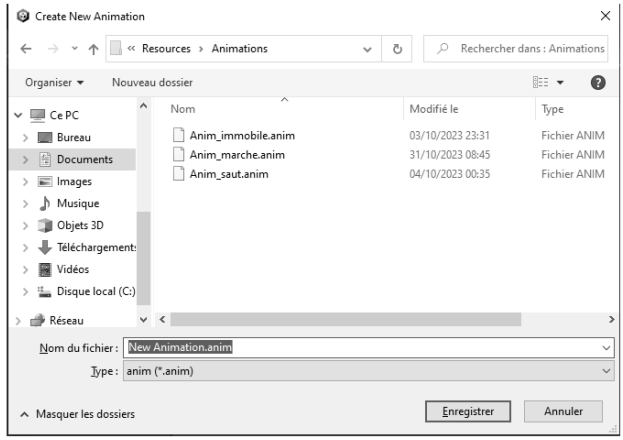


Dans les débuts du jeu vidéo, les consoles étaient limitées en ressources, il fallait donc optimiser au maximum pour avoir un jeu à la fois complet et avec des animations de qualité. C'est ce qui a notamment amené Ubisoft à créer le héros du jeu Rayman sans bras ni jambes. En plus de gagner du temps dans l'animation qui était faite à la main à l'époque, ils pouvaient réaliser des animations fluides qui ne prenaient que peu de ressources puisqu'il n'y avait que les mains et les pieds à animer. Aujourd'hui, il est toujours important de bien optimiser son jeu, mais les contraintes sont moins grandes qu'avant en termes de ressources. Nous allons maintenant présenter la création des animations avec Unity.

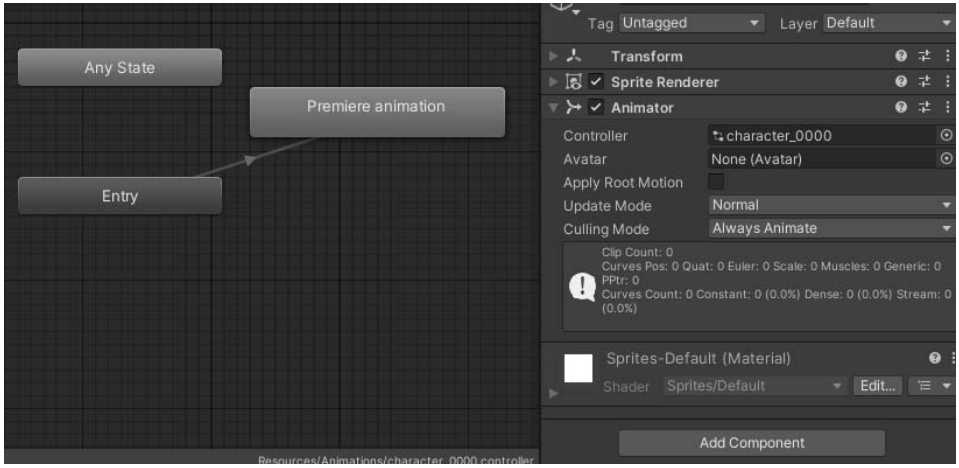
1.2 Création rapide

Il existe une manière rapide pour créer un GameObject avec un sprite animé à partir de plusieurs sprites. Il faut sélectionner les sprites qui composeront l'animation voulue et les glisser dans la scène ou la **Hierarchy**. Unity proposera de sauvegarder l'animation dans le dossier voulu.





Le GameObject est désormais créé avec un **Component** appelé **Animator** qui sera nommé à partir du premier sprite de l'animation et contenant l'animation sauvegardée.

