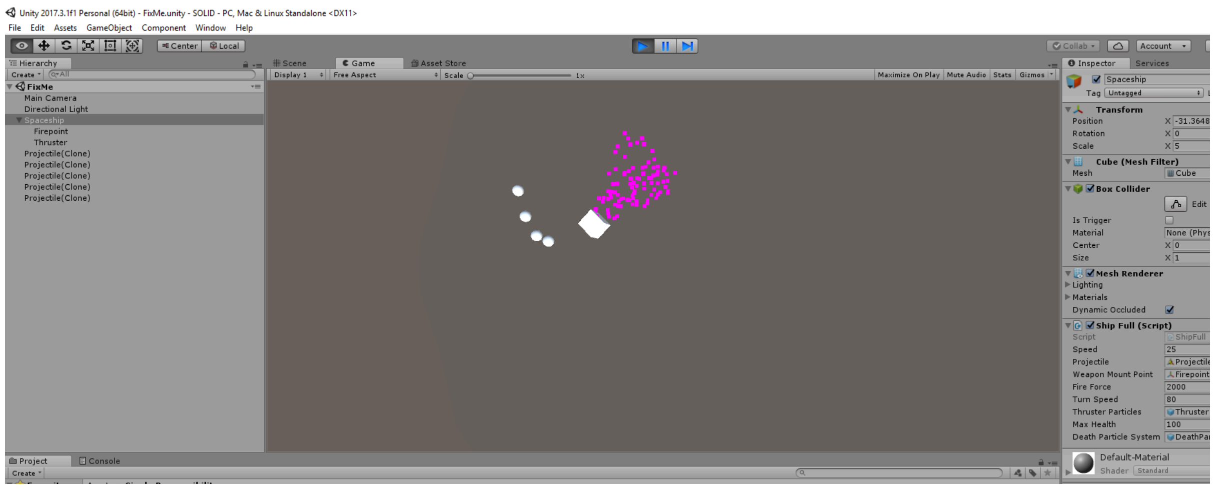
Exercises - Game Architecture

Applying The Single Responsibility Principle

Import the SOLID package from the course website into Unity. In the SRP folder, open the FixMe scene that contains a spaceship object together with some scripts and prefabs. The spaceship can be moved using keyboard input and projectiles can be fired with the mouse. All of the game logic is contained within the ShipFull script on the spaceship.

Apply the single responsibility principle to the ShipFull script so that the code becomes easier to understand and extend. After applying the principle, the spaceship should contain multiple script components instead of just one.

Hint: You'll need separate components for ShipEngine, ShipInput, ShipHealth, ShipParticles and ProjectileLauncher to cover the same functionality as the ShipFull script.



Spaceship
Tag Untagged
Transform
Position X -31.3648
Rotation X 0
Scale X 5
Cube (Mesh Filter)
Mesh Cube
Box Collider
Is Trigger
Material None (Phys
Center X 0
Size X 1
Mesh Renderer
Dynamic Occluded ✓
Ship Full (Script)
Script ShipFull
Speed 25
Projectile Projectile
Weapon Mount Point Firepoint
Fire Force 2000
Turn Speed 80
Thruster Particles Thruster
Max Health 100
Death Particle System DeathPar
Default-Material
Shader Standard

Extending The Implementation

a) Extend the fixed project so that the spaceship can fire two projectiles side-by-side instead of just one.

Consider: How did you achieve this? Did you have to change any code in order to extend the functionality of your game?

Hint: You may need more than one Projectile Launcher and FirePoint.

b) Extend the implementation so that it includes another ship. This spaceship should not react to any input. Shoot the new spaceship with your own (while making an evil laugh). Make sure that the spaceship makes a particle explosion when it dies!

Consider: Again, do you need to modify any code, or can this functionality be achieved in a simpler/cleaner manner?

Hint: You will need to check for null in the scripts that try to access components that you might want to remove!

c) Extend the implementation so that this idle ship fires a projectile every second without any player input. Move your own spaceship into the crossfire and watch it explode!

Hint: You'll need to create a new script component, AutoFire, with a Shoot coroutine that calls an OnFire event every second. Subscribe to this event in the ProjectileLauncher class and remember appropriate null checks.

d) Make your spaceship invulnerable! Move it into the crossfire and laugh as the other spaceship cannot hurt you anymore. As for a) and b), this should be achieved without writing any code (other than making sure you have appropriate null-checks).
e) Extend on the implementation by adding different spaceships with different components and properties.

Apply SOLID and Design Patterns to your Game Project
In the slides you will find a video demonstration for each SOLID design principle, as well as a link describing common game design patterns. Go through the material and make sure that you understand how the principles and patterns can be applied in Unity, then consider how the principles can be applied to your game project, and what design patterns best fit your game architecture.