UNITY - TP1

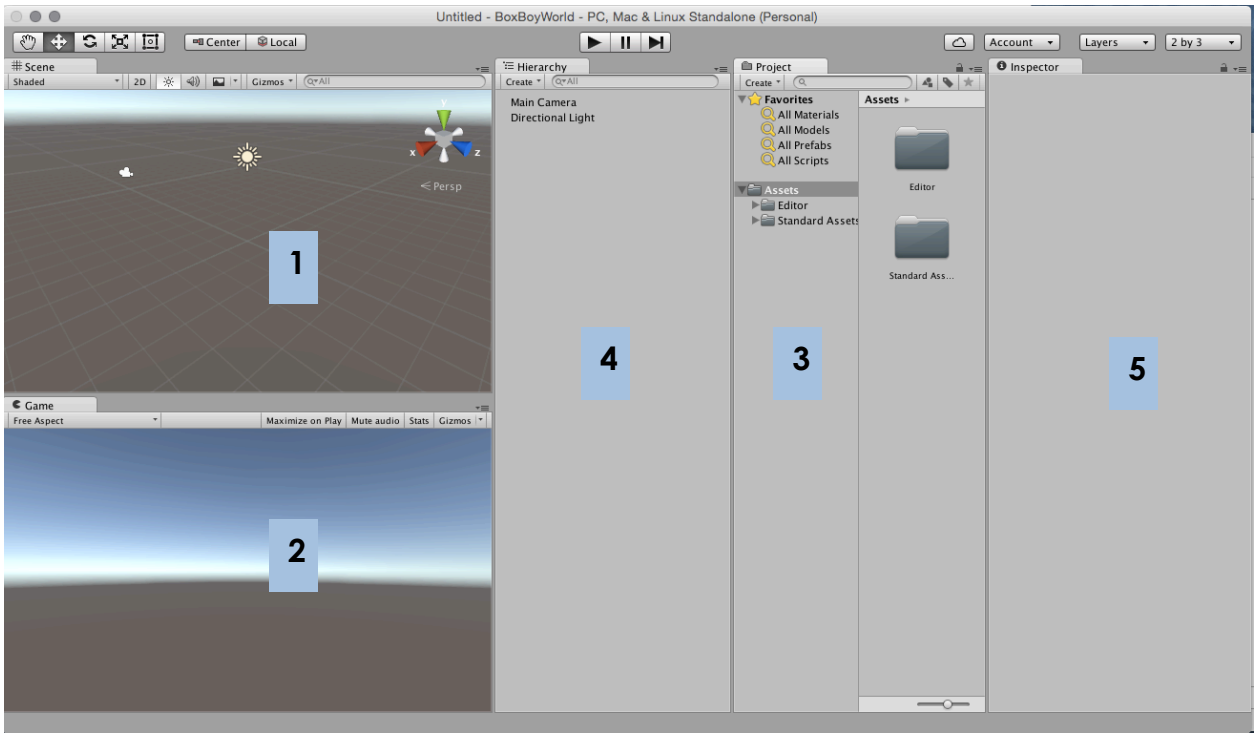
Ce TP est une introduction à l'interface et aux caractéristiques de base d'Unity (www.unity3d.com). Unity est un moteur 3D permettant de créer de jeux, des applications interactives, des animations en mettant à disposition du développeur des nombreux outils. Il est disponible pour PC et Mac, et peut publier les applications dans de multiples plate-formes telles que PC, Mac, Web, iOS, Android,….

L'interface d'Unity, très intuitive, permet d'avoir accès aux différentes ressources disponibles pour la création de l'application. Ouvrez Unity et créez un nouveau projet. Il est possible de sélectionner ensuite les packages que vous voulez inclure dans votre projet (aucun pour l'instant). Si vous avez oublié un package, vous pourrez toujours l'importer plus tard.

