# **Engine 2017**

**Version 1.114 Build 14**

by Ethan Vrhel

# Table of Contents

1. Introduction to the Engine 4
   1. Introduction 4
   2. Features 4
2. Programming 5
   1. Introduction 5
   2. The Main Class 5
   3. The Engine GUI Classes 9
      1. GameWindow 9
      2. ImageHandler 11
   4. Engine Classes 13
      1. Animation 13
      2. AnimationHandler
      3. AnimationLoader
      4. Entity
      5. EntityHandler
      6. GameEvent
      7. GameEventHandler
      8. GuiElementHandler
         1. ButtonAnimationHandler
         2. GuiElement
         3. GuiAnimatedImage
         4. GuiButton
         5. GuiImage
         6. GuiText
      9. InputListener
      10. Light
      11. LightHandler
      12. Luxel
      13. MapEntityHandler
      14. MapLoader
      15. Material
      16. MaterialHandler
      17. MouseInputListener
      18. NavigationMesh
      19. PhysicsHandler
      20. RayTrace
      21. RenderLoop
      22. ResourceHandler
      23. Trigger
          1. TriggerGameEvent
          2. TriggerLoad
          3. TriggerMove
      24. TriggerHandler
      25. TypingHandler
   5. Utility Classes
      1. ClockTimer
      2. FramesCounter
      3. KeyBinds
      4. RenderMonitor
      5. ResourceMonitor
      6. RichText
3. Resources
   1. Introduction
   2. Textures
   3. Animations
   4. Sound
4. Map Editing
   1. Introduction
   2. Setup
   3. Brush
   4. TiledBrush
   5. Decal
   6. Trigger
   7. Entity
   8. Logic
   9. Prefabs
   10. Referencing
5. Legal
   1. Licensing

# 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

# Programming

## Introduction

The Engine allows for minimal programming, or heavy programming for more advanced games. The Engine comes equipped with many baseline java classes to be referenced for an easier experience.

## The Main Class

The Main class is the class that is called on the startup of the engine. It sets up all handlers needed for running the engine along with loading the resources into memory.

### Class Overview

**package main**

### Variables

**public static final double ENGINE\_VERSION**The version of the engine.

**public static final int ENGINE\_BUILD**The build of the engine.

**public static final String ENGINE\_NAME**The name of the engine.

**public static final String ENGINE\_VERSION\_NAME**The name of the engine including the version.

**public static boolean DEBUG**Toggles debug mode.

**public static final boolean OCCLUSION**Toggles occlusion.

**public static final int UPDATE\_RATE**The game update rate.

**public static final boolean DRAW\_TRIGGERS**Toggles drawing trigger brushes.

**public static final boolean DRAW\_NAVMESH**Toggles drawing navigation meshes.

**public static final boolean DRAW\_WIREFRAME**Toggles drawing in wireframe mode.

**public static final double RENDER\_REFRESH\_RATE**The maximum frames drawn per second.

**public static double renderRefreshBias**The multiplier of the RENDER\_REFRESH\_RATE.

**public static boolean stablizeFPS**Toggles FPS stabilization.

**public static boolean overrideScale**Toggles the scaling override.

**public static double resolutionScaleX**Used for scaling the game on different resolutions along the X axis.

**public static double resolutionScaleY**Used for scaling the game on different resolutions along the Y axis.

**public static File logfile**The base log output file.

**public static PrintStream out**The PrintStream to the output file.

**public static String debugMessage**The debug message when in debug mode.

**public static String loadMessage**The message when the map is loading.

**public static boolean loadingMap**Whether the game is loading a map.

**public static boolean lossyRendering**Toggles lossy rendering.

**public static int lossFactor**  
The loss factor of lossyRendering

[Game Handlers]

