Namespace

Classes

AudioManager

Manages audio settings for the game, including music and sound effects (SFX) volume.

Bullet

Player projectile class. Uses Monobehavior to interact with enemies and move across the screen. Interacts with enemy HealthManagers to deal damage. Implements `Start`, `FixedUpdate`, and `OnTriggerEnder2D` Unity functions.

CameraController

Moves the camera around to follow the player. Implements `FixedUpdate()` to update its position smoothly.

CursorObj

Keeps track of the cursor's position in world space, and aligns it to the game grid.

Deparent

Utility class which moves the children of a GameObject out of the GameObject when it is initialized.

EndzoneTrigger

Implements Unity's `OnTriggerEnter2D` to detect when a player reaches the endzone, then makes Game Manager load the next level.

EnemyManager

Manages the behavior and interactions of an enemy character in the game. This includes pathfinding and attacking the player. Implements Unity's `FixedUpdate` to perform actions per frame, and also uses the IWeapon strategy for flexibility.

EnemySpawnpoint

Container to hold an EnemyManager GameObject

ExplodeWeapon

An implementation of IWeapon which choots out a burst of bullets when used.

GameManager

Singleton which oversees the game.

HealthManager

Adds health to a GameObject, allowing it to be damaged, and destroying it when health is set to zero.

HelpTile

IWeapon
: Defines the Weapon Strategy, which is used by EnemyManager. Weapons keep track of their own cooldowns, and have some interfaces so the enemy can act depending on the weapon state (eg. run away if weapon is not ready).

Item
: container class to hold an ID for serialization.

LeaderboardDisplay
: Queries the backend to get and display the leaderboard data. Uses Unity's `Start` to do this on scene load.

LevelEditor
: Manages the editor scene, including selecting the tile to paint, serialization, and making sure levels are of a valid format (eg. tiles must not overlap). Implements Unity's `Start`, `Update`, and `FixedUpdate` to interact with the user per-frame

LevelModel

LevelRequester
: Makes calls to the backend to load levels.

LevelResponse

LevelSerializer
: Serializes levels: Turns a grid of Tile s into a list of positions and their associated tile IDs.

MeleeWeapon
: An implementation of IWeapon which directly damages the player when used, but has a short range.

Menu
: Controls Scene loading and opening options within the main menu.

ObjectIndex

PaletteItem
: Manages the tile painting feature when in the LevelEditor. Uses `Update` to "paint" the level scene per frame the mouse is down, and communicates with LevelEditor to update the level state accordingly.

PlayerController
: Implements movement and shooting for the player, as well as player interactions with water tiles. Uses Unity's `Update` and `FixedUpdate` to respond to input per frame.

Projectile

Projectile launched by enemies, implements Unity's `FixedUpdate` to move across the screen per frame, and implements `OnTriggerEnter2D` to define collision logic.

RangedWeapon
An implementation of IWeapon which shoots a Projectile at the player.

SceneAudio

ScoreModel

ScoreResponse

SerializationTest
Class implementing unit tests for level serialization/deserialization.

Startzone

Tile
A struct so objects interacting with Tile can easily get its `GameObject` and `Transform`, eg. when serializing. GameObjects can't hold onto values otherwise, so sometimes Data classes are necessary in Unity.

Transition
Animates menus.

WaterTile
Slows the player down when a Tile with this Monobehavior collides with the player.

Class AudioManager

Namespace: Global

Manages audio settings for the game, including music and sound effects (SFX) volume.

```
public class AudioManager : MonoBehaviour
```

Inheritance

object ← AudioManager

Fields

manager

```
public static AudioManager manager
```

Field Value

AudioManager

mixer

```
public AudioMixer mixer
```

Field Value

AudioMixer

Methods

LoadSettings()

Loads the saved music and SFX volume settings from PlayerPrefs and updates the AudioMixer.

```
public void LoadSettings()
```

SetGMusicVol(float)

Sets the global music volume and updates the corresponding AudioMixer parameter.

```
public void SetGMusicVol(float vol)
```

Parameters

`vol` float

The desired music volume level (range 0 to 1).

