

Technical Design Document

**Zephyr: A State-Based, Event-Driven, Domain-Specific Language for 2D, Top-Down, Action Role-Playing Games**

Version 1.0

|  |  |  |
| --- | --- | --- |
| **Position Title** | **Name** | **Signature** |
| Programmer | Tyler Wolverton |  |
| Advisor | Matt Butler |  |

* ***Delete all instructional information before turning in final version.***
* *Replace the image above with an appropriate image.*
* *Update the Header and Footer with the correct information.* 
  + *Lock the date to when the version is completed, not the date of its opening.*

Table of Contents

[Table of Figures 3](#_Toc63281931)

[Introduction 4](#_Toc63281932)

[Overview 5](#_Toc63281933)

[Scope 5](#_Toc63281934)

[End Product 5](#_Toc63281935)

[Development Schedule 7](#_Toc63281936)

[Schedule Detail 7](#_Toc63281937)

[Total Hours 7](#_Toc63281938)

[Technology Sources 8](#_Toc63281939)

[Theory 9](#_Toc63281940)

[Previous Work 9](#_Toc63281941)

[Artifact 10](#_Toc63281942)

[Implementation 10](#_Toc63281943)

[Architecture 10](#_Toc63281944)

[Syntax 10](#_Toc63281945)

[Results 11](#_Toc63281946)

[Playtest Results 11](#_Toc63281947)

[Profiling Results 11](#_Toc63281948)

[Example Images Section 12](#_Toc63281949)

[Example Image 1 12](#_Toc63281950)

[Example Image 2 13](#_Toc63281951)

[Example Image 3 14](#_Toc63281952)

[Example Citations Section 15](#_Toc63281953)

[Example Citation 1 15](#_Toc63281954)

[Example Citation 2 15](#_Toc63281955)

[Example Citation 3 15](#_Toc63281956)

[Bibliography 16](#_Toc63281957)

# Table of Figures

[Figure 1: Example Image 1 13](#_Toc61607850)

[Figure 2: Example Image 2 14](#_Toc61607851)

[Figure 3: Example Image 3 15](#_Toc61607852)

# Introduction

*Add your usual thesis introduction. Should be a short paragraph between 3 – 5 sentences in length, use common industry terms, and reference important existing algorithms or games.*

Zephyr is a domain-specific language that defines state machines that can listen to and fire events to interact with a custom C++ game engine. Source scripts are compiled into bytecode when the game loads on initial startup or via a hotkey. The bytecode is interpreted by a virtual machine (VM) at runtime whenever a script event is fired. The language is demonstrated with a 2D action role-playing game (RPG) with enemies to fight, keys to collect to unlock doors, and quests to complete, similar to games such as The Legend of Zelda (NES) and Dungeon Explorer.

# Overview

*Add a more detailed overview to this section explaining why this project is interesting to do and describe the level of detail needed in this document in order to assure the success of the project.*

## Scope

*Give a brief statement of the scope of the project from a technical point and view. Tie this to the design vision and introduce the overarching development plan for this project. Insert as much detail as necessary.*

* Compiler and VM
* Designer friendly syntax
* 2D top-down, action RPG demo game to show off scripts
* Map editor was not in scope

## End Product

*Describe the pieces that are going to be built in order to develop the final game as defined in the Game Design Document. This sets expectations for amount of work necessary.*

*Gameplay*

* *List separately all game mechanics that need to be implemented*

*Game Objects*

* *List separately all game objects that need to be implemented*
* *For example:*
* *Player Character(s)*
* *Enemies (list with description)*
* *Pick-ups*
* *Etc.*