**private static GameWindow window**

**private static EntityHandler entityHandler**

**private static InputListener listener**

**private static Player player**

**private static ResourceHandler resourceHandler**

**private static MapLoader mapLoader**

**private static TriggerHandler triggerHandler**

**private static Thread playerThread**

**private static MouseInputListener mouseListener**

**private static GuiHandler guiHandler**

**private static Menu menu**

**private static RenderMonitor rMonitor**

**private static ButtonAnimationHandler bAnimationHandler**

**private static MapEntityHandler mapEntityHandler**

**private static PhysicsHandler physicsHandler**

**private static GameEventHandler gameEventHandler**

**private static GameMain gameMain**

**private static FramesCounter fpsCounter**

**private static AnimationHandler animationHandler**

**private static LightHandler lightHandler**

**private static int threadsRunning**The number of threads running.

**public static double player\_x**

**public static double player\_y**

**public static boolean gamePaused**Toggles if the game is paused or not

**public static boolean isMultiplayer**Is the game a multiplayer game

### Methods

**public static void main(String args[])**The main method; throws InterruptedException.

**public static mapExitCode(int exitCode)**Handles the map exit codes.

**public static int getThreadsRunning()**Returns the number of threads running

**public static GameWindow getGameWindow()**

**public static InputListener getInputListener()**

**public static EntityHandler getEntityHandler()**

**public static ResourceHandler getResourceHandler()**

**public static TriggerHandler getTriggerHandler()**

**public static String getMapName()**

**public static Player getPlayer()**

**public static void println(String str)**Prints to the in game console, log file, and System.out.

**public static void println(String str, Color c)**Same as previous method, but contains a color.

**public static getPrefix(String str)**Gets the prefix of a string.

**public static String getNextLogName()**Gets the next available log file name.

**public static void addPlayer(Entity entity)**Sets the player entity.

**public static int loadMap(String mapname, double xbias, double ybias)**Loads a map with a bias on the player spawns; returns an exit code.

**public static Thread getPlayerThread()**

**public static GuiElementHandler getGuiElementHandler()**

**public static MouseInputListener getMouseListener()**

**public static Menu getMenu()**

**public static RenderMonitor getRenderMonitor()**

**public static ButtonAnimationHandler getButtonAnimationHandler()**

**public static MapEntityHandler getMapEntityHandler()**

**public static PhysicsHandler getPhysicsHandler()**

**public static GameEventHandler getGameEventHandler()**

**public static FramesCounter getFpsCounter()**

**public static AnimationHandler getAnimationHandler()**

**public static LightHandler getLightHandler()**

## The Engine GUI Classes

### Overview

These classes control the low-level rendering.

### GameWindow

extends JFrame

Renders everything.

