# **Real Engine Map Documentation**

**Version 1.3.1.00 Build 39**

by Ethan Vrhel

# Table of Contents

1. Introduction to the Engine
   1. Introduction
   2. Features
2. Map Editing
   1. Introduction
   2. Setup
   3. Brush
   4. TiledBrush
   5. LowDetailBrush
   6. ReflectiveBrush
   7. Decal
   8. Trigger
   9. Entity
   10. Logic
   11. Prefabs
   12. Referencing
   13. Additional Tips

# Introduction to the Engine

## Introduction

The Engine is a versatile two-dimensional engine for running simple games. It allows for flexible creating with minimal programming which is used for programming game events and the graphical user interface. The Engine can also be heavily modified as it is directly apart of the source code along with the game code.

## Features

* Physics
* Triggerable Game Events
* Animations
* Graphical User Interfaces
* Map Editor
* Material System

# Map Editing

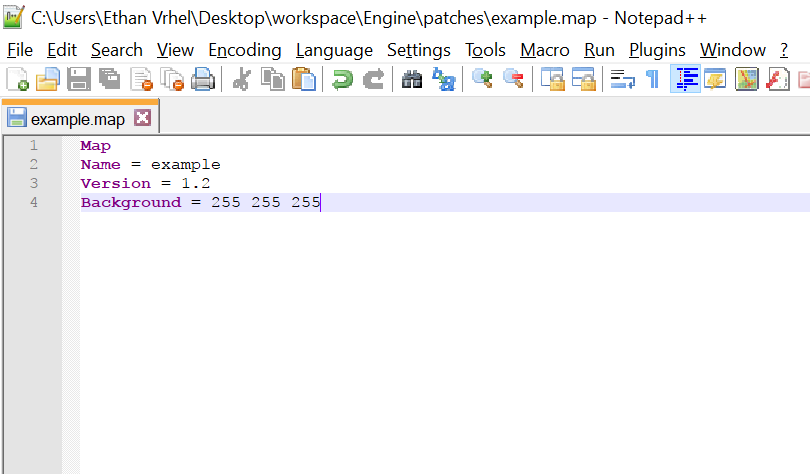
## Introduction

The engine comes equipped with a file parser that reads in from map files. It allows for many paths for level design as well as being able to implement programmable features within the game. It is recommended to use Notepad++ as the editor along with the .xml format to provide verification of keywords as it provides a better experience for level designers. All maps must be stored in maps/ and prefabs in maps/prefabs/. In this version the startup level is maps/test.map. The player will automatically go up brushes that are less than or equal to 16 units relative to the players position to act as a form of stairs. However, having a brush with a change less than this can cause the player to jitter. Some of the figures in this documentation might not contain newer features, so it is recommended to read the documentation thoroughly.

## Setup

A map file needs to be setup in its header. At the top of the type of the file must be defined; in this instance, the type is Map. Under the map identifier you have the option to use the ForceSeperate modifier. Without this, the map will not work on resolutions other than 1920x1080 so it is HIGHLY recommended to use it. The next part of setting up a map is defining the name, which must match the file name. You must also setup the minimum engine version as so: Version: [Version Number]. Although this is not necessary it is highly recommended to do so as it may cause errors on lower versions of the engine. The pure numerical version of the engine is the same as removing the dots from the actual version as shown in the settings window or a log file. The final part of setting up the map is defining the background color (which is optional). Everything in the map is defined in the format [map tool] = [arguments]. See figure 1 for an example.

Fig 4.1:

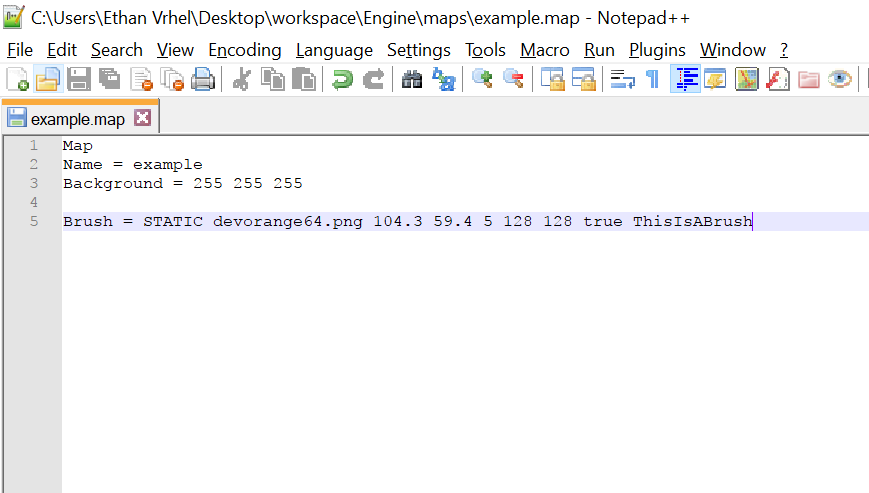


## Brush

The brush map tool is what makes up the basic geometry of the map. Typing “Brush” on a line will define a brush. The brush tool has arguments to specify the entity type, texture, location, size, mass, and solidity. See table below for argument lists. A STATIC brush indicates that the brush will remain stationary, and a DYNAMIC brush

