Developer Manual

Team 5 Guys

A red and black logo

Description automatically generated

Contents

[Environment Setup 2](#_Toc180780967)

[High Level Overview 3](#_Toc180780968)

[Context Diagram 3](#_Toc180780969)

[Class Diagrams 3](#_Toc180780970)

[Enemies 3](#_Toc180780971)

[Towers 4](#_Toc180780972)

[Level Manager 5](#_Toc180780973)

[Round Manager 6](#_Toc180780974)

[User Interface 7](#_Toc180780975)

[Scenes (Prefab Equivalent) 8](#_Toc180780976)

[Introduction to Scenes 8](#_Toc180780977)

[Creating a Scene 8](#_Toc180780978)

[Documentation 9](#_Toc180780979)

[Test Plan 9](#_Toc180780980)

[Static and Dynamic Binding 10](#_Toc180780981)

[Patterns 10](#_Toc180780982)

[Copy Right 11](#_Toc180780983)

# Environment Setup

* Download and install “Godot Engine - .NET” version 4.3-stable from: <https://godotengine.org/download>

A screenshot of a phone

Description automatically generated

* 1. Installation instructions can be found here: <https://docs.godotengine.org/en/stable/about/faq.html#how-do-i-install-the-godot-editor-on-my-system-for-desktop-integration>
* Download and install the latest version of git from: <https://git-scm.com/downloads>
* Clone the GitHub repository using:
  1. $ git clone <https://github.com/flightcontrol40/CS-383-Project.git>
  2. $ git clone [git@github.com:flightcontrol40/CS-383-Project.git](mailto:git@github.com:flightcontrol40/CS-383-Project.git)
* Launch Godot and import using the “Import” button
  1. A screenshot of a computer

     Description automatically generated
* After importing Godot should automatically open the project in the editor, and anytime you launch Godot afterwards the project should appear in the “Project Manager”

# High Level Overview

## Context Diagram

A diagram of a computer

Description automatically generated

## Class Diagrams

### Enemies

A screenshot of a computer

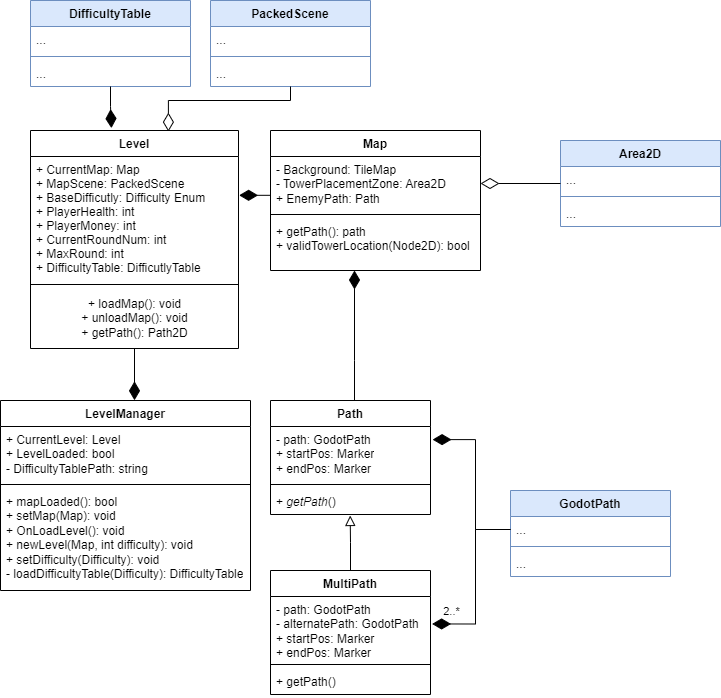
Description automatically generated

### Towers

A diagram of a tower

Description automatically generated

### Level Manager



### Round Manager

A computer screen shot of a computer

Description automatically generated

### User Interface

A diagram of a menu

Description automatically generated

# Scenes (Prefab Equivalent)

## Introduction to Scenes

There is no concept of a prefab, there are however scenes. Scenes allow us to get the general structure and functionality of an object figured out using Godot’s node system. Once the scene is created you can use the scene to define other scenes or use it in your game.

The general structure for Kill Mo’ Chickens will be:

Each feature will have some collection of scenes associated with it. Each feature’s scenes will be connected to a main scene that will mediate a connection for the scenes where needed.

