阮中楠的 RPGMV 开发笔记

作者：阮中楠（B 站同名 UP 主）

目录

一、定稿说明

该版本已经定稿，不再更新。新版本的《阮中楠的 RPGMV 开发笔记》将会更新到 github，gitee，CSDN，简书，B 站等各大技术性平台。请在上述平台上面搜索“阮中楠”或“RuanZhongNan”，以便得到最新版的“笔记”。

感谢你的支持，愿 mv 开发界越来越好！

以下为本文的金山文档电子版阅读链接：

https://www.kdocs.cn/l/skwnSanY3L9J

二、序言

本教程提供基础教学和进阶教学。感谢 18 数应 1 雷文龙、19 计技(中)刘云龙、同学的技术支持。

本文会将大部分比较简单的东西全部略过，对于那些在网上有现成的教学的技术，本文只提供网址链接，不再复述。本文只讲解 RPGMV 这款开发引擎的基本用法。本文会对当前 B 站中的一部分教学视频进行一定的复述，以帮助读者查阅知识点。本文会以日记的形式来记录相关的知识点。

本教程适用人群：不会编程却又想做游戏的同学；会简单的编程技术，但是不会做大项目的同学；想学习 JavaScript 在网页前端的运用的同学；想复习 RPG 基本制作的同学。

（对序言进行更新）

（一）本文重构声明

为什么本文需要重构呢？随着作者阅历的进展，本教程的组织结构混乱，且内容落后，急需一次重构。重构的工作量非常大，且重构后的行文方式会发生重大变化，会更加倾向于用专业术语来描述内容。

未来将会提供给大家更完善的内容，和更方便的阅读手段。

（二）作者介绍

B 站同名 UP 主，阮中楠。曾获 2019 年 全国大学生数学建模竞赛 广西赛区 区二等奖。

（三）排版约定

本教程排版约定：宋体五号，首行缩进两字符，第一次出现的知识点名称、变量名等专有词汇会用双引号括起来，重点知识点会用**加粗**的形式标识。

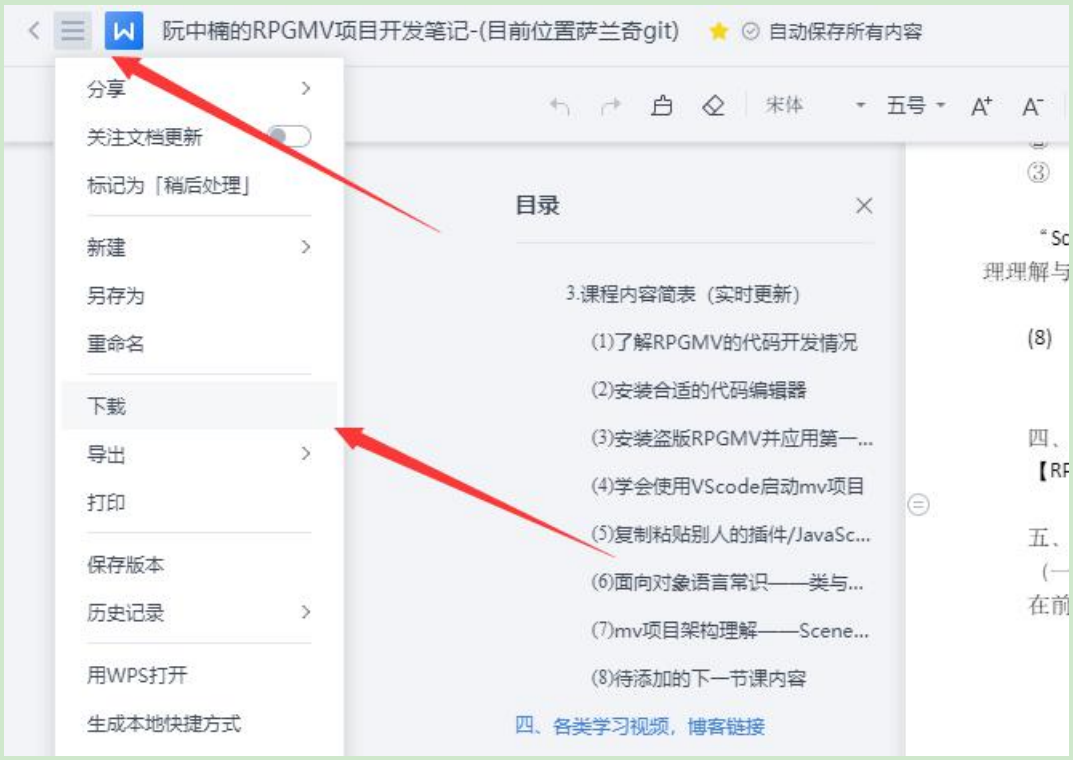
涉及到 RPGmv 项目的源码，会用***斜体加粗***的形式来表示。

（四）推荐书目（本条目暂时记录推荐的数目群）
《JavaScript 设计模式》
《RPG Maker MV 游戏制作基本外功篇:从操作到完成游戏一镜到底,马上就会!》
https://item.jd.com/69901403590.html

RPG Maker MV 遊戲製作 基本外功篇：從操作到完成遊戲一鏡到底，馬上就會！
作者： siakoMobi
出版社：拓客
出版日期：2018/07/25
語言：繁體中文
ISBN：9789865002312
規格：平裝 / 384 頁 / 17 x 23 cm / 普通級 / 全彩印刷 / 初版
出版地：中国中国台湾

（五）本文电子版开源链接
本文采用“金山文档”进行开源，该电子版链接为免登录链接，电子版访问链接如下：
https://www.kdocs.cn/l/skwnSanY3L9J

在本文档的网页版中，可能会因为显示问题而无法显示出以脚注形式展现的绝大部分的网址链接，为了显示并访问这些支撑性、拓展性的网址链接，请点击下载。下载选项的示意图如下：



三、RPGMV 软件使用简单教程-RPG 游戏制作教程(雷文龙)-2019.2.24
本部分对雷文龙的教程进行复述与重构。（待重构）

（一）作者介绍

雷文龙 18 数应 1 班

（二）认识 RPG 与制作攻略

从初识 rpg 到现在，已经差不多一年了，这一年里面曾放弃——又重新入坑

总的来说，这段经历认识了很多，首先 rpg 游戏可以一个人自己完成（不像 unity 3d 一样，需要团体）游戏终究还是游戏，娱乐娱乐，别荒废了学习，其次，一个好的 rpg 游戏不仅需要时间雕刻，而且还得有独特的眼光，设置一段内容精彩的剧情，能吸引广大玩家的眼球

做出一个 rpg 游戏后，一般将会有两个选择

1. **设置成收费**，可以去 steam 上登陆，出售自己的游戏（这样子的话，素材，图片，脚本必须要自己制作，公共素材基本不能用作商业游戏，不然人家作者会告侵权的）
2. **设置成免费的**，去 66rpg 论坛发布游戏，让广大群众免费玩，当然你可以让玩家自行决定，如果玩家觉得好玩，肯自己赞助打赏（类似主播一样），这样的方式是可以的

其他的就不多说了，以后会慢慢明白的，下面开始介绍 rpg 游戏开发引擎

Rpg 游戏常见又容易的开发引擎有 4 个（rpg maker mv，rpg maker xp，rpg maker vx，rpg maker Vx ace）各有各的长处，看你们的兴趣，用哪个都可以。常见又容易的 rpg 游戏开发引擎就这几个，当然如果你想挑战更高难度的，可以去学习 unity 3d 和虚幻 4，这两个是 3d 游戏和 2d 游戏的基础开发引擎，操作难度一般，努力学习肯定懂得（电脑低配置的，不建议用这 2 个，我的电脑就是用 unity 3d，用到现在经常蓝屏，卡得一批 0.0 说多都是泪）

Rpg 游戏开发引擎，我用的是 rpg maker vx ace，一开始我使用 mv 的，但是后来慢慢发现，va 更适合我，所以就改用 va（建议你们用 mv，mv 画质好，是编程界大佬总结 xp，vx，va 三个应用，拿其优点，创造了 mv），等最后展示一个 rpg 游戏的截图，是个不知名大佬的作品，用 mv 硬是做出了 2d 游戏的巅峰，已经接近 3d 游戏，与其他 mv 做出游戏相比，都是渣渣，真羡慕，也不知道自己什么时候才能做出一个这样的游戏，用自己的课余时间，朝着自己的想前进的方向努力吧。

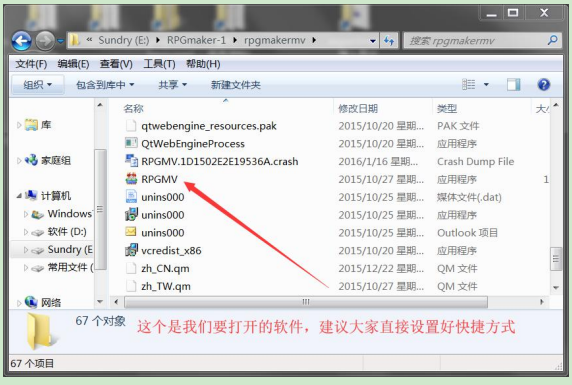
下面我开始介绍 mv（网上也有教程，如果不喜欢我写的，可去网上参考，放心，我不介意的 T.T【是我太丑的原因吗】）

在这里提供“RPG Maker MV v1.0.1”的下载链接：

http://www.xue51.com/soft/6586.html(附带安装简介)

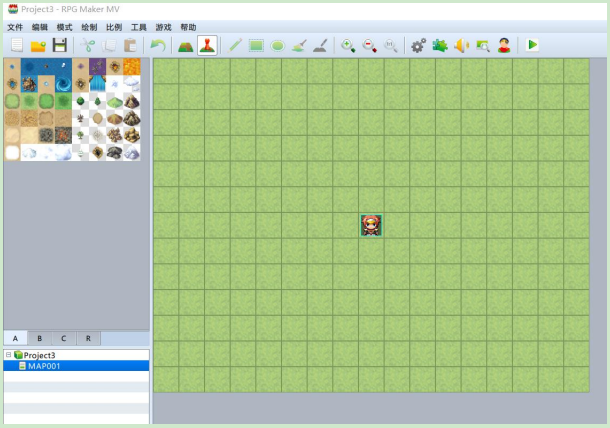
https://pan.baidu.com/s/1ZSqywn2f4lUxhv5BT__jAg(百度云下载链接)

本软件免安装，中文版，直接解压文件即可打开。现在打开文件夹，找到名为“RPGMV”的软件，打开即可。



打开 mv 后，会提示创建一个新工程，点击创建工程，接下来就是开始你的表演，各种秀操作。

创建好工程后，就会出现画面：



首先先熟悉 mv 里面的按键，

（从左边开始第一个）

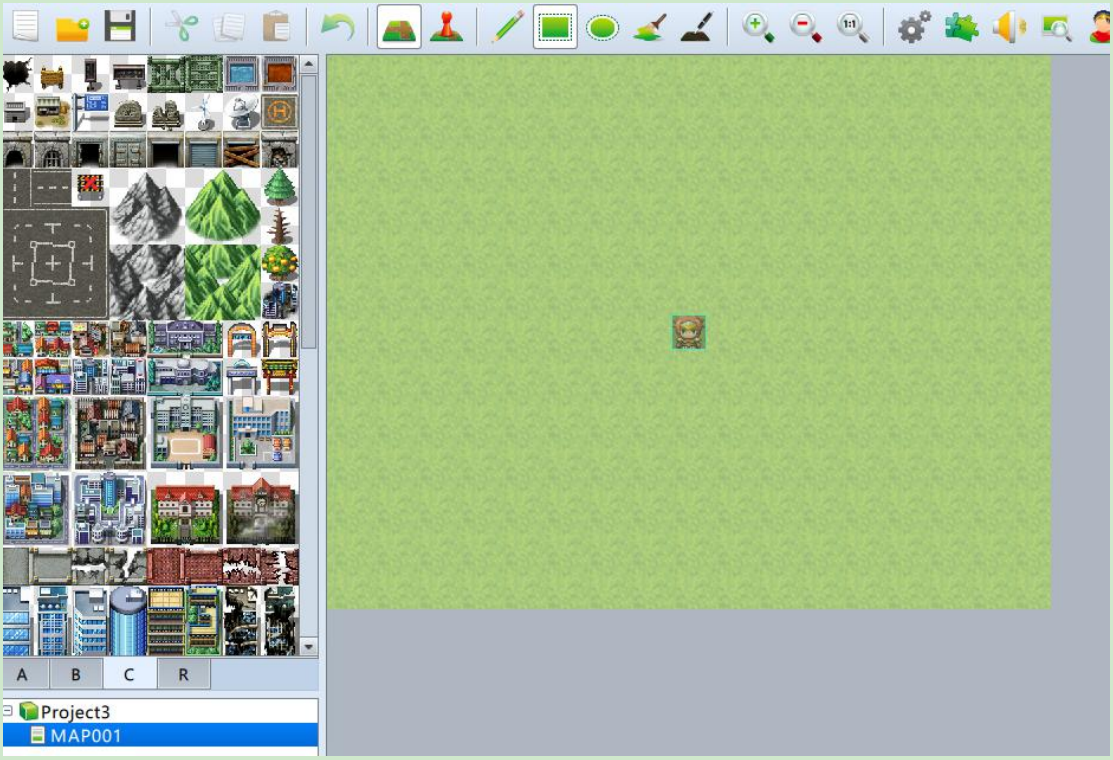
1. 新建工程
2. 打开工程
3. 保存工程
4. 4 到 6 不用管，基本用不上
5. 是返回上一次的操作，相当于误操作恢复
6. 绘图，即绘制地图
7. 事件的处理与安排
8. 10 到 17 是画笔的调整和地图屏幕的放大与缩小
9. 数据库
10. 插件的使用方法
11. 音乐的添加
12. 事件查找
13. 人物的生成
14. 游戏的测试

mv 里面，最烦的就是数据库的处理和事件的安排，至于地图的绘画是很简单的，只要有绘画基础的，基本都能制作，插件得看个人努力了，去互联网寻找插件或者去论坛找，当然如果精通 ruby 的，可自行编做插件，（大佬请无视）

