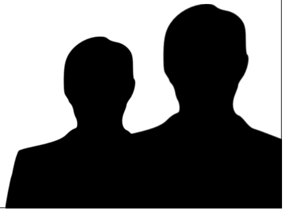
Welcome To Laser Defender



This Section is Work in Progress

We're working hard to make this section available as quickly as we can. Hope you enjoy the preview :-)

We'll be adding videos as we make them in the coming days.

What Laser Defender Teaches

Animation basics

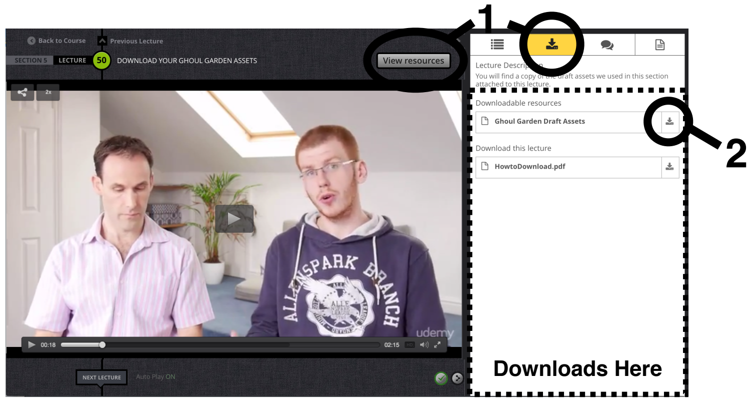
Using Trigger colliders

Layers and Sorting Layers

Introduction to Particle Systems

GDD

Your Laser Defender Assets



Section notes

Importing the menu system

Importing the menu system

- Open our previous game and import the menu system
- Create a unity package
- Import package into Laser Defender
- Alternatively, we can use the unitypackage from the section bundle at the beginning of this section

Import a Menu System

Create a unity package from your previous game

(or use the one provided in the asset pack)

Import it into Laser Defender

Importing the menu system

Imported our menus from a previous game or from the bundle

We added a blank scene to begin

Ready to go!

A Starship we can control

A Starship we can control

Find a suitable sprite asset

Import into our game

Create a Player Controller Script to move it

Restrict the movement to the playspace

Finding a suitable asset

Found on http://opengameart.org

We use Kenney's (http://www.Kenney.nl) Public

Domain sprite assets:

http://opengameart.org/content/space-shooter-redux

Importing the sprite into our game

Add it to Unity's assets

Change the mode to sprite (2D and UI)

drag it into our scene

Add a PlayerController script to the ship

Add a PlayerController script to the ship

Move the ship with the arrow keys

Make the speed adjustable in the editor

The PlayerController script

Uses **`Input.getKey()`** and **`transform.position`**

Make sure movement is independent of framerate

using **`Time.deltaTime`**

Restricting the Spaceship's position

Restricting the Spaceship's position

We don't want the spaceship to go outside the playspace

We will check the position when moving and restrict it to something sensible

Restricting the Position

Use **Mathf.clamp()** to restrict movement

Use **Camera.ViewportToWorldPoint()** to work out the boundaries of the playspace

Creating the enemies

Creating the enemies

Creating the enemy prefab

Create an EnemySpawner that will generate enemies at runtime

Make the EnemySpawner generate a single enemy on start

Create the Enemy Prefab

Use the bundled art assets to create an enemy prefab

Building the spawner I

Spawner is an empty GameObject with a script attached to it.

The script has a reference to the Enemy prefab

The Start Method calls `Instantiate()` to create an enemy

Building the spawner II

We child the new Enemy to an EnemyFormation

This keeps our scene hierarchy tidy and helps us find what we want

The Spawner script will need a reference to that object too.