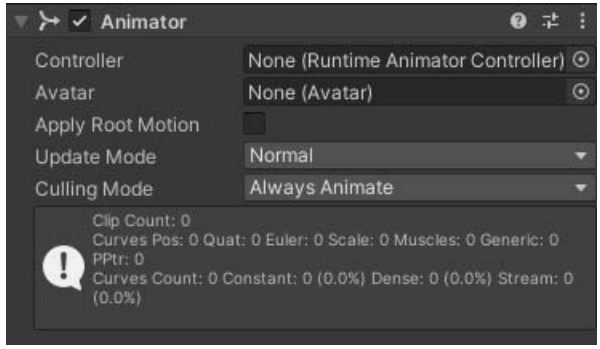


Cette méthode est utile pour créer rapidement des objets animés simples qui n'auront qu'une animation ou la structure de base d'un objet animé, et les modifier pour ajouter plus de détails.

1.3 Animator

Un **Animator** (ou animateur en français) est un composant permettant de créer et gérer des animations pour des personnages ou des objets dans un jeu. Pour être plus précis, il permet :

– de créer des animations en faisant des mouvements d'objet et/ou de personnage ou en changeant d'état pour activer des animations ;
– d'utiliser des **State Machine** (ou « machine à état » en français) pour définir les différentes transitions entre les animations pour automatiquement activer des animations en fonction de ce qu'il se passe sur la scène ;
– d'utiliser des paramètres pour gérer les transitions de la **State Machine** telles que les booléens, les float ou les strings.



Le composant **Animator** a besoin d'un Animation Controller pour gérer les animations du GameObject auquel il est lié. Ce contrôleur peut être créé dans la fenêtre **Project** dans le sous-menu **Create - Animator Controller**. Il est aussi possible de lier un « Avatar » à l'**Animator** qui devra être importé dans Unity sous le format .obj, . fbx ou tout autre format provenant d'un logiciel de modélisation.

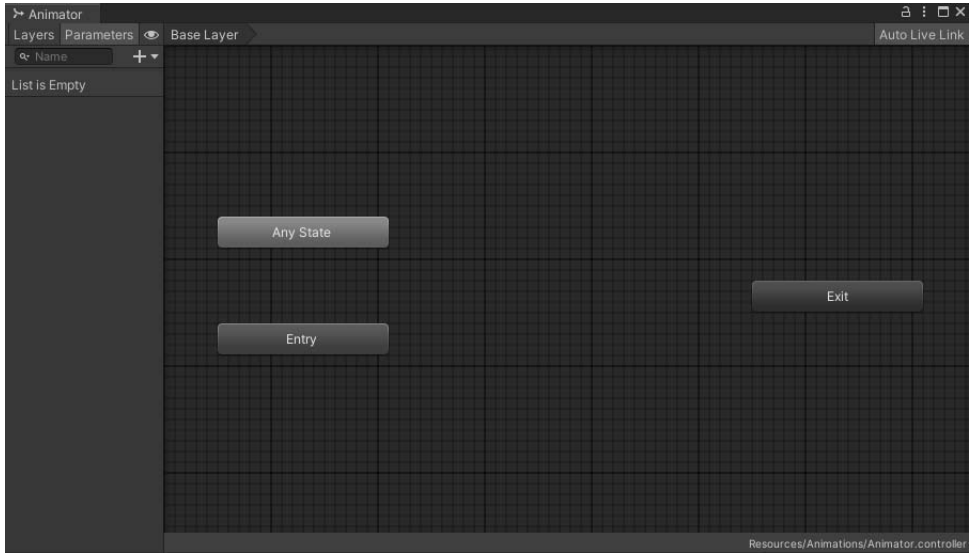
Le champ **Update Mode** permet de définir le mode de mise à jour de l'animation parmi les choix suivants :

– **Normal** qui actualise l'animation en même temps que la fonction Update.
– **Animate Physics** qui actualise l'animation en même temps que la fonction FixedUpdate (en même temps que les calculs de la physique).
– **Unscaled Time** qui actualise l'animation quelle que soit l'échelle de temps. Elle continuera même lorsque le jeu est en pause (échelle de temps à 0).

