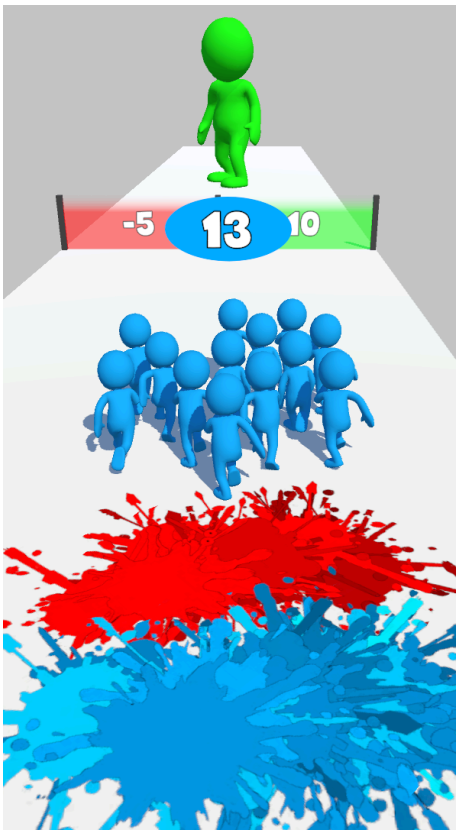
FOREWORD



Thank you for purchasing the Merge Tower Defence Template!

This is a special asset. You might think you've acquired an ordinary prototype of a hyper-casual game, but in reality, you have a convenient tool for creating a gaming hit in your hands. Once I supported and promoted a similar project for a major product company, and over the two years of working on it, I had some ideas how to make such a game exciting, undemanding, and easily expandable. Additionally, I have collaborated extensively with a motion design team that created promotional videos for game marketing, helping me understand how to make the project clear and user-friendly for those without programming skills.

I sincerely hope that this prototype will be a valuable asset for you in any gaming endeavor. :)

SUPPORT

I really look forward to your feedback and inquiries! Feel free to write to me at alex.of.navahrudak@gmail.com - I'll be happy to help with any issues related to the asset.

When seeking assistance, please include the invoice number received upon purchasing the asset in your email.

G&P

OVERVIEW

Runner Clash Template is a simple, undemanding, and easily expandable template for creating hyper-casual games. Games of this genre easily garner thousands of downloads because their rules are extremely simple, and the gameplay looks visually appealing and captivating. Perfect for passing the time on the bus on your way to work.

To create your game based on this prototype, you don't need to be a professional programmer. Most game objects are created and configured directly in the editor - all that's left is to prepare the game scene.

FEATURES

- An animated stickman model that helps you start making your game immediately. You can use it in your ready-to-play version or replace it with other characters you have.
- The template contains no demanding components (such as colliders, Rigidbody, etc.), so to customize levels, you simply need to add environment models and materials.
- The template is based on an easy-to-understand Event-Driven Architecture (EDA).
- A well-designed set of managers controls the game objects, requiring no additional customization.
- To prepare the game for play, simply add the GameCore component to the scene. Yes, that's all!
- All global settings are stored in a single file (you can create and save an unlimited number of such settings files).
- Five custom animations: Idle, Run, Fight, Death, Dancing.
- The game is adapted for mobile devices:
  - ➔ Heavyweight functions like GetComponent<>() are called only when absolutely necessary,
  - ➔ The Update() method is encountered only once throughout the entire project,
  - ➔ Object pooling is used for spawning projectiles, enemies, and effects,
  - ➔ As I've said before, there are no demanding components like colliders or Rigidbody in the template,
- etc.

MANAGERS / CONTROLLERS

GameController launches the game and initializes all other managers. Requires GameData configuration file.

InputManager processes the player's input and triggers the appropriate actions, which then initiate various game events.

FightController manages all combat processes involving crowds and bosses.
public class FightController : MonoBehaviour

CameraController controls the game camera making it follow the player's crowd on a certain distance.

FxSpawner works as a Fabric for effects spawning.

LevelLoader manages the level loading process.

UIMessageController is used for handling game messages.

COMPONENTS

BaseUnit component serves as the foundation for all units in the game. It contains base logic that is common to all active units.

Crowd stores specific crowd data and includes game logic to control child units.

Gates stores specific gate data.

GemCounter is a simple component used for displaying the gem count label.

Level stores specific level data, initializes it's units and checks if the level has been completed.