Creating the enemies

Creating the enemy prefab

Create an EnemySpawner that will generate enemies at runtime

Make the EnemySpawner generate a single enemy on start

Creating Enemy Positions

Creating Enemy Positions

Create a position within the EnemyFormation

Use **`OnDrawGizmos()`** to show the position

Turn the position into a prefab

Change the spawning script to keep track of positions

Create the Position

Child an empty game object to EnemyFormation

Add a script and use **`OnDrawGizmos()`** to show the position while editing

Create your formation

Turn the position into a prefab

Add several positions from their prefab to the EnemyFormation

Be creative about the formation

Spawning multiple enemies

Loop over every child object

Grab their transform

Spawn an enemy on top of every position

Creating Enemy Positions

Create a position within the EnemyFormation

Use **`OnDrawGizmos()`** to show the position

Turn the position into a prefab

Change the EnemySpawner script to spawn an enemy on every position

Moving the enemy formation

Moving the enemy formation

Add a Gizmo to show the formation while editing

On the Enemy formation’s update, move it left or right to leave the player no space to hide

Make sure that the formation doesn’t leave the bounds of the playspace

Showing the Formation in the Editor

We can use Gizmos again

We define the width and height of the Formation

We draw lines around the boundary

Move the formation side to side

On every `Update()` change the position in the EnemyFormation script

When it reaches the edge, reverse the direction of travel

Moving the formation

Use **`transform.position`** to move the formation

We use **`Time.deltaTime`** to ensure the movement is smooth

Moving the enemy formation

Add **Update()** to the EnemyFormation to move it side to side

Add a Gizmo to show the formation while editing

Make sure that the formation doesn’t leave the bounds of the playspace

Fixing the formation movement

Fixing the formation movement

There is an issue when the enemy formation gets stuck against one of the edges

We need to fix it for the game to be playable

Why does this happen?

Figure out why the enemy formation gets stuck on the edge

What is the assumption that we are making in the code that causes this issue?

Fixing the issue.

We changed the conditional so that instead of toggling **movingRight**, we set it to the right value.

This fixes the issue.

Spawning projectiles

Spawning Projectile

Player object should spawns laser when [space] is pressed

Create a laser prefab

We use **`Instantiate()`** to create a new one

We give the projectile velocity

Create a laser prefab

Create a laser prefab from the bundled assets

When the player presses [space] create a new instance of the laser prefab

Creating the laser prefab

Add sprite, create prefab

Add public field to Player and pass in the prefab

When the space key is pressed, `Instantiate()` a new laser from its prefab

Making the laser shot move

Add a Rigidbody2D to the laser prefab

Fix the angle and remove drag and gravity

From the player set `laser.rigidbody.velocity`

As a bonus, offset alternative shots to make it look like the ship has two guns

Limiting the firing rate

Use `GetKeyDown()` and `GetKeyUp()` to call `InvokeRepeating()` and `CancelInvoke()`

To avoid multi-shot bug, make sure that the initial delay is greater than 0.0 for `InvokeRepeating()`

Destroying the laser shots

Add a Trigger Collider outside of the playspace

Attach a Shredder script that destroys all the objects that enter the trigger

Add a BoxCollider2D to the laser prefab

Spawning Projectile

Player object should spawns laser when [space] is pressed

Create a laser prefab

We use **`Instantiate()`** to create a new one

We give the projectile velocity

?

Spawning Projectile Review

Shooting enemies

Shooting Enemies

Enemies will respond to the projectile hitting them.

We use *Kinematic Rigidbody Triggers* for the enemies

On trigger, enemy takes damage according to projectile component

Defining the projectile behaviour

Create a **Projectile** script that has a public **damage** field

Add the script as a component of our lasers

Detect laser collisions

Log a message when a projectile hits an enemy.

Bonus points if you can figure out how to log messages **only** when hit with projectiles

Hint: **`gameObject.GetComponent<Projectile>()`** will return the Projectile component, if it exists

Getting the damage from the lasers

We use **`OnTriggerEnter2D()`** to detect collisions

We check that the thing we bumped into has a Projectile component.

If it does, we damage ourselves and call the **`Hit()`** method of the projectile

Shooting Enemies

Enemies will respond to the projectile hitting them.