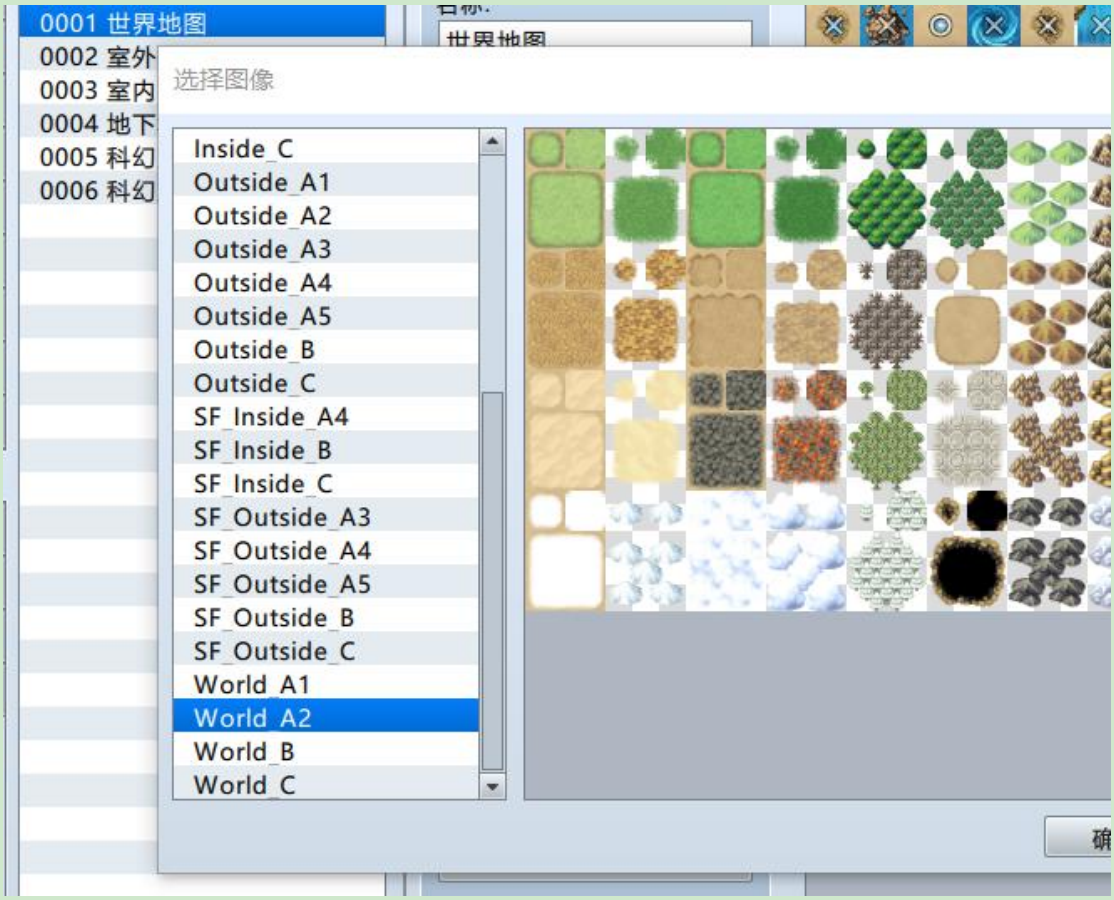
（三）地图的绘画

接下来介绍地图的绘制和图块组的添加

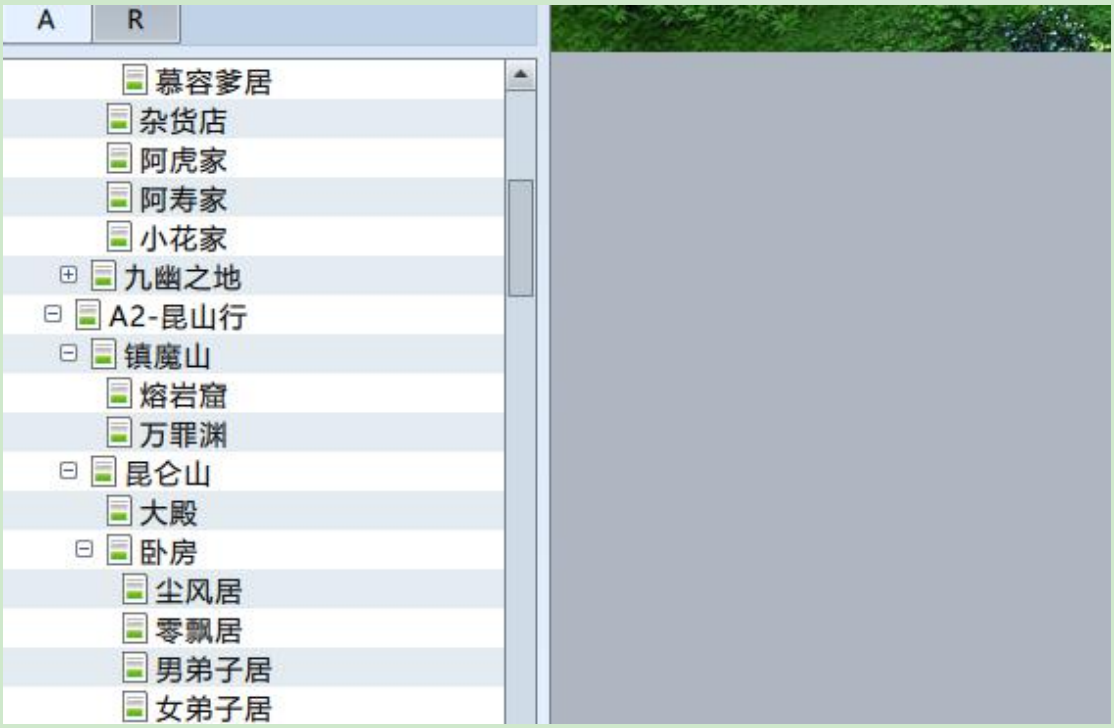

角色
职业
技能
道具
武器
防具
敌人
敌群
状态
动画
图块组
公共事件
系统
类型
用语
图块组
0001 世界地图
0002 室外
0003 室内
0004 地下城
0005 科幻室外
0006 科幻室内
更改最大值...
基本设置
名称:
世界地图
模式:
世界类型
图像
A1 (动画):
World_A1
A2 (地面):
World_A2
A3 (建筑):
A4 (墙壁):
A5 (普通):
B:
World_B
C:
World_C
D:
E:
A
B
C
通行
通行（4方向）
梯子
草木繁茂处
柜台
伤害地形
地形标记
注释
确定
取消
应用


A
B
C
R
Project3
MAP001

仅仅依靠 mv 自带的图块组是绘制不出精美的地图的，这时候就需要去互联网或 66rpg 论坛上搜素材，大多数都是别人绘制的，大部分是免费素材，少部分是收费的，找到图片后，然后就放到文件夹 ing——tilesets 里面，新建的工程都会帮忙创建一个文件夹的，接下来就去 mv 数据库图块组里面，把新素材添加到地图里



