COMPSCI 715
Advanced Computer Graphics

Unity Basics

Today's Mission

1. How does a game engine like Unity work?
2. What are game objects and components?
3. How do you apply this to your own project?

Schedule 1st Half

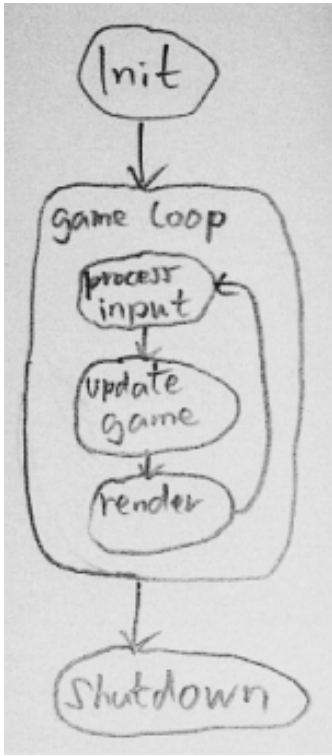
Week	Activities	Assignments
1	Tue: course outline, Wed: project topics, Thu: feedforward learning (guest lecture)	Register teams
2	Tue + Thu: exergaming (guest lectures), Wed: academic writing overview, abstracts	Abstract (1.5%)
3	Tue + Thu: Unity, Wed: writing introductions	Introduction (2.5%)
4	Tue: feedforward (guest lecture), Thu: Unity, Wed: writing about related work	Related work (2.5%)
5	Team meetings, Wed: design & implement.	1st prototype (1.5%)
6	Team meetings, Wed + Thu: demos (2.5%)	2nd prototype (1.5 %)

Mid-semester break. So far 12% of individual assignments.

Unity Resources

- Unity Documentation: http://docs.unity3d.com/
- Unity Tutorials: http://unity3d.com/learn/tutorials/modules
- Many video tutorials on YouTube, e.g. https://www.youtube.com/watch?v=9Xr5Rc9Rw6I&list=PLmQnFpk1W81tyuEySbOJ4bG6Z1BrS_0hi
- Thanks to Michael Ivanov for some illuminating figures: http://www.slideshare.net/sasmaster/unity3d-programming-5725801

