**Asteroids ECS**

by Leandro Wainberg

Unity version: 2020.3.30f1

ECS version: 0.51.1-preview.21

<https://github.com/leandronw/AsteroidsECS>

This game was developed with a hybrid approach, using both Entities/Components/Systems and Monobehaviors.

Main entities in the game are:

* Player: Is represented by a spaceship. Can shoot bullets when a WeaponData component is present. Can pick power-ups (weapons or shield). Gets destroyed by asteroids, UFOs or UFO’s bullets (only when no Shield is active). Is currently controlled with the keyboard (W to thrust, A and D to turn, H to jump to hyperspace, Space to shoot). There’s only one Player at a time, although most systems are prepared to handle more players simultaneously.
* UFO: Can shoot bullets that will kill the Player. Gets destroyed by asteroids and Player’s bullets. Has a very basic AI that tries to avoid asteroids.
* Asteroid: Can have any of 3 sizes: Big, medium and small. Floats around the screen with random velocity (depending on size). Gets destroyed by any bullet or collision with Player/UFO. When it’s destroyed, two smaller ones are spawned.
* Power-Up: Static object that only collides with Player. When picked, grants either a Shield (temporary) or a Weapon (permanent).

Monobehaviors

# GameManager

Singleton that is responsible for gameplay rules and spawning of main entities:

* Spawns Player when game starts and assigns input controller to it.
* Keeps count of remaining lives on Player’s death, and respawns Player until no more lives are left (Game Over).
* Checks if level was cleared (no more enemies left) and starts a new level, with asteroids and power-ups.
* Spawns UFOs after a certain time interval, which decreases over time.
* Destroys everything from previous game when a new game starts.
* Invokes events for the UI:
  + OnGameStarted()
  + OnGameEnded()
  + OnLivesChanged(int newValue)
  + OnCountdownStarted(float duration)

# GameUI

Listens to events from ECS and GameManager, and shows visual feedback:

* Player lives
* Game Over
* Shield timer
* Countdown

# SfxPlayer

Listens to sound events from ECS and direct calls from Monobehaviors and plays audio clips/loops.

# GameArea

Singleton that keeps track of game world’s edges. Updates itself when screen size changes.

Systems

# ***PlayerKeyboardInputSystem*** (InitializationSystemGroup)

Reads from <PlayerKeyboardComponent> which keys control the player, and writes to <PlayerInputData> the actions the player is doing.

# ***PlayerInputHandlingSystem*** (InitializationSystemGroup)

Reads <PlayerInputData> and acts according to what actions the player is doing:

* Turns player left/right (writes <Rotation>).
* Enables or disables thrust (writes <PhysicsVelocity> and makes child “Thrust” particle system enable/disable)
* Shoots weapon. This spawns a new “Bullet” entity, reading all the necessary info from <WeaponData>. Writes <WeaponData> by setting ElapsedTimeSinceLastShot to 0.
* Adds <JumpToHyperspaceTag> if player just pressed the hyperspace button.

# ***HyperspaceSystem*** (InitializationSystemGroup)

For all players that have <JumpToHyperspaceTag> change their position to a random point inside the Game Area and removes <JumpToHypespaceTag>.

# ***CollisionDetectionSystem*** (FixedStepSimulationSystemGroup)

Listens to trigger events from the Physics system, and adds <CollisionData> to collided entities.

# ***CollisionHandlingSystem*** (SimulationSystemGroup)

Reads <CollisionData> from all Entities that collided and acts accordingly:

* Bullets get destroyed.
* UFOs get a <DestroyedTag>.
* Asteroids get a <DestroyedTag>.
* If Player collided with Power-Up, adds a <PickedTag> to Power-Up. If collided with anything else, check if has Shield active, and if not, adds a <DestroyedTag> to Player.

# ***PlayerDestroySystem*** (LateSimulationSystemGroup)

Destroys Player entities with <DestroyedTag>. Spawns a VFX prefab at player’s position.

# ***UFODestroySystem*** (LateSimulationSystemGroup)

Destroys UFO entities with <DestroyedTag>. Spawns a VFX prefab at UFO position.

# ***AsteroidsDestroySystem*** (LateSimulationSystemGroup)

Destroys Asteroid entities with <DestroyedTag>. Spawns a VFX prefab at asteroid’s position. If <AsteroidSizeComponent> is Big or Medium, creates a new <AsteroidsSpawnRequest> for 2 smaller asteroids.