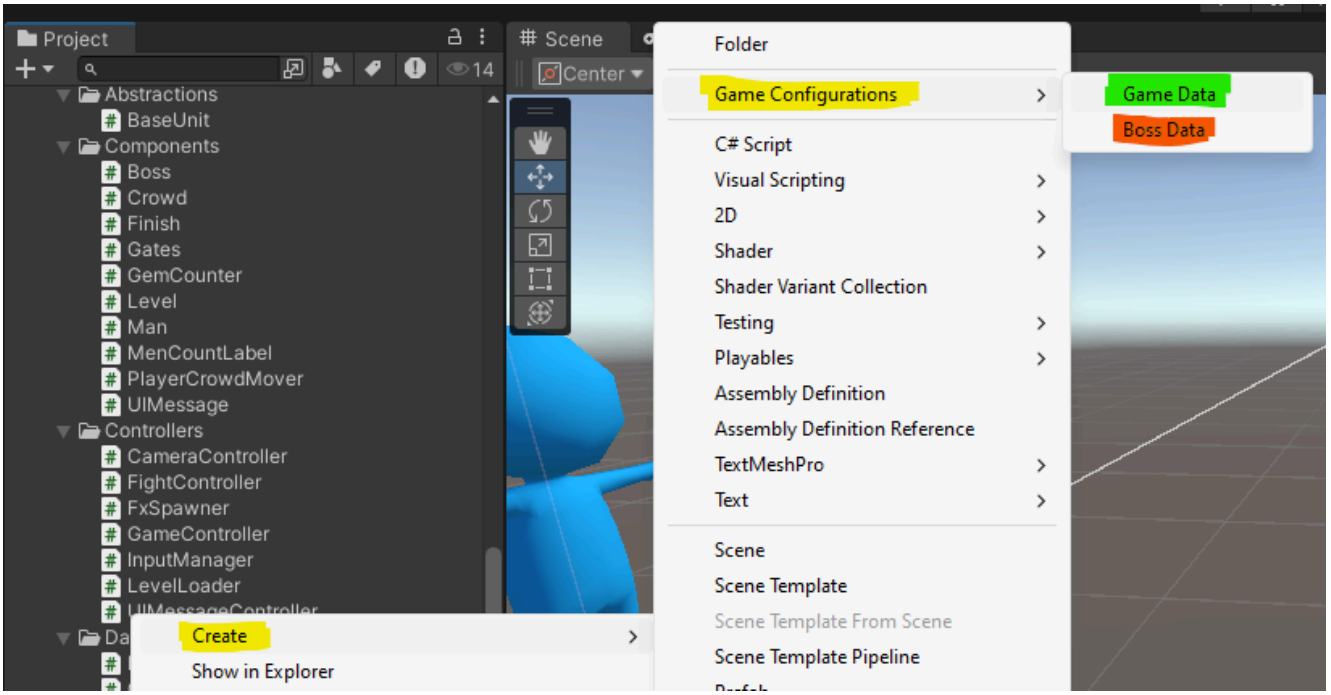
MenCountLabel shows a current number of men in a specific crowd.

PlayerCrowdMover moves the Player Crowd according to the InputManager signals.

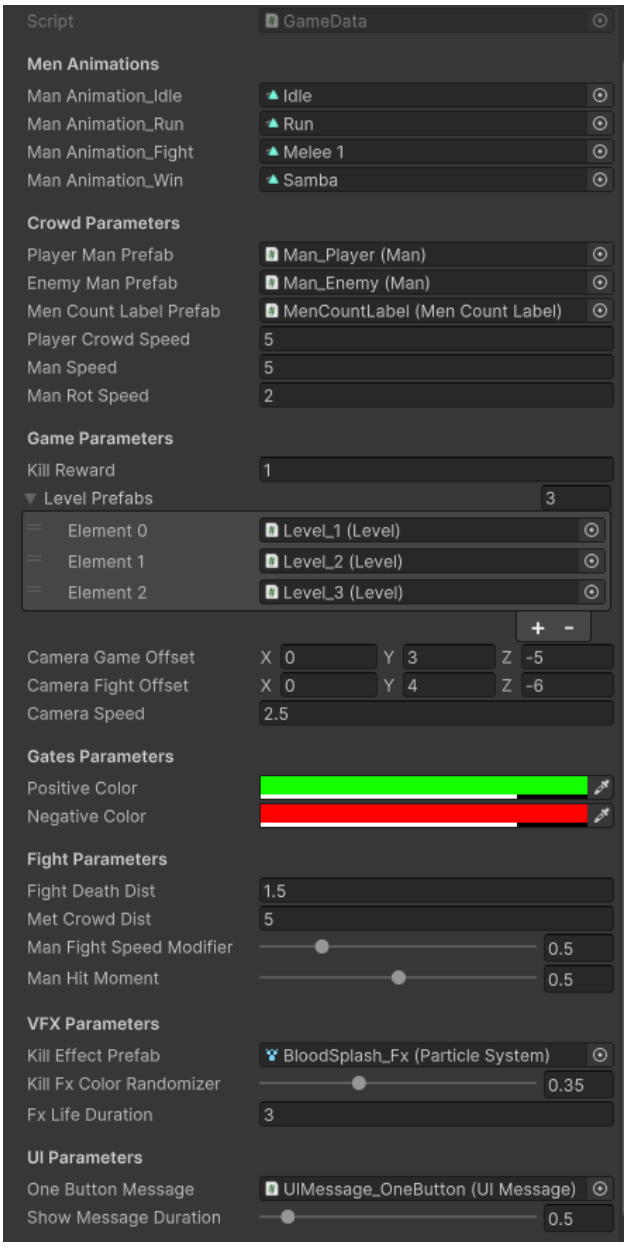
UIMessage is a component for a simple UI message for it's initializing.