We use *Kinematic Rigidbody Triggers* for the enemies

On trigger, enemy takes damage according to projectile component

Enemies shooting back

Enemies shooting back

Enemies will randomly shoot back with a tuneable frequency

Make enemies shoot at the player

In a similar way to the player shooting, make the enemies shoot at the player

At first, shoot a projectile on every `Update()` call

Bonus: Reduce the shooting frequency by using `Time.deltaTime` and `Random.value`

When projectile hits the player, log out to console

Creating the enemy projectile

Identical to player prefab

Check that projectile has Rigidbody2D and Collider

Also includes the Projectile script

Getting hit by the enemy

Player will need a *Kinematic Rigidbody Trigger* and an **`OnTriggerEnter2D()`** method

Still good idea to check for a Projectile component

Tuning the Frequency

We calculate the probability of firing in a given frame and fire if appropriate

Probability of firing depends on how long has elapsed and the intended frequency.

p(fire this frame) = time elapsed x frequency

Use **Random.value** to fire given a probability

Enemies shooting back

Enemies will randomly shoot back with a tuneable frequency

Controlling Collisions Using Layers

Controlling Collisions Using Layers

Player shoots itself when firing!

Lasers hit each other!

We need the player's projectile not to collide with itself or the player

We need the enemy projectiles to not collide with enemies or each other

Create and setup the layers

Create the Enemies, Friendlies, EnemyProjectiles and FriendlyProjectiles layer

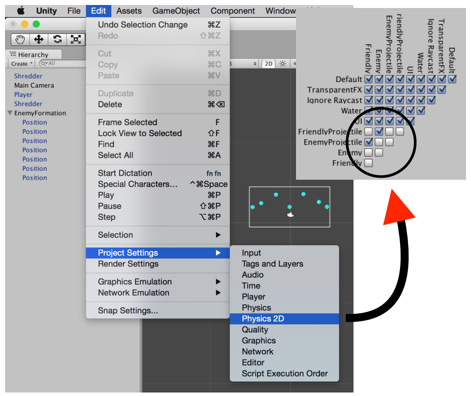
Add the appropriate objects to them

Using layers to stop projectiles colliding

Tag enemy projectiles prefab with 'EnemyProjectile' tag in inspector

Set the FriendlyProjectile, Friendly and Enemy tags appropriately

Go to **Edit > Project Settings > Physics 2D** and uncheck the collision box between things that shouldn't collide



	Friendly	Enemy	EnemyProjectile	riendlyProjectile	UI	Water	Ignore Raycast	TransparentFX	Default
Default	✓	✓	✓	✓	✓	✓	✓	✓	✓
TransparentFX	✓	✓	✓	✓	✓	✓	✓	✓	
Ignore Raycast	✓	✓	✓	✓	✓	✓	✓		
Water	✓	✓	✓	✓	✓	✓			
UI	✓	✓	✓	✓	✓				
FriendlyProjectile	☐	✓	☐	☐					
EnemyProjectile	✓	☐	☐						
Enemy	☐	☐							
Friendly	☐								

Detecting enemies have been destroyed

Detecting enemies have been destroyed

We need to know when all enemies are dead

We use the childCount property of a transform on the positions - an empty position is a dead enemy

We re-spawn the enemies when that happens

Using `Transform.childCount`

This lets us know if the enemy ship is still here

Need to loop over every positions

Keep logic in **`AllMembersAreDead()`** method

Re-spawn enemies

When all enemies are dead, respawn a fresh batch of enemies.

Extract the spawning code from the Start() method

Don’t duplicate but call a new method instead

Detecting enemies have been destroyed

We need to know when all enemies are dead

We use the childCount property of a transform on the positions - an empty position is a dead enemy

We re-spawn the enemies when that happens

Spawning Enemies One By One

Spawning Enemies One by One

Enemies spawning all at once look bad.

Spawn enemies until formation is full one by one instead

Add a spawn delay between each enemies for effect

Create NextFreePosition()

Create a method that will return the next free position

Use the template of **AllMembersDead()** but instead return the Position gameobject and Null

Recursion!

Recursion is when a function calls itself

We're calling **SpawnUntilFull()** from **SpawnUntilFull()** using invoke

We can do that

?

Understanding Transform Relationships

Position animation for a new enemy

Position Animation for a new enemy

- Enemies should animate in, rather than appear
- Create an Animator and Animation Controller
- Create states to represent arriving and flying
- Add the appropriate animation

Explore the animation package

- See if you can make your own unique animation for the incoming enemy
- Why don’t you explore the other options for curves in the animator?