地图你想怎么绘制就怎么绘制，地图大小是可以随便调整的，最近流行末世地图，复古地图，有想法的同学可挑战绘画一下

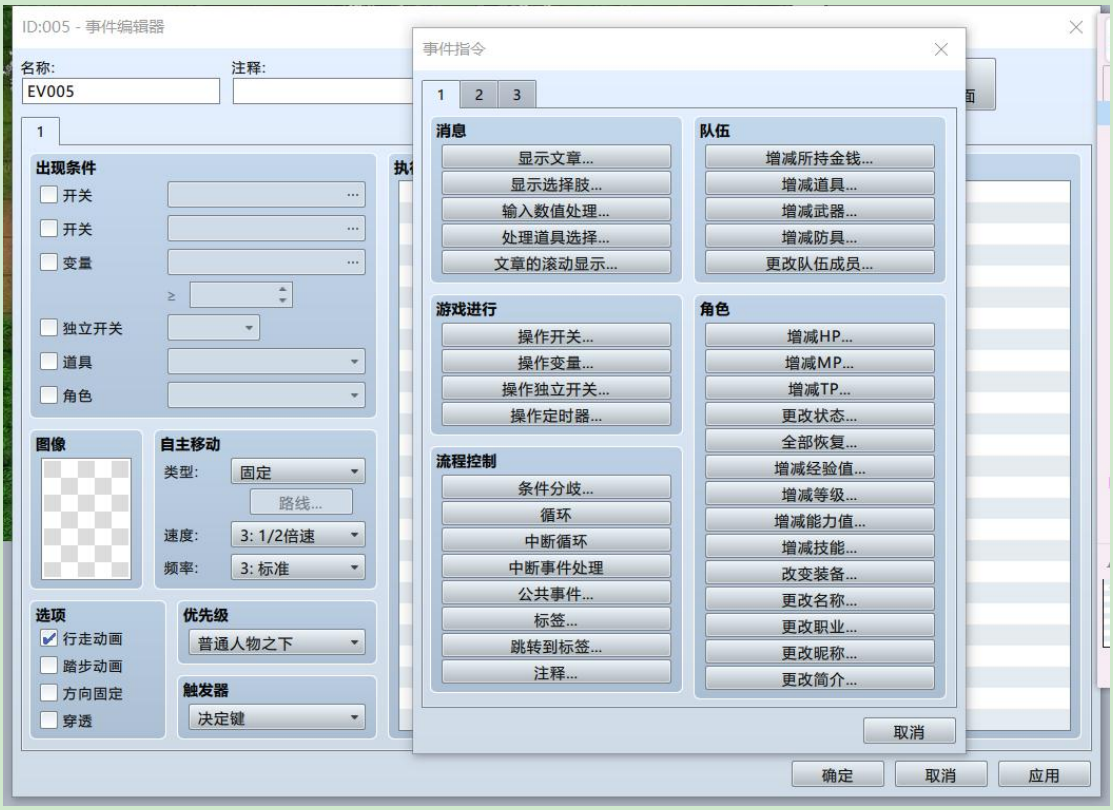


游戏就是由一个又一个的地图衔接构成，地图是游戏必不可少的元素，也是吸引玩家的的一个因素

人物的设置看自己喜好，都可以调整

（四）事件的安排

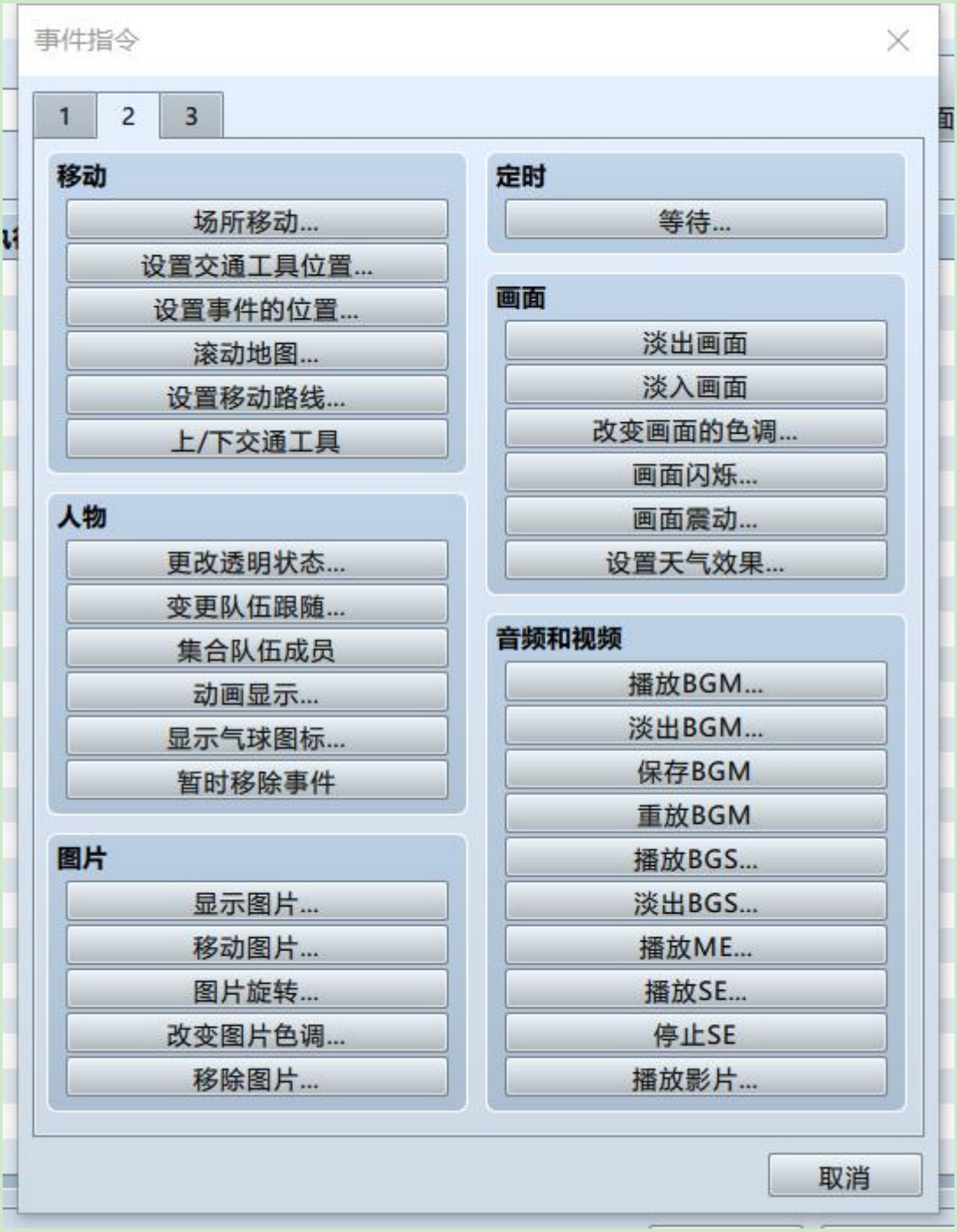
说起事件，同学们不妨会疑问，事件和游戏有什么关系，



我打个比方来说，假如要设置一段情节，主角从悬崖上面摔下来，中途遇到一个绝世强者洞府的狗血剧情（0.0 当然你别去悬崖上面尝试，毕竟你不是主角，没有光环在手，而且主角是有外挂的，而你啥都没有）

线路：首先设置一段对话，主角在和哪个不长眼的在打架，打不过他，被逼无奈跳崖——醒来发现自己没死，在一洞府前面（这时候就要控制主角的移动路线，强制移动）——然后主角的自言自语，走进洞府，得到远古传承，开始不一般的人生

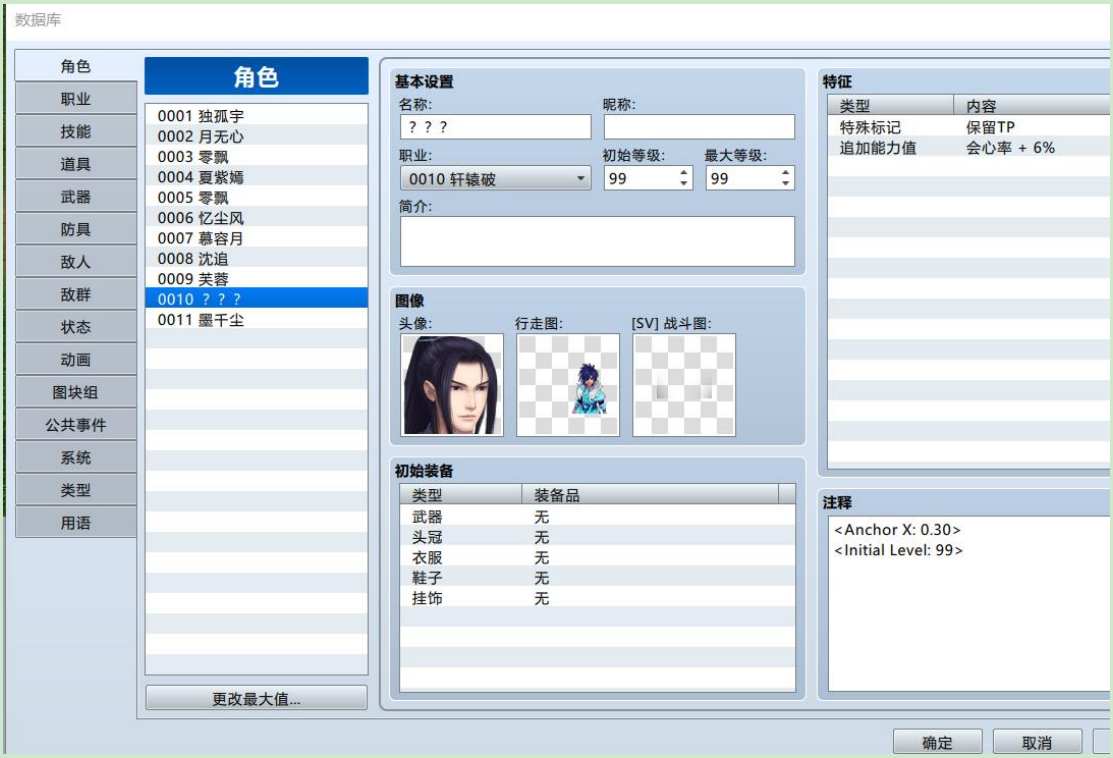

事件指令
1
2
3
场景控制
战斗处理...
商店处理...
名称输入处理...
开启选单画面
开启保存画面
游戏结束
返回标题画面
系统设定
更改战斗BGM...
更改胜利的ME...
更改败北的ME...
更改交通工具BGM...
变更禁止存档设置...
变更禁止菜单设置...
更改禁止遇敌设置...
变更禁止队伍设置...
更改窗口颜色...
更改角色图像...
更改交通工具图像...
地图
变更地图名称显示...
更改图块组...
更改战斗背景...
改变远景图...
获取指定位置信息...
战斗
增减敌人HP...
增减敌人MP...
增减敌人TP...
更改敌人的状态...
敌方完全恢复...
敌人出现...
敌人变身...
显示战斗动画...
强制战斗行动...
战斗中断
高级
脚本...
插件指令...
取消

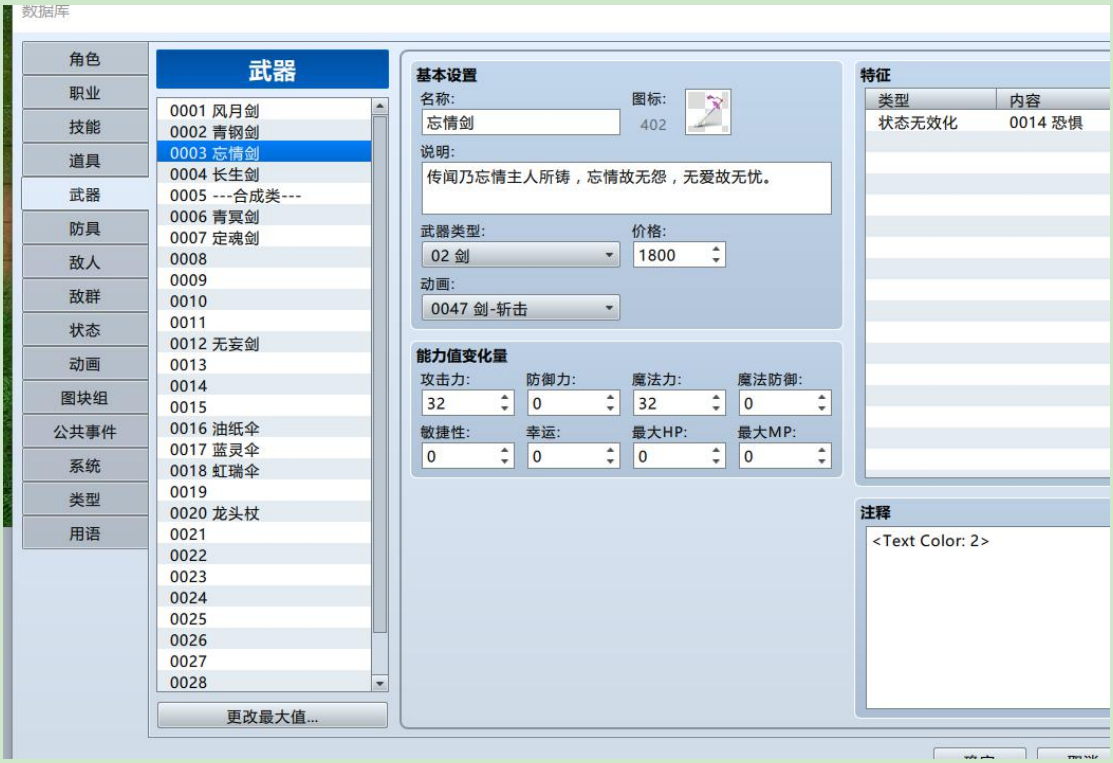


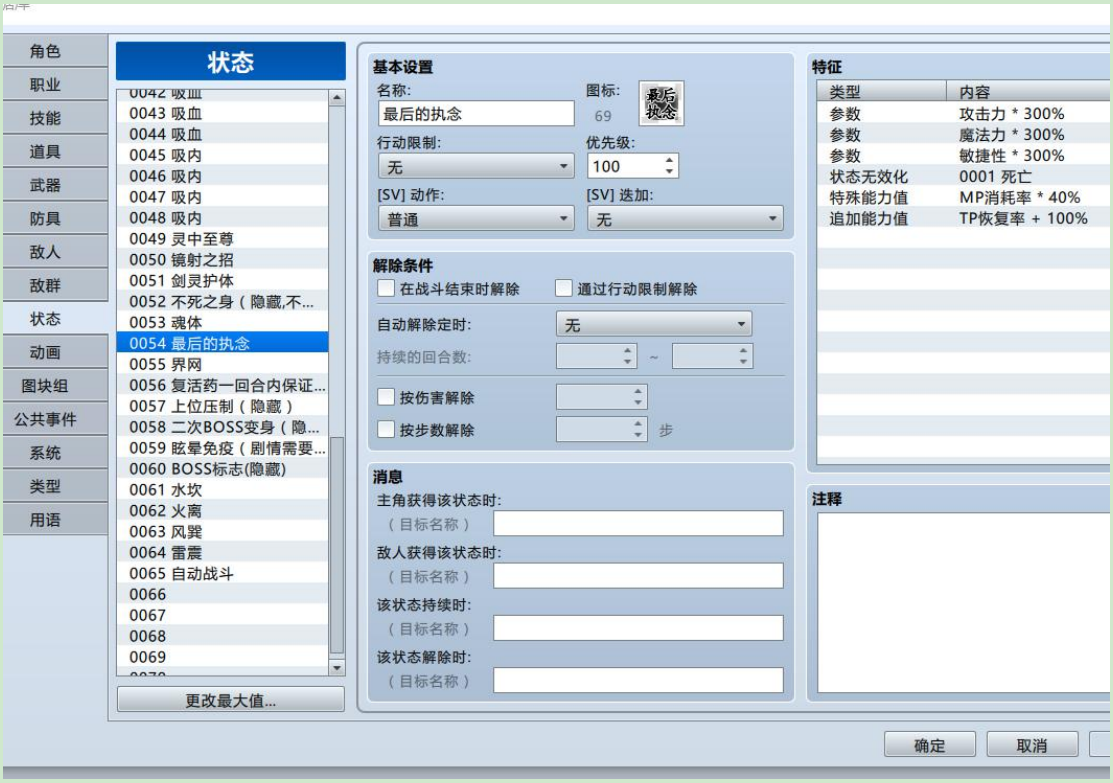
事件分为很多种，具体等同学们空余时间去体验一下，事件是游戏里面最关键的部分，人物的各种对话，打架等等