L'INTERFACE UNITY

Il est possible de modifier l'aspect de l'interface. Vous pouvez glisser les onglets des vues pour les réorganiser. Vous pouvez choisir le layout de la figure ci-dessous en passant par le menu ***Window > Layouts > 2 by 3***

1- La vue « Scène » est la zone d'edition d'Unity. Dans cette zone on construit graphiquement chaque scène de l'application.
2- La vue « Jeu » permet d'avoir un aperçu de l'application sans sortir de l'éditeur
3- La vue « Project » est la bibliothèque de ressources « assets » dans Unity. Il est possible d'importer des modèles 3D, de textures, créer des scripts. Pour rajouter un objet à la scène, il faut simplement le glisser de la vue Project à la vue Scène.
4- La vue « Hierarchie » liste tous les objets de la scène courante. Elle implémente le graphe de scène
5- La vue « Inspector » permet, lorsqu'un objet est sélectionné, de voir ses propriétés. Il montre aussi les paramètres de configuration pour certains outils.



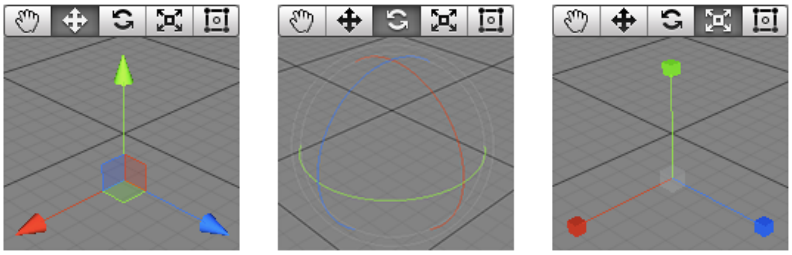
Modes de visualisation
Par défaut, la vue « Scène » est une visualisation 3D en perspective de la scène. Les vues orthographiques sont disponibles par des boutons et accessibles par le Gizmo, la boîte avec des cônes, que vous avez surement déjà remarquée :

Si vous cliquez sur la boite vous changerez au mode Perspective. Si vous cliquez sur les cônes, vous serez sur une vue 2D : le vert pour Top, le rouge pour Right, le bleu pour Front. Vous accéderez à toutes les options en faisant clic droit dans la boite.

Navigation et modification des objets
Cette barre d'outils vous permet d'explorer la scène et les objets. Il est aussi possible d'utiliser les touches Q, W, E et R.

Q – Hand Tool – cet outil vous permet de vous déplacer dans la scène. Il a plusieurs modes : ALT+clic gauche pour changer l'orientation, ALT + clic droit vous permettra de zoomer (cela doit fonctionner également avec la molette de la souris) et MAJUSCULES vous permettra d'aller plus vite.
W – Translate Tool – cet outil vous permet de déplacer l'objet sélectionné
E – Rotate Tool – cet outil vous permet de modifier l'orientation de l'objet sélectionné
R – Scale Tool – cet outil vous permet de modifier la taille de l'objet sélectionné.
L'outil rectangle permet de modifier les élements de l'UI (User Interface)

Lorsque vous utilisez les outils sur un objet, un gizmo apparaitra sur lui, vous permettant de modifier l'objet selon vos envies. Vous verrez les flèches correspondant à chaque axe dans l'espace 3D. Les valeurs de la transformation que vous effectuez seront affichées dans la vue Inspector.



http://docs.unity3d.com/Manual/PositioningGameObjects.html

Organisation de la bibliothèque

Un jeu va contenir plusieurs scènes et assets. Il est donc judicieux d'organiser la bibliothèque en plusieurs dossiers pour faciliter l'accès. La vue Projet contient tous les éléments du jeu : modèles, textures, sons, scripts, etc. Pour créer un dossier, utilisez le bouton Create dans la vue Projet, ne passez pas par l'explorateur windows ou le finder ! Vous pouvez glisser et déposer les items dans vos divers dossiers. Vous pouvez également importer et exporter des « packages » (une collection d'assets) en utilisant le clic droit.

L'hiérarchie

L'hiérarchie contient une liste de tous les objets utilisés dans la scène courante. Elle permet de sélectionner rapidement un objet sans avoir à le chercher dans la vue Scène (il peut être caché par d'autres objets par exemple). Lorsque l'objet est sélectionné dans la vue Hiérarchie il le sera aussi dans la vue Scène. Deux clics sur l'objet permettent de le centrer dans la scène. La vue Inspector, quant à elle, affichera les propriétés de l'objet sélectionné.