An introduction to Mecanim

For the Enemy prefab, got to the **Inspector > Add Component** then search for **Animator**

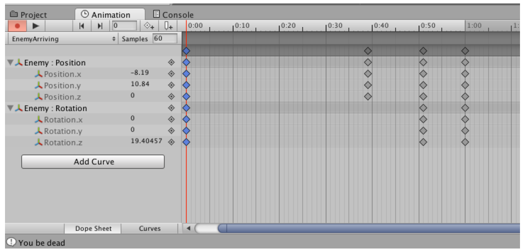
In the Animator, create states by right clicking in the workspace and selecting **New State > Empty**

Create transitions between states by right-clicking a state and selecting **Make Transition**

Creating an animation

Drag and drop an enemy into the scene

Window > Animator to show the animation tool



EnemyArriving   Samples 60

Enemy : Position	
Position.x	-8.19
Position.y	10.84
Position.z	0
Enemy : Rotation	
Rotation.x	0
Rotation.y	0
Rotation.z	19.40457

You be dead

Add the animation to a state

Add your newly created EnemyArriving animation to the Arriving state in the inspector when in the animator tool

Right click and select the EnemyArriving state as default.

Congrats, your enemies are animating

Try to tweak the animation to be a little smoother

See if you can create an idle or formation animation

Creating a starfield

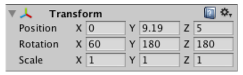
Creating a Starfield

- The background looks a little barren
- Let’s add a starfield with parallax effect to give some sense of depth
- We can use a Particle System to do this.

Adding our first particle effect

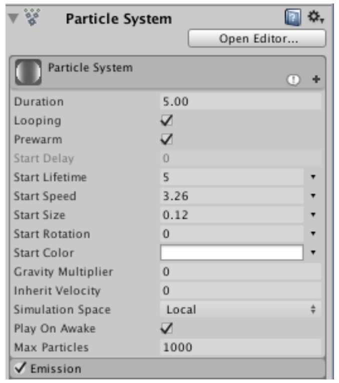
- Create a new Particle System Object in the hierarchy
- Position it so that the particles are moving down and towards the camera
- Tune the lifetime and size of the particles to make the effect look like stars

First Particle System Settings



Transform

	X	Y	Z
Position	0	9.19	5
Rotation	60	180	180
Scale	1	1	1

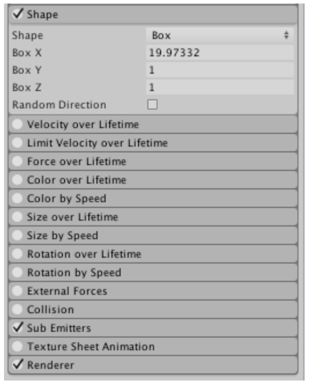


Particle System

Particle System

Duration	5.00
Looping	✓
Prewarm	✓
Start Delay	0
Start Lifetime	5
Start Speed	3.26
Start Size	0.12
Start Rotation	0
Start Color	⬜
Gravity Multiplier	0
Inherit Velocity	0
Simulation Space	Local
Play On Awake	✓
Max Particles	1000

✓ Emission



✓ Shape

Shape	Box
Box X	19.97332
Box Y	1
Box Z	1
Random Direction	☐

- Velocity over Lifetime
- Limit Velocity over Lifetime
- Force over Lifetime
- Color over Lifetime
- Color by Speed
- Size over Lifetime
- Size by Speed
- Rotation over Lifetime
- Rotation by Speed
- External Forces
- Collision
- ✓ Sub Emitters
- Texture Sheet Animation
- ✓ Renderer

Explore Particle Effects

Explore the various options of a particle effect

See if you can create a smoke signal

See if you can create a plasma torch

See if you can create a thruster effect

Combine with animation for awesomeness!

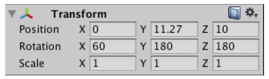
Parallax?