（五）数据库

数据库，顾名思义就是游戏各种数据的一个载体，设置角色，职业，装备，技能，敌人的一个地方


数据库
角色
职业
技能
道具
武器
防具
敌人
敌群
状态
动画
图块组
公共事件
系统
类型
用语
角色
0001 独孤宇
0002 月无心
0003 零飘
0004 夏紫嫣
0005 零飘
0006 忆尘风
0007 慕容月
0008 沈追
0009 芙蓉
0010 ？？？
0011 墨千尘
更改最大值...
基本设置
名称:
？？？
昵称:
职业:
0010 轩辕破
初始等级:
99
最大等级:
99
简介:
图像
头像:
行走图:
[SV] 战斗图:
初始装备
类型
装备品
武器 无
头冠 无
衣服 无
鞋子 无
挂饰 无
特征
类型
内容
特殊标记 保留TP
追加能力值 会心率 + 6%
注释
<Anchor X: 0.30>
<Initial Level: 99>
确定
取消


数据库
角色
职业
技能
道具
武器
防具
敌人
敌群
状态
动画
图块组
公共事件
系统
类型
用语
武器
0001 风月剑
0002 青钢剑
0003 忘情剑
0004 长生剑
0005 ---合成类---
0006 青冥剑
0007 定魂剑
0008
0009
0010
0011
0012 无妄剑
0013
0014
0015
0016 油纸伞
0017 蓝灵伞
0018 虹瑞伞
0019
0020 龙头杖
0021
0022
0023
0024
0025
0026
0027
0028
更改最大值...
基本设置
名称:
忘情剑
图标:
402
说明:
传闻乃忘情主人所铸，忘情故无怨，无爱故无忧。
武器类型:
02 剑
价格:
1800
动画:
0047 剑-斩击
能力值变化量
攻击力:
32
防御力:
0
魔法力:
32
魔法防御:
0
敏捷性:
0
幸运:
0
最大HP:
0
最大MP:
0
特征
类型
内容
状态无效化 0014 恐惧
注释
<Text Color: 2>



主角想拥有什么样的能力，什么样的外挂，都可以设置，装备的种类，属性也是可以调的

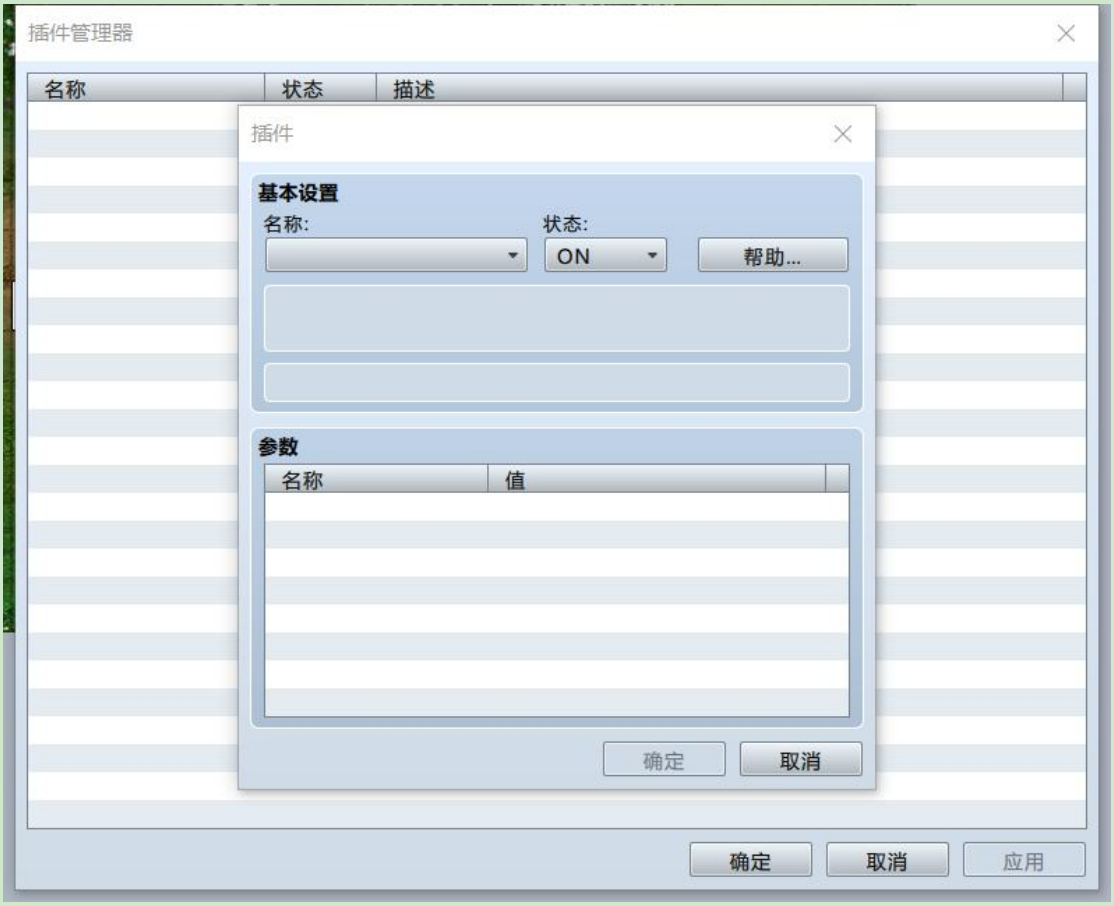
状态自己设置，技能动画也是一样，自己设置，各种炫酷，吊炸天的动画是可以弄得，只要有素材，看个人想法，反正 mv 充满了一切可能。

（六）插件(脚本)

插件也称脚本（脚本喊得舒服一点），做游戏的必须要跟上社会的脚步，搞出独具一格的特色，单单依靠 mv 自带的功能，是做不出好的游戏的，最多只能自己玩玩

比如如果要让装备能强化，附魔，镶嵌宝石，mv 自带功能是做不出的，菜单的美化，自定义背景，添加更多的菜单选项，都需要脚本的配合。（mv 的脚本系统我是觉得挺垃圾的，用得不舒服，没有 vxa 的脚本用得好，但是

vxa 画质没有 mv 好，各有各的追求，想用哪个都可以，并不妨碍做游戏）



初始的 mv 是没有插件的，也就是说，插件全部都要依靠自己寻找。互联网和 66rpg 论坛都有，自己可去寻找，插件是要 js 后缀的，然后将其插件移动至文件夹 js 里面，这个文件夹是专门用来放插件的，工程里面打开插件管理，然后就能使用插件了，插件的种类很多很多

各种各样的功能可以让游戏更加协调，元素更丰富，更能吸引玩家的兴趣。

简单基础的介绍就这么多，其实 rpg 也不是很难，很快就能入手精通的，看好你们。

四、针对零基础人员的学习内容讲解(RPGMV 游戏开发网课课程大纲)

这一部分为零基础的人，简单讲解一下，学习 RPGMV 游戏开发大概需要学习那些内容。以下内容按照顺序编排，但本文的目录内容不会按照一下内容的顺序编排。

本部分还会给出一些编程常识的整理，但是这些编程常识仅仅罗列出 mv 开发所需要的插件开发编程常识，并不完全适用于其他语言。

（一）针对于 RPGMV 内容相关，相近的学习内容(课程教学内容大纲)

学会使用 RPGMV 这款软件，并制作简单的游戏。

了解 RMMV 圈子的常见学习资源。

学会使用 VScode 软件。

学会配置 localhost 本地服务器。

掌握基本的编程常识。

~~了解面向对象语言的基本常识。~~

掌握类的写法、方法重写、对象实例化、类的继承。

了解 JavaScript 项目的类继承的最优化写法、了解

了解 MV 项目的代码层次结构。

学会使用 JSDoc 写注释。

掌握“插件”的定义。
了解 JavaScript 的 ES5 标准下的语法。
能够模仿经典插件写代码。

（二）基本编程语言常识
代码编辑器的识别与选取。
变量的定义与赋初值。
变量的类型。
加减乘除运算的实现。
变量装载数值的特点理解。（即变量数值交换的特点）
变量的输出。
for 循环、while 循环。
逻辑运算符与逻辑语句。
数组。
函数式编程，形参，实参，与返回值。
变量的作用域与生存期。

（三）面向对象语言的编程常识
类的定义。
对象的声明与实例化。
构造方法。
成员变量的声明，调用，与添加。
子类与父类的概念。
多态：类的继承。
多态：方法重写。

（四）针对于河池学院计算机协会，游戏开发项目，基础教学的开课课程及课程内容简表和上课说明

1. 课程说明

课程适用人群：本课程面向任何想制作简单电脑版 2D RPG 类型游戏的同学。
课程周期：即日起至 2021 年 7 月。
上课模式：。全网课模式，无任何线下教学。
上课主教：阮中楠。
课程目标：学会使用 RPGMV 这款软件，并用模仿常见的插件代码，修改 mv 项目，最后制作简单的游戏。
预备知识：自行观看***夜神基础课***和***编辑操作篇***这两个 B 站视频，看完本文的***雷文龙教程***。对编程技术无任何前置要求。
学习方式比例：大约为“30%的网课+20%的博客文章+30%的本文档阅读+20%的引导”

2.

3. 在教学中的和常见用语说明(各类专有名词说明)

“***SIAKO.Mobi***”，指的是一个在台湾的 RPGMV 游戏开发界的开发者。
“***鳗驼螺***”，指的是一个在“简书”创建“RPG Maker MV”专题并写了几篇插件博客教程的开发者[1]。
“***Project1***”，指的是 RPG Maker 系列的游戏引擎开发者论坛[2]。

[1] 简书.RPG Maker MV 专题 https://www.jianshu.com/c/78532c0aef87
[2] Project1 https://rpg.blue/

“***夜神基础课***”，指的是 B 站 UP 主“夜神说话”[3]的 RPGMV 使用基础教学系列视频[4]。

“***编辑操作篇***”，指的是 SIAKO.Mobi 针对 RPGMV 的基础使用教学系列视频[5]。

“***渐进篇***”，指的是 SIAKO.Mobi 在 B 站的系列课程：【SIAKO.Mobi】RPG Maker MV Plugin Scripting 脚本教学【渐进篇】[6]。

“***进阶篇***”，指的是【SIAKO.Mobi】 RPG Maker MV Plugin Scripting 脚本教学【进阶篇】[7]。

“***主文档***”，一般指的本文档，表现形式为本文档的“金山文档”共享电子版形式[8]。

“***RPGMV***”，“***RMMV***”，指的是 RPG Maker MV 这款 2D 游戏开发引擎，就是我们教学常用的软件。

“***mv***”，“***mv 项目***”指的是由 RPGMV 生成的基于 JavaScript ES5 版本的 桌面应用端/移动端 项目。

“***VScode***”，指的是 Visual Studio Code 这款文本编辑器。

4. 课程内容简表（实时更新）

(1) 了解 RPGMV 的代码开发情况

本节课要求同学们直接去看看别人的打代码情况，直接去了解怎么将代码植入 mv 项目。

本节要求：了解 RPGMV 插件开发的大概情况，了解在 RPGMV 项目开发下，打代码的基本方式。

参考链接：渐进篇：1-3 节。

(2) 安装合适的代码编辑器

为了让同学尽快熟悉 RPGMV 的代码编写，本节课需要同学学会安装专业化的代码编辑器，尽快熟悉编辑平台。

本节要求：学会在自己电脑上下载、安装、并汉化软件 VScode。

参考链接：

VScode 官网：https://code.visualstudio.com/

汉化软件 VScode 的教程：https://www.cnblogs.com/shapaozi/p/9651168.html

(3) 安装盗版 RPGMV 并应用第一节课程内容

本节课要求同学搭建好 RPGMV 这个平台，并开始模仿第一节课的内容来亲自写代码

本节要求：

① 下载并安装 RPGMV 盗版。（有条件可以购买并下载 RPGMV 正版）

② 模仿课程，写代码 console.log("fuck"); 并输出结果。（并将结果截图发给课程负责人以检查学习情况。）

③ 看渐进篇：4-6 节。

额外技能要求：学会电脑版 QQ 的录制视频功能，即功能快捷键：Ctrl+Alt+S。学会这种较高效率的沟通方式。

RPGMV 盗版下载方式：

① 在“河计协活动通知群[9]”内的群文件夹“RPG 游戏开发课(阮中楠)(雷文龙)”内下载。

② 见本文的“RPGMV 下载与其帮助链接”部分。

---

[3] 之前该 UP 主的名称叫做“夜神不说话”，改名可能会对各位读者在找人时遇到困难。

[4] 【RPGmakerMV 教程】从零开始制作自己的游戏 | 夜神的游戏制作课堂汇总 https://www.bilibili.com/video/BV1Ds411d7EM

