CE318/CE818: High-level Games Development

Lecture 10: 2D Games

Diego Perez

dperez@essex.ac.uk
Office 3A.527

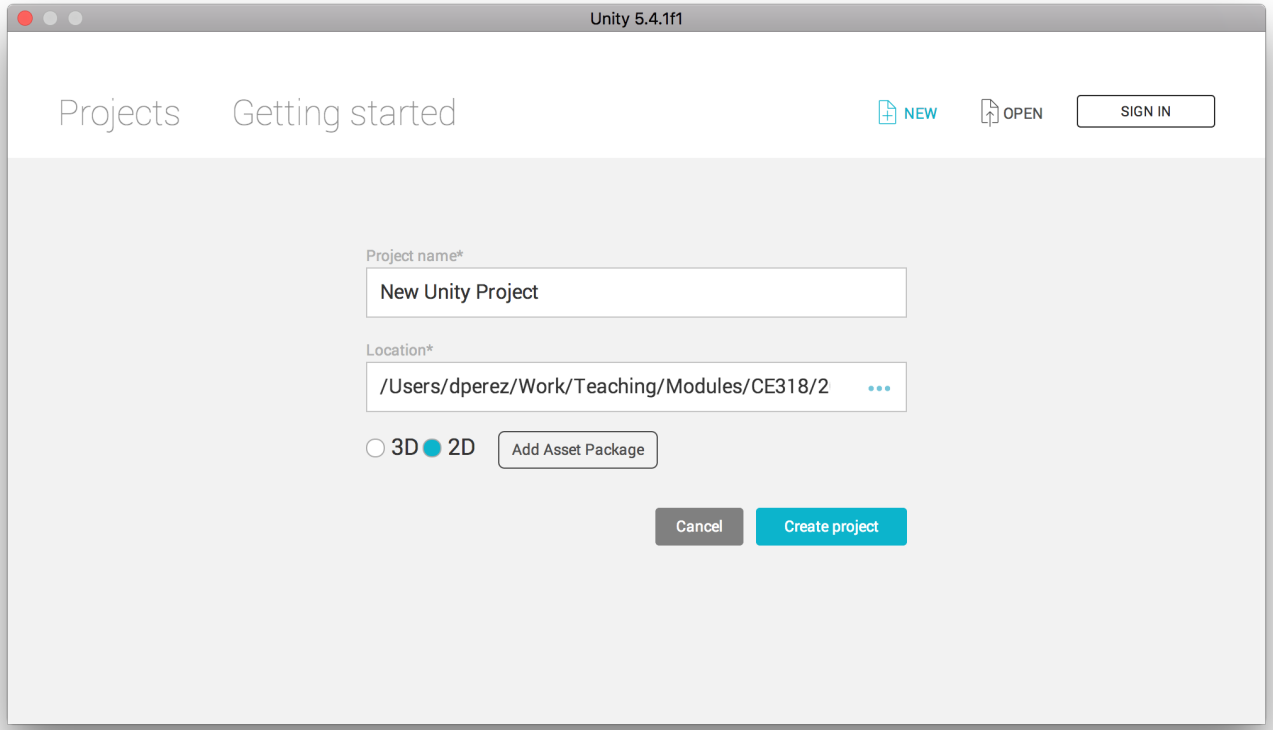
2016/17

Outline

Outline

The 2D Mode

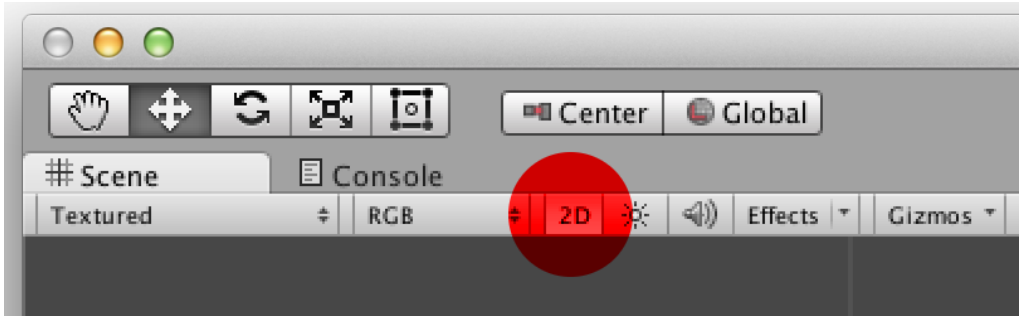
When creating a project in Unity, it's possible to set up the editor for 2D Mode:



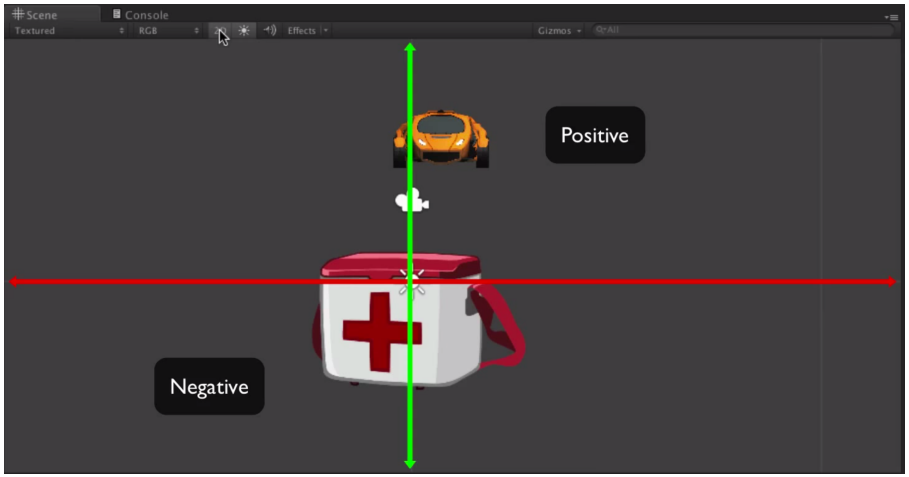
In this mode, Unity automatically sets some parameters for 2D (such as textures and sprites).

The Scene View - 2D Mode

The Scene View also has a toggle that switches the view between 2D and 3D.



When the 2D view is set, the *depth* coordinate is ignored, making 2D game development much easier. The camera and the view are locked in an orthographic view:



2D Graphics

Graphic objects in 2D are known as **Sprites**.

Sprites are essentially just standard textures but there are special techniques for combining and managing sprite textures for efficiency and convenience during development.

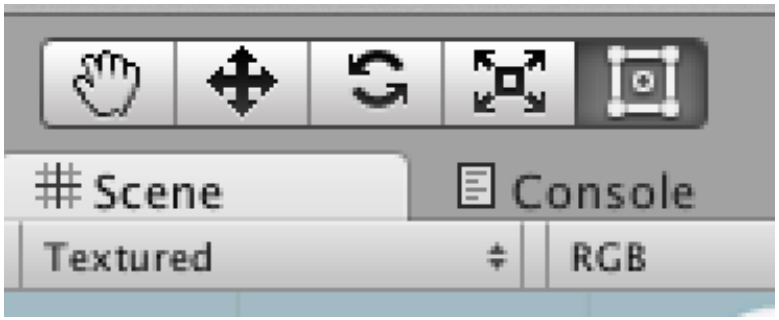
Unity provides a *Sprite Editor* for manipulating sprites. Of special use is to edit an image to separate it into several components. For instance, to keep the arms, legs and body of a character as separate elements within one image.

Sprites are rendered with a **Sprite Renderer** component rather than the Mesh Renderer used with 3D objects.

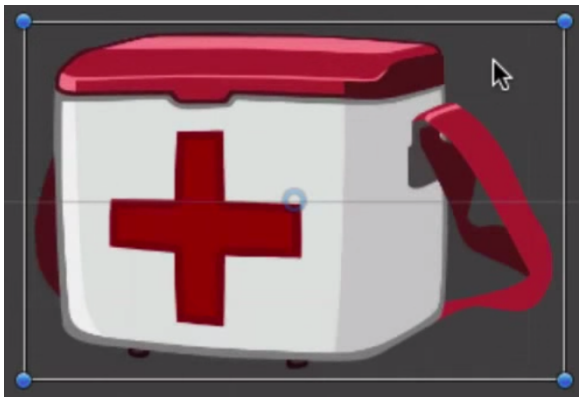
2D Transformations

There are four controls at the top left corner to manipulate sprites in the scene:

