Project Brief

A domain-specific language for game modifications

Student Name: Daniel Best

Supervisor Name: Julian Rathke

Problem:

The modding community of many video games has grown significantly in recent times, with many developers providing extensive APIs which user-generated scripts can interact with to change elements of the game as they see fit. These may be as simple as an in-game soundboard, to as complex as an entirely new GUI that replaces the original UI of the game. Since games are played by all manner of people, the modding community of any given game can be incredibly diverse, teenagers may develop simple mods as their first foray into programming, or a seasoned programmer could make use of their skills to design a complex mod to enhance their game experience. Many games now support mods, from Roblox to Civilization and World of Warcraft. Mods are written in extension languages, i.e. a language that allows for the creation of programs that are invoked by the host application (game) – no language is more prevalent in this role than Lua, a dynamically typed scripting language designed primarily with this goal in mind. However, it was not designed with game modifications specifically in mind, just as a general-purpose extension language. Other languages can fulfil this role in games, such as C and Java, however these languages were never designed with the goal of being an extension language in mind; Lua is by far the most dominant choice in the industry. Therefore, the question remains, can a new domain-specific language (hereafter referred to as DSL) be created that is more suited to the task of game modification development, and as such makes the development of both simple and complex game mods easier for these modding communities?

Goal:

The goal of the project is to research game modifications in order to gain an understanding of what features, and especially what syntax, would aid modders (both novice and advanced) in creating better mods. This research will then be applied by designing and implementing a new DSL that improves upon the already existent solutions of Lua and other languages. The details of this language, including its type system and program semantics, will be documented in full, and any design decisions taken will be explained and justified in the final report. Since the modding language of a game is fixed by the developers of the said game, this new language will need to act as a *wrapper* that interprets source code, and then converts it into the appropriate source code for the game in question, as opposed to the usual binary output file that may be expected. Finally, several mods will be developed in order to show off the capability and suitability of the created solution.

Scope:

Due to the dominance of Lua, the DSL will initially be designed as a wrapper for that language. As Lua's interpreter itself is written in ANSI C (because it designed to easily embed itself into applications via a C API) it may also be possible to easily convert this proposed DSL into C source code using Lua as an intermediary. Support for any other scripting languages used for mods in games will be considered on a case-by-case basis dependant on the time remaining once a rigorous DSL that works for Lua-supported games has been designed and implemented.