[5] 编辑操作篇 https://www.bilibili.com/video/BV1cx411z71B

[6] 渐进篇 https://www.bilibili.com/video/av23706785/

[7] 进阶篇 https://www.bilibili.com/video/BV1zx411q76q

[8] 本文的金山文档电子版链接 https://www.kdocs.cn/l/skwnSanY3L9J

[9] 河计协活动通知群 QQ 号：931650213

(4) 学会使用 VScode 启动 mv 项目

本节课要求同学们，学会用专业的方式来启动 RPGMV 项目，学会脱离 RPGMV 来独立启动项目。

本节要求：严格按照本文及 B 站教程来实现该操作。

具体教学见本文的“其他技术实现”->“如何使用 VScode 来调试 RMMV 项目(VScode+Live Server+Debugger for Chrome+Launch.json)”部分[10]。

(5) 复制粘贴别人的插件 JavaScript 代码，并在 mv 项目中运行

本节课要求同学们实际运行一次别人写的代码，感受一下插件代码在 RPGMV 项目中的强大作用。

本节要求：学会“插件”与 RPGMV 的接口编写格式。

具体教学见“简书鳗驼螺”的第一篇教程[11]。

(6) 面向对象语言常识——类与对象

本节课要求同学们了解面向对象语言的一些常识，不要求掌握。至于其他的基础编程常识，要求自学。

本节要求：

了解类的定义

了解方法的定义

了解在 JavaScript，类的定义方式

了解对象的实例化

了解对象对“成员变量”的调用方式

了解 JavaScript 是“基于对象”语言而不是严格的“面向对象语言”

(7) mv 项目架构理解——SceneManager 类的大概工作原理及核心执行方法的简要讲解

本节课要求同学们开始了解 mv 项目中，一个相当重要的核心工作方法群，SceneManager 类及其附属方法。本节课会稍微介绍 SceneManager 类的主要功能和其附属方法的一部分功能。

本节课所需要阅读的全部代码，需要自己手动下载。该仓库内的大部分代码都已做好基本的注释。代码仓库链接地址，在本文档内搜索“**开源的 mv 分类代码**”词条即可。从仓库内下载代码的方式恕不讲解与教学。

本节要求：

① 学会下载从开源的 gitee 仓库内下载代码。

② 记住 SceneManager 类的主要功能。

③ 了解 SceneManager 类常用的几个核心方法。

“SceneManager 类的主要功能”与“SceneManager 类常用的几个核心方法”的具体内容，见“MV 源码原理理解与工作原理解释”->“SceneManager 类常用方法原理的简要解释”部分。

(8) 插件代码的模仿与运行——制作一个启动画面

本节课要求同学们模仿“鳗驼螺实例教程 2”的代码，做出一个自己的启动画面

本节要求

[10] 如何使用 VScode 来调试 RMMV 项目 https://www.bilibili.com/video/BV11A411n7Lq

[11] 简书.鳗驼螺.【RPG Maker MV 插件编程】【实例教程 1】怎样编写一个插件？ https://www.jianshu.com/p/0bd8b462dac1

https://www.jianshu.com/p/8b67041f02d5

(9) RPGMV 开发圈子

本节课要求了解一个国内主要的 RPGMV 游戏开发者的圈子——Project1[12]。

(10) 了解 MV 项目源码的主体架构方式

本节课需要了解 mv 项目的源码各个主要类的关系，需要大家去 GitHub 上面搜索 Drill_up 并下载他的 RPGMV 插件说明书[13]，阅读完他的全部说明书。从 GitHub 上面下载的文件的方式，在这里不做解释，需要各位自己学。

(11) 待添加的下一节课内容

五、各类学习视频，博客链接

【RPGmakerMV 教程】从零开始制作自己的游戏 | 夜神的游戏制作课堂汇总[14]。

素材网站推荐

【MV 插分享】https://ysrpg.lofter.com
【哔哩哔哩云书主页】https://space.bilibili.com/164546413
【云书临时博客】http://rpgyunshu.cn
【RPGCZ 站】https://www.rpgcz.cn
【Project1 论坛】https://rpg.blue
【爱给网】https://www.aigei.com
【绘世界】http://yms.main.jp
【效果音】https://soundeffect-lab.info
【森の奥】https://fayforest.sakura.ne.jp
【睡工坊】https://hime.be/rgss.html
【VE】https://victorenginescripts.wordpress.com
【DKRPG】https://dekitarpg.com
【taroxds Blog】https://taroxd.github.io/rgss
【74RX】https://mdc-light.jpn.org/TYPE74RX-T/material.html
【VCS 素材馆】http://www.vita-chi.net/sozai1.htm

图标有很多种获取途径，你谷歌搜 “ ツクール MV アイコン ” 就有了
比如这里，总结了一些现有的可商用图标：https://storyinvention.com/tsukuru-icon-matome/
至于如何拼接，PS 不太准确的话就用这个方法：https://fungamemake.com/archives/3387（直接浏览器自带谷歌翻译就行）
推荐你用这个老外姐姐的：
https://forums.rpgmakerweb.com/index.php?threads/avys-icon-workshop-update-jan-2021.21623/
很全，自己做着玩完全就够用了，如果要求高一点就只能自己画了，

六、RPGMV 下载与其帮助链接

（一）下载链接

[12] Project1.RPGMV 讨论区 https://rpg.blue/forum.php?mod=forumdisplay&fid=605
[13] GitHub.DrillUp/drill_plugins
https://github.com/DrillUp/drill_plugins/tree/master/%E6%8F%92%E4%BB%B6%E8%AF%A6%E7%BB%86%E6%89%8B%E5%86%8C
[14] https://www.bilibili.com/video/av3330603?from=search&seid=13280652077082629986

1. 盗版 1.01 版

在前面的教程中，一律使用“**RPG Maker MV v1.0.1**[15] [16]”。本软件免安装，中文版，直接解压文件即可打开。现在打开文件夹，找到名为“RPGMV”的软件，打开即可。

2. 盗版 1.61 版

感谢 UP 主“硕明云书”[17]的分享，感兴趣的读者可以去观看其视频。

链接：https://pan.baidu.com/s/1uugSlp5hK2UM35I5Voa9eA

password：alft

3. 正版

现价格为 204 元。一般会在西方节日内打折，如感恩节、圣诞节。有时候春节也会打折。

https://store.steampowered.com/app/363890/RPG_Maker_MV/

（二）RPGMV 官方简中教学文档（与其他的杂项记录）

http://miaowm5.github.io/RMMV-F1/

（目前已失效）

如果你要找素材：素材汇总，但不提供微皮嗯…

https://docs.qq.com/doc/DY09kd0ttRmFEQlZW

如果你要学习 mv 的基础使用：夜神的 mv 大讲堂

https://rpg.blue/thread-386462-1-1.html

如果你想阅读 mv 的官方说明文档：rpg maker mv 帮助文档汉化

https://miaowm5.github.io/RMMV-F1/

如果你要学习 yep 系列插件的使用：简书-yep 系列插件中文说明文档

https://www.jianshu.com/p/96a3f87a06f5

如果你想用一点代码执行命令：rmmv 脚本指令大全

http://suo.im/4NrqKE

（已使用工具缩短网址）

如果你想深入研究 mv 的核心 js：汪汪大佬的 mv 核心脚本注释

http://suo.im/5bd1SK

（已使用工具缩短网址）

一些有用的网址（四）

【套娃式强固加密工程】

https://rpg.blue/forum.php?mod=viewthread&tid=479864&page=1&extra=#pid2878275

【PhotoshopCC 便携版附视频教程】

---

[15] 下载链接 http://www.xue51.com/soft/6586.html (附带安装简介)

[16] 下载链接 https://pan.baidu.com/s/1ZSqywn2f4lUxhv5BT__jAg (百度云下载链接)

[17] 硕明云书 B 站个人空间 https://space.bilibili.com/164546413

pan.baidu.com/s/1gFd8QQutUYrDkLoTUc6xVA 提取码: ve6e
【在线生成脸图（使用前请阅读用户条款）】
https://charat.me
敌人等级计算器：
http://yanfly.moe/tools/enemylevelcalculator/
======

看内容：https://www.jianshu.com/nb/13204998 学会里面的 RPGMV 教程 学会写插件。
RMMV 插件编程实例教程
https://rpg.blue/forum.php?mod=viewthread&tid=395487&page=1&authorid=2647944

七、RPG 事件的运用
（一）地图的切换
地图和地图的链接是用“事件”来完成的。

（二）地图怪物的生成与重新自动生成

八、RPG 内部数据库的修改与设置
（一）人物头像图片的更换
在 MV 中，绝大部分图片的格式为 **png** 格式。在数据库中，我们在“角色”选项卡里面更换图片。先制作图片，针对人物头像的大小有固定的参数：**像素 144*144（有时，145*145 也是可以加载的）。**将图片导入到文件夹：E:\RPG project\Project1\img\faces 即可。注意图片的像素大小和图片本身的格式。

（二）地图图块组的制作 与 规格说明
单个图块组像素大小“**48*48**”，

九、其他技术的实现
（一）本节阅读前言
本节主要讲解一些在游戏开发过程中用到的另类技术，大部分会跳出 RPGMV 这个软件框架。这些技术有些和 mv 开发密切相关；有些则关系不大，可有可无；有些则会极大地提高开发效率等。

（二）游戏字体的替换
暂时不写。

（三）地图背景音乐的添加与替换
找到你的项目文件夹的 bgm 文件夹(这里以我的文件夹为例：E:\RPG project\Project1\audio\bgm)，将你要添加的音乐文件放到 bgm 文件夹即可。注意更改文件的格式，一份音频文件需要两份不同格式的音频文件才能被 MV 识别。分别是.ogg 格式和.m4a 格式。音频格式的转换用格式工厂完成。由于使用格式工厂转换文件比较简单。在这里不提供格式工厂转换文件格式的具体教程[18]。右键点击地图。点击“编辑”选项卡，修改背景音乐。

（四）开发者无视地图障碍
在游戏运行的时候，按住 Ctrl 键实现地图障碍物的无视。

（五）成品游戏的安装包生成与打包

[18] 格式工厂的下载链接：http://www.pcfreetime.com/formatfactory/CN/index.html

先**部署**文件，部署完成后，再对部署文件进行“**打包**”。这里说的打包并不是简单地压缩文件，而是用专业的软件对其进行整合压缩，最后生成专业的文件[19]。做好的文件打包后，解压以后，才可以生成 eve 格式的游戏打开端口。

（六）问题解释“为什么不能直接通过点击 index.html 文件的方式来启动 mv 项目？”

因为浏览器打开 index.html 文件时，使用的是 file 协议，而不是 ajax 所支持的协议，而且 mv 项目本身一定会导入.json 文件，这就共同触发了“跨域浏览”问题。当前浏览器的 file 协议无法访问本地.json 数据。

解决这种问题的方式，这里笔者介绍两个：

① 使用 ajax 所支持的协议来打开 html 文件。

② 设置浏览器，使其能够支持 file 协议的浏览。

关于“使用 ajax 所支持的协议来打开 html 文件”，指的是，可以使用 http 的协议来访问该文件。具体做法是：搭建一个本地的服务器，进而访问文件。原理是如此的，具体实现操作可以参考本文的“如何使用 VScode 来调试 RMMV 项目(VScode+Live Server+Debugger for Chrome+launch.json)”。

关于“设置浏览器，使其能够支持 file 协议的浏览”，可以参考“让 Chrome 浏览器支持本地访问数据”。

（七）让 Chrome 浏览器支持本地访问数据

经典的 ajax 所支持的协议不包括“file”协议，也就是本地访问文件。为了让 Chrome 可以直接调用本地的数据，可以按照以下的设置实现。

找到装在电脑内的 Google Chrome，并在其“属性”状态栏内的“快捷方式->目标”栏中添加代码：“ --allow-file-access-from-files”[20]。注意在填写时要注意前面要加上空格。

如果控制台报出以下的错误，那么就可以判断是跨域浏览的问题了。

Access to XMLHttpRequest at ‘文件地址’ from origin 'null' has been blocked by CORS policy: Cross origin requests are only supported for protocol schemes: http, data, chrome, chrome-extension, https.