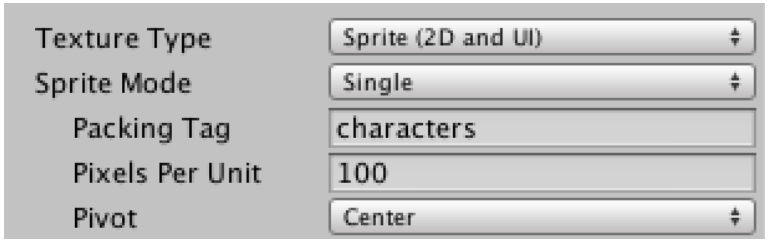
- Drag (Hand icon, shortcut Q): To move around the scene.
- Translate (Shortcut W): Changes position of a sprite.
- Rotate (Shortcut E): Rotates a sprite.
- Scale (Shortcut R): Scales a sprite.



- The last button (Shortcut T) allows to make all these changes on *Scene View* as shown.

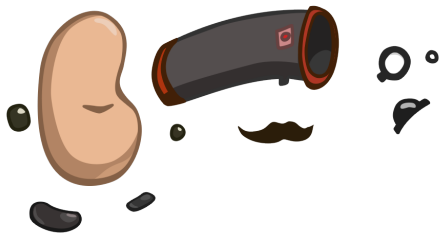


The Sprite Type



When an image is **imported** as a sprite, the *Texture Type* must be set to *Sprite (2D and UI)*.
The *Sprite Mode* can take two values:

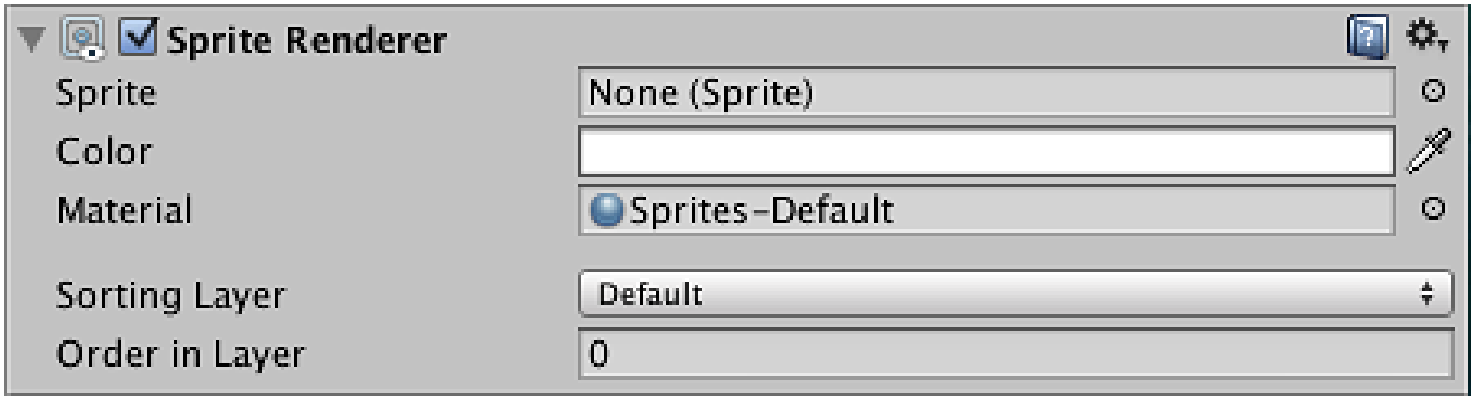
- Single: The image is composed of 1 only element.
- Multiple: The image contains multiple elements in the same file.



Pixels to Units defines the number of pixels from the sprite that will correspond to 1 unit in world space.
Pivot determines the position of the pivot in the sprite. The pivot is the point in the image where the sprite's local coordinate system originates.

The Sprite Renderer

The Sprite Renderer component is in charge of drawing the sprite on the screen.



- Sprite: The Sprite object to render.
- Color: Vertex color of the rendered mesh.
- Material: Material used to render the sprite.
- Sorting Layer: The layer used to define this sprites overlay priority during rendering.
- Order in Layer: The overlay priority of this sprite within its layer. Lower numbers are rendered first and subsequent numbers overlay those below.