SetGSFXVol(float)

Sets the global sound effects (SFX) volume and updates the corresponding AudioMixer parameter.

```
public void SetGSFXVol(float vol)
```

Parameters

`vol` float

The desired SFX volume level (range 0 to 1).

UpdateSliders(Slider, Slider)

Updates the UI sliders to reflect the current music and SFX volume levels.

```
public void UpdateSliders(Slider musicSlider, Slider sfxSlider)
```

Parameters

`musicSlider` Slider

The UI slider for adjusting music volume.

`sfxSlider` Slider

The UI slider for adjusting SFX volume.

Class Bullet

Namespace: Global

Player projectile class. Uses Monobehavior to interact with enemies and move across the screen. Interacts with enemy HealthManagers to deal damage. Implements `Start`, `FixedUpdate`, and `OnTriggerEnder2D` Unity functions.

```
public class Bullet : MonoBehaviour
```

Inheritance

object ← Bullet

Class CameraController

Namespace: Global

Moves the camera around to follow the player. Implements `FixedUpdate()` to update its position smoothly.

```
public class CameraController : MonoBehaviour
```

Inheritance

object ← CameraController

Class CursorObj

Namespace: Global

Keeps track of the cursor's position in world space, and aligns it to the game grid.

```
public class CursorObj : MonoBehaviour
```

Inheritance

object ← CursorObj

Fields

pos

```
public static Vector2 pos
```

Field Value

Vector2

self

```
public static CursorObj self
```

Field Value

CursorObj

spriteRenderer

```
public SpriteRenderer spriteRenderer
```

Field Value

SpriteRenderer

trfm

```
public static Transform trfm
```

Field Value

Transform

Class Deparent

Namespace: Global

Utility class which moves the children of a GameObject out of the GameObject when it is initialized.

```
public class Deparent : MonoBehaviour
```

Inheritance

object ← Deparent

Class EndzoneTrigger

Namespace: Global

Implements Unity's `OnTriggerEnter2D` to detect when a player reaches the endzone, then makes Game Manager load the next level.

```
public class EndzoneTrigger : MonoBehaviour
```

Inheritance

object ← EndzoneTrigger

Class EnemyManager

Namespace: Global

Manages the behavior and interactions of an enemy character in the game. This includes pathfinding and attacking the player. Implements Unity's `FixedUpdate` to perform actions per frame, and also uses the IWeapon strategy for flexibility.

```
public class EnemyManager : MonoBehaviour
```

Inheritance

object ← EnemyManager

Methods

ResetPath()

```
public void ResetPath()
```

Class EnemySpawnpoint

Namespace: Global

Container to hold an EnemyManager GameObject

```
public class EnemySpawnpoint : MonoBehaviour
```

Inheritance

object ← EnemySpawnpoint

Methods

GetEnemy()

```
public GameObject GetEnemy()
```

Returns

GameObject

Class ExplodeWeapon

Namespace: Global

An implementation of IWeapon which choots out a burst of bullets when used.

```
public class ExplodeWeapon : IWeapon
```

Inheritance

object ← IWeapon ← ExplodeWeapon

Inherited Members

IWeapon.attackCD , IWeapon.attackRange , IWeapon.attackDamage

Fields

projectilePrefab

```
public GameObject projectilePrefab
```

Field Value

GameObject

Methods

Ready()

Get whether the weapon is Ready or not, typically when the cooldown is zero.

```
public override bool Ready()
```

Returns

bool

Whether the weapon is ready or not

Usable(GameObject)

Get whether the weapon is Usable, typically when the cooldown is zero AND the player is close enough.

```
public override bool Usable(GameObject target)
```