#### Variables

**GraphicsDevice device**

**Window window**

**DrawPane pane**

**public JOptionPane optionPane**

**private int leftBounds**

**private int rightBounds**

**private int topBounds**

**private int bottomBounds**

**private static GraphicsDevice gd**

**private static DisplayMode dm**

**public static final int XRES\_GL**The resolution of the monitor.

**public static final int YRES\_GL**The resolution of the monitor.

**public Color backgroundColor**

**public boolean drawBackground**

**public float fontSize**

**private int xRes**

**private int yRes**

**private String console[]**The debug console.

**private Color consoleColor[]**

**private String typedConsole**

**private int typeInterval**

**private int typeFrame**

**private boolean consoleOpen**

**private boolean borderless**Whether the game window is borderless or not

**private ImageHandler imgHandle**

**private short num**

**private long renderUpdates**

**public boolean lazyConsole**

#### Constructors

**public GameWindow(String name, int x, int y, boolean borderless, int cacheSize)**

#### Subclasses

**private class DrawPane extends JPanel**The panel that is drawn on.

#### Methods

**public void paint()**Redraws everything.

**public void addToCache(int nx, int ny, int nw, int nh, Color nc)**Unused.

**public void addToCache(int nx, int ny, Image image, Color c)**Adds an image to the render cache.

**public void addToCache(int nx, int nt, String text)**Unused.

**public int getTopBounds()**

**public void setTopBounds(int bounds)**

**public int getBottomBounds()**

**public void setBottomBounds(int bounds)**

**public int getRightBounds()**

**public void setRightBounds(int bounds)**

**public int getLeftBounds()**

**public void setLeftBounds(int bounds)**

**public int getScreenResX()**

**public int getScreenResY()**

**public ImageHandler getImageHandler()**

**public void addToConsole(String str, Color c)**Adds text to the debug console of color c.

**public void type(String key)**

**public RenderingHints getRenderingHints()**

### ImageHandler

#### Variables

**private boolean[] s**

**private int[] x**

**private int[] y**

**private int[] w**

**private int[] h**

**private Color[] c**

**private Image[] im**

**private String[] text**

**private float[] size**

**private Font[] font**

**private int cacheSizeDyn**

**private int numImage**

**private int cacheSize**

**private Random r**

#### Constructors

**public ImageHandler()**

**public ImageHandler(int cacheSize)**

#### Methods

**public void addToCache(int nx, int ny, int nw, int nh, Color nc)**Unsued.

**public void addToCache(int nx, int ny, Image image, Color c)**Adds an image to the render cache.

**public void addToCache(int nx, int ny, Strint ntext)**Unused.

**public void removeFromCache(int i)**Removes from the render cache.

**public void clearCache()**Clears the render cache.

**public int getX(int i)**

**public int getY(int i)**

**pubic int getW(int i)**

**public int getH(int i)**

**public Color getColor(int i)**

**public Image getImage(int i)**

**public int getDynSize()**

**public boolean isInUse(int i)**

**public int getNumImage()**

**public int getCacheSize()**

**public String getText(int i)**

**public float getSize(int i)**

**public Font getFont(int i)**

## Engine Classes

### Overview

These classes are the core of the entire engine and handle everything except game specific programming.

### Animation

**implements Runnable**

#### Variables

**private String name**The name of the animation

**private Image[] images**The frames of the animation

**private int frame**

**private int rate**The rate of the animation

**private boolean runAnimation**

**private boolean autoAnimate**

#### Constructors

**public Animation(String name, Entity entity, Image[] images, int rate)**

**public Animation(String name, GuiElement element, Images[] images, int rate)**

**public Animation(String name, Images[] images, int rate)**

#### Methods

**public void run()**

**public Image nextFrame()**Returns the next frame of the animation

**public Image frame(int frame)**Returns a frame

**public Image currFrame()**Returns the current frame

**public void setFrame(int frame)**

**public void update(Entity e)**Updates an entity with current frame

**public void update(GuiElement g)**Updates a guiElement with current frame

**public void setRate(int rate)**Sets the animation rate

**public boolean isAnimating()**

**public void shouldRun(boolean runAnimation)**

**public String getName()**

**public String toString()**

**public static void animate(Entity e, Animation a)**Animates entity e with animation a

**public static void animate(GuiElement g, Animation a)**Animates guiElement g with animation a

# Map Editing

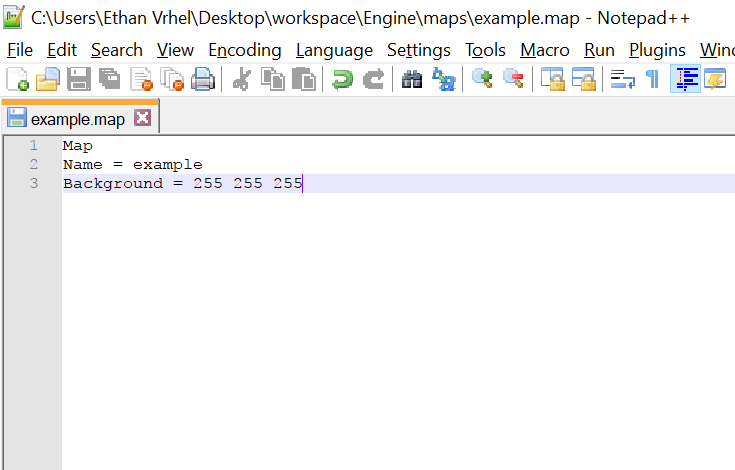
## Introduction

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 as it provides a better experience for level designers.