Unity Introduction



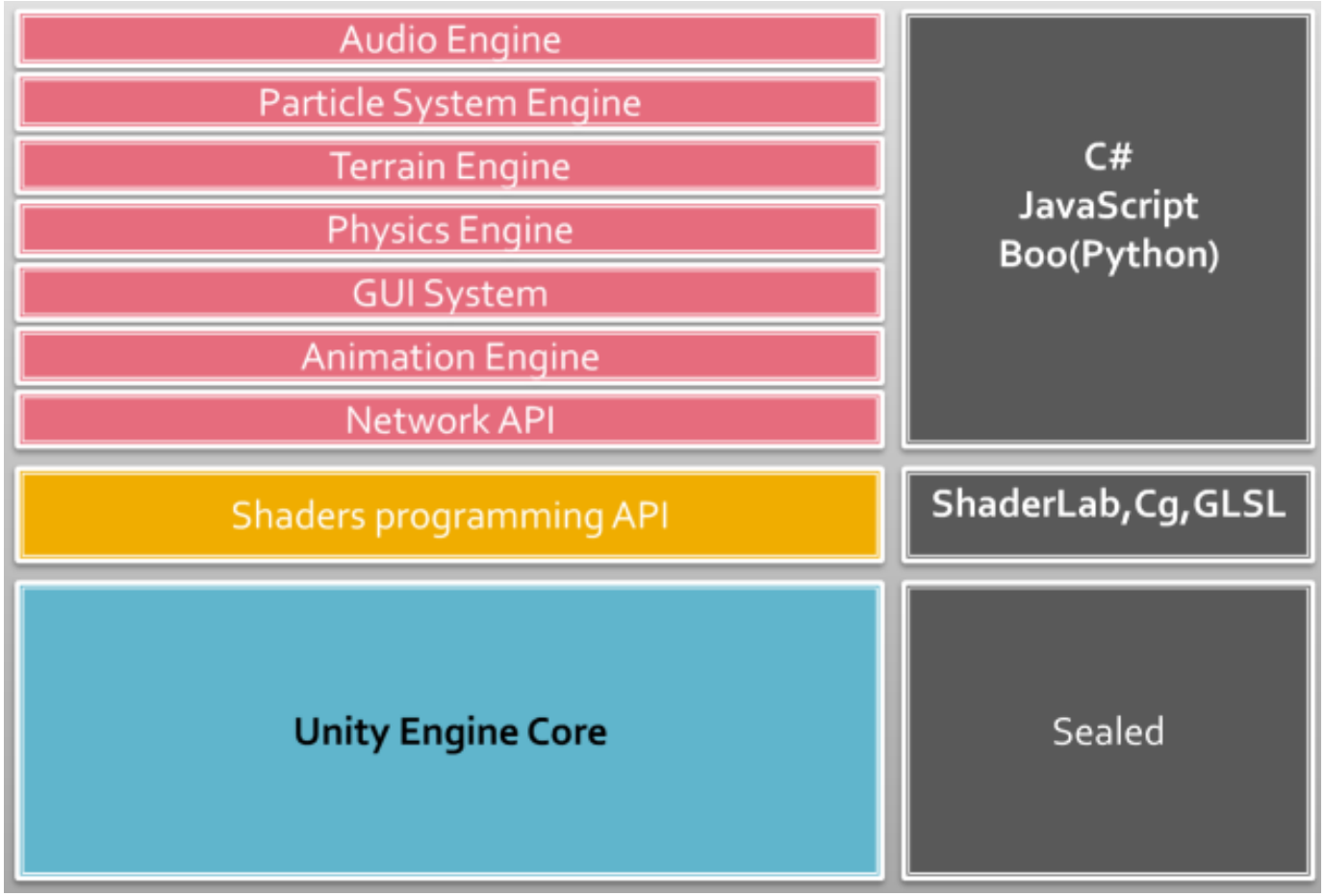
A game engine

- Abstraction: sits on top of OpenGL (ES), DirectX, …
- Complete: provides all features you need for a game, e.g. graphics, physics, sound, input, networking…
- Inversion of control: the engine runs the game
  - Specific game content/features/behavior are plugged into and managed by the engine
  - Don't call us, we call you: engine calls event handlers

