UPDC

Unity to PokEngine Data Converter

Technical Document

Oleg Loshkin

December 1, 2019

TODO: more diagrams, add glossary, working bibliography, figure table, add labels to images

Contents

1 Introduction

- **Project's Context**
  As part of the game programming cursus at SAE Institute Geneva, for the **technical module GPR5100.2**, students of the second year must **assist students from the third year in completing their bachelor's project.**
  This year, the third year's **PokFamily team** develops a **video game for the Switch** and PC using a tailored **in-house engine**.
  Second year students must **assist them** by creating various **tools they will need** in order to create their game.
  This document describes the functioning of the **Unity to PokEngine Data Converter** tool, **UPDC** for short.

- **Project's Goals**
  - Create a useful tool that the PokFamily team will use to create their video game.
  - Learn to work in a non-academic environment in a team that depends on the student's performance.

- **Specific Problem**
  The PokFamily team uses the **Unity engine as an external editor**.
  The PokFamily team needs a tool to **convert ScriptableObjects to a JSON(meaning needed)** format **readable by the PokEngine** parser.

2 Analysis

2.1 Requirements

This project's requirements have two origins:

- **Academic requirements**
  - The task given by the team has been understood and done in time.
  - The tool is maintained by the student after the tool's completion.
  - The tool must be user-friendly.
  - The student understands how to manage data.
  - The student understands how a game engine interfaces with a game engine editor.
  - The student has organized himself and his work in a way to facilitate the work of others.
  - The tool's performance is reasonable.
  - The implementation is appropriately sophisticated.
  - The student understands the implications of non-academic teamwork.
- **Pragmatic requirements**
  - Convert ScriptableObject files to files readable by the PokEngine's parser.
  - The user must be able to interact with the tool via Unity.
  - The code must satisfy the quality and style expected by the team. C++ coding style is defined in the Coding Style Document. C# coding style is defined in UnityWorkOrganization document.
  - The student must communicate with the team appropriately and be dependable.
  - Allow other tools to export data via the UPDC.

2.2 Technologies Used

- **PokEngine**
  The PokEngine is the game engine developed by the PokFamily team. The engine is written with C++ standard 2017 and C++ standard 2014 for code running on the Nintendo Switch.
  The engine has a parser that is capable of reading JSON files. This parser is used to import data exported from Unity with UPDC.

- **Unity 2019.1.10f**
  Unity 2019.1.10f is used as an external editor.

- **Visual Studio 2017**
  Visual Studio 2017 is used for development of the PokEngine.

- **Git**
  github.com is used for versioning for the PokEngine source code. gitlab.com is used for versioning for the Unity prototype source code. Git bash is used for most interactions with the git framework. Merge conflicts are solved manually via text editor.

2.3 Interaction with Overall Project

WIP: rework: diagram of my tool in relation to unity + poke + other tools. written explanation as well

2.4 Other Unity Tools Interactions

1. Enemy Pattern Tool (Adam): allows to export enemy waves. TODO: brief explanation of tool, data that is exported

2. Spline tool (Cédric): allows to export Vector3's that define a spline. TODO: idem

3. Chunk loading/unloading system (Séb): allows to export chunks. TODO: idem

3 Execution

3.1 Solution

1. UPDC editor is accessible via the MainMenu/Tools/Unity to PokEngine Data Converter. TODO: show don't tell, diagram

2. The editor automatically detects .asset files located in folders located in Assets/Data. TODO: add diagram with folder structure

3. One or more assets can be selected for batch exporting. TODO: image

4. The destination folder for exportation can be selected. TODO: image

TODO: UML diagram, Model-View-Controller diagram, files in folder diagram, I/O diagram for tool and where it takes I and O to/from

3.2 Interface Wireframe

TODO: make bigger, label

3.3 Data Representation

TODO: data can be optimized with export as readable bool, image

3.3.1 Generic

Generic convertible data is saved as a .json
No values are expected in generics.

3.3.2 Splines

Splines are saved as a .pokspline
The contents is an array of vectors of variable size. The array is named "points". The vector's values are named "x", "y" and "z".

3.3.3 Chunks

WIP

3.3.4 Enemy Waves

WIP

3.4 Planning

The project uses a Scrum approach for day-to-day work. As such, here is the rough planning for the next six weeks available for the completion of this tool project. These six weeks are split between three two-week Sprints.

1. First Sprint goal:
   (a) Prototype the tool UI and export a few select ScriptableObject types.
2. Second Sprint goal:
   (a) Implement data conversion for few remainder of ScriptableObject types.
   (b) Allow importation of JSONs PokEngine side.
   (c) Optimize code.
3. Third Sprint goal:
   (a) Polish the UI.
   (b) Genericise and clean the code.

4 Summary

WIP TODO Adjust doc following feedback from Elias