CONFIGURATIONS

There are only two configuration files: GameData and BossData. You can create them by following the path *Right Mouse* / *Create* / *Game Configurations* / *Game Data* *(or BossData)*.



Game Data stores the general game settings.



Men Animations - a set of common animations for units.

Crowd Parameters - contains unit prefabs, counter label prefab and also some specific settings for the player's crowd.

Game Parameters - contains a looped set of levels and camera settings. Kill Reward field requires a number of gems you can get for every dead enemy.

Fight Death Dist - a distance at which a clash is detected.

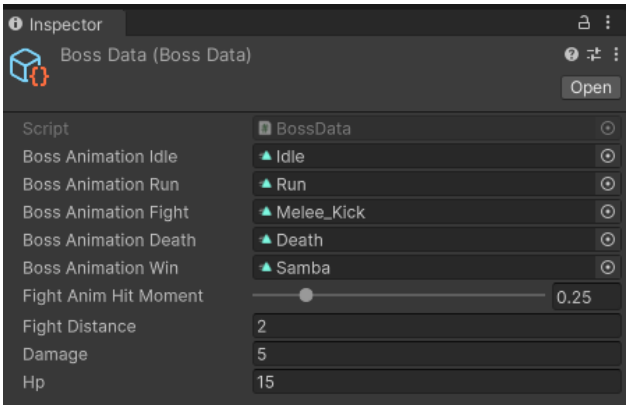
Met Crowd Dist - a distance at which a crowd fight is started.

Man Fight Speed Modifier - a modifier that requires a man's speed during a fight, relatively to his running speed.

Man hit moment - a specific moment in the fight animation when a hit can be detected (it is used in Boss Fights).

Kill Fx Color Randomizer - this parameter requires the color difference between the blood splashes of each killed man.

Boss Data stores settings for a boss.



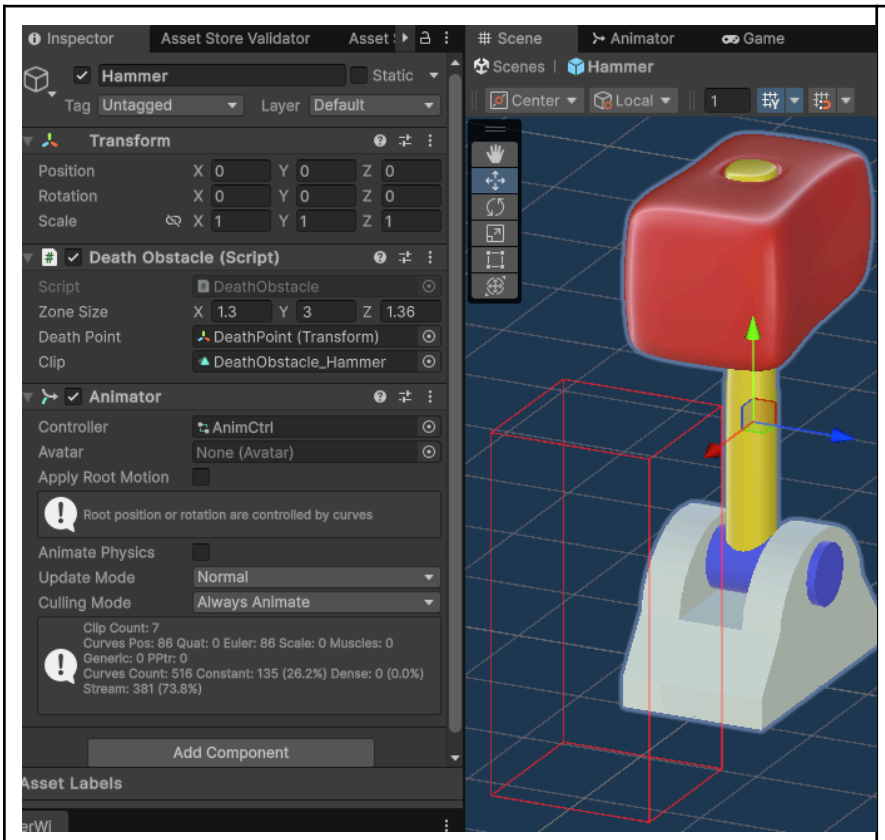
Fight Anim Hit Moment - a specific moment in the fight animation when a hit can be detected.

Fight Dist - a distance at which a boss fight is started.

ALSO PLEASE KEEP IN MIND : There are some constant values in `ValuesCounter`

VERSION 1.1.0

There are some new prefabs in the **'DeathObstacles'** folder. These objects kill the player's men if they enter the *death zone* (a red wireframe cube in the screenshot). To control them, use the **DeathObstacle** component.



Zone Size – Defines the Death Zone where men are killed.

Death Point – The central bottom point of the Zone Size. You can attach it as a child object to a moving part of the obstacle or keep it static.

Clip – An animation that defines the obstacle's behavior.

Note: Use the `StartKillProcess()` and `EndKillProcess()` events to define the periods when the killing behavior is active.