UNOFFICIAL RPG MAKER MV MODDING DOCUMENTATION

Version 0.0.1

JUNE 5, 2017

DamionWhite

Introduction

To start off I want to say that English isn't my native language, so if you find any strange sentence structures, misused words, or if a sentence could be worded better, feel free to contact me.

I also won't cover every aspect of RPG Maker MV, as I do not have any experience using MV and very little experience using older versions. I am, at the moment of writing, only interested in the experience of backwards engineering the MV engine, and the knowledge that comes by doing so.

So, this document is **not** for those who wish to learn how to use RPG Maker, but for those who wish to know more about the engine and how it works, and how to modify it.

Why am I writing this?

As I was looking into the source files of a RPG Maker project to figure out how to add a simple function I discovered that the RPG Maker Engine is somewhat more complex than I initially guessed. So, I decided to search around the web and the included help documentation for some information about making plugins, the different classes and their methods, but I didn't get much wiser.

The RPG Maker MV Help html file (press **F1** while working on a project to open) has a JS Library section which looks pretty helpful. It documents some classes and objects with their methods and properties. Feel free to look into it as it contains some interesting information. For me however it just felt incomplete as it does not cover any of the Javascript files in the project/js directory. These are the files I am more interested in, as they seem more essential.

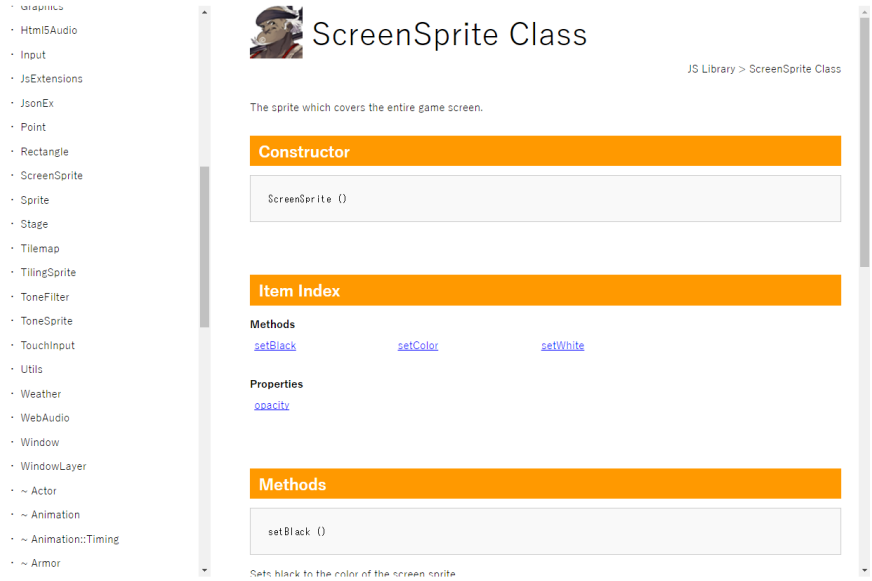

Figure 1 - help file opened in Chrome

The Javascript files seem to be well structured and not too complex at first sight, they are however large (some even have over 10,000 lines). I am hoping to cover most of the essential parts in this document, as well to guide you in exploring the possibilities of developing plugins. For me it is a first time doing something of this scale, so I really hope I can be of any kind of help.

And feel free to contact me as feedback is always appreciated!

Who is writing this?

I am a 20-year-old student who has a lot of free time on his hand. I love to use my free time to "research" certain aspects of game development, design and computer technology on YouTube (aka being too lazy to research yourself), and on those rare occasions I feel inspired, I will do my own research on certain aspects.

I recently started my course as an application developer, this however hasn't been much more than developing some webpages, some simple Javascript apps and some basic PHP MVC. I am also not a programming ninja, I still have much to learn, and lack experience and knowledge in some topics. But I see this as a learning process, and this will be one step I'll take into the world of programmers.

Table of Contents

Preparations

Before you read this document, I will strongly advise you to get some experience with Javascript and programming in general as I won't go in much detail on the basic, as I assume that the reader has some knowledge and experience. A quick google search on "Javascript guide" will give you plenty of resources.

I will also make use of the dev console in RPG Maker MV, this is the same dev console used in Google Chrome. All you have to do is press **F8** while testing a game in RPG Maker. If you have experience using the console while developing for the web then you'll be able to use most, if not all, methods of debugging and testing you are used to. So, feel free to console.log an object and look at it guts, or execute a method and see what happens. The console is a powerful tool if you know how to use it correctly.

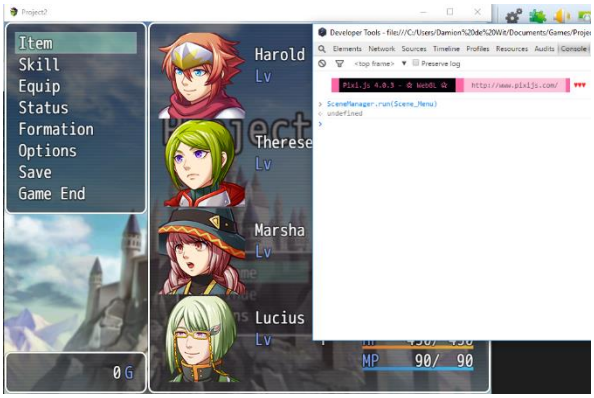

Figure 2 - Using the console to make the menu appear on the title screen

Preparing a project

First off, we need a project to experiment with. Go ahead and make a new project and name it whatever you want. I noticed that when you create a new project RPG Maker includes a lot of basic files which I won't need. A 'blank' project is almost 400 MB, more than half of this is because of the default music and graphic files imported into a project. I don't need 300 MB of music files and graphics, so let's get rid of those first if they bother you, I recommend using the resource manager in RPG Maker to do so. Below I will note all folders in the img folder which are greater than 20 MB, you might want to clean up those folders. It will also be worth to delete some background music files.