Parameters

`target` GameObject

Returns

bool

Whether the weapon is usable

Use(GameObject)

What to do when the weapon is used, eg. spawn projectiles, damage the player directly, etc.

```
public override IEnumerator Use(GameObject target)
```

Parameters

`target` GameObject

Returns

IEnumerator

IEnumerator: This happens asynchronously as a Unity coroutine, so it must return an IEnumerator

Class GameManager

Namespace: Global

Singleton which oversees the game.

```
public class GameManager : MonoBehaviour
```

Inheritance

object ← GameManager

Fields

GameStart

```
public static UnityEvent GameStart
```

Field Value

UnityEvent

Instance

```
public static GameManager Instance
```

Field Value

GameManager

Properties

LevelSerializer

```
public LevelSerializer LevelSerializer { get; }
```

Property Value

LevelSerializer

Methods

EndGame()

```
public void EndGame()
```

LoadLevel(string)

Loads a level based on the

```
public void LoadLevel(string levelString)
```

Parameters

`levelString` string

The level to load, serialized as a string

SetupLevel(bool)

Queries the backend using LevelRequester , then loads the level based on the serialized level strings returned.

```
public void SetupLevel(bool skipped = false)
```

Parameters

`skipped` bool

If set to true, increments the player's score

UnloadLevel()

Deletes all GameObjects associated with the level.

> Remark: Switching levels doesn't actually involve changing scenes, only resetting certain states and loading/destroying GameObjects

```
public void UnloadLevel()
```

Class HealthManager

Namespace: Global

Adds health to a GameObject, allowing it to be damaged, and destroying it when health is set to zero.

```
public class HealthManager : MonoBehaviour
```

Inheritance

object ← HealthManager

Fields

MaxHealth

Health cap and the amount of health the GameObject starts with.

```
public int MaxHealth
```

Field Value

int

Properties

Dying

Dying is set to true if the GameObject is in the process of being destroyed, eg. when a death animation is playing.

```
public bool Dying { get; }
```

Property Value

bool

Health

```
public int Health { get; set; }
```

Property Value

int

Class HelpTile

Namespace: Global

```
public class HelpTile : MonoBehaviour
```

Inheritance

object ← HelpTile

Class IWeapon

Namespace: Global

Defines the Weapon Strategy, which is used by EnemyManager. Weapons keep track of their own cooldowns, and have some interfaces so the enemy can act depending on the weapon state (eg. run away if weapon is not ready).

```
public abstract class IWeapon : MonoBehaviour
```

Inheritance

object ← IWeapon

Derived

ExplodeWeapon, MeleeWeapon, RangedWeapon

Fields

attackCD

All weapons have an attack cooldown

```
protected float attackCD
```

Field Value

float

attackDamage

The amount of damage to do when the weapon hits.

```
protected int attackDamage
```

Field Value

int

attackRange

Get the attack range of the weapon. Enemies need to know this to get in range of the player.

```
public float attackRange
```

Field Value

float

Methods

Ready()

Get whether the weapon is Ready or not, typically when the cooldown is zero.

```
public abstract bool Ready()
```

Returns

bool

Whether the weapon is ready or not

Usable(GameObject)

Get whether the weapon is Usable, typically when the cooldown is zero AND the player is close enough.

```
public abstract bool Usable(GameObject target)
```

Parameters

`target` GameObject

Returns

bool

Whether the weapon is usable

Use(GameObject)

What to do when the weapon is used, eg. spawn projectiles, damage the player directly, etc.

```
public abstract IEnumerator Use(GameObject target)
```

Parameters

`target` GameObject

Returns

IEnumerator

IEnumerator: This happens asynchronously as a Unity coroutine, so it must return an IEnumerator

Class Item

Namespace: Global

container class to hold an ID for serialization.

```
public class Item : MonoBehaviour
```