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Brush | String | Tells the parser that it should treat this line as a Brush entity |
| = | String | Tells the parser that the arguments of the Brush tool is next |
| Type | String | The entity type [STATIC, DYNAMIC] |
| Texture | String | The texture of the brush under root folder resources/textures/ |
| x | double | The x position of the brush |
| y | double | The y position of the brush |
| z | integer | The z position of the brush |
| w | integer | The width of the brush |
| h | integer | The height of the brush |
| solid | boolean | The solidity of the brush |
| mass | integer | The mass of the brush (use only with DYNAMIC brushes) |
| kinetic | boolean | Whether the brush is affected by gravity (use only with DYNAMIC brushes) |
| name | String | The name of the brush, no spaces. Optional, used for referencing |

Fig 4.2  
An example of a Brush

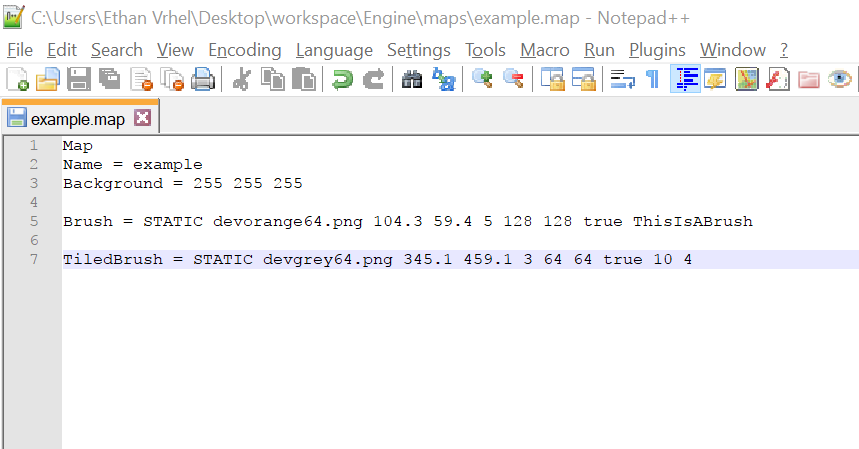


## TiledBrush

The TiledBrush is a way of placing down mass Brushes by tiling a texture with a width and a height.

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| TiledBrush | String | Tells the parser that it should treat this line as a TiledBrush entity |
| = | String | Tells the parser that the arguments of the TiledBrush tool is next |
| Type | String | The entity type [STATIC, DYNAMIC] |
| Texture | String | The texture of the brush under root folder resources/textures/ |
| x | double | The x position of the brush |
| y | double | The y position of the brush |
| z | integer | The z position of the brush |
| w | integer | The width of the brush |
| h | integer | The height of the brush |
| solid | boolean | The solidity of the brush |
| tx | integer | The tiling width of the brush |
| ty | integer | The tiling height of the brush |

Fig 4.3:



## LowDetailBrush

The LowDetailBrush map tool is what makes up the basic geometry of the map, but depending on the LOD options, the brush may or may not render

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Brush | String | Tells the parser that it should treat this line as a Brush entity |
| = | String | Tells the parser that the arguments of the Brush tool is next |
| Type | String | The entity type [STATIC, DYNAMIC] |
| Texture | String | The texture of the brush under root folder resources/textures/ |
| x | double | The x position of the brush |
| y | double | The y position of the brush |
| z | integer | The z position of the brush |
| w | integer | The width of the brush |
| h | integer | The height of the brush |
| solid | boolean | The solidity of the brush |
| LODnumber | integer | The level of detail the brush is on (1-3) |
| mass | integer | The mass of the brush (use only with DYNAMIC brushes) |
| kinetic | boolean | Whether the brush is affected by gravity (use only with DYNAMIC brushes) |
| name | String | The name of the brush, no spaces. Optional, used for referencing |

## ReflectiveBrush

The ReflectiveBrush map tool can create reflective surfaces. Cannot be a dynamic brush.