## 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. The next part of setting up a map is defining the name, which must match the file name. The final part of setting up the map is defining the background color. Everything in the map is defined in the format [map tool] = [arguments]. See figure 1 for an example.

Fig 4.1:

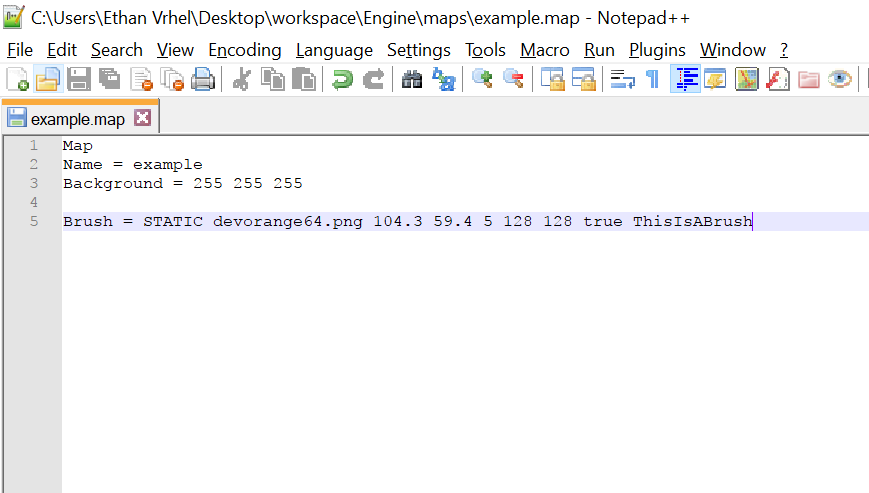


## Brush

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.

|  |  |  |
| --- | --- | --- |
| 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 |
| name | String | The name of the brush, no spaces. Optional, used for referencing |

Fig 4.2  
An example of a Brush

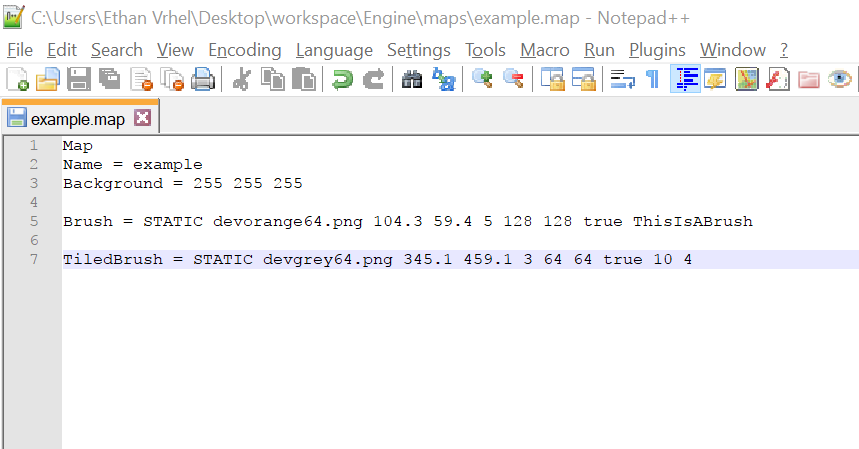


## TiledBrush

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:



## 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 |

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

|  |  |  |
| --- | --- | --- |
| 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 |
| Dekay | integer | The delay of the trigger |
| Normal Args | (variable) | Normal trigger brush arguments |

### TriggerStopTrigger

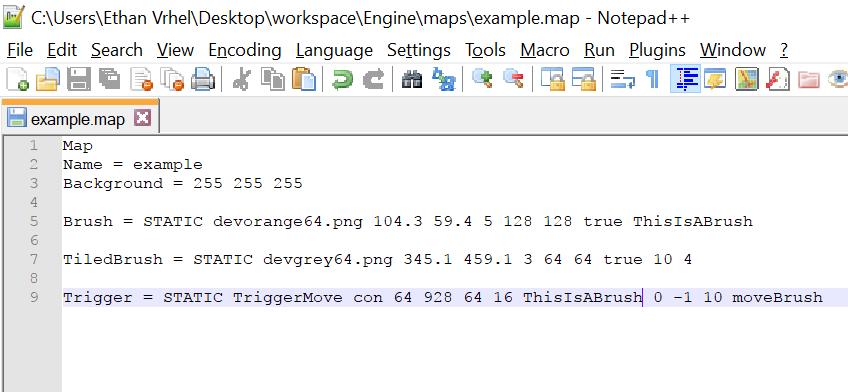
|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| Target Trigger | String | The name of the target trigger entity to disable |
| Normal Args | (variable) | Normal trigger brush arguments |