Inheritance

object ← Item

Fields

id

```
public int id
```

Field Value

int

Class LeaderboardDisplay

Namespace: Global

Queries the backend to get and display the leaderboard data. Uses Unity's `Start` to do this on scene load.

```
public class LeaderboardDisplay : MonoBehaviour
```

Inheritance

object ← LeaderboardDisplay

Class LevelEditor

Namespace: Global

Manages the editor scene, including selecting the tile to paint, serialization, and making sure levels are of a valid format (eg. tiles must not overlap). Implements Unity's `Start`, `Update`, and `FixedUpdate` to interact with the user per-frame

```
public class LevelEditor : MonoBehaviour
```

Inheritance

object ← LevelEditor

Fields

TILE_BRICK

```
public const int TILE_BRICK = 2
```

Field Value

int

TILE_END

```
public const int TILE_END = 1
```

Field Value

int

TILE_ENEMY

```
public const int TILE_ENEMY = 5
```

Field Value

int

TILE_LAVA

```
public const int TILE_LAVA = 4
```

Field Value

int

TILE_START

```
public const int TILE_START = 0
```

Field Value

int

TILE_WATER

```
public const int TILE_WATER = 3
```

Field Value

int

brushType

```
public int brushType
```

Field Value

int

mapSerialization

```
public string mapSerialization
```

Field Value

string

origin

```
public Vector2 origin
```

Field Value

Vector2

paletteHover

```
public static bool paletteHover
```

Field Value

bool

self

```
public static LevelEditor self
```

Field Value

LevelEditor

tileIDsMap

Maps positions in the scene to Tile IDs for serialization

```
public Dictionary<Vector2, int> tileIDsMap
```

Field Value

Dictionary<Vector2, int>

tileObjMap

Maps positions in the scene to GameObjects to control game state

```
public Dictionary<Vector2, Tile> tileObjMap
```

Field Value

Dictionary<Vector2, Tile>

tileSize

```
public float tileSize
```

Field Value

float

Methods

RequestBrush(int)

```
public void RequestBrush(int id)
```

Parameters

`id` int

Class LevelModel

Namespace: Global

```
[Serializable]
public class LevelModel
```

Inheritance

object ← LevelModel

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

author

```
public string author
```

Field Value

string

level_name

```
public string level_name
```

Field Value

string

serialized_level

```
public string serialized_level
```

Field Value

string

Class LevelRequester

Namespace: Global

Makes calls to the backend to load levels.

```
public class LevelRequester : MonoBehaviour
```

Inheritance

object ← LevelRequester

Methods

GetLevel()

Returns a random level from the levels which have been received by `RequestLevels`.

```
public LevelModel GetLevel()
```

Returns

LevelModel

LevelModel: a struct containing a Level's name, creator, and serialized string

QueryAllLevels()

Querys a certain number levels from the backend at once.

```
public IEnumerator QueryAllLevels()
```

Returns

IEnumerator

IEnumerator: this is required for Unity's coroutines feature, which lets us make backend calls asynchronously

Class LevelResponse

Namespace: Global

```
[Serializable]
public class LevelResponse
```

Inheritance

object ← LevelResponse

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

data

```
public List<LevelModel> data
```

Field Value

List<LevelModel>

message

```
public string message
```

Field Value

string

statusCode

```
public int statusCode
```

Field Value

int

Class LevelSerializer

Namespace: Global

Serializes levels: Turns a grid of Tile s into a list of positions and their associated tile IDs.

```
public class LevelSerializer : MonoBehaviour
```

Inheritance

object ← LevelSerializer

Methods

LoadField(string, GameObject, Transform)

Loads a level by turning the serialized string into `GameObject`s under the `parent` parameter, then moves the player to starting location provided by `playerTransform`

```
public void LoadField(string data, GameObject parent, Transform playerTransform)
```

Parameters

`data` string

Serialized level data

`parent` GameObject

Parent under which to create the level's `GameObject`s

`playerTransform` Transform

Where to move the player

SaveField(GameObject)

Serializes and uploads a `GameObject` `parent` to the backend.

```
public void SaveField(GameObject parent)
```

Parameters

`parent` GameObject

The `GameObject` to be serialized and saved

SerializeLevel(GameObject)

Serializes a GameObject `parent` representing the level into a string which can be stored and later deserialized.

```
public string SerializeLevel(GameObject parent)
```

Parameters

`parent` GameObject

Unity `GameObject` representing the parent of all of the tiles in a level (like the html root element).