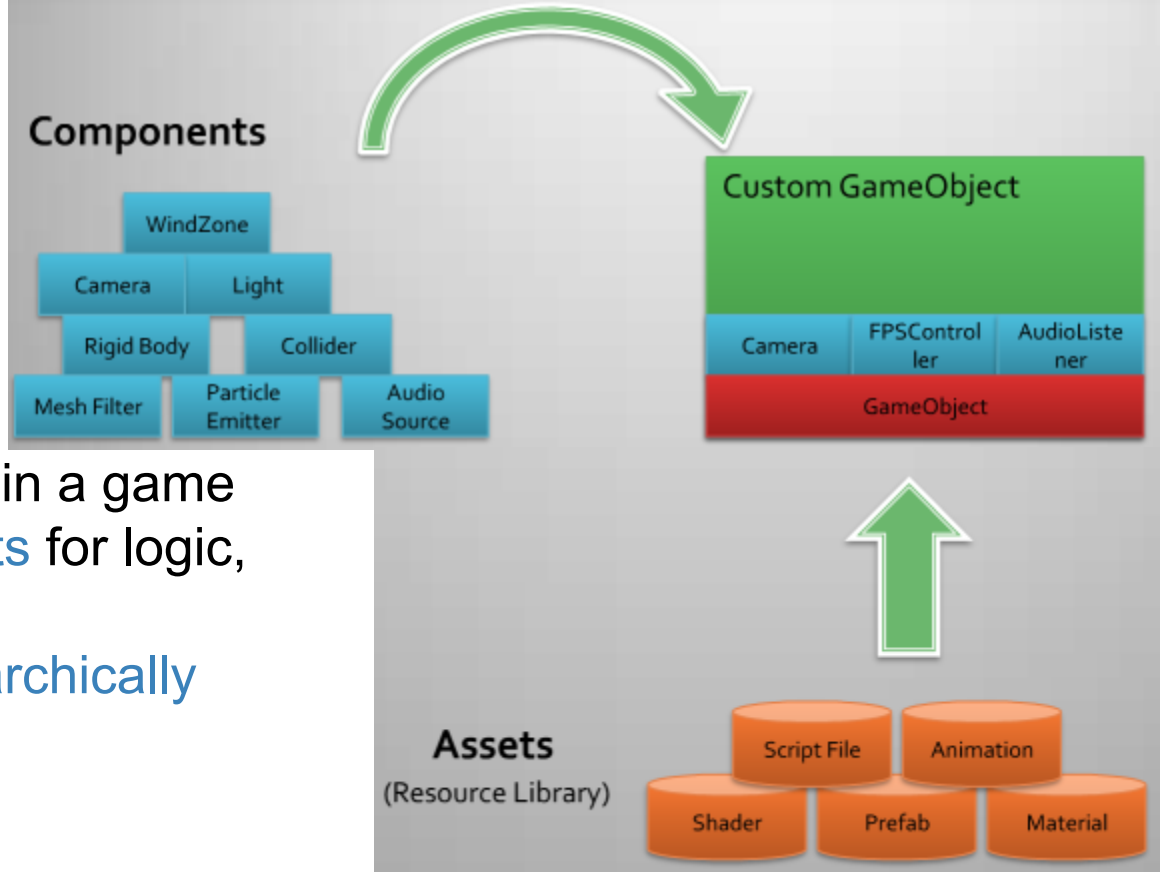
Why Unity?

- Free version available, lots of free resources
- Multi-platform: supports most mobile, desktop & console OSs, browser plugin