*HUD & UI*

* *All GUI attributes implemented*
  + *List each aspect of the UI separately*

*Etc.*

* Tech
  + Syntax
  + Scanner
  + Parser
  + Interpreter ( VM )
  + ZephyrScriptDefinition, ZephyrObject
  + Entity system
  + VS code plugin
  + Core game systems to demonstrate language
  + Hot recompile functionality
  + GameAPI methods to communicate with engine
  + Doxygen docs for GameAPI methods
* Gameplay
  + Controls
    - WASD moves
    - Arrows attack
    - Space interacts with objects and NPCs
  + Game Objects
  + Enemies
    - Blob
    - Splitting blob
  + NPCs
    - Old man
  + Pickups
    - Keys
    - Triple shot power up
* Maps
  + XML based maps
    - Define tiles
    - Define entities
* UI
  + Current quest shown in top left of screen

# Development Schedule

## Schedule Detail

*Use a table or other suitable method to list out the major milestones and approximately how must time was spent on each.*

## Total Hours

*Put the total hours here and any other summary information that is appropriate.*

# Technology Sources

*Describe the external tools and technology used during production. May be omitted if nothing applies.*

VSCode plugin

Doxygen

# Theory

*Include as many sections as necessary to cover all theoretical background information that is considered a prerequisite to the thesis. Describe important math and algorithms.*

*This heading and the following three are interchangeable and optional. Choose the combination that best fits your specific thesis.*

* Compiler
  + Scanner
  + Parser
    - Pratt Parser
* Virtual Machine
  + Bytecode
  + Stack-based interpretation
* State-based AI
* Event System
* Syntax or designer friendliness?

# Previous Work

*Include as many sections as necessary to describe existing implementations or games that relate to the thesis.*

* Describe Naughty Dog’s scripting language

# Artifact

*Add sections under this heading as appropriate to describe the artifact.*

*This heading and the following three are interchangeable and optional. Choose the combination that best fits your specific thesis.*

# Implementation

*Add sections under this heading as appropriate to describe the artifact.*

* Compiler
  + Scanner
    - Text into tokens
  + Parser
    - Tokens into bytecode, saved in ZephyrDefinition
    - Break down specific magic I do to make revised language work
  + Error handling with entities
  + Recompiling without shutting down app
* Virtual Machine
  + Bytecode generation
  + Stack-based interpretation
* Explain how ZephyrObject works
  + State-based entity update
  + How scripts use the event system

# Architecture

*Add sections under this heading as appropriate to describe the artifact.*

* Diagram of the full compilation process

# Syntax

* Types
* State, Function, Variable, Conversation definitions
* Screenshots of code

# Results

*Add sections under this heading as appropriate to describe any data, test results or conclusions.*

*This heading and the following three are interchangeable and optional. Choose the combination that best fits your specific thesis.*

# Playtest Results

*Add sections under this heading as appropriate to describe any data, test results or conclusions.*

* Write up results from LD playtest

# Profiling Results

*Add sections under this heading as appropriate to describe any data, test results or conclusions.*