Folders which are at least 20 MB

Animations	*52 MB*
Battlebacks1	*59 MB*
Battlebacks2	*37 MB*
Titles1	*20 MB*

The JS Folder

This document will mostly focus on the JS Folder and its content. This folder can be found in the root of your project folder, which is easily opened in RPG Maker (*Game > Open Folder*). In this folder, we find 8 JavaScript files and 2 sub folders. This documentation will not cover the libs folder or its content.

main.js

The entry point of our program. There isn't happening much here. main.js starts with a method call to PluginManager.setup() with $plugins as its parameter.

$plugins is defined in plugins.js.

PluginManager is defined in rpg_managers.js.

The next statement sets the window.onload property to a function which calls SceneManager.run(Scene_Boot). It basically starts the first screen. You can pass any Scene Object.

Window.onload is a build-in method which I won't cover, if you are interested in it you can do a google search.

SceneManager and its run method are defined in rpg_managers.js.

Scene_Boot is a Scene object defined in rpg_scenes.js.

plugins.js

This file is generated by RPG Maker and is best not to edit directly.

This file contains $plugins which is an array containing data for each active plugin.

I recommend to only read this file when needed.

rpg_core.js

This is a file containing 8000+ lines. It seems to have methods for decrypting data, converting datatypes and handling all kind of object information. This file seems far more complex than most other included JavaScript files. I probably won't need to use any of these functions so I won't cover this file in my documentation (for now) as other files seem more usable.

> Graphics

rpg_managers.js

This file contains everything related to managers. Managers seem to be the ones who are doing most off the work. There are different kind of managers. You've got managers for sounds, graphics, data, plugins and more.

- DataManager
- ConfigManager
- StorageManager
- ImageManager
- AudioManager
- SoundManager
- TextManager
- SceneManager
- BattleManager
- PluginManager

rpg_objects.jp

Skimming through this file I can see a lot of different uses. But one pattern that I notice is that every Class in here uses the same Game_ prefix. Among the Class I saw Classes which handle battles, scrolling the map, gaining items and much more.

> Game_Temp

```
Some Shitty Shit Comment
```

> Game_System
> Game_Timer
> Game_Message
> Game_Switches
> Game_Variables
> Game_SelfSwitches
> Game_Screen
> Game_Picture
> Game_Item
> Game_Action
> Game_ActionResult
> Game_BattlerBase
> Game_Battler
> Game_Actor
> Game_Enemy
> Game_Actors
> Game_Unit
> Game_Party
> Game_Troop
> Game_Map
> Game_CommonEvent
> Game_CharacterBase
> Game_Character
> Game_Player
> Game_Follower
> Game_Followers
> Game_Vehicle
> Game_Event
> Game_Interpreter

rpg_scenes.js

Scenes are your screens, maps and menus. These seem to be easily made.

- Scene_Base
- Scene_Boot
- Scene_Title
- Scene_Map
- Scene_MenuBase
- Scene_Menu
- Scene_ItemBase
- Scene_Item
- Scene_Skill
- Scene_Equip
- Scene_Status
- Scene_Options
- Scene_File
- Scene_Save
- Scene_Load
- Scene_GameEnd
- Scene_Shop
- Scene_Name
- Scene_Debug
- Scene_Battle
- Scene_Gameover

rpg_sprites.js

Sprites are graphical objects to be drawn to a scene.

> Sprite_Base
> Sprite_Button
> Sprite_Character
> Sprite_Battler
> Sprite_Actor
> Sprite_Enemy
> Sprite_Animation
> Sprite_Damage
> Sprite_StateIcon
> Sprite_StateOverlay
> Sprite_Weapon
> Sprite_Balloon
> Sprite_Picture
> Sprite_Timer
> Sprite_Destination
> SpriteSet_Base
> SpriteSet_Map
> SpriteSet_Battle

rpg_windows.js

- Window_Base
- Window_Selectable
- Window_Command
- Window_HorzCommand
- Window_Help
- Window_Gold
- Window_MenuCommand
- Window_MenuStatus
- Window_MenuActor
- Window_ItemCategory
- Window_ItemList
- Window_SkillType
- Window_SkillStatus
- Window_SkillList
- Window_EquipStatus
- Window_EquipCommand
- Window_EquipSlot
- Window_EquipItem
- Window_Status
- Window_Options
- Window_SavefileList
- Window_ShopCommand
- Window_ShopBuy
- Window_ShopSell
- Window_ShopNumber
- Window_ShopStatus
- Window_NameEdit
- Window_NameInput
- Window_ChoiceList
- Window_NumberInput
- Window_EventItem
- Window_Message
- Window_ScrollText
- Window_MapName
- Window_BattleLog
- Window_PartyCommand
- Window_ActorCommand

> Window_BattleStatus
> Window_BattleActor
> Window_BattleEnemy
> Window_BattleSkill
> Window_BattleItem
> Window_TitleCommand
> Window_GameEnd
> Window_DebugRange
> Window_DebugEdit