Unity Overview



GameObjects



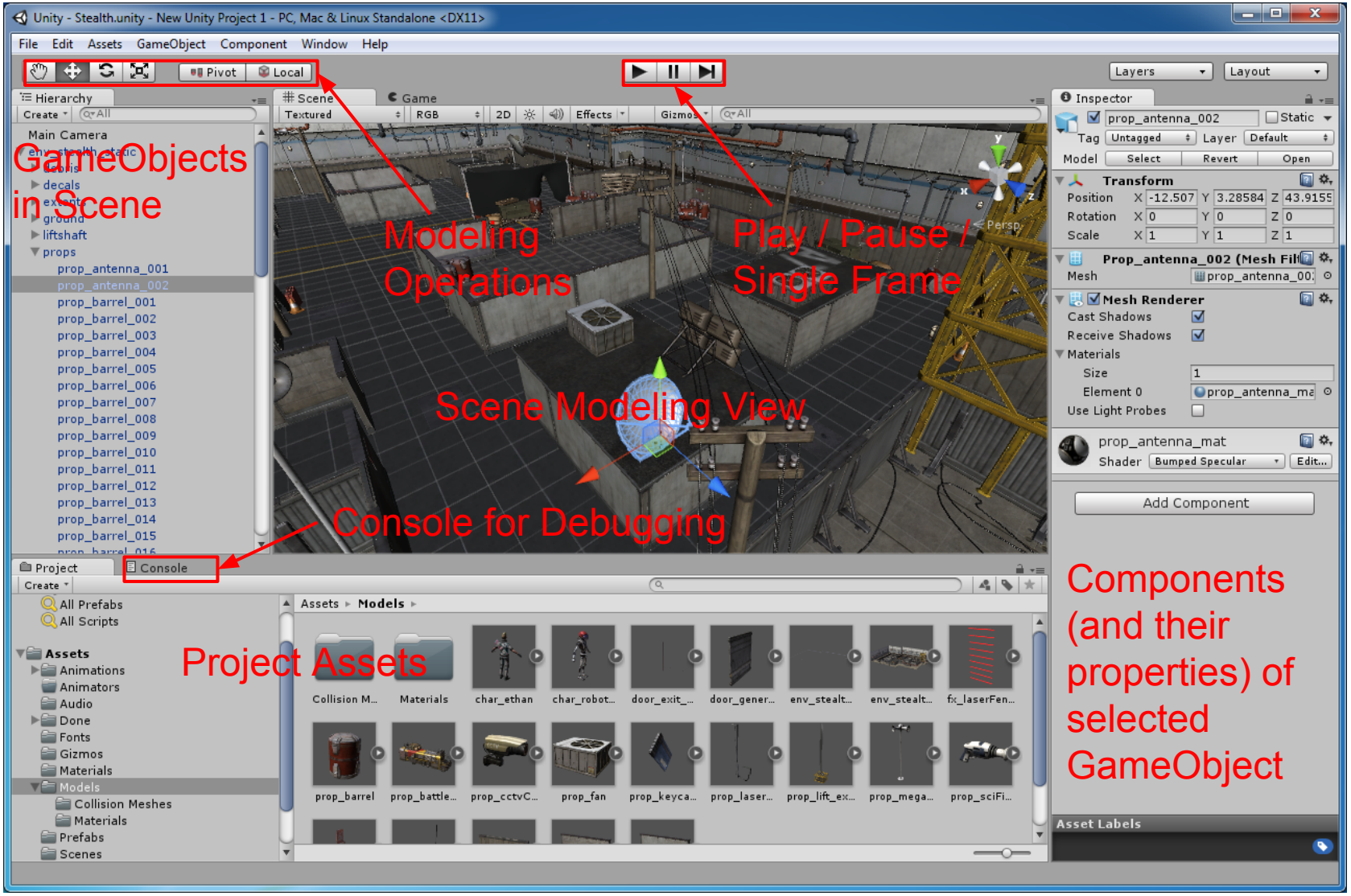
Games consist of them

- Think of visible objects in a game
- But also invisible objects for logic, state etc.
- Can be organized hierarchically in a Scene

What a GameObject can do depends on its Components

- Technically Components are themselves objects
- Are just associated with GameObject and can reference it
- Give a GameObject more features by adding components, e.g. visual appearance, physics, dynamic behavior
- Knowing Unity's capabilities means knowing the different components

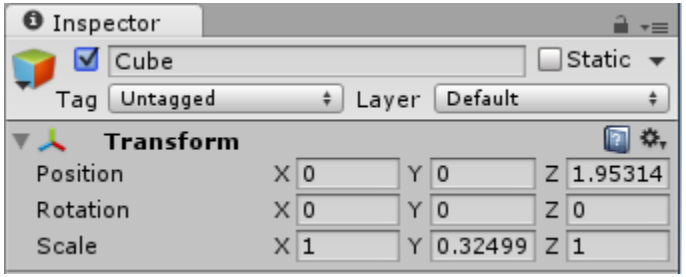
Unity GUI Overview



Creating & Transforming Objects

Use menu GameObject -> Create (e.g. Create Other -> Cube) or just drag & drop from Assets

- Object appears in current Scene
- Combine objects by dragging into other object in Hierarchy
- Name it, enable/disable it
- Put object on Layer to organize groups of objects
- Tag objects to retrieve them more easily



Transform Component

- Every GameObject has it
- Defines public properties: Position, Rotation, Scale
- Grid cells usually used to mark 1 meter