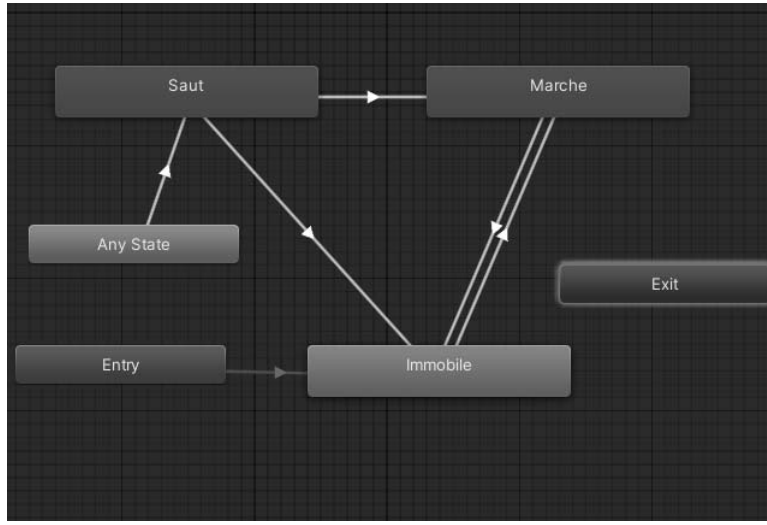
Enfin, le **Culling Mode** permet de déterminer si l'animation est exécutée lorsque l'objet est en dehors de la vue de la caméra. Les options incluent **Always Animate** (Toujours animer), **Cull Update Transform** (Masquer mise à jour transformée) et **Cull Completely** (Masquer complètement).

1.4 Animator Controller

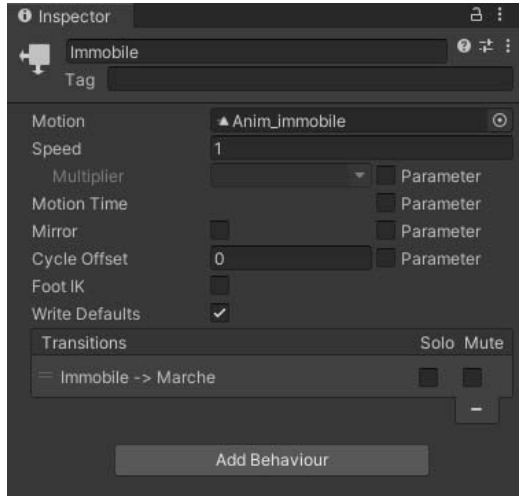
L'**Animator Controller** est l'objet qui sera associé au composant **Animator** et qui va gérer les états, les transitions et les animations de ce dernier. Il peut être créé dans le dossier du projet en y faisant un clic droit → **Create - Animator Controller**. Une fois l'élément créé et ouvert, une nouvelle fenêtre apparaît : la fenêtre **Animator**.