As you move, objects in the distance seem to move less than those nearby

Good technique to give a sense of depth artificially, even in 2D

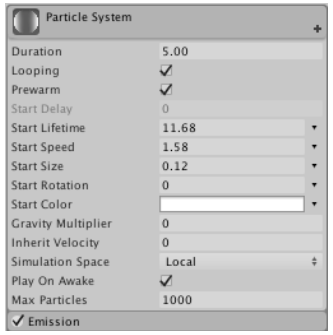
Relative speed of objects is important

Second Particle System Settings



Transform

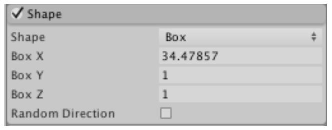
Position	X 0	Y 11.27	Z 10
Rotation	X 60	Y 180	Z 180
Scale	X 1	Y 1	Z 1



Particle System

Duration	5.00
Looping	✓
Prewarm	✓
Start Delay	0
Start Lifetime	11.68
Start Speed	1.58
Start Size	0.12
Start Rotation	0
Start Color	⬜
Gravity Multiplier	0
Inherit Velocity	0
Simulation Space	Local
Play On Awake	✓
Max Particles	1000

✓ Emission



✓ **Shape**

Shape	Box
Box X	34.47857
Box Y	1
Box Z	1
Random Direction	☐

Play Tune Till Awesomeness

Play around with the shape and properties of your particle system until it looks great

Congrats on your first Particle Systems!

Used two to create a sense of depth

Background decoration, but changes the feel of the game a lot

?

Animation and Particle Systems

Keeping Score

Keeping score

Requires Some object to keep track of the scores

We'll create a ScoreKeeper that we attach to the score

When an Enemy dies, we'll call the ScoreKeeper

The ScoreKeeper

Will track scores and update the UI (for now)

Attached to the score Text UI element

We can use the **`GameObject.Find()`** function to get the Score game Object and the **`GetComponent()`** method to get the ScoreKeeper from the EnemyController

Create the UI for the score

Create a Score text visible to the player that will keep track of the score

Make sure that it renders at the right place for the target resolution

Creating a Score UI

Create a UI Canvas

Add a text element

Style it and use a font from http://dafont.com

Make sure it renders at the right place by selecting the right anchor and placement

Create the ScoreKeeper Script

Should be attached to the Score text

Has two methods: **`Score(int points)`** and **`Reset()`**

Will change the score Text whenever the score is updated.

The ScoreKeeper Script

Attached to the Score UI

We need to call it when the enemies die

`GameObject.Find(“Score”).GetComponent<ScoreKeeper>()`

can be used to recover the ScoreKeeper

Add a public field in our enemy script to keep the value in points of an enemy

Send the value from our enemies to our ScoreKeeper

Congrats, we’re now Keeping score

Created an Object to keep the score

Attached it to our UI element

When an Enemy dies, we call it and send our points to the ScoreKeeper

Sound effects for fun and profit

Sound Effects for fun and profit

Will make a huge difference to our game

Easy enough to do

We’ll look at playing sounds independently of an object, so that we can play a death sound for the enemies

Add sound to your game

Add sound to the enemies and the player so that a sound plays when:

- The player fires
- The enemies fire
- An enemy dies

Adding sound to the game

Modify scripts to plays sounds using

`AudioSource.PlayClipAtPoint()`

Add sound assets to unity

Connect clips to scripts

We now have sound!

Sprites rendering order

Sprite Rendering Order

Changes which sprites are drawn on top

Lets missiles from the player be drawn below the ship when instantiated

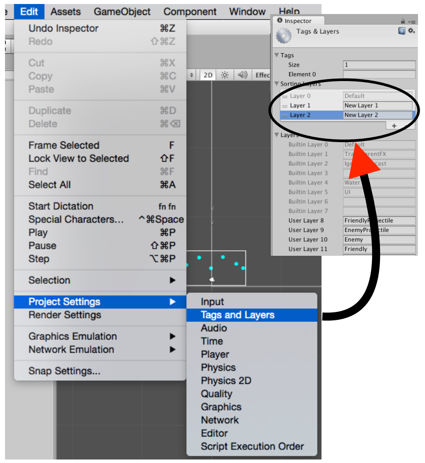
Not affected by z position

Sprites rendering

Could be controlled with distance from camera

Powerful and flexible to control with layers

We use layers to make our game look better by drawing projectiles below their guns