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Brush | String | Tells the parser that it should treat this line as a Brush entity |
| = | String | Tells the parser that the arguments of the Brush tool is next |
| Texture | String | The texture of the brush under root folder resources/textures/ |
| x | double | The x position of the brush |
| y | double | The y position of the brush |
| z | integer | The z position of the brush |
| w | integer | The width of the brush |
| h | integer | The height of the brush |
| solid | boolean | The solidity of the brush |
| reflectivity | double | The reflectivity of the brush |
| tileX | int | The tiling on the x plane |
| tileY | int | The tiling on the y plane |

## Decal

This should be used to a minimal as its use is nearly obsolete. It is the same as a brush but it always rendered behind the normal brush faces.

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Decal | String | Tells the parser that it should treat this line as a Decal entity |
| = | String | Tells the parser that the arguments of the Decal tool is next |
| Type | String | The entity type [STATIC, DYNAMIC] |
| Texture | String | The texture of the brush under root folder resources/textures/ |
| x | double | The x position of the decal |
| y | double | The y position of the decal |

## Trigger

The trigger brush is the most important brush in the editor. Without it, almost no gameplay would be possible. The trigger brush consists of many trigger types that can move objects, load levels, and trigger game events which are programmed into the game beforehand.

|  |  |
| --- | --- |
| Trigger Type | Purpose |
| TriggerLoad | Loads another map |
| TriggerMove | Moves another entity |
| TriggerStopTrigger | Disables another trigger brush |
| TriggerStartTrigger | Enables another trigger brush |
| TriggerGameEvent | Triggers a game event |
| TriggerSound | Plays a sound |
| TriggerShader | Enables a shader |
| TriggerGravity | Changes the gravity |

Base trigger arguments:

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Trigger | String | Tells the parser that it should treat this line as a Trigger entity |
| = | String | Tells the parser that the arguments of the Decal tool is next |
| Type | String | The entity type [STATIC, DYNAMIC] |
| Frequency | String | The frequency type, either true/false for once or con for a continuous update |
| x | double | The x position of the brush |
| y | double | The y position of the brush |
| Specific Trigger  Arguments | (variable) | The specific trigger arguments of a Trigger Type |
| Name | String | The name of the trigger brush with no spaces |

### TriggerLoad

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| Map Name | String | The name of the map to load |
| Offset x | integer | The offset of the map spawn along the x axis |
| Offset y | integer | The offset of the map spawn along the y axis |
| Normal Args | (variable) | Normal trigger brush arguments |

### TriggerMove

The entity that is going to be moved must be initialized as type DYNAMIC instead of STATIC.

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| Target Entity | String | The name of the target entity to move |
| Move x | double | The movement along the x axis |
| Move y | double | The movement along the y axis |
| Delay | integer | The delay of the trigger |
| Normal Args | (variable) | Normal trigger brush arguments |

### TriggerStopTrigger

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| Target Trigger | String | The name of the target trigger entity to disable |
| Normal Args | (variable) | Normal trigger brush arguments |

### TriggerStartTrigger

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| Target Trigger | String | The name of the target trigger entity to enable |
| Normal Args | (variable) | Normal trigger brush arguments |

### TriggerGameEvent

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| Target Event | String | The name of the game event to trigger |
| Normal Args | (variable) | Normal trigger brush arguments |

### TriggerSound

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| Sound | String | The name of the sound file to play |
| SoundType | String | The type of the sound (GENERIC, MUSIC, EFFECT, \*) |
| Normal Args | (variable) | Normal trigger brush arguments |

### TriggerShader

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| Shader Name | String | The name of the shader to enable |
| Normal Args | (variable) | Normal trigger brush arguments |

### TriggerGravity

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| Gravity | double | The gravity to be set |
| Normal Args | (variable) | Normal trigger brush arguments |

### TriggerHurt

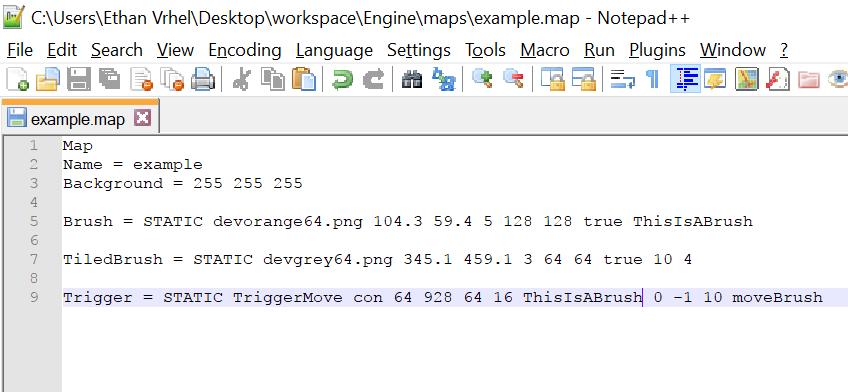
|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| Damage | integer | The damage to be dealt |
| Normal Args | (variable) | Normal trigger brush arguments |