（八）如何使用 VScode 来调试 RMMV 项目(VScode+Live Server+Debugger for Chrome+launch.json)

本小节内容涉及到了插件开发的内容，但是并不涉及具体代码编写，只涉及到简单的软件设置。

在实际开发的时候，会遇到这样的一个开发情况：我需要先对插件设置断点，然后再启动游戏触碰断点，随后再跟着进程一步一步的摸清楚代码的运行情况。虽然 RPGMV 本身会提供一个调试工具，按 F8 可以开启调试工具，但是这个工具是先启动游戏，再启动调试，然后我们才可以给目标代码设置断点。这个顺序不对，这样的顺序也不符合我们常见的软件开发，那我们就必须找到一个新的方法来启动 MV 项目代码。因此本小节问题还可以这样称呼：**如何在不使用 RPGMV 这款软件来独立启动其附属的 MV 项目代码？**

下面介绍基本流程。下面的安装流程和相关操作可以打乱一下顺序。

① 下载代码编辑器 VScode，从官网上下载[21]，全称为“Visual Studio Code”。

② 下载浏览器 Chrome，从官网先下载[22]，全称为“Google Chrome”。

③ 在 VScode 中下载软件插件“Live Server”。

④ 在 VScode 中下载软件插件“Debugger for Chrome”。

⑤ 在 MV 项目下找到由 VScode 自动创建的文件夹“.vscode”内找到“launch.json”并设置它。

⑥ 复制粘贴 Debugger for Chrome 提供的代码并作出小改动，更改本地服务器端口号和调试器名称。

⑦ 找到装在电脑内的 Google Chrome，并在其“属性”状态栏内的“快捷方式->目标”栏中添加代码：–remote-debugging-port=9222 。注意在填写时要注意前面要加上空格。

具体教学可以参考以下两个 B 站教程：

原版：https://www.bilibili.com/video/BV1jJ411c7zK

[19] http://www.k73.com/glzq/217985.html?tdsourcetag=s_pcqq_aiomsg

[20] https://blog.csdn.net/weixin_41697143/article/details/80784701

[21] VScode 官网 https://code.visualstudio.com/

[22] Google Chrome 下载官网 https://www.google.cn/chrome/

阮中楠改编版：https://www.bilibili.com/video/BV11A411n7Lq

下面介绍启动流程：

① 在 VScode 下启动 Live Server 并开启本地服务器。

② 在 VScode 中找到 debugger 栏，并选择 debug 的工具。

③ 按 F5 运行/调试。

接下来介绍基本原理，并提出几个容易迷糊的问题。

1. 为什么需要两个 VScode 插件？单独用一个 Debugger for Chrome 不行么？

我们的目的是为了在自己的电脑上面运行 MV 项目，说稍微专业一点就是：在自己的电脑上面的本地服务器上启动能够访问本地文件的前端代码。而 Debugger for Chrome 插件本身不提供“建立本地服务器”的功能，所以我们才要额外下载一个“Live Server”插件来建立并启动本地服务器。

2. 为什么要先启动 Live Server 再使用 Debugger for Chrome？

只有先建立本地服务器才可以使用调试。建立本地服务器的最本质要求是，因为 MV 项目的本质是一个前端网页游戏，还是一个会调用本地的.json 数据库文件的前端网页游戏，所以才需要建立本地服务器。只有本地服务器才可以调用本地文件。

3. 为什么要额外设置 VScode 自动生成的 launch.json？

launch.json 的本质是 VScode 提供给项目的“调试工具设置清单”。只有先设置好这个清单内容，告诉 VScode 我需要怎么样的调试工具，我们才可以用 VScode 启动/调试 MV 代码。

（九）如何编写控制 Live Server 插件的 settings.json 文件并改变其端口号

在启动我们本地的 web 项目时，有时需要对 Live Server 插件提供的端口号进行调整，比如说将预设值 5500 调整成自己想要的其他数值。这时我们需要去 settings.json 文件内进行修改。

假设你已经学会了用 Live Server 启动 web，并假设我们要设置的端口号为 4399。首先，我们在本项目的.vscode 文件夹内新建一个与 launch.json 文件并列的 settings.json。

在 settings.json 中填写以下字段：

```
{
  "liveServer.settings.port": 4399,
  "liveServer.settings.root": "/"
}
```

我们不仅要修改 settings.json 的值，还要同步修改 launch.json 内"url"属性的值为"http://localhost:4399"，这样我们才算是修改好了 Live Server 的端口号。

本段内容在 CSDN 上已有原创博客文章[23]。

（十）如何用 VScode+Gitee 的工作环境来实现 MV 项目代码的版本控制

具体的操作可以看次视频教程[24]。下面给出网上的操作教程：

[23] CSDN.阮中楠 https://blog.csdn.net/qq_39438464/article/details/113783740

[24] B 站.《简单介绍 VScode+Gitee 的工作环境配置,演示此工作流程》 https://www.bilibili.com/video/BV1cC4y1b7u4

① 主教程[25]。
② 如何配置本地 SSH 公钥[26]。
③ 设置 VScode 的上传设置选项[27]。

（十一）如何使用基于 NodeJS 的 JSDoc 制作自己写的 mv 项目插件代码的 API 网页版说明文档

在插件开发中，我们会写出很多的代码。为了使其他人更好地阅读我们的代码，除了在代码中写上注释，还有生成 API 接口文档的方式来帮助他人阅读代码。对于 JavaScript 这种弱类型的语言来说，就更需要写注释了。对于 mv 项目来说，大部分的插件代码都没有写注释，那我们要本着负责的态度，对自己的代码负责，就更需要写清楚自己代码的注释。

在生成 API 接口文档之前，我们要先了解 JSDoc 是什么？JSDoc 不仅是一个 API 接口生成工具，更是一种注释语法，它类似于 JavaDoc，更确切地讲，JSDoc 就是从 JavaDoc 引申而来的。在使用 JSDoc 工具之前，我们要先学会其注释语法。事实上，VScode 本身就自动对 JavaScript 支持 JSDoc，学习成本会大幅下降。这种注释语法，笔者要求各位自己自学[28]。

在 CSDN 中，绝大多数的 JSDoc 使用教程都在使用极其熟悉的 npm 包导入方式，要想使用 npm，就必须先安装 NodeJS 环境[29]。安装完 NodeJS 后，JSDoc 的安装和使用就极其容易了[30]。

下面介绍基本流程：

① 去 NodeJS 官网[31]下载 windows installer 版本的安装包。
② 在你的 nodejs 目录下创建两个目录，分别是 node_cache 和 node_global。在 cmd 执行下面这两个命令：npm config set prefix “你的安装位置\nodejs\node_global” 和 npm config set cache “你的安装位置\nodejs\node_cache”。
③ 在环境变量的配置界面配置 NODE_PATH，值填：你的安装位置\nodejs\node_global\node_modules
④ 修改 Path 中含有 npm 值的式子，修改为：你的安装位置\nodejs\node_global\
⑤ 在 cmd 输入 npm install -g jsdoc
⑥ 找到自己写插件代码所储存的上一层级文件夹目录，在此目录中打开 cmd
⑦ 在此 cmd 中输入：jsdoc -r 文件夹名称
⑧ 在新生成的 out 文件夹内点击 index.html 文件即可检查自己所写插件代码的 API 接口文档。

十、插件的设置

十一、基于 JavaScript 的 RPGmv 项目插件编写

如果你已经熟练地掌握了 RPGmv 这款软件的运用，你多少会感觉到：这款软件的局限性太大了，很多东西实现不了。为了突破这些局限性，不少游戏开发者深入项目源码，用编写插件的方式来实现更多样化的功能。

（一）本章前言与阅读建议

在这里，将会涉及到很多复杂的内容，会涉及到 RPGmv 的插件开发和 PixiJS 的源码分析。建议读者具有以下的编程基础：**懂得基本的 C 语言编程技术，懂得函数式编程，能够熟练地掌握 for 循环的用法；了解面向对象编程的特性，能够用 Java 写出简单的类，懂得继承、多态、重写、重构等特性。**

我们使用的计算机编程语言是 JavaScript，语言版本稳定在 ES5。MV 项目的 JS 版本是 ES5。

---

[25] 简书.《vscode+码云（gitee），用 git 进行源代码管理--初级入门，超级小白也会用》 https://www.jianshu.com/p/8cd8491a611d
[26] CSDN.《在 VSCode 中使用码云(Gitee)进行代码管理》 https://blog.csdn.net/watfe/article/details/79761741
[27] CSDN.《vscode 的 git 冲突后报错怎么解决?Git:You have not concluded your merge（MERGE HEAD exist)、未能推送 refs 到远端》 https://xunmi.blog.csdn.net/article/details/104570265
[28] JSDoc 在线手册 http://www.dba.cn/book/jsdoc/
[29] CSDN.Nodejs+npm 详细安装 https://blog.csdn.net/qq_39308408/article/details/97754889
[30] CSDN.用 JSDoc 生成 js 文档 https://blog.csdn.net/qq_44810574/article/details/89194784
[31] node https://nodejs.org/en/download/

建议读者学会学会使用“VScode”这款软件，学会用 VScode 来安装插件[32]，独立地汉化此软件，并学会使用 Beautify 插件。

这里使用的是来自 steam 的正版 RPGmv 软件，不再是盗版的 RPG Maker MV v1.0.1。

（二）读者的身份转换声明

此时的读者已经开始涉足于插件开发，并且已经学会使用很多常用的插件了，如 MOG、Yep、Drill 等。但是还不会熟练地写插件。

（三）mv 项目的性质与定位

1. 属于游戏前端而不是单纯的网页应用前端

“mv”，“mv 项目”指的是由 RPGMV 生成的基于 JavaScript ES5 版本的 桌面应用端/移动端 项目。这个定位相当于“游戏前端”而不是常见的“网页前端”。这两个前端有交集，就是 html 的基础。游戏前端主要用的是 canvas 标签，而网页前端主要使用的是 HTML5+CSS3+JavaScript6 的前端三剑客，和相关的 Ajax、Vue 等框架。这些框架和 mv 项目基本上无关。

（笔者当时花了好多时间才搞清楚 mv 项目的具体定位和学习方向，走了很多弯路。）

2. 属于 pixi.js+canvas 的技术栈，而不是单纯的 html+canvas 的技术栈

mv 项目是用 pixi.js 的 API 来实现 canvas 绘图的，而不是用 html5 提供的 canvas 标签 API 来绘图的。因此，大部分的 canvas 教程，都不能很好地契合我们的 mv 项目，因为大部分的 canvas 教程一定会涉及到 html 标签的其他写法，而 mv 项目几乎不是用其他多余的 html 标签。这事实上造成了技术栈的割裂。

3. 以 canvas 为主体的 mv 项目在技术栈上的窘境

使用纯 canvas 作为游戏的技术栈，就意味着，无法使用 html+css 的方式来做动画效果，也无法使用 DOM 元素的 API，自然也无法使用基于 DOM 对象的大部分框架，甚至连最繁琐的，原生的 JavaScript 动态效果都是用不了。

这意味着，我们要在 canvas 标签内部实现完几乎全部已经成熟的功能，诸如鼠标点击事件，意味着我们要重复造轮子。

（四）开源的 mv 分类代码

作者稍微整理的 mv 代码分类与翻译[33]，目前保存在以下的开源仓库内[34]。

https://gitee.com/HechiCollegeComputerAssociation/rpgmv_project

作者用 JSDoc 的方式注释了一部分的方法，为了方便阅读，可以选择下载 out 文件夹并运行 index.html 文件来阅读 API 文档。

https://gitee.com/HechiCollegeComputerAssociation/rpgmv_project/blob/master/out/index.html