# ***AsteroidsSpawnSystem*** *(*InitializationSystemGroup)

For all <AsteroidsSpawnRequest> instantiates new asteroids and adds a <NeedsInitTag> to them. New asteroid’s <PhisicsVelocity> is set to the value stored in <AsteroidsSpawnRequest>, which is the velocity of the bigger asteroid that was just destroyed.

# ***AsteroidsInitializeSystem*** (SimulationSystemGroup)

Writes the <PhysicsVelocity> of all asteroids with <NeedsInitTag>, using their <AsteroidRotationData>, <AsteroidSpeedData> and previous velocity. <NeedsInitTag> gets removed.

# ***LifetimeSystem*** (SimulationSystemGroup)

Decreases value of <LifetimeComponent> for all entities, destroys them when it reaches 0. Used for bullets and VFX explosions.

# ***WeaponSetDefaultSystem*** (InitializationSystemGroup)

Keeps a reference to the default <WeaponData> and <WeaponEquipRequest>, which are in the scene hierarchy, and assigns them to any Player with no <WeaponData>.

# ***WeaponPickedSystem*** (SimulationSystemGroup)

All picked Weapon Power-Ups (those that have <PickedTag>, <WeaponPowerUpTag>, <WeaponData>, <WeaponEquipRequest> and <CollisionComponent> get destroyed and their <WeaponData> and <WeaponEquipRequest> are transferred to the Player who picked them, which is referenced in the <CollisionComponent>.

# ***WeaponEquipSystem*** (SimulationSystemGroup)

For Players with <WeaponEquipRequest>, it first destroys any existing Weapon children the Player may have, then instantiates a new Weapon (prefab referenced in <WeaponEquipRequest>), and attaches it to the Player as child. <WeaponEquipRequest> gets removed.

# ***ShieldPickedSystem*** (SimulationSystemGroup)

All picked Shield Power-Ups (those that have <PickedTag>, <ShieldPowerUpTag>, <ShieldData> and <CollisionComponent>) get destroyed and their <ShieldData> is transferred to the Player who picked the, which is referenced in the <CollisionComponent>. Also a new <ShieldEnableRequest> is added to the Player.

# ***ShieldEnableSystem*** (InitializationSystemGroup)

Players with <ShieldEnableRequest> and <ShieldData> get their previously equipped Shield destroyed and a new one is instantiated from <ShieldData>. <ShieldEnableRequest> gets removed.

# ***ShieldDepleteSystem*** (SimulationSystemGroup)

Players with <ShieldData> have their shield TimeRemaining decreased. When it reaches 0 their <ShieldData> is removed and the child Shield is destroyed.

# ***UFOAttackSystem*** (InitializationSystemGroup)

UFOs with <UFORotatingWeaponData> ElapsedTimeSinceLastShot greater than 1/ BulletsPerSecond shoot their weapon spawning EnemyBullets, and set their ElapsedTimeSinceLastShot to 0.

# ***UFOMovementSystem***

When UFOs with <UFOBrainComponent> ElapsedTimeSinceLastChange is greater than MinTimeSinceLastChage, they use the position of nearby asteroids to calculate a new direction, and then calculate a new <PhisicsVelocity> from <UFOSpeedData>. The algorithm to calculate the new direction is explained in the “UFO Movement” file.

### *WrapAroundEdgesSystem* (LateSimulationSystemGroup)

Teleports all entities with <WrapAroundEdgesComponent> that are leaving the bounds of the game world to the opposite side.

# ***EventDispatcherSystem*** (LateSimulationSystemGroup)

Consumes event entities and invokes C# delegates that can be listened by Monobehaviors or other systems. Events are:

* OnPlayerDestroyed(float2 position)
* OnUFODestroyed(float2 position)
* OnAsteroidDestroyed(float2 position, AsteroidSize size)
* OnShieldEnabled(float duration)
* OnShieldDepleted()
* OnWeaponEquipped()
* OnHyperspace(float2 fromPosition, float2 toPosition)
* OnSoundPlayed(SoundId sound)
* OnSoundLoopStarted(SoundId sound)
* OnSoundLoopStopped(SoundId sound)