System and Unit Test Report

Team Name: Tool-Assisted Speedrun
Product Name: SuperTuxKart TAS Tools
Date: March 8, 2022

Sprint 1

Most user stories and tasks for this sprint were dedicated to infrastructure and ensuring that as developers we could build the game and have access to the source code and debugging tools.

One of these user stories did have a provable result however.
User Story 2 from sprint 1: As developers, we require that our application will have read/write access to the game's memory.

Scenario:

1. Run game.
2. Run Injector.exe, and verify that a text box pops up displaying a message.

Sprint 2

Here we shall ignore the two user stories that continued the infrastructure tasks from sprint 1. There is not much the user can test to verify those after all.

Secondly, user story 3 [As a TAS creator, I want to be able to run my scripts from anywhere in the game] has not been fully implemented by the release. It is mostly done, only crashing from running a script in the Exit menu. But as is standard in Scrum, this means the story is still incomplete.

A. User story 2 from sprint 2: As a programmer and TAS creator, we need a consistent and well documented TAS Scripting Syntax.

Scenario:

1. In the release code, User can go to the Docs/ directory and open the “TAS scripting syntax” pdf file.

Sprint 3

A. User Story 1 from Sprint 3: As a TAS creator, I want to send my TAS scripts to the parser and see that script executed in-game.

B. User Story 2 from Sprint 3: As a TAS creator, I want to be able to perform any possible movement in-game by writing some script.

These can both be tested by the following scenario.

Scenario:

1. The user can follow the TAS syntax documentation in order to create a script that performs basic movement. I.e. accelerate forward, then drift right, then brake, etc.

2. The user can run that script using parser.py after launching the game and verify that those movements did indeed occur, and quite consistently at that.

C. User Story 3 from Sprint 3: As a TAS creator, I want the application to provide feedback if I make a syntax mistake in the script.

Scenario:

1. Users can create a valid, working script.
2. Change a value in some line to get an invalid script.
3. Run the game.
4. Run the invalid script using the parser, and note an error shows up in the command line.
5. This error will display which line has the error, and some info on the line itself.

Sprint 4

A. User story 1 from sprint 4: As a TAS creator, I want to see examples of what scripts look like so I have a reference for making my own.

Scenario:

1. Any user can open a .peng script from the scripts/ directory from the release code.

2. Then the user can run that script using the parser after launching the game, and observe what lines from the script cause which actions in-game.
3. By using the playspeed feature to slow down movements in-game, it shouldn't be too complicated for the user to figure out what's happening.

B. User story 2 from sprint 4: As a TAS creator, I want to be able to set map properties from my TAS scripts e.g. number of AI karts, map difficulty, etc. (player character, number of laps, and difficulty are implemented)

Scenario:

1. When the user creates a script, in the header they can add any of the fields [kart_name, num_laps, difficulty] followed by a value.
2. Then the user can run the script using the parser after launching the game and observe that the fields have been accepted. (Assuming the field was valid)

C. User story 3 from sprint 4 [As a TAS creator, I want to be able to run my scripts from anywhere in the game.] is still unimplemented, since running a script from the Esc menu still crashes the game.