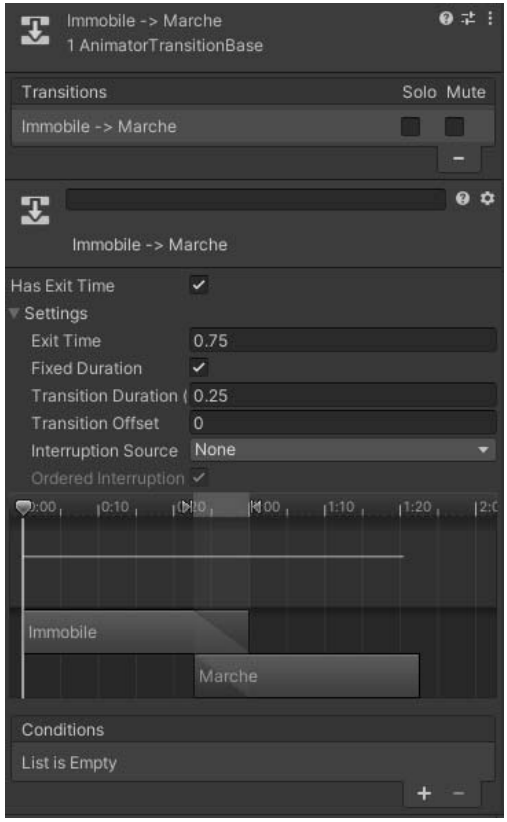
Cette fenêtre permet de visualiser les différents états que peut prendre une animation et d'en gérer les transitions. Trois éléments seront présents par défaut lors de la création d'un **Animator Controller** : **Entry**, **Exit** et **Any State**. **Entry** correspond au point de départ de l'animation. La première animation sera liée à **Entry**. **Exit** correspond à la fin de l'animation. Il n'est pas nécessaire de l'utiliser car les animations sont souvent vouées à être animées en permanence. Enfin, l'élément **Any State** correspond à une activation d'une animation, quel que soit l'état dans lequel il se trouve. Il peut par exemple être utilisé pour déclencher des animations de saut ou de dégâts sur un personnage car ces animations peuvent être déclenchées depuis un état **Immobile** ou **Marche** si la condition pour la transition vers cet état est vraie.



Sur l'illustration se trouve un exemple d'un graphe d'état pour un personnage. Pour créer un état, il faut réaliser un clic droit → **Create State - Empty**. Le premier état créé est toujours lié à **Entry** et sera la première animation jouée. Il sera de couleur orange pour se démarquer des autres états. Chaque état contient un nom, une animation ainsi que d'autres paramètres permettant de gérer la vitesse, l'orientation, la transition, etc.

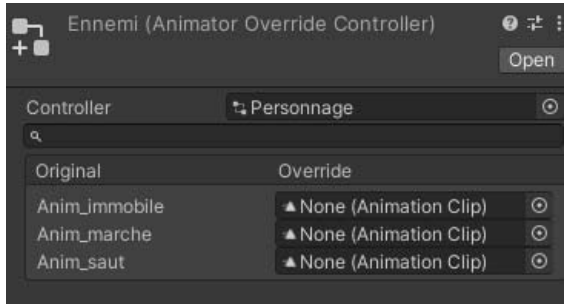


Pour créer une transition entre deux états, il faut faire un clic droit sur un état, sélectionner **Make Transition** et pointer vers l'état voulu. En cliquant sur la flèche représentant la transition, il est possible d'en gérer les paramètres pour définir le délai de transition, le chevauchement des deux animations lors de la transition et la condition permettant de déclencher cette transition.



1.5 Animator Controller Override

Il existe un autre type de **Controller** appelé **Animator Controller Override** qui servira à éviter tout doublon d'**Animator** pour des graphes communs tels que l'animation de personnages qui ont globalement le même comportement mais pas le même aspect visuel. Cet objet peut être créé dans le dossier du projet en faisant un clic droit → **Create - Animator Controller Override**. Ce dernier demandera la référence du **Controller** à surcharger pour pouvoir remplacer une ou plusieurs animations tout en gardant tout le graphe d'états.



1.6 Animation

Une **animation** est une séquence de changements de propriété et/ou d'images qui évolue au fil du temps. Ces changements créent du mouvement physiquement et/ou visuellement afin de rendre le jeu plus dynamique. Il est possible d'avoir une animation pour les personnages, les objets, le décor, mais aussi pour les éléments du menu.

Avec l'objet **Animation** de Unity, il est possible d'associer une séquence d'images issue de sprite 2D. Pour ce faire, il faut importer toutes les images nécessaires à l'animation une à une ou importer une *spritesheet* et la découper en plusieurs parties qui seront considérées comme plusieurs images individuelles (cf. chapitre Création et habillage de scènes - sous-section Traiter une image contenant plusieurs images).