### 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 |

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 |

|  |  |  |
| --- | --- | --- |
| 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

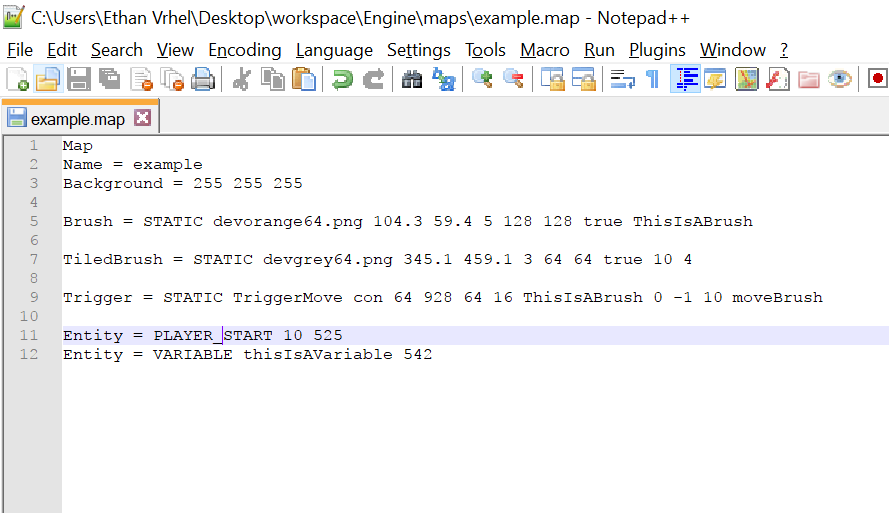
|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal trigger brush arguments |
| x | double | The player start on the x axis |
| y | double | The player start on the y axis |

### VARIABLE

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

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

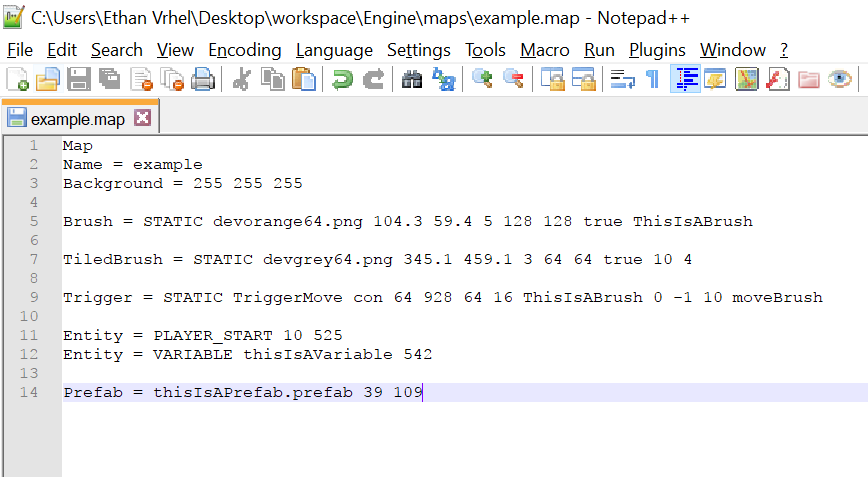
|  |  |  |
| --- | --- | --- |
| Argument Order | Data Type | Definition |
| Normal Args | (variable) | Normal Logic arguments |
| Key | String | The key name |
| TriggerName | String | The name of the desired trigger brush to be invoked |

## 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

