Engine Overview:

My ECS Engine implementation utilizes Sparse Sets for components optimizing for fast lookup, insertion, and deletion. A similar approach is implemented in the popular ECS framework EnTT. Components are stored in an unordered map using the component type as the key. Systems, responsible for all game logic and are similarly stored in an unordered map.

Events are also handled through components and systems. Event data is stored in components and then processed by a corresponding system.

Scenes Overview:

The game comprises three main scenes: MenuScene, PlayScene, and DeathScene.

MenuScene acts as the starting point, leading to PlayScene or an exit option.PlayScene is of course where the game is played. Finally, we have the DeathScene which is reached upon the Players death where the time survived is displayed with an opportunity to try again.

Systems Overview:

Arm System:

Controls the aiming arm/line for player and enemy tanks, managing arm direction and projectile creation.

Button System:

Manages all button-related functionalities, detecting clicks and triggering corresponding events.

Damage System:

Processes damage events and handles health regeneration, deleting entities when health drops below 0.

Enemy System:

Governs enemy behavior and wave spawning.

Physics System:

Handles collision detection and response, as well as the kinetics of rigid bodies.

Player System:

Manages player-specific behavior.

Render System:

Responsible for rendering all elements on the screen.

Timer System:
Iterates on all timer components, managing time-related tasks.

Components Overview:

Arm Component:
Contains information for arms and projectile creation.

BoxCollider Component:
Holds box bounds.

Button Component:
Holds button information and type.

CircleCollider Component:
Holds circle radius.

CollisionEvent Component:
Created upon collision, holding information for collision response.

Damage Component:
Holds damage value.

DamageEvent Component:
Created upon collision between Entity holding health component and one holding damage component. Holds information for damage response.

Homing Enemy Component:
Holds information needed for homing enemy type.

Tank Enemy Component:
Holds information needed for tank enemy type.

Health Component:
Manages entity health and flags entities for deletion below 0 health.

Label Component:
Holds string to be printed.

Player Component:
Holds information needed for player behavior.

Render Component:
Holds information to render the entity.

Rigidbody Component:
Holds physics related information.

Timer Component:
Stores time.

Transform Component:
Holds entity positions.

Wave Component:
Contains information for spawning enemy waves.