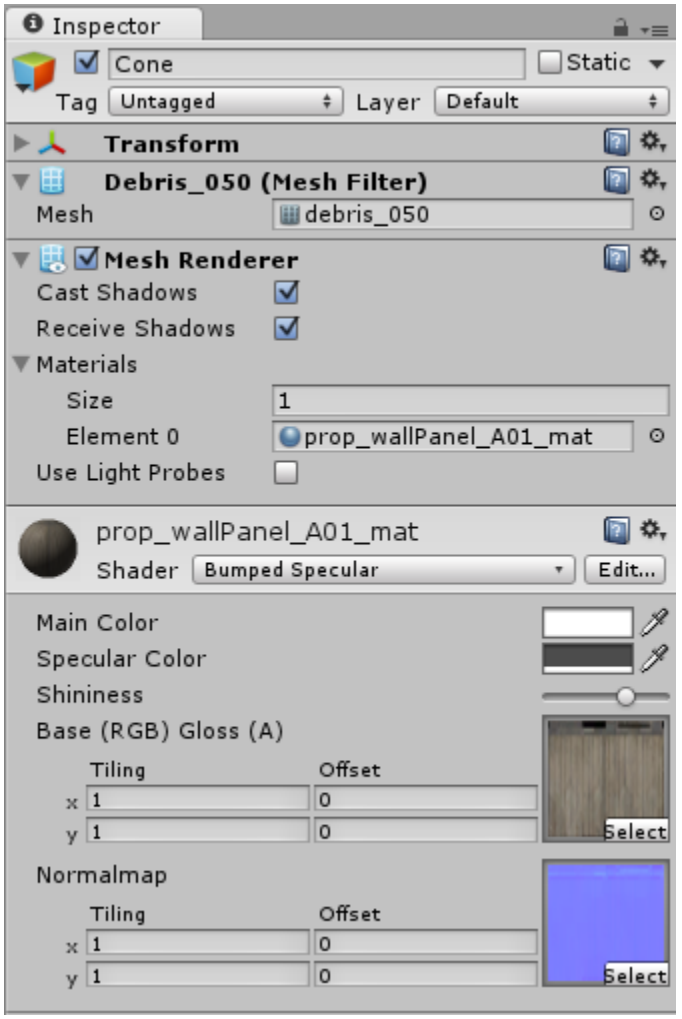
Visual Appearance

MeshFilter

- Selects a mesh for the object from Assets
- Can import new assets from asset packages or files

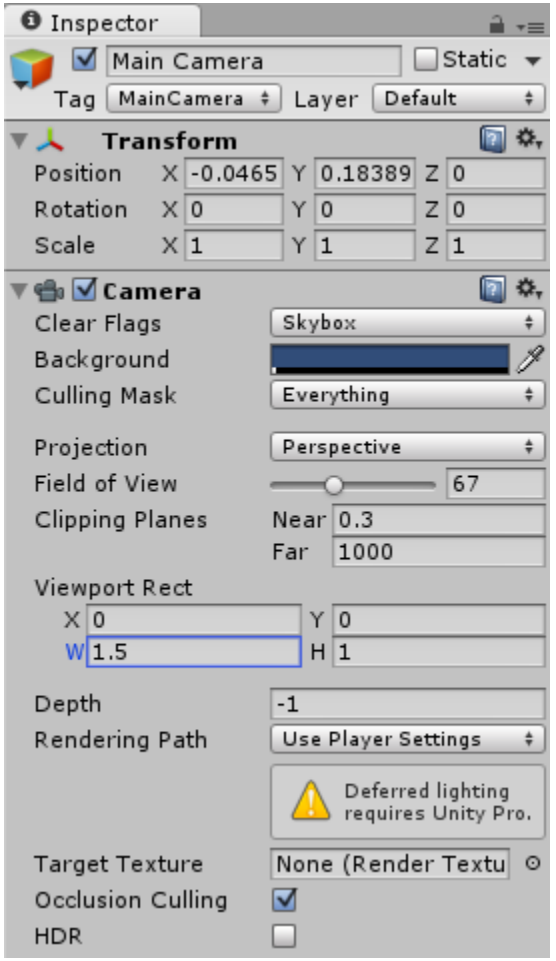
MeshRenderer

- Select Material for object, e.g. colors, textures, reflective properties...
- Select Shader to use, e.g. Diffuse, Specular...