Outline

2D Physics

Unity has a separate physics engine for handling 2D physics so as to make use of optimizations only available with 2D. The fundamentals in both physics systems (2D and 3D) are equivalent. Most 2D physics components are simply flattened versions of the 3D equivalents, but there are a few exceptions. These components are:

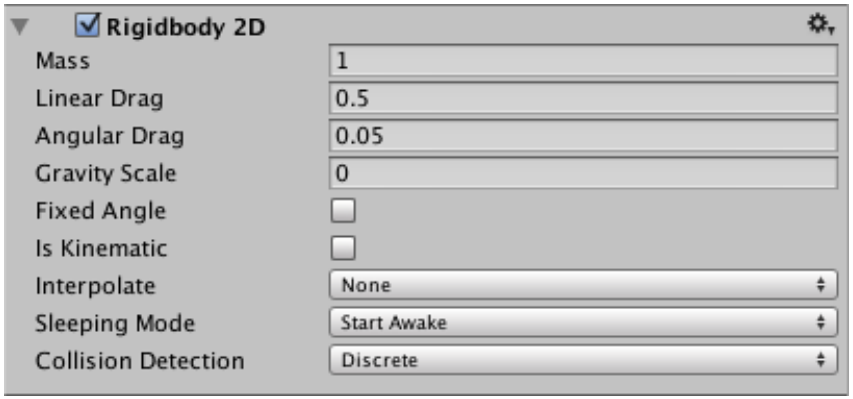
- Rigidbody 2D.
- Collider 2D.
- Hinge Joint 2D.
- Spring Joint 2D.
- Distance Joint 2D.

2D versus 3D physics:

- 2D and 3D physics can be used together (they can be present in the same scene) but they won't interact with each other.
- There is no concept of depth in 2D: All 2D physics occur in the XY plane, with $Z = 0$. All physics interactions take places at Z axis position 0.
- Game objects in 2D can only move along the XY axis and rotate around the Z axis.

Rigidbody 2D

Attaching a rigidbody 2D component to a game object places the object under the control of the physics 2D engine: it will respond to forces applied through the scripting API.



- Mass: mass of the object.
- Linear Drag: Drag coefficient affecting positional movement.
- Angular Drag: Drag coefficient affecting rotational movement. Both drag parameters slow the speed of the movement, working against the force that moves the object.
- Gravity scale: Degree to which the object is affected by gravity.
- Fixed angle (`bool`): Can the rigidbody rotate when forces are applied?
- IsKinematic (`bool`): Is the rigidbody moved by forces and collisions? See next slide.

Kinematic game objects

If a game object is kinematic, the rigidbody **does not** react to gravity or collisions. A kinematic body is meant to be moved **only** by changing its transform.

This check can be set dynamically on scripts:

```
1 rigidbody2D.isKinematic = false;
```

This is typically used to keep an object under non-physical script control most of the time but then switch to physics in a particular situation.

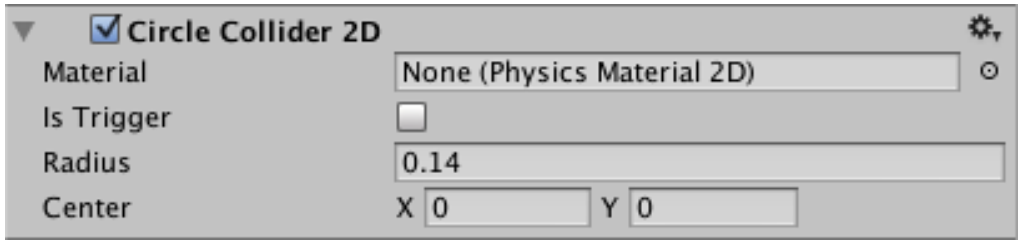
For example: a player might normally move by walking (better handled without physics) but then get catapulted into the air by an explosion or strike. Physics can be used to create the catapulting effect if you switch off "Is Kinematic" just before applying a large force to the object.

Note: kinematic bodies may have a collider component attached, and then it will react to collisions.