（五）对 MV 代码的基本常识讲解

1. 项目调试方式

按 F8 进行 debug。

2. update 方法原理

update 一帧运行一次、refresh 一般是需要的时候、调用一下刷新。

[32] 这里所说的“插件”指的是 VScode 这个文本/代码编辑器的插件，而不是 RPGmv 游戏项目的插件。

[33] 该源码翻译，大部分的整理与翻译并不是由仓库主人完成的，而是从其他论坛下载到的。作者仅仅是在此基础上做了一下改动，并增加了正版 mv 所具有的类。

[34] 若该仓库的链接失效，请联系 B 站 UP 主阮中楠并发私信进行反馈。UP 主会及时更新最新版本仓库链接。

3. 各类的定义方式——混合的构造函数/原型方式

mv 项目的绝大多数类采用“混合的构造函数/原型方式”来定义[35]。

4. 各类的继承方式——寄生组合式继承

mv 项目中的绝大多数类采用“寄生组合式继承”的方式来完成类的继承[36]。

定义：所谓寄生组合式继承，即通过借用构造函数来继承属性，通过原型链的混成形式来继承方法。其背后的基本思路是：不必为了指定子类型的原型而调用超类型的构造函数，我们所需要的无非就是超类型原型的一个副本而已。本质上，就是使用寄生式继承来继承超类型的原型，然后再将结果指定给子类型的原型。（引自《JavaScript 高级程序设计》[37]）

其中，我们常见的 Object.create()方法是 ES5 中原型式继承的规范化[38]。

结合上面两个部分的内容，作者有了以下的理解：

方法的定义与继承均在 prototype 原型链完成，属性这是在 function 类本身中定义，且继承的时候也是用构造函数来完成属性的继承。

5. Object.create()实现继承的例子

由于作者水平有限，尚不能理解透彻该部分的继承。这里粘贴了和 mv 项目几乎相同的继承例子，作为本小节内容[39]：

```
// Shape - 父类(superclass)
function Shape() {
  this.x = 0;
  this.y = 0;}
// 父类的方法
Shape.prototype.move = function(x, y) {
  this.x += x;
  this.y += y;
  console.info('Shape moved.');};
// Rectangle - 子类(subclass)
function Rectangle() {
  Shape.call(this); // call super constructor.}
// 子类续承父类
Rectangle.prototype = Object.create(Shape.prototype);
Rectangle.prototype.constructor = Rectangle;
var rect = new Rectangle();
console.log('Is rect an instance of Rectangle?',
  rect instanceof Rectangle); // true
console.log('Is rect an instance of Shape?',
  rect instanceof Shape); // true
rect.move(1, 1); // Outputs, 'Shape moved.'
```

[35] CSDN.javascript 定义类或对象之混合的构造函数与原型方式 https://blog.csdn.net/iteye_9339/article/details/81473212
[36] CSDN.JavaScript 实现继承的几种方法 https://blog.csdn.net/xgy123xx/article/details/106019671
[37] https://blog.csdn.net/qq_35718410/article/details/91412908
[38] https://blog.csdn.net/weixin_36465540/article/details/90176318
[39] MDN.Object.create() https://developer.mozilla.org/zh-CN/docs/orphaned/Web/JavaScript/Reference/Global_Objects/Object/create

6. 待整理的部分

2021.8.5 得到了许多新的结论，这些结论的整理，可能会极大地更改整个文档的内容组织方式。

我需要更正一下，在 20 年上半年的结论，有些东西讲的不对。

通过阅读一下文章，我对 JavaScript 的继承有了一下认识：

MDN:
https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Global_Objects/Object/create# 用 _object.create 实现类式继承
https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Global_Objects/Function/call# 使用 _call_ 方法调用父构造函数
https://developer.mozilla.org/zh-CN/docs/Learn/JavaScript/Objects/Inheritance
https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Inheritance_and_the_prototype_chain

CSDN:
https://blog.csdn.net/weixin_36465540/article/details/90176318
https://blog.csdn.net/qq_35718410/article/details/91412908
https://blog.csdn.net/lixiaosenlin/article/details/108140634
https://blog.csdn.net/xgy123xx/article/details/106019671

上述的这 8 篇文章，可以说我是从 2020 年 2 月份，一直反反复复看到现在，结合 rpgmv 的代码，现在我给出以下的结论：

1：在 JavaScript ES5 版本中，继承写法的最优解是“寄生组合式继承”，使用 call(this)方法来实现子构造方法 调用 父构造方法，实现“属性”的继承；使用 Object.create 和 prototype.constructor 的方式实现原型链指向，实现“方法”的继承。
2：在 JavaScript ES6 版本中，使用语法糖可以同时实现“属性”和“方法”的继承。
3：mv 源码中的 call(this)写法目的是为了实现继承，mv 插件的 call(this)写法目的是为了实现方法功能拓展。
4：mv 插件编写原理是：使用 call(this)方法调用原来的方法，做出功能拓展，利用 PluginsManager 的 document.appendChild 方法实现代码的导入，并利用 V8 引擎实现的词法作用域与变量原理实现 JavaScript 的方法覆盖，最后实现 mv 源码调用 mv 插件的方法。
5：mv 插件写法的本质是直接修改 mv 源码。
6：根据 4，可得：mv 插件冲突 是 mv 插件编写原理 的具体体现。
7：更正：mv 插件冲突的原因是 4，而不是阮中楠在 2020 年 2 月指出的“基于原型链继承”的特点而导致冲突。

目前我对 mv 继承的理解是这样的，如果理解不对，还请大家指正：

mv 采用了“寄生组合式继承”，即—— 通过借用构造函数来继承属性，通过原型链的混成形式来继承方法。

以 Scene_Base 为例：

```
function Scene_Base() {
    //调用 初始化
    this.initialize.apply(this, arguments);
}
//设置原形
Scene_Base.prototype = Object.create(Stage.prototype);
//设置创造者
Scene_Base.prototype.constructor = Scene_Base;
//初始化
Scene_Base.prototype.initialize = function() {
    //舞台 初始化 呼叫(this)
    Stage.prototype.initialize.call(this);
    //活动标志 关闭
    this._active = false;
    //设置 淡入记号
    this._fadeSign = 0;
    //设置 淡入的时间
    this._fadeDuration = 0;
    //设置 淡入精灵
    this._fadeSprite = null;
};
```

Scene_Base 类通过 Stage.prototype.initialize.call(this);的方式，来继承来自父类 Stage 的属性。

通过 Scene_Base.prototype = Object.create(Stage.prototype); 和 Scene_Base.prototype.constructor = Scene_Base; 的原型链方式继承来自父类 Stage 的方法。

针对 call(this)写法，应该分成两种情况来考虑：

1：用于实现继承。“在一个子构造函数中，你可以通过调用父构造函数的 call 方法来实现继承”

2：用于实现功能拓展，类似于“修饰者模式”，以 Drill_BattleCamera.js 为例：

```
var _drill_BCa_sys_initialize = Game_System.prototype.initialize;
Game_System.prototype.initialize = function() {
    _drill_BCa_sys_initialize.call(this);
    this._drill_cam_enable = true ;
    this._drill_cam_speed = DrillUp.g_BCa_speed;
    this._drill_cam_ftime = DrillUp.g_BCa_ftime;
    this._drill_cam_limit_width = DrillUp.g_BCa_limit_width;
    this._drill_cam_limit_height = DrillUp.g_BCa_limit_height;

    this._drill_BCa_X = {}          // 缩放x
    this._drill_BCa_X.cur = 0;      //  cur = -0.1，则缩放为0.9
    this._drill_BCa_X.move = 0;     //
    this._drill_BCa_X.time = 0;     //
    this._drill_BCa_Y = {}          // 缩放y
    this._drill_BCa_Y.cur = 0;      //
    this._drill_BCa_Y.move = 0;     //
    this._drill_BCa_Y.time = 0;     //
    this._drill_BCa_R = {}          // 旋转
    this._drill_BCa_R.cur = 0;      //
    this._drill_BCa_R.move = 0;     //
    this._drill_BCa_R.time = 0;     //

    this._drill_BCa_flip = {};          //翻转控制
    this._drill_BCa_flip.lock = false;  //
};
```

使用 _drill_BCa_sys_initialize.call(this); 调用原来的方法，再执行拓展后的逻辑，进而实现功能拓展，客观上应用了“修饰者模式”

（六）对全局变量的讲解

DataManager 专门生成一些全局变量，而这些全局变量又会被定义为各种实体对象。我们可以去对象类里面找到我们要输出的对象属性值。

（七）常见的插件代码组织方式

1. 立刻执行函数写法

很多插件的的代码组织方式几乎是这样的：(function () { /* code */ })(); 整个插件代码通篇下来就是两个个括号，(……)();。这种代码组织方式可以被称为“立刻执行函数写法”。

javascript 中没用私有作用域的概念，如果在多人开发的项目上，你在全局或局部作用域中声明了一些变量，可能会被其他人不小心用同名的变量给覆盖掉，根据 javascript 函数作用域链的特性，可以使用这种技术可以模仿一个私有作用域，用匿名函数作为一个“容器”，“容器”内部可以访问外部的变量，而外部环境不能访问“容器”内部的变量，所以(function(){…})()内部定义的变量不会和外部的变量发生冲突，俗称“匿名包裹器”或“命名空间”。

（引用至：https://blog.csdn.net/iteye_19474/article/details/82580396?utm_medium=distribute.pc_relevant.none-task-blog-OPENSEARCH-3.control&depth_1-utm_source=distribute.pc_relevant.none-task-blog-OPENSEARCH-3.control）

更加严格的称呼是：立即执行函数表达式。

（八）自定义一个窗口

如何直接做一个窗口类[40]？调用重写了很多构造函数。这是无插件指令的写法。

Scene_Splash.prototype.create 方法、Scene_Splash.prototype.start 方法、Scene_Splash.prototype.stop 方法、Scene_Splash.prototype.terminate 方法、Scene_Splash.prototype.update 方法等（当然除了构建器部分外，其它方法都是可选的）。下面就在 Scene_Splash 类中重写这些方法。

有一个 App 叫做 ： MV 游戏厅

http://www.51zixue.net/JavaScript/79183.html

显式地调用 window 的 close 方法，实现关闭。

现在学习这个教程的修改方式：

https://m.gamer.com.tw/home/creationDetail.php?sn=3857290

一个实例：

函数别名[41]。

（九）插件教程为什么修改类名？

什么时候不修改类名？什么时候修改类名？当我们修改里面的系统原本就有的对象的内容时，我们就不修改类名。当我们完全地创造一个新的东西时，我们就修改类名。

当我们仅仅只是在先有功能上添加一个新的功能时，我们需要在插件内复制大部分的代码，重写原来的类。

当我们要创建一个原来在游戏内从来没有的一个对象时，我们就需要写一个新的类名。由于新的类在功能上与原有的类有重复的部分，所以可以复制原有的类，并修改其类名。

图像：想把图像放到正中间，可以考虑 this.centerSprite()方法[42]。

（十）Sprite 类的形参理解

[40] 简书.鳗驼螺.【RPG Maker MV 插件编程】【实例教程 2】制作一个启动画面 https://www.jianshu.com/p/8b67041f02d5

[41] 函数别名教程 https://www.bilibili.com/video/av13028923?p=3

[42] 鳗驼螺.4.标题画面 https://www.jianshu.com/p/527a82a2fd6a

《针对“【SIAKO.Mobi】RPG Maker MV Plugin Scripting 脚本教学【渐进篇】 13 节[43]”和“【RPG Maker MV 插件编程】【实例教程 4】玩转标题画面 41 above”的导入图片示例的具体写法对比研究——Sprite 类形参的写法》

我们按照 SIAKO.Mobi 的写法，写出了一下的代码，但是运行的时候总是出现“Cannot set property'bitmap' of undefined”的报错：

```
this.titleFrontImage = new Sprite();
this.titleFrontImage.bitmap = ImageManager.loadTitle2(titleFrontImage);
this.addChild(this.titleFrontImage);
```

以下这段代码是可以运行的 42 above。

```
var temp = ImageManager.loadTitle2(titleFrontImageName);
this.temp_image = new Sprite(temp);
this.addChild(this.temp_image);
```

我现在猜测，是在 new 一个 Sprite 类对象的时候，形参写错了，我参考了以下的源码得出的猜想：

```
//在 Scene_Title 类的源码中。系统针对 Sprite 类形参的写法是，另外在 new 一个 Bitmap 类对象。其中，Bitmap 类对象的参数为 Graphics.width, Graphics.height
this._gameTitleSprite = new Sprite(new Bitmap(Graphics.width, Graphics.height));

//在 Scene_Title 类的源码中。系统源码的在填写 Sprite 类的形参是直接填写一个图片。
this._backSprite1 = new Sprite(ImageManager.loadTitle1($dataSystem.title1Name));
```