* Discuss optimizations made to strings if I get time to do that

# Example Images Section

## Example Image 1

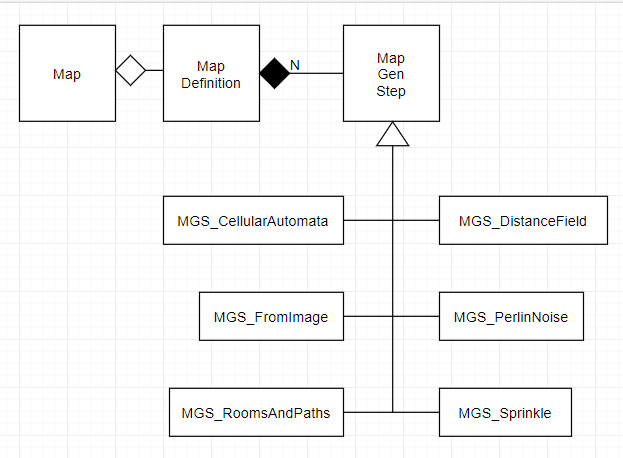


Figure 1: Example Image 1

## Example Image 2

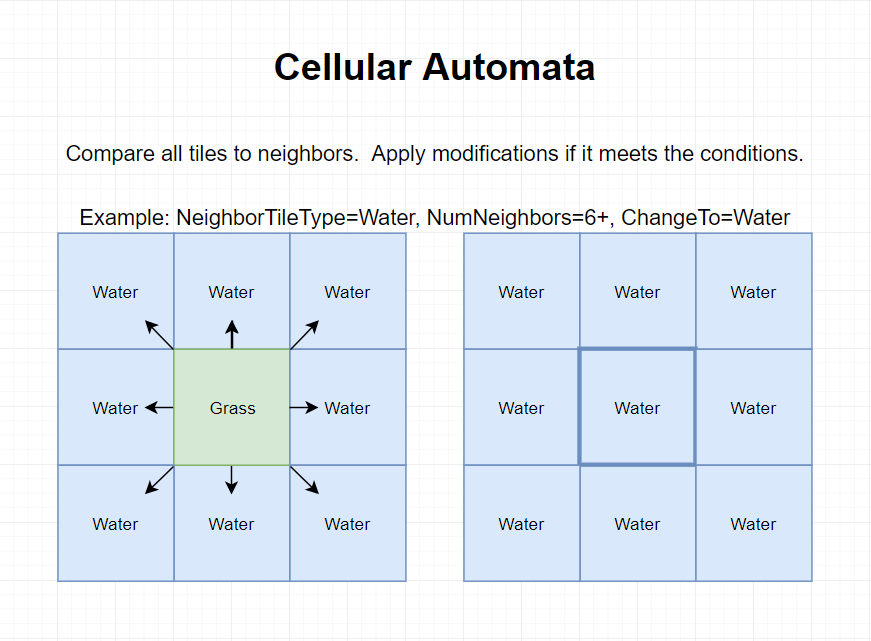


Figure 2: Example Image 2

## Example Image 3

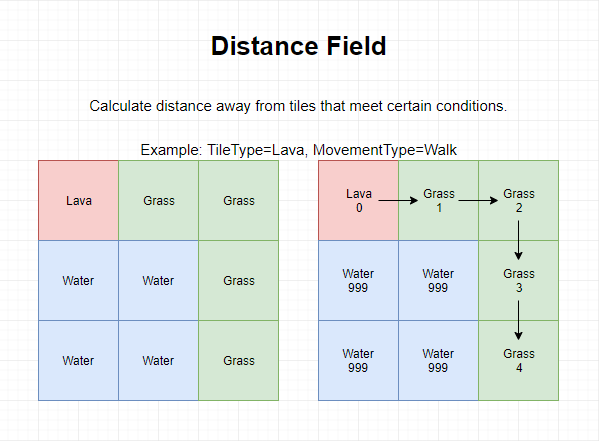


Figure 3: Example Image 3

# Example Citations Section

## Example Citation 1

Perlin Noise is a means of producing randomized heat maps using numbers in the negative one to positive one range [1].

## Example Citation 2

*Tiled* [2] is a map editor with many features.

## Example Citation 3

The *Tanagra* [3] map editor also has many features.

# Bibliography

|  |  |
| --- | --- |
| [1] | Wikipedia, "Perlin noise," 28 February 2020. [Online]. Available: https://en.wikipedia.org/wiki/Perlin\_noise. [Accessed 28 March 2020]. |
| [2] | T. Lindeijer, "Tiled: Map Editor," 26 March 2020. [Online]. Available: https://www.mapeditor.org/. [Accessed 28 March 2020]. |
| [3] | G. Smith, J. Whitehead and M. Mateas, "Tanagra: A Mixed Initiative Level Design Tool," 2010. [Online]. Available: https://people.engr.ncsu.edu/dlrober4/gamesreading/papers-s11/4-5.smith.10.pdf. [Accessed 28 March 2020]. |