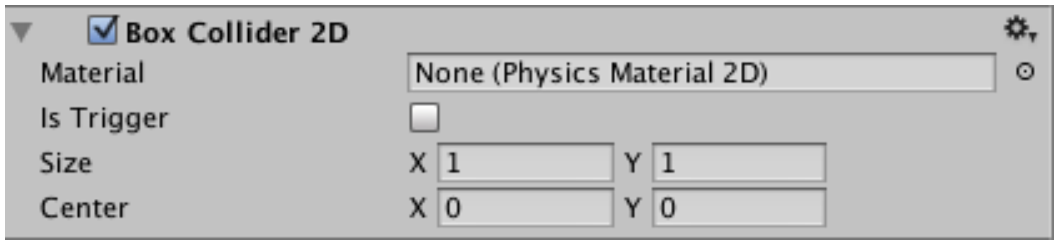
Collider 2D (1/3)

2D Colliders allow game objects to have physical shape in the scene's 2D environment. They define an approximate shape of the object that will be used by the physics engine to determine collisions with other objects. The colliders that can be used with a rigidbody2D are:

- Circle Collider 2D: A circle with a given position and radius in the local coordinate space of a Sprite.

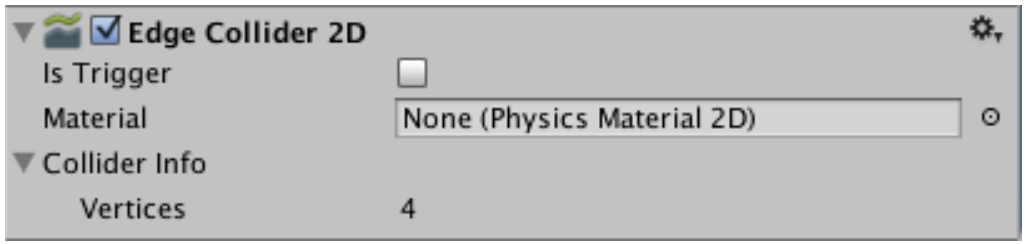


- Box Collider 2D: A rectangle with a given position, width and height in the local coordinate space of a Sprite. Important: the rectangle is axis-aligned (edges are parallel to the X or Y axes).

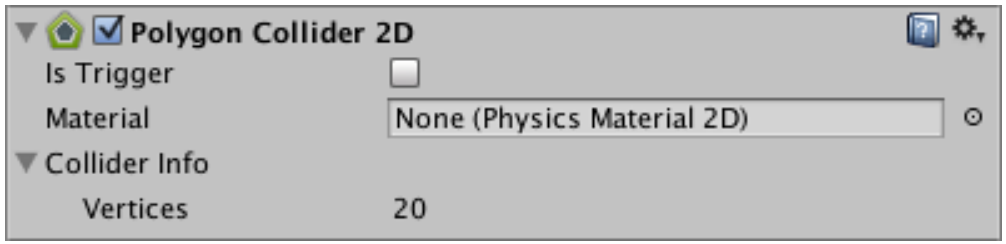


Collider 2D (2/3)

- Edge Collider 2D: The colliders shape is defined by a freeform edge made of line segments. it can be used to fit the shape of the Sprite graphic with great precision. Also, a good use for this is to make a single solid surface, instead of a series of 2D colliders, even if several sprites are used to create the surface.



- Polygon Collider 2D: The colliders shape is defined by a freeform edge made of line segments that **must** completely enclose an area.



The Polygon Collider 2D can be edited manually but it is often more convenient to let Unity determine the shape automatically. You can do this by dragging a sprite asset from the Project view onto the Polygon Collider 2D component in the inspector.

Collider 2D (3/3)

Colliders set as **triggers** will **not** participate in physical collisions. If a collision happens with a trigger 2D collider, the following messages will be sent:

- `void OnTriggerEnter2D (Collider2D other)`: Sent when another object enters a trigger collider attached to the game object.
- `void OnTriggerExit2D (Collider2D other)`: Sent when another object leaves a trigger collider attached to the game object.
- `void OnTriggerStay2D (Collider2D other)`: Sent each frame where another object is within a trigger collider attached to the game object.

