Modular Complex System Brief

Documentation for My Modular Complex System

# System Overview

Please provide a brief overview of the modular system that you are creating. This should include the **purpose of the system** that you are created (i.e. what technical need or problem is it solving?).

-RPG skills system.

Using this tool, you will be able to create skills by giving the name, mana cost, type of skill (for example a damage, healing, debuff/buff), choose its range of effect or have it be single target, pick a max level.

# Packaging

Please describe how your program will be packaged for delivery for use by others.

This can come in one or many forms which is up to your choosing:

* Unity
  + A set of source files (.cs files)
  + A Unity package (.unitypackage)

# Quick Start – Using the Modular Complex System

## Integrating the System

Please list and describe each step that a fellow programmer would have to bring your complex system into their own program or even another library.

You should be able to provide these instructions to someone unfamiliar to your project but has enough technical knowledge to be familiar with the overall process

Add system to unity project:

-Download scripts/package

-Add scripts/package to your unity project

-drag asset into hierarchy

Create a skill:

-press play

-press create skill

-input what is needed to create a skill

-press create

Add skill to gameobject/player/enemy:

-Attach “assign skills” script to gameobject

-select an existing skill and assign it to a key/button(not sure)

## Building Upon the System

Please describe how a fellow programmer could build on your system in order to use your modular complex system.

For example, your system may provide a base type to inherit from in order for other programmers to provide additional logic that is tailored for their game.

-This system would allow programmers to create their own custom skills at runtime by interacting with UI. Then you can assign created skills to gameobjects in the inspector.

# Underlying Mathematical Operations and Algorithms

## Mathematical Operations

Please list and explain the reason for any notable **mathematical operations** that your modular complex system will need to undertake. The use of vector math alone is not notable – it is very common in game development to make use of vectors.

For example, it is notable and worth mentioning that your project will calculation *barycentric coordinates* to create a software renderer that will interpolate values for use in its fragment shader stage.

## Advanced Algorithms

Additionally, please list and explain the reason for any notable **advanced algorithms** that your modular complex system will need to undertake.

For example, it is notable and worth mentioning that your program will implement *behavior trees* as a part of creating a robust set of designer-friendly tools for authoring behavior trees.

# Additional Third-Party Libraries

Please identify and provide a link and license for each third-party library used to implement this modular complex system.

When discussing each third-party library, discuss why it is included rather being completely implemented by yourself.

# Research Material

Optionally, please include links or references to other research material that helped you develop this modular complex system.

Appendix I – Technical Design

# Technical Requirements

The technical requirements are the needs that must be met for your project to perform as required (i.e. meet the needs of the client or user of your modular complex system).

* Target Frame Rate
  + You may choose to vary this in different situations (i.e. 60fps gameplay, 30fps cutscenes)
    - 60fps
* Target Render Resolution
  + You may specify more than one, if applicable
  + This should serve as the baseline for how you author your user interface
  + Omit this if rendering is uninvolved
* Memory Usage Budget
  + Please specify if any memory will need to be dynamically allocated (and deleted/GC’d)
* Target Device and/or Hardware
  + If specifying a PC, please provide an approximation of the capabilities of the PC.
    - Should be able work on low spec PCs

# Technical Design

The technical design of a project is a provides a high-level understanding of how your project will be engineered to meet the requirements laid out above.

* Major Systems and/or Data Types
  + (Please provide a highly level description of how the systems or data types interact with each other)
  + Create/edit/delete skill system
    - Should be able to call create a skill by calling a function and giving it the necessary parameters
    - Can call a function that will increase skill level
    - Can call a function that changes a specified skill
    - Should be able to delete a skill
  + Some system that is attached to a gameobject of your choosing that can select the skills you want that gameobject to have
    - Can select one or more of existing skills to give gameobject
    - Can use skill
* Global and/or Persistent State
  + Global Variables
    - Type, damage, name, range, max level
  + Private Variables
    - current level, exp, max level cap
  + More than one character can have skills
* Unity (if applicable)
  + Scripting Runtime
  + Render Pipeline
  + Tags and Layers
    - Might need to add a tag to gameobjects you would like to target. For example, if you want your skill to know if an enemy is in range then you would set its tag to “enemy”

## Chart, box and whisker chart Description automatically generatedClass Diagram

# Technical Risks

If there any risks or further considerations that provide cause for concern, please list them here.

# Credits

Trey Gleason © 2022