Returns

string

String representing the serialized level

Class MeleeWeapon

Namespace: Global

An implementation of IWeapon which directly damages the player when used, but has a short range.

```
public class MeleeWeapon : IWeapon
```

Inheritance

object ← IWeapon ← MeleeWeapon

Inherited Members

IWeapon.attackCD , IWeapon.attackRange , IWeapon.attackDamage

Methods

Ready()

Get whether the weapon is Ready or not, typically when the cooldown is zero.

```
public override bool Ready()
```

Returns

bool

Whether the weapon is ready or not

Usable(GameObject)

Get whether the weapon is Usable, typically when the cooldown is zero AND the player is close enough.

```
public override bool Usable(GameObject target)
```

Parameters

`target` GameObject

Returns

bool

Whether the weapon is usable

Use(GameObject)

What to do when the weapon is used, eg. spawn projectiles, damage the player directly, etc.

```
public override IEnumerator Use(GameObject target)
```

Parameters

`target` GameObject

Returns

IEnumerator

IEnumerator: This happens asynchronously as a Unity coroutine, so it must return an IEnumerator

Class Menu

Namespace: Global

Controls Scene loading and opening options within the main menu.

```
public class Menu : MonoBehaviour
```

Inheritance

object ← Menu

Fields

errorText

```
public GameObject errorText
```

Field Value

GameObject

panel

```
public GameObject panel
```

Field Value

GameObject

usernameInput

```
public TMP_InputField usernameInput
```

Field Value

TMP_InputField

Methods

CloseOptions()

```
public void CloseOptions()
```

LoadScene(string)

```
public void LoadScene(string game)
```

Parameters

`game` string

OpenOptions()

```
public void OpenOptions()
```

PauseGame()

```
public void PauseGame()
```

PlayGame(string)

```
public void PlayGame(string game)
```

Parameters

`game` string

Quit()

```
public void Quit()
```

UnpauseGame()

```
public void UnpauseGame()
```

Class ObjectIndex

Namespace: Global

```
public class ObjectIndex : ScriptableObject
```

Inheritance

object ← ObjectIndex

Fields

objects

```
public List<GameObject> objects
```

Field Value

List<GameObject>

Class PaletteItem

Namespace: Global

Manages the tile painting feature when in the LevelEditor. Uses `Update` to "paint" the level scene per frame the mouse is down, and communicates with LevelEditor to update the level state accordingly.

```
public class PaletteItem : MonoBehaviour
```

Inheritance

object ← PaletteItem

Class PlayerController

Namespace: Global

Implements movement and shooting for the player, as well as player interactions with water tiles. Uses Unity's `Update` and `FixedUpdate` to respond to input per frame.

```
public class PlayerController : MonoBehaviour
```

Inheritance

object ← PlayerController

Fields

moveSpeed

```
public float moveSpeed
```

Field Value

float

self

```
public static PlayerController self
```

Field Value

PlayerController

trfm

```
public static Transform trfm
```

Field Value

Transform

Methods

EnterWater()

```
public void EnterWater()
```

ExitWater()

```
public void ExitWater()
```

Class Projectile

Namespace: Global

Projectile launched by enemies, implements Unity's `FixedUpdate` to move across the screen per frame, and implements `OnTriggerEnter2D` to define collision logic.

```
public class Projectile : MonoBehaviour
```