Create Appropriate Sorting Layers

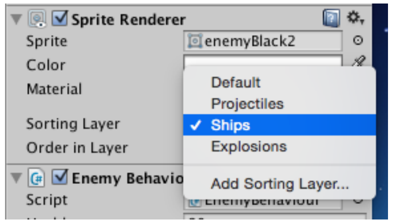
So that:

- Player Projectiles render below the Player
- Enemy Projectiles render below the Enemies
- Player Projectiles render above the Enemies

Make sure you change the Sprite Renderers to have the right sorting layers.

Change the Render layer for a Sprite

The Sorting Layer for a sprite can be changed in the inspector


Sprite Renderer
Sprite: enemyBlack2
Color
Material
Sorting Layer: Default / Projectiles / ✓ Ships / Explosions / Add Sorting Layer...
Order in Layer
Enemy Behaviour
Script: EnemyBehaviour

Laser Defender, now with layers

We changed the way sprites were rendered by using sorting layers

Lets us draw sprites on top of each other independently of distance from camera

Completely **separate an independent** from normal Layers, which are used for physics

?

Score, Sounds and Sprite Order

Polishing the menu system I

Polishing the menu system I

- Replacing the menu style
- Passing the score to the ends

Change the style of the menu system

- Use a custom Font & colours
- Make sure to change the hover colours to match

Pass the score to the final scene

Change the ScoreKeeper to static methods.

Display the score in the end Scene.

Make sure we call **ScoreKeeper.Reset()**

Polishing the menu system I

Replacing the menu style

Passing the score to the end scene

Polishing the menu system II

Polishing the menu system II

Adding our own music to the game

Adding a background starfield

Adding atmospheric music

Found on http://opengameart.org

We use Clearside's (http://clearsidemusic.com) music: http://opengameart.org/users/clearside licensed under Creative Commons

Changing the music on scene load

Want different music on menu and scene

Add music to music player, then use **`OnLevelWasLoaded()`** to check when it's appropriate to play which music track

Add a starfield effect to the menu

Use Particle Systems to give the impression the player is flying through space in the main menu

Quick and easy option is to copy the Particle System from the Game scene

Menu System Ready to go Chief!

We now have a musical menu :)

We're showing off some particle effects as soon as the game is loaded

Nice priming for the game itself

?

End of Section Quiz

Improving the player animation

Improving the player animation

- Use triggers for the animator
- Give player visual feedback on their key presses
- Create a better Idle animation

improving the enemy animations

Improving the enemy animations

- Enemies in formation should not be static
- Add firing animation to warn the player
- Enable thrusters when arriving
- Disable firing until in formation
- Add an animation to the formation to make it more exciting

Improving the projectile animations

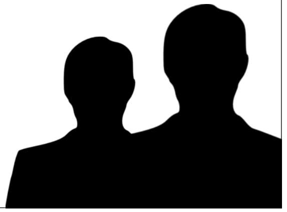
Improving the projectile animations

- Explosions agains ships look nicer
- Create an explosion prefab and instantiate before projectile death
- Add to right layer so it renders above ships

Improve the other projectile

- But do something else.
- Use a different explosion using a Particle System
- Add sound to match the effect

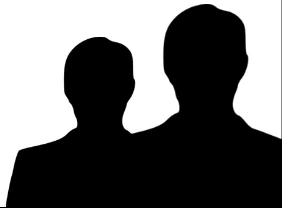
Make it yours and share



Make it yours and Share

- Make it yours!
- Add different enemy types
- We use a fraction of the sprites bundled - Use more
- Add asteroids that hurt enemies and player alike
- Let the player take a few hits and change the sprite to show the damage, ditto for enemies
- Add loot drop from dead enemies - Health pickups?

Recap & What’s Next



Recap & What’s Next

- New in your toolkit
  - Trigger Colliders
  - Sprite Animations
  - Particle Systems
  - Physics Layers
  - Sorting Layers