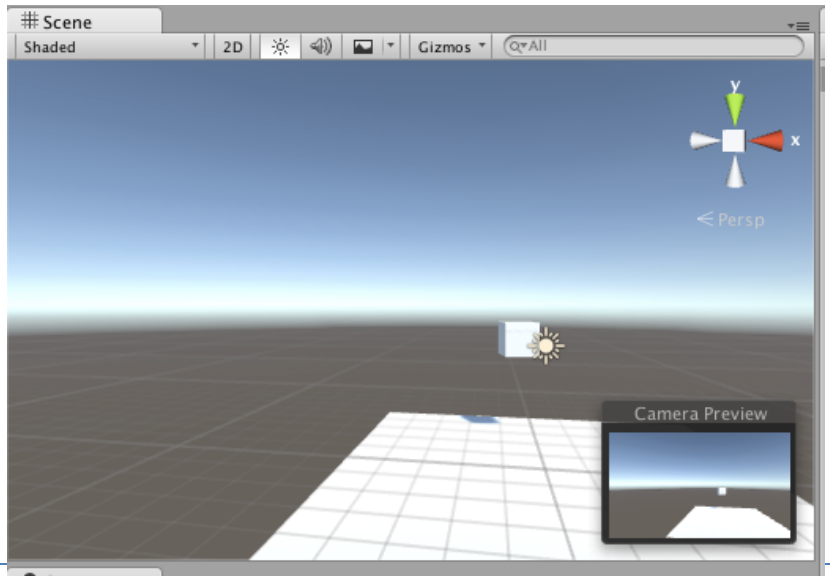
CONSTRUCTION D'UNE SCENE

Vous aurez une scène vide avec une caméra et une source de lumière, visibles dans la vue Scène et dans l'hiérarchie. Sauvegardez la scène en passant par le menu ***File > Save Scene***. Donnez lui le nom « BoxBoy ». Lorsque vous souhaiterez créer une nouvelle scène, cliquez sur ***File > New scene***.

Maintenant, nous allons ajouter quelques objets à notre scène :
* ***Game Object 3D Object > Plane***. Cela vous permettra de créer un plan 2D.
* ***Game Object > 3DObject > Cube,*** vous permettra de créer, sans surprise, un cube

Très souvent, si vous ne voyez pas les objets, cela veut dire que la caméra ne regarde pas dans la bonne direction. Sélectionnez la caméra. Vous pouvez voir la pyramide de vue. Si les objets ne sont pas dans la pyramide ils ne seront pas visibles dans l'aperçu Caméra. Vous pouvez naviguer dans la scène pour les localiser et modifier l'emplacement de l'objet ou celui de la caméra, soit avec la souris, soit en modifiant les valeurs dans l'Inspector..

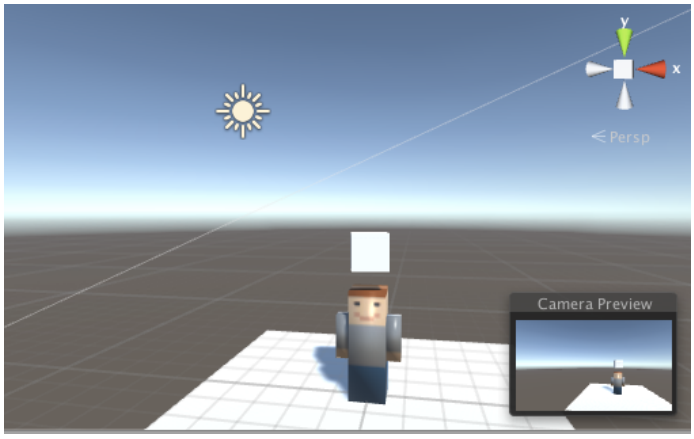
Il est important de comprendre qu'une transformation sur un objet modifie sa relation spatiale avec la caméra. Il est parfois donc intéressant de modifier la caméra plutôt que l'objet ! (le soleil tourne autour de la terre ou la terre tourne autour du soleil ?) . Si vous sélectionnez ***Game Object > Move to view*** , l'objet sélectionné se déplacera vers la caméra. Egalement pour la caméra, vous pouvez modifier votre emplacement dans la scène et cliquer ensuite sur ***Game Object > Align with view*** pour mettre à jour la caméra.