Inheritance

object ← Projectile

Properties

Direction

```
public Vector3 Direction { get; set; }
```

Property Value

Vector3

Class RangedWeapon

Namespace: Global

An implementation of IWeapon which shoots a Projectile at the player.

```
public class RangedWeapon : IWeapon
```

Inheritance

object ← IWeapon ← RangedWeapon

Inherited Members

IWeapon.attackCD , IWeapon.attackRange , IWeapon.attackDamage

Fields

projectilePrefab

```
public GameObject projectilePrefab
```

Field Value

GameObject

Methods

Ready()

Get whether the weapon is Ready or not, typically when the cooldown is zero.

```
public override bool Ready()
```

Returns

bool

Whether the weapon is ready or not

Usable(GameObject)

Get whether the weapon is Usable, typically when the cooldown is zero AND the player is close enough.

```
public override bool Usable(GameObject target)
```

Parameters

`target` GameObject

Returns

bool

Whether the weapon is usable

Use(GameObject)

What to do when the weapon is used, eg. spawn projectiles, damage the player directly, etc.

```
public override IEnumerator Use(GameObject target)
```

Parameters

`target` GameObject

Returns

IEnumerator

IEnumerator: This happens asynchronously as a Unity coroutine, so it must return an IEnumerator

Class SceneAudio

Namespace: Global

```
public class SceneAudio : MonoBehaviour
```

Inheritance

object ← SceneAudio

Fields

musicSlider

```
public Slider musicSlider
```

Field Value

Slider

sfxSlider

```
public Slider sfxSlider
```

Field Value

Slider

Methods

PlaySound(AudioSource)

```
public void PlaySound(AudioSource sound)
```

Parameters

`sound` AudioSource

SetMusicVol(float)

```
public void SetMusicVol(float vol)
```

Parameters

`vol` float

SetSFXVol(float)

```
public void SetSFXVol(float vol)
```

Parameters

`vol` float

Class ScoreModel

Namespace: Global

```
[Serializable]
public class ScoreModel
```

Inheritance

object ← ScoreModel

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

name

```
public string name
```

Field Value

string

score

```
public int score
```

Field Value

int

Class ScoreResponse

Namespace: Global

```
[Serializable]
public class ScoreResponse
```

Inheritance

object ← ScoreResponse

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

data

```
public List<ScoreModel> data
```

Field Value

List<ScoreModel>

message

```
public string message
```

Field Value

string

statusCode

```
public int statusCode
```

Field Value

int

Class SerializationTest

Namespace: Global

Class implementing unit tests for level serialization/deserialization.

```
public class SerializationTest : MonoBehaviour
```

Inheritance

object ← SerializationTest

Methods

RunTest()

Tests serialization by verifying a level is deserialized, then serialized back into the same string.

```
public void RunTest()
```

Class Startzone

Namespace: Global

```
public class Startzone : MonoBehaviour
```

Inheritance

object ← Startzone

Fields

Instance

```
protected static Startzone Instance
```

Field Value

Startzone

Class Tile

Namespace: Global

A struct so objects interacting with Tile can easily get its `GameObject` and `Transform`, eg. when serializing. GameObjects can't hold onto values otherwise, so sometimes Data classes are necessary in Unity.

```
public class Tile : MonoBehaviour
```

Inheritance

object ← Tile

Fields

obj

```
public GameObject obj
```

Field Value

GameObject

trfm

```
public Transform trfm
```

Field Value

Transform

Class Transition

Namespace: Global

Animates menus.

```
public class Transition : MonoBehaviour
```

Inheritance

object ← Transition

Fields

Instance

```
public static Transition Instance
```

Field Value

Transition

Methods

HideScreen()

```
public void HideScreen()
```

ShowScreen()

```
public void ShowScreen()
```

Class WaterTile

Namespace: Global

Slows the player down when a Tile with this Monobehavior collides with the player.

```
public class WaterTile : MonoBehaviour
```

Inheritance

object ← WaterTile