Camera

- Camera Component makes an object a camera
- Typical camera properties:
  - Perspective/Orthographic Projection
  - Field of View: wide or narrow
  - Clipping planes: near/far visibility
  - Viewport aspect ratio
  - Culling mask: what to draw
  - Clear flags: sky color
- Note: Cogwheel -> Reset



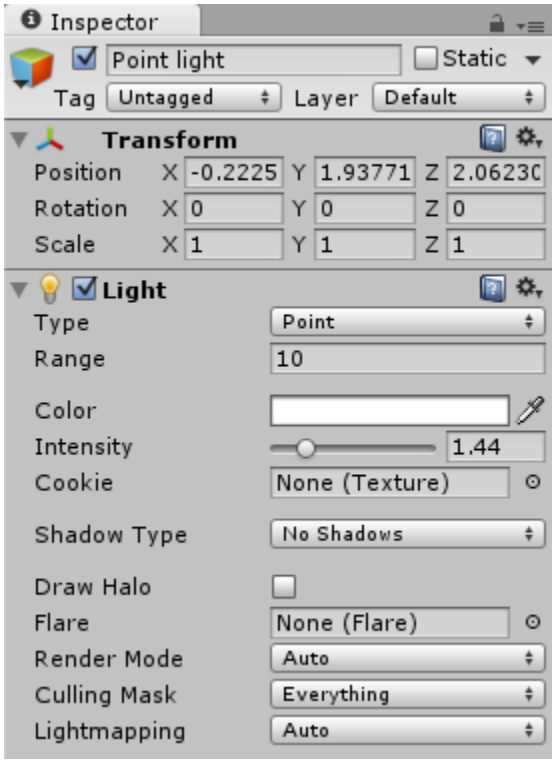
Other Visible Objects

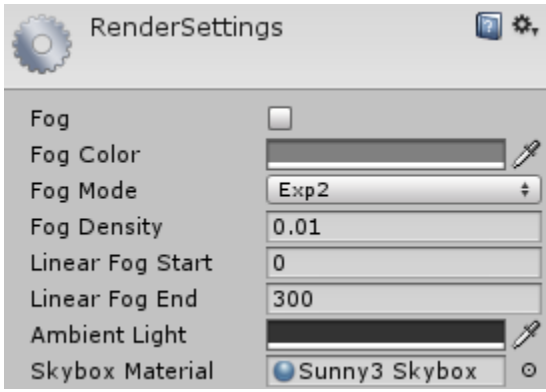
Light Component

- GameObject -> Create Other -> Directional / Point / Spot Light …
- Define light properties, e.g. color, intensity

SkyBox Component

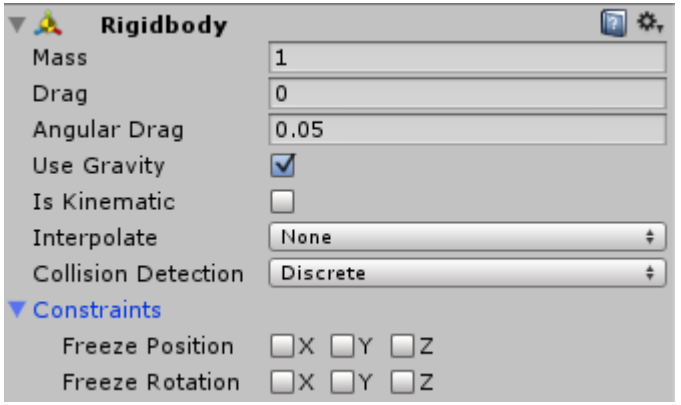
- Textured environment around your scene
- Edit -> Render Settings -> Skybox Material (or add SkyBox component to camera)
- Get Skybox material from Assets -> Import Package -> Skyboxes





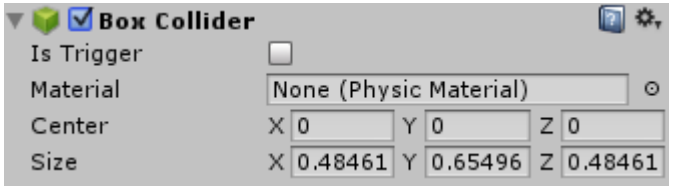
Physics

RigidBody Component



- Makes object subject to physical forces, e.g. gravity or impact
- Define physical properties, e.g. mass & drag
- Define physical constraints, e.g. object can only move vertically
- Test by clicking play to start game engine

Collider Components



- E.g. BoxCollider, SphereCollider etc.
- Detects collision between objects
- Physics only works if all involved objects have colliders