Importation d'assets

1- Téléchargez le fichier boxboy.zip et décompressez-le sur le bureau
2- Maintenant glissez le dossier boxboy (contenant boxboy.fbx et texture.png) dans l'onglet Assets
3- Ajustez dans la vue Inspector la taille du boybox (j'ai appliqué une échelle de 50 dans chaque axe)

Modifiez vos objets de manière à avoir BoxBoy sur le plan, le cube sur BoxBoy, visibles par la caméra et éclairés par la lumière. N'hésitez pas à changer de vue pour mieux localiser les objets les uns par rapport aux autres.

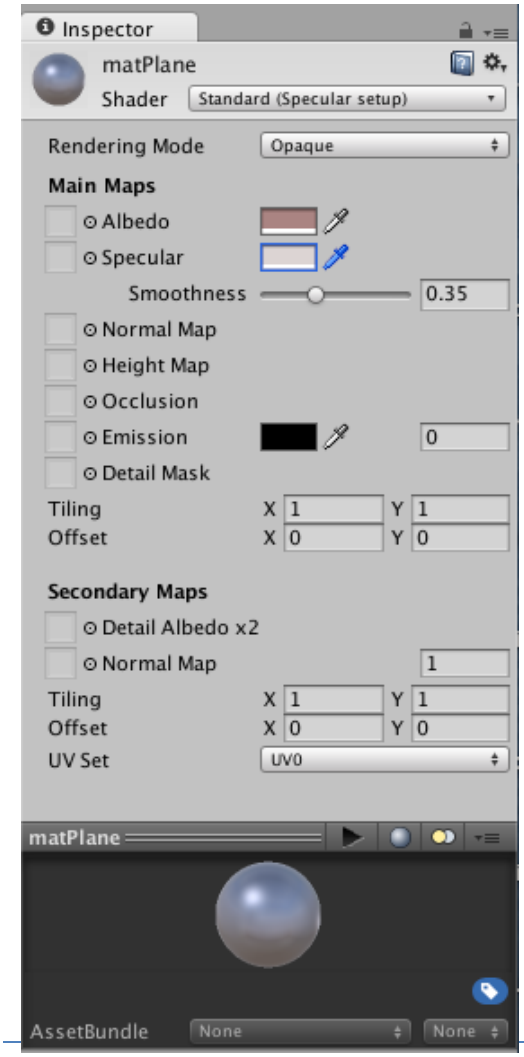


LES MATERIAUX

Il est possible de donner un aspect particulier aux objets grâce aux matériaux. Pour définir un nouveau matériau, faites clic droit sur l'onglet Assets, puis choisissez ***Create > Material***.

Vous pouvez tester les differents shaders et jouer avec les couleurs et les différents paramètres. Les Materials sont liés aux GameObjets. Les Shaders contiennent du code (la méthode permettant de réaliser le rendu de l'objet), les programmes vertex et fragment nécessaires au rendu et des propriétés. Le Material, lui, vous permet d'ajuster ces propriétés et d'assigner les assets (textures, couleurs, cubemap,…).

Glissez ensuite le matériau sur l'objet Plane dans l'hiérarchie pour l'appliquer. Créez un nouveau matériau et appliquez-le au cube.



LES SCRIPTS

Les scripts sont une partie essentielle d'Unity car ils permettent de définir les comportements des objets. Unity accepte des scripts Javascript et C#. Nous utiliserons principalement JavaScript pour créer un script qui contrôlera un objet grâce aux flèches du clavier. Créez un script vide en utilisant ***Assets > Create > Javascript***. Faites deux clics sur le nom du script pour le modifier en ouvrant l'editor MonoDevelop.

Tous les scripts ont une méthode start() et une méthode update(). La méthode start() est exécutée une seule fois, lorsque l'objet est crée tandis que la méthode update() est exécutée une fois par frame. Notre script doit vérifier si l'on a appuyé sur les touches, il sera donc placé dans la méthode update() :

```
function Update() {
        transform.Translate(Input.GetAxis("Horizontal"),-Input.GetAxis("Vertical"),0) ;
}
```

Pour modifier la position d'un objet, il est nécessaire de changer les valeurs de sa « transform ». La transform est l'entité qui stocke la position, rotation et échelle d'un objet. La fonction Translate de l'entité transform permet de modifier la position et reçoit 3 paramètres, le déplacement dans chaque axe.