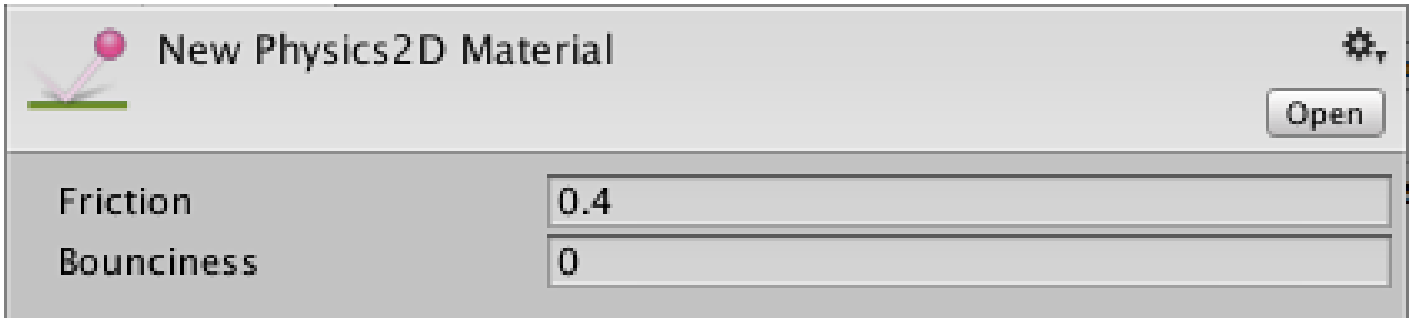
However, a 2D collider not set up as a trigger will participate in physical collisions, sending the following messages:

- `void OnCollisionEnter2D (Collision2D other)`: Sent when an incoming collider makes contact with this object's collider.
- `void OnCollisionExit2D (Collision2D other)`: Sent when a collider on another object stops touching the object's collider.
- `void OnCollisionStay2D (Collision2D other)`: Sent each frame where a collider on another object is **touching** the object's collider.

As with standard collisions, one of the objects **must** have a rigidbody2D. Typically, the trigger is static and the rigidbody2D moves and goes through it.

Physics Material 2D

A Physics Material 2D is used to adjust the friction and bounce that occurs between 2D physics objects when they collide.

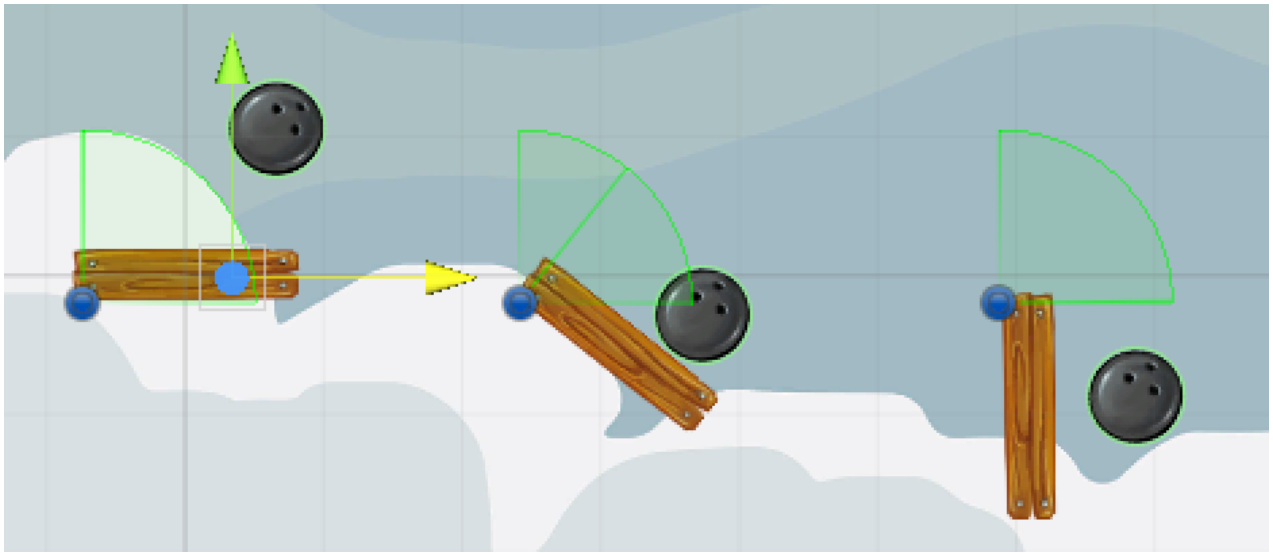


- Friction: Coefficient of friction for this collider.
- Bounciness: The degree to which collisions rebound from the surface. A value of 0 indicates no bounce while a value of 1 indicates a perfect bounce with no loss of energy.

