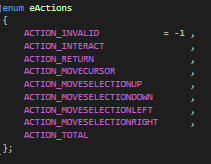
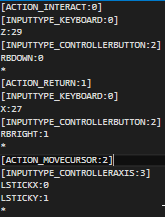
Strikefighter Additional Technical Documentation

Author: Joeb Rogers

Preface

This document shall be used to compile any information that will be useful for additional programmers/designers/external people that need to work with the game source. If there is a particular aspect of the codebase that you believe requires additional documentation, then please get in touch with me via the contact information at the bottom of the document with the details you would like to be added.

Input System



The body of the input system is driven by “actions”, which are defined in “Actions.hpp” (Left Image). Valid actions are 0-based and their index is important to remember for defining input maps.

The actions should be mapped to game input within each state via a .inputmap file (Stored in the /Input folder). An example is shown in the image on the right. An input map file is made up of individual action definitions; it is important to note that the order is very important, everything that defines an index number should be written in ascending order for it to work correctly.

Each action definition is represented as follows:

[Action Name:Action Index] – Name and index as defined in the enum file.

[Input Type:Input Index] – Name and index of the input type as defined in the input enum.

Input Key:Index – The input key represents what input button has been pressed and is purely for readability purposes, such as Z, XButton, MouseLeft. The Index should be defined as the index for that key as specified by the SDL2 library.

\* - Every action definition should be ended with a lone asterisk on it’s own line.

You can have as many input keys as you want under each input type heading. You should only specify one action header at a time. Also notice that headers are surrounded by square brackets. A colon is the delimiter for all definitions.

You should specify the input map to load in each game state’s “enter” method.

Image Assets

Images are currently stored in the /Textures folder. This folder is subject to additional organisation as the number of assets and their usage increases.

Currently, asset paths are defined in the “Constants.hpp” file under the “Const Strings” region. You should be able to figure out which assets are what based on their file names. This is a temporary solution until a robust system for UI building has been created.

These file names can be changed accordingly with new assets as appropriate. Please be aware that they may need to have changes made to their defined positions if the new image is of a different size to the old image. To find this, you should work backwards through the “setUpObjects” method of the game state to find out where it’s currently being stored and set. This will be the case until a proper system is built for UI positioning, defining and scaling.

Contact

Name: Joeb Rogers

Email: [joebrogers@pixcelstudios.uk](mailto:joebrogers@pixcelstudios.uk)