Input est une classe et la fonction GetAxis retourne une valeur entre -1 et 1. Dans le cas de l'axe horizontal, la touche flèche gauche est indiquée par -1 et la droite par 1. Les axes sont prédéfinis dans la configuration d'entrée. Si vous souhaitez modifier les noms et les raccourcis clavier, allez dans ***Edit > Project Settings > Input***.

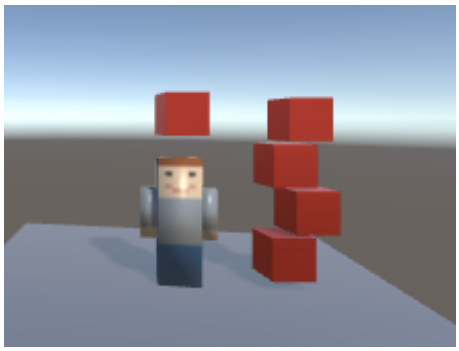
Maintenant que le script est prêt, faites un glisser-déposer du script dans BoxBoy de votre scène ! Exécute votre jeu en cliquant sur le bouton Play, vous devez pouvoir contrôler le personnage grâce aux flèches. Si vous avez effectué des rotations sur BoxBoy, vous devez modifier le script pour suivre son repère.

Important : n'oubliez pas que les changements que vous faites pendant que le jeu tourne seront perdus lorsque le jeu s'arrêtera. Vérifiez donc toujours que vous n'êtes pas en mode aperçu du jeu lorsque vous réalisez des modifications. Si vous activez l'option « Maximize on play » de l'onglet Game, cela vous arrivera peu ☺

AJOUT DE COMPOSANTS

Nous allons ajouter un composant RigidBody au cube. Ce type de composant donnera à l'objet un comportement « physique » c'est-à-dire sous l'influence de la gravité, car pour l'instant, il reste en levitation !. Sélectionnez le cube et puis, dans l'onglet Inspector, cliquez sur addComponent. Sélectionnez ensuite ***Physics > Rigidbody***. Le composant doit être visible dans la vue Inspector. Déplacez le cube pour qu'il se trouve sur la tête du bonhomme. Exécutez le jeu et observez le comportement du cube.

DUPLICATION



Lorsque l'on duplique un objet, toutes ses caractéristiques sont aussi dupliquées. Sélectionnez le cube et appuyez sur Ctrl+D (disponible aussi dans le menu Edition ou dans le menu contextuel de l'objet dans la vue Hiérarchie). Créez plusieurs cubes et déplacez-les en les empilant les uns sur les autres. Exécutez le jeu, vous verrez les cubes interagir en suivant les lois de la physique.

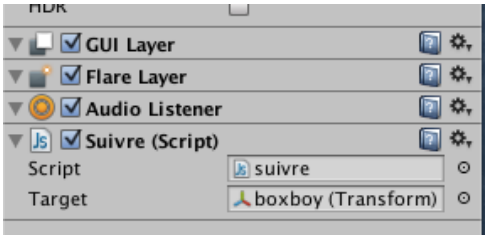
VARIABLES EXPOSEES

Les variables exposées sont des variables accessibles en dehors du script. Elles sont visibles dans l'Inspector et peuvent être connectées entre elles avec un simple glisser-déposer ! Nous allons tester cette fonctionnalité pour permettre à la caméra de suivre BoxBoy. Il existe une fonction Unity qui nous permet de récupérer la transformation d'un objet pour pouvoir « le regarder », la fonction transform.LookAt(). Créez un nouveau script appelé *suivre*, qui va recevoir une transformation qui sera ensuite appliquée à l'objet auquel le script est associé :

```
var target : Transform;

function Update () {
   transform.LookAt(target);
}
```

Attachez le script à la caméra, la variable target devient disponible. Dans la vue Hierarchy, glissez BoxBoy vers la variable target. Testez votre jeu !



Le comportement des objets est celui que vous attendiez ? Quelle est la différence entre la connexion de variables en utilisant le script précédent et la connexion des objets via le graphe de scène (relation boxboy-lumière) ?