### TriggerBreak

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| BreakableBrush | String | The target breakable brush |
| Normal Args | (variable) | Normal trigger brush arguments |

### TriggerGravityDecay

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| gravity | double | The target gravity |
| decayFactor | double | The amount to be decremented value per tick |
| rate | integer | The rate at which the gravity is decayed |
| Normal Args | (variable) | Normal trigger brush arguments |

Fig 4.4

## Entity

The entity within a map is different than an entity within the engine. Entities within maps are merely points that can serve special functions and do not have a x, y coordinate. To differentiate between the two, use “map entity” for map entities and “game entity” for entities within the engine.

|  |  |
| --- | --- |
| Map Entity Type | Purpose |
| PLAYER\_START | Defines the default spawn point of the player when the map is loaded |
| VARIABLE | A value that can be referenced in other map definitions |
| GEM | A collectable that can be placed around the map |

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Entity | String | Tells the parser that it should treat this line as an Entity |
| = | String | Tells the parser that the arguments of the Entity tool is next |
| EntityType | String | The type of desired map entity |

### PLAYER\_START

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal entity arguments |
| x | double | The player start on the x axis |
| y | double | The player start on the y axis |

### GEM

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal entity arguments |
| x | double | The gem’s location on the x axis |
| y | double | The gem’s location on the y axis |
| Name | String | The gem’s name (must be unique per level to avoid counting gems twice) |

### VARIABLE

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal entity arguments |
| name | String | The variable name |
| value | double | The value of the variable |

### TRACK

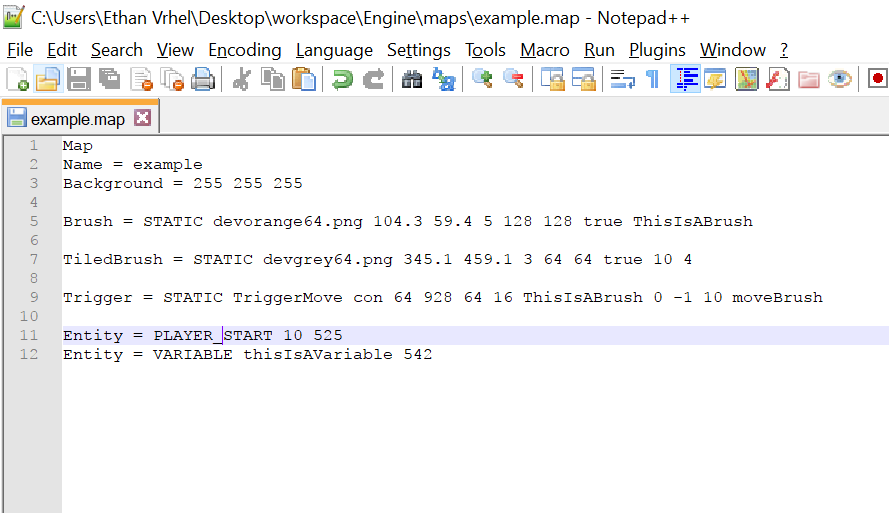
|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal entity arguments |
| type | String | The plane of the type (XPLANE or YPLANE) |
| movement | double | The movement of the track |
| delay | int | The delay in milliseconds before the track moves again |
| entity | String | The name of the target entity to move |
| ignorePlayer | boolean | Whether the track ignores the player as a wall |

### PARENT

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal entity arguments |
| child | String | The name of the map element to be parented |
| parent | String | The name of the map element that is to be the parent |

For referencing of variables, see Referencing in section 4.10

Fig 4.5



## Logic

Logic types within the map editor control various functions based off of different keys.