## Creating a Scene

1. Open the Scene drop down menu

A screenshot of a computer

Description automatically generated

1. Click “New Scene” or press Ctrl+n

A screenshot of a computer

Description automatically generated

1. Select a root node based on the scene’s functionality

A screenshot of a computer program

Description automatically generated

1. Add nodes, as normal, to the new scene to give it the desired structure and functionality

A screenshot of a computer program

Description automatically generated

For the oral exam you will need at least one scene.

More info about scenes can be found at:

[Godot Scenes and Nodes Documentation](https://docs.godotengine.org/en/stable/getting_started/step_by_step/nodes_and_scenes.html)

# Documentation

All C# and GDScript files, classes, and functions should have comments. Make sure these comments follow the documentation standards provided in the coding standards document.

For the oral exam you will need to have at least one scene that has good documentation in a separate readme file, and should be able to answer:

* What question are you answering with your documentation?
* Who is the intended audience for this question?

# Test Plan

All features need a comprehensive test plan implemented for them. To do this, provide at least one test case for every non-trivial function. Each test plan will need 1 stress test and 30-unit tests.

For the oral exam you should have completed this test plan and be able to answer:

* What are you testing?
* Why did you choose these tests?
* Either explain an example of when a test case later found a bug in your code by things a teammate added later or why you chose a test case so a teammate would know if they broke your code.

# Static and Dynamic Binding

For the oral exam you should be able to:

* Be able to show a class in your code where there could be either static or dynamic binding.
* Write some mock code that would show how you would set the static and dynamic type of the variable and know which dynamically bound function gets called if the dynamic type gets changed.

More information can be found at:

* [Dr. BC's Dynamic Binding Video](https://vandalsuidaho-my.sharepoint.com/personal/jbeeston_uidaho_edu/_layouts/15/stream.aspx?UniqueId=692d7f0e%2Dcf02%2D5014%2D3432%2D7e08954b134b&Translate=false&referrer=StreamWebApp%2EWeb&referrerScenario=AddressBarCopied%2Eview%2Ec39ff92c%2Dcf70%2D4c15%2Da069%2D04344d2d3957)
* [Dr. BC's Dynamic Binding Lecture Slides](https://vandalsuidaho-my.sharepoint.com/personal/jbeeston_uidaho_edu/_layouts/15/onedrive.aspx?id=%2Fpersonal%2Fjbeeston%5Fuidaho%5Fedu%2FDocuments%2FCS383%2FSlides%2F10%5FStaticAndDynamicTypes%2Epdf&parent=%2Fpersonal%2Fjbeeston%5Fuidaho%5Fedu%2FDocuments%2FCS383%2FSlides&ga=1)

# Patterns

For the oral exam you should be able to:

* Name the pattern (or patterns if you chose 2 small patterns) you used
* Explain why you chose the pattern
* Draw a class diagram of the pattern
* Explain situations where this pattern would be a bad choice

More Information can be found at:

* [Design Patterns page from Source Making](https://sourcemaking.com/design_patterns)
* [Dr. BC's Pattern Lecture Slides](https://vandalsuidaho-my.sharepoint.com/:b:/g/personal/jbeeston_uidaho_edu/Ebf8jnMp3UlCgTnF1JBc4McBLU14G0RquDDznRKe4gKJqA?e=S712dO)
* [Dr. BC's Singleton Lecture Slides](https://vandalsuidaho-my.sharepoint.com/:b:/g/personal/jbeeston_uidaho_edu/Ed9GOCJh9bdKiOaWA4Qee98B6DzakAwKOcS5qpZEWwoi7A?e=uQscdU)
* [Dr. BC's Decorator Lecture Slides](https://vandalsuidaho-my.sharepoint.com/:b:/g/personal/jbeeston_uidaho_edu/ES0pDsv-EeNEnh_-BNLIqrQBdemCnXIC-Y6HbM3V8rXU0g?e=lGIK0W)

# Copy Right

For the oral exam you will need to be able to:

* Show an example where you violate copyright law, and explain why it violates copyright
* Argue that you are protected under fair use.
* Answer the questions:
  1. What did you have to do to integrate it with the code you wrote?
  2. What are the legal imlications if you market your code with the re-used portion?

More Information can be found at:

* [Dr. BC's Copyright Lecture Slides](https://vandalsuidaho-my.sharepoint.com/personal/jbeeston_uidaho_edu/_layouts/15/onedrive.aspx?id=%2Fpersonal%2Fjbeeston%5Fuidaho%5Fedu%2FDocuments%2FCS383%2FSlides%2F44%5FCopyright%2Epdf&parent=%2Fpersonal%2Fjbeeston%5Fuidaho%5Fedu%2FDocuments%2FCS383%2FSlides&ga=1)