CREATION D'UN TERRAIN

Créez une nouvelle scène dans votre projet.
Les terrains en Unity sont des maillages plats que l'on peut sculpter. Pour ajouter un nouveau terrain cliquez sur ***Game Object > 3D Object >Terrain.*** Sélectionnez le terrain, la vue Inspector change et affiche les outils permettant de modifier le terrain.

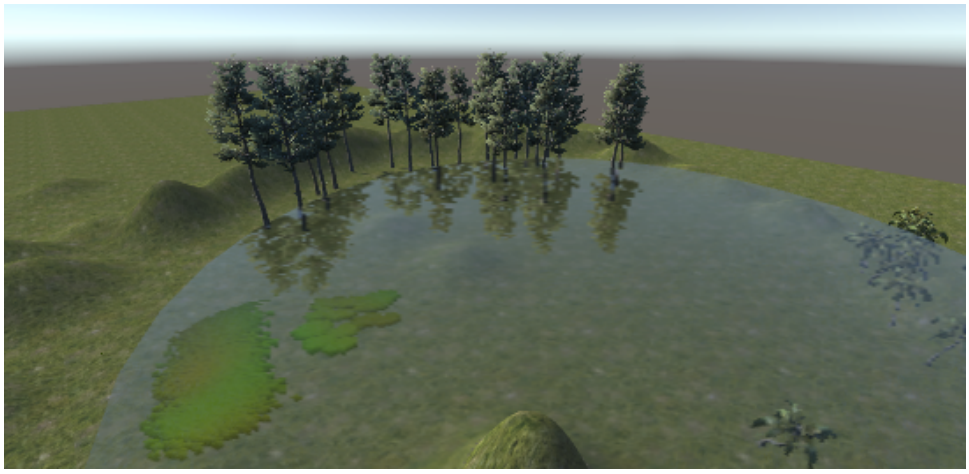
Dans le panel TerrainScript vous trouvez une barre d'outils avec plusieurs boutons :

- Raise/Lower : pour soulever ou enfoncer la géométrie du terrain en utilisant un pinceau
- Set Height : pour dessiner le terrain avec une hauteur limite
- Smooth : permet de lisser le terrain
- Paint Texture : permet d'ajouter des textures sur la surface du terrain
- Place Trees : permet de rajouter des arbres
- Paint Detail : permet de dessiner les détails du terrain comme de l'herbe
- TerrainSettings : paramètres du terrain

Créez divers reliefs sur votre terrain avec les outils Raise/Lower et Smooth. Il est conseillé de commencer par les zones importantes comme les montagnes et puis raffiner les détails Appliquez ensuite une texture. La première texture sera appliquée au terrain tout entier, les suivantes seront ajoutées en sur-couche. Cliquez sur ***Edit Textures > Add Textures*** pour voir la fenêtre de sélection de textures. Choisissez la texture GrassHill par exemple. Si aucune texture n'est disponible, importez le package Environment en faisant clic droit sur l'onglet Assets.
Améliorer votre scène en important des arbres (Place Trees), en ajoutant de l'herbe et de zones rocheuses en utilisant l'outil PaintDetail. Pour tous les outils vous pouvez modifier la brosse et la taille du pinceau.