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Logic | String | Tells the parser that it should treat this line as an Logic |
| = | String | Tells the parser that the arguments of the Logic tool is next |
| LogicType | String | The type of desired logic type |

|  |  |
| --- | --- |
| Logic Type | Purpose |
| InvokeTrigger | Forcefully triggers a trigger brush |

|  |  |
| --- | --- |
| Key Type | Purpose |
| time | Logic based off of a timer |

### Timer Key

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| timer | String | Tells the parser that it should treat this key as a timer |
| : | String | Tells the parser that the next arguments are related to the timer |
| [timer beginning] | String | The time at which to start the timer |
| : | String | Tells the parser that the next arguments are related to the timer beginning |
| time | double | The time waited in seconds |

### InvokeTrigger

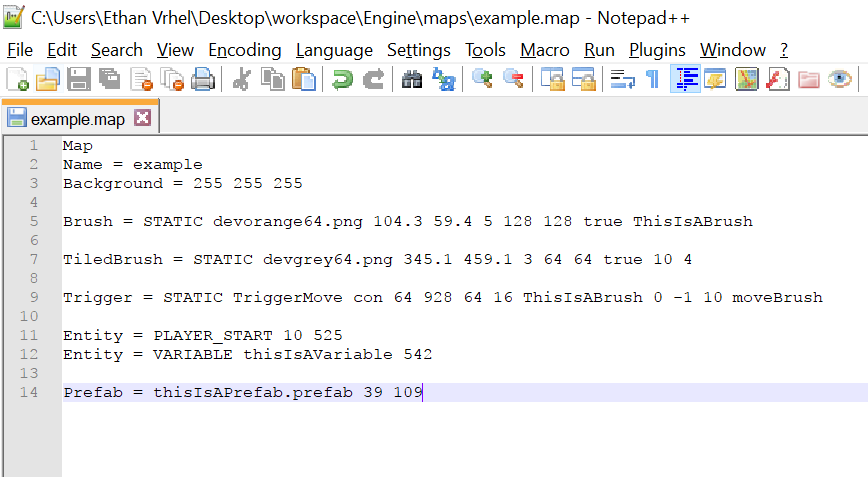
|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal Logic arguments |
| Key | String | The key name |
| TriggerName | String | The name of the desired trigger brush to be invoked |

## Prefabs

Prefabs are separate map files that can be referenced from any other map file. Prefabs carry the prefab extension and contain a header just like a map file. However, the header of a prefab file must be “Prefab”. Inside the prefab, the definitions act just like a map. Prefabs are stored in maps/prefabs/.

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Prefab | String | Tells the parser that it should treat this line as a Prefab |
| = | String | Tells the parser that the arguments of the Prefab tool is next |
| prefabName | String | The name of the prefab |
| x | integer | The offset of the prefab along the x axis |
| y | integer | The offset of the prefab along the y axis |

Fig 4.6



## Referencing

Advanced map features can be used by using referencing. References are placed inside entity definitions as replacements for values. They are defined with a “%” at the start.

|  |  |
| --- | --- |
| Reference Type | Purpose |
| Random | References a pseudorandom number generator |
| Variable | References a variable |

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| referenceType | String | Tells the parser that it should treat this value as a reference |

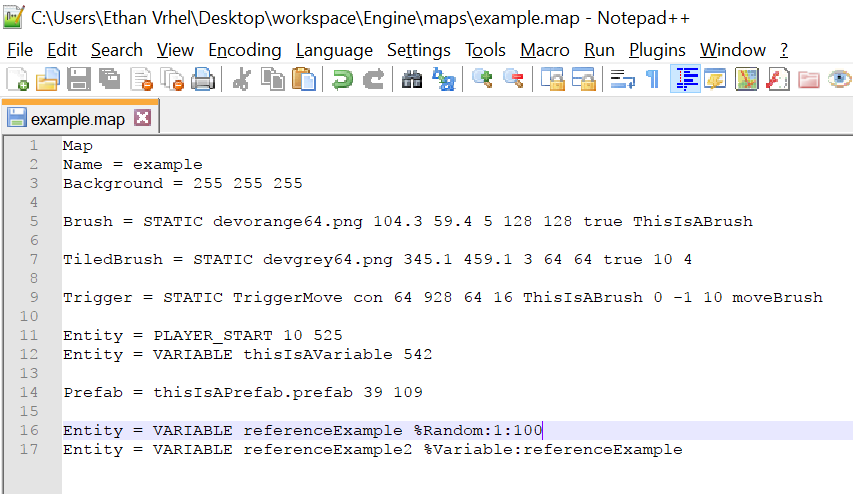
### Random

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | The normal arguments for a reference |
| value1 | integer | The minimum generated number |
| value2 | integer | The maximum generated number |

### Variable

|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | The normal arguments for a reference |
| variableName | String | The name of the referenced variable |

Fig 4.7



## Additional Tips

* Split large brushes into multiple parts to increase performance
* Limit the amount of ReflectiveBrushes in a visible area to increase performance (having multiple in the map outside of the viewport shouldn’t affect too much)
* Use LowDetailBrushes to help lower-end PCs run faster at a graphical cost