Hinge Joint 2D

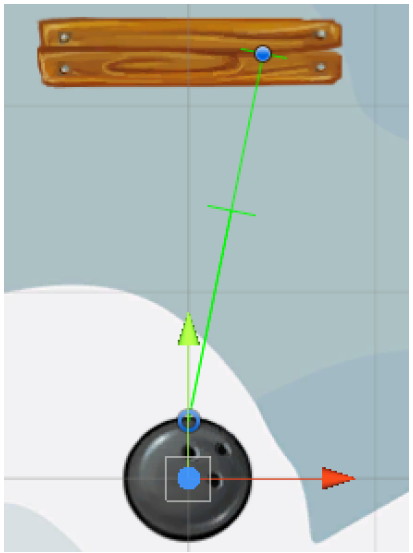
The Hinge Joint 2D component allows a Sprite object controlled by rigidbody physics to be attached to a point in space around which it can rotate.

The rotation can be left to happen passively (in response to a collision, say) or actively powered by a motor torque provided by the joint itself. The hinge can also be set up with limits to prevent it from making a full rotation.



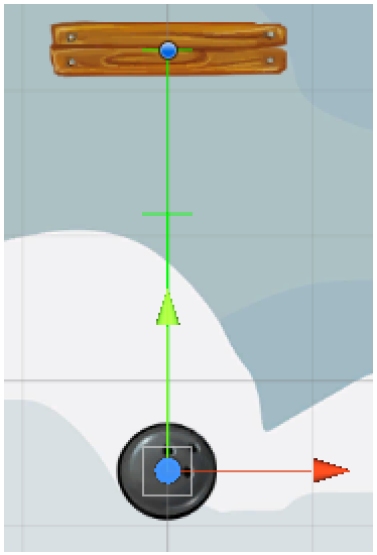
Spring Joint 2D

The Spring Joint 2D component allows two Sprite objects controlled by rigid-body physics to be attached together as if by a spring. The spring will apply a force along its axis between the two objects, attempting to keep them a certain distance apart.



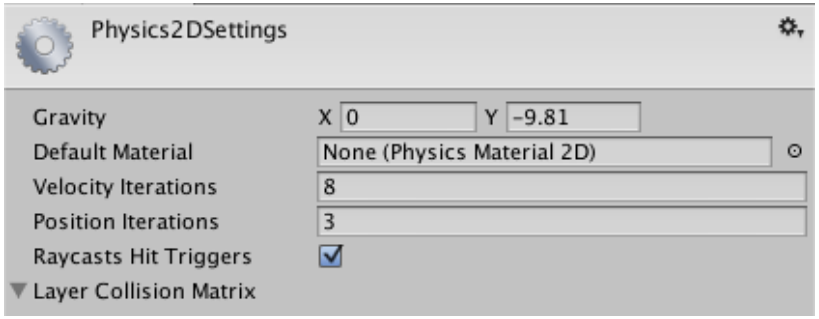
Distance Joint 2D

The Distance Joint 2D component allows two Sprite objects controlled by rigidbody physics to be attached together to keep them a certain distance apart. Note that unlike the Spring Joint 2D, the Distance Joint applies a hard limit rather than a gradual force to restore the desired distance.



Physics 2D Settings - The physics 2D Manager

The physics 2D Manager allows to edit the global Physics 2D settings that affects all sprites in the game.
Edit $\rightarrow$ *Project Settings* $\rightarrow$ *Physics2D*.



- Gravity: The amount of gravity applied to all Rigidbody2D objects. Generally, gravity is only set for the negative direction of the Y-axis.
- Default Material: The default Physics Material 2D that will be used if none has been assigned to an individual collider.
- Position/Velocity iterations: The number of iterations made by the physics engine to resolve position/velocity changes. Higher numbers result in more accurate physics but at the cost of CPU time.
- Raycasts Hit Triggers: If enabled, any Raycast that intersects with a Collider marked as a Trigger will return a hit. If disabled, these intersections will not return a hit.

Outline

Lab Preview

Assignment Part II and Merry Christmas.