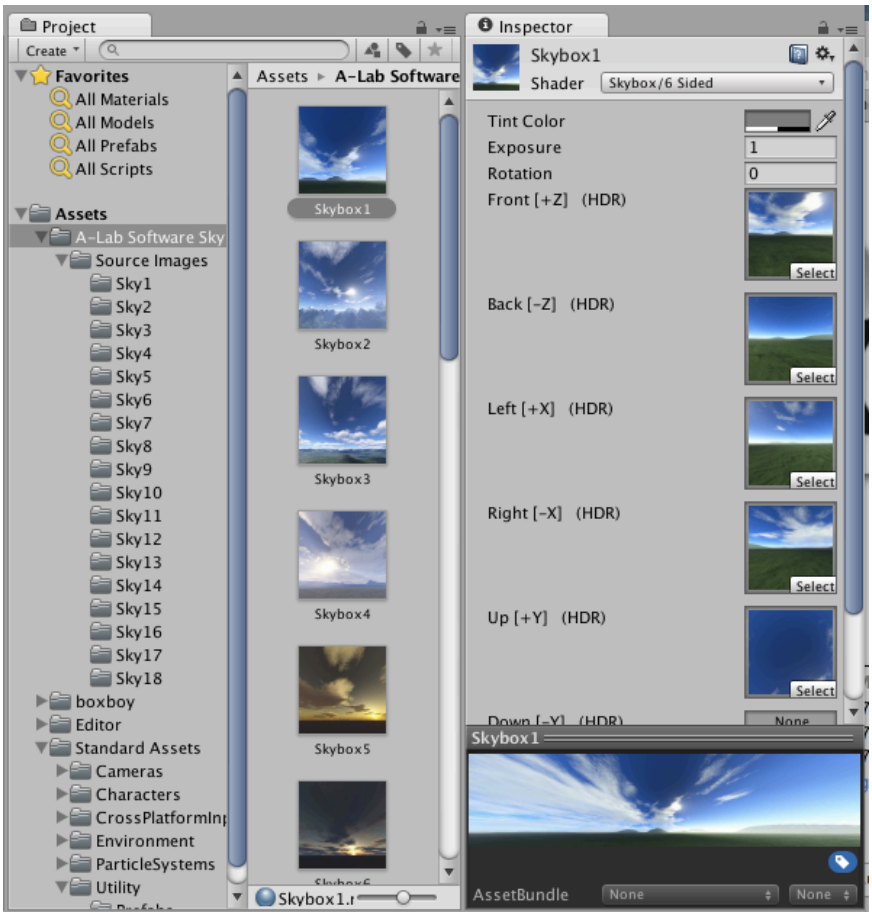
L'EAU

Rien de plus simple ☺ Importez le package Water (basic) en utilisant le menu contextuel dans la fenêtre Project ou le menu ***Assets > Import Package***. Placez ensuite l'objet sur une surface plane. L'eau est une « texture animée », elle ne déborde pas, vous ne pourrez pas créer des cascades par exemple. Vous pouvez améliorer l'effet en créant une bordure avec les outils de terrain ou avec de l'herbe :

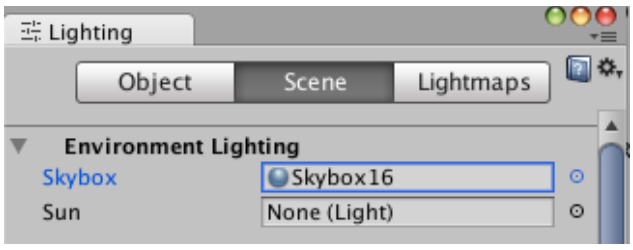


LE CIEL (SKYBOX)

Nous allons améliorer notre environnement en utilisant une skybox, c'est-à-dire un cube avec 6 images différentes permettant de simuler le ciel et l'horizon. Importez le package A-Lab_Software_Skyboxes pour avoir accès à plusieurs skybox[1].



Allez ensuite dans le menu Window > Lighting. Dans l'onglet Scene, choisissez la skybox de votre choix, le tour est joué !



Il est également possible de créer votre propre skybox. Pour cela, créez un nouveau matériau et sélectionnez le shader Skybox/6 Sided dans l'Inspector. Spécifiez ensuite les 6 textures en utilisant le bouton Select pour chaque partie ou en glissant l'image correspondante à chaque case.

[1] *http://www.alabsoft.com/unity3d.shtml*

C'est beau n'est-ce pas ? ☺

CONTROLE DE LA CAMERA

Créez un script vous permettant de contrôler la caméra avec les flèches. Si elle va un peu vite, il est conseillé de s'assurer que les objets se déplacent à une vitesse plus intuitive en l'exprimant en mètres par seconde. Pour ce faire, il faut multiplier la valeur retournée par Input.GetAxis() par Time.DeltaTime et par la vitesse de mouvement que nous souhaitons :

```
function Update () {
        var speed = 5.0;   // déplacer l'objet 5 m par seconde
        var x = Input.GetAxis("Horizontal") * Time.deltaTime * speed;
        var z = Input.GetAxis("Vertical") * Time.deltaTime * speed;
        transform.Translate(x, 0, z);
}
```

Bravo ! Vous avez complété la première étape de la découverte Unity ☺ Envoyez une capture d'écran de vos deux scènes à nancy.rodriguez@lirmm.fr.

BON WEEKEND ET À VENDREDI PROCHAIN !