现在更改了一下代码的写法，就通过了：

```
this.titleFrontImage = new Sprite( ImageManager.loadTitle2(titleFrontImageName) );
this.addChild(this.temp_image);
```

我觉得，在 new 一个 Sprite 类对象的时候，形参应该要填写好具体的图像。而不是像 SIAKO.Mobi 那样不填写参数格式。我重新写好了一段代码，发现不是 SIAKO.Mobi 写法的问题，而是我的写法问题。

先 new 一个 Bitmap 位图对选哪个，用 ImageManager.方法导图，然后再把 Bitmap 位图对象填写形参至 Sprite 对象内。或者针对于现存的 Sprite 对象，直接对 Sprite.bitmap 进行赋值。

（十一）在 Sprite 类及其子类添加图片

一般我们在 Sprite 类的 create 方法内写导入图片，我们要先用 this 的方式来添加一个新的变量作为导入的图片，然后用导入图片的方法导入图片即可。随后 Sprite 类就会出现这个图片。为了美观，可以在 create 方法内部顺便定义好图片的坐标值。

为什么要用 this 的方式添加变量而不是用 var 的方式呢？作者做了一个实验，发用 this 的方式定义的新变量才可以正常使用。当前无法解释其原理。

我们在导入图片的时候，还要使用到 addChild 方法，将图片正式添加到 Sprite 类中，这样才可以把局部变量变成全局变量。

（十二）TilingSprite 满版精灵类的使用——实现图像的移动

《针对 SIAKO.Mobi 14 节课——满版图像》

TilingSprite 类是 pixi 源码里的类，称为“满版精灵类”。为了使得导入的图像可以持续的移动，我们要先给图像确定中心，然后才能移动。

用 TilingSprite.prototype.move 方法来确定图像的位置和大小，其中，大小的参数我们通常会用 Graphics.width

[43] SIAKO.Mobi 13 节 https://www.bilibili.com/video/av23706785?p=13

和 Graphics.height 的方式。其本质是 Graphics 静态类调用图像的 width 和 height 属性。

我们在 TilingSprite 对象的所在场景的 update 方法[44]内，通过调整此对象的 origin 成员变量附带的 x，y 坐标值，来实现 TilingSprite 对象在场景内部的坐标移动。我们之所以使用 origin 成员变量是因为 origin 在 TilingSprite 中是专门使用与图像移动的一个坐标控制点。

在网课中，专门提到了 move 方法和 origin 成员变量的相互使用，这两个用法必须同时使用。暂时无法解释其必要性。

（十三）在已有的菜单栏中创建新的窗口——写一个自己的窗口类

《SIAKO.Mobi15 节课——在菜单栏窗口中创建一个新的窗口[45]》

主要的算法是，在已知的“窗口场景”的内部里面，再添加一个新的窗口。

在 Scene_Menu.prototype.createCommandWindow 方法内部，添加新的窗口。输出数据的方式是使用 Window_Base 窗口基类的 drawTextEx 方法来输出一行字体。可以输出字符串。数据的输出方式也是当做是字符串输出。如果我们要把数值当做字符串输出时，我们可以用 String()方法来进行类似于其他语言的“强制类型转换”。输出字体一般写在场景类的 refresh 方法内部。

网课里面用的是重写 Window_Gold 类，最重要的是重写一个窗口类的以下方法：定义、initialize、windowWidth、windowHeight、refresh[46]。我们写的窗口类是以 Window_Base 作为父类的，而不是选择 Window 类。现在无法解释。

（十四）创建一个新的场景类

《SIAKO.Mobi16 节课——新建场景，嵌入窗口》

主要算法：先新建一个场景类，然后再把窗口显示在场景内部。SIAKO.Mobi 指出，不可以用 addChild 方法代替 addWindow 方法。但是在我们自己的实验中，是可以实现互换的。

我们新建的一个场景类，是用 Scene_MenuBase 作为父类的，试了一下用 Scene_Base 作为父类，发现背后的场景是全黑的。并不是原来的半透明效果。

试了一下用 Stage 类作为父类，发现直接报错。说明 Stage 类并没有具体实现一些方法。

新建的场景类主要重写 Scene_Menu 菜单场景、继承于 Scene_MenuBase 菜单基类场景。因为我们需要使用到菜单场景自带实现的半透明背景图，所以一般会选择 Scene_MenuBase 作为基类。主要重写的方法有：定义、initialize、create、start。如果需要场景刷新，可以看情况选择重写 update 和 refresh 方法。

（十五）建立自己的游戏启动场景

《SIAKO.Mobi17 节课——建立自己的游戏启动场景》

首先要建立一个新的场景，主要重写 Scene_Gameover 场景类。重写的方法有：initialize、create、start、update、stop、terminate。

在 create 方法中、这个场景需要开始执行播放背景音乐，创建启动界面图。故需要在自建的 Scene_Splash 场景类添加 playSplashBGM 和 createSplashSecneImage 方法。

在 playSplashBGM 方法中，要先关闭背景音乐。故需要 AudioManager.stopBgs()和 AudioManager.stopMe()方法。随后开始声明音乐类型。实际上完成播放的核心方法仅有 AudioManager.playBgm 方法。AudioManager.stopAll()方法停止全部的音频工作。

createSplashSecneImage 方法给 Scene_Splash 类添加一个新的图片精灵对象。1：新建一个 Sprite 类精灵类图片对象。2：用 ImageManager.loadSystem 方法导入图片。3：用 addChild 将图片添加进入 Scene_Splash 类。

在 update 方法中，需要监听用户的输入流。因此需要重写 isTriggered 方法。使之可以监控键盘和屏幕的输入流。update 方法需要转移场景到 Scene_Title 标题场景类内部。

其余的大部分代码都不需要重写。只需要保留在类中即可。他们会自动执行。

---

[44] update 方法更新每一帧，而图像移动是更新每一帧的。故图像的更新写在场景类的 update 方法内。

[45] SIAKO.Mobi 15 节 https://www.bilibili.com/video/av23706785?p=15

[46] refresh 方法是在更新的时候才调用。和 update 不一样。

（十六）更改一个窗口的背景图片，并设置其位置、大小、透明度等参数

这里会有一个误区：“我们能修改菜单窗口背景图，其他的窗口都不能修改其背景。”这个误区一直“指导”我们来修改 Scene 类的背景图。我不记得这个误区是不是从 SIAKO.Mobi 的视频里面学来的，反正我印象深刻。凡是一个 Scene 都可以通过 addChild 方法来添加一张图片。注意，我们只是添加了一张图片。Scene 可以使用 addChild 方法，是毋庸置疑的。核心证据是最深层的 Stage 类实现了 PIXI 的 Container 容器类。所以 Container 类的子类均可以使用 addChild 方法[47]。用这种追根溯源的方式，我们甚至可以看到任何的窗口类都可以使用 addChild 的证据[48]以及精灵类可以使用 addChild 的证据[49]。

我们直接修改的对象是来自 Window 基类的_windowBackSprite 变量。_windowBackSprite 表示的就是一个窗口的背景，这个背景就是以一个 Sprite 精灵的方式来描述的。我们只需要给这个精灵对象赋值即可。

但是这里的赋值又会出现一个问题，我们不能 new 一个新的 Sprite 来直接代替好这个_windowBackSprite。而是要在_windowBackSprite.bitmap 里面，添加位图对象。**给精灵的位图添加对象，添加位图，而不是精灵。**目前我无法解释，为什么要这样写，反正程序就是达不到效果。具体的写法可以如下：

```
var backImage_Bitmap = ImageManager.loadSystem('2020_temp');
this._windowBackSprite.bitmap = backImage_Bitmap;
```

现在一般默认用 ImageManager. 图片管理者的方式来导入位图。这个方法是最稳妥的。在上面的教程中已经提及到过。

一般我们是先采取导入图片，再对 Sprite 对象进行调整。mv 的 Sprite 类继承并重写了来自 PIXI 的 Sprite 类。从源码易知，可以更改 tint、alpha、scale 等参数。scale 参数是修改一个 Sprite 对象大小的核心要素[50]。不能直接用 width 和 height 的方式来直接修改一个 Sprite 对象。这个没有效果。

我更愿意用 alpha 来代替使用 opacity。因为 opacity 的本质是修改一个 Sprite 的透明度，其实就是封装了 alpha 变量[51]。这个用法是直接调用 PIXI 的定义的。

待检验：用 setFrame 方法来实现导入图片的大小修改。（本方法由 QQ ： 2335937510 提供）

（十七）利用$gameActors 输出一个角色所具有的技能

严格来说，人物角色和技能是被专门地放在两份.json 数据库文件的。我们肯定要找到人物技能和人物之间的关系的。按照这种思路，我们肯定要用人物来输出人物附属的技能。

在 DataManager 类中，明确说明了$gameActors 是 Game_Actors 类的成员。

$gameActors = new Game_Actors();

---

[47] Scene_Base.prototype = Object.create(Stage.prototype);说明了任何场景类均实现了 Stage 类。
Stage.prototype = Object.create(PIXI.Container.prototype); 说明了 Stage 类实现了 PIXI 的容器类。

[48] Window.prototype = Object.create(PIXI.Container.prototype);说明了 Window 类实现了 PIXI 的容器类。

[49] @extends PIXI.Container 在 PIXI 源码内，这说明了 Sprite 类继承了容器类。

[50]
```
width: {
    get: function () {
        return Math.abs(this.scale.x) * this.texture.orig.width;
    },
    set: function (value) {
        var sign = utils.sign(this.scale.x) || 1;
        this.scale.x = sign * value / this.texture.orig.width;
        this._width = value;
    }
},
```
说明了 Sprite 类的宽高属性本质上就是 scale 和其材质 texture 的操作。

[51]
```
Object.defineProperty(Sprite.prototype, 'opacity', {
    get: function() {
        return this.alpha * 255;
    },
    set: function(value) {
        this.alpha = value.clamp(0, 255) / 255;
    },
    configurable: true
});
```
说明了 opacity 的本质就是封装了 Sprite 的 alpha 值。

接下来在 Game_Actors 类中，我们发现其本质就是包装了 Game_Actor 类，其中的 Game_Actors.prototype.actor 方法告诉我们了如何使用这个$gameActors 来访问具体存放角色数据的方式——在其形参内填写角色的编号。值得一提的是，我们要通过具体的调试才能准确判别角色编号。有可能会出现数组下标为 0 时，所指向的并不是第一个角色的情况。

```
Game_Actors.prototype.actor = function(actorId) {
    if ($dataActors[actorId]) {
        if (!this._data[actorId]) {
            this._data[actorId] = new Game_Actor(actorId);
        }
        return this._data[actorId]; }
    return null; };
```

在 Game_Actor 类中，我们可以使用其 Game_Actor.prototype.skills 方法来输出其技能。在这里，$gameActors 调用到了$dataSkills。易知，skills 方法输出的是这个角色的全部技能，这个技能是以一个数组的形式输出的。

```
Game_Actor.prototype.skills = function() {
    var list = [];
    this._skills.concat(this.addedSkills()).forEach(function(id) {
        if (!list.contains($dataSkills[id])) {
            list.push($dataSkills[id]); }
    });
    return list; };
```

综上所述，输出一个角色的核心代码写法为：$gameActors.actor(1).skills()[1].name 这里的 name 表示的是技能名而不是角色名。注意数组下标的选取。易知，这种写法类似于一个二维数组。

值得说明的是，直接调用.json 数据库的方式是$gameActors[index]，直接把“$gameActors”当做是一个数组名，访问数组内容，即数据库内部的内容。而不是一个对象来访问其成员方法。

十二、基于 JavaScript ES5 语言版本与“开闭原则”的 MV 代码编写

（一）本章前言与阅读建议

本节写的代码，更多的是写一个类，一个系统，一个超脱于插件的代码群。这里会给出一些写代码的设计技巧。

（二）读者的身份转换声明

此时的读者已经不再是那种依赖于写插件的人了，应该多多少少意识到插件编写方式在软件工程中的不规范性。该考虑插件导入原理和 index.html 文件中各个.js 文件的导入顺序原理了。

此时的读者不应该直纠结于插件的效果，还应该深究各种 mv 源码运行原理。

（三）用 css 实现动态变化（待细化）

GameCanvas

使用:hover 伪类来实现一个动态变化。使用 css 的 id 选择器来针对性的完成变化。

在 mv 项目中查到的，html 元素的 id 号：

GameCanvas

UpperCanvas

十三、MV 源码原理理解与工作原理解释

（一）本章前言与阅读建议

本节不写代码，只讲一些深层次的理论和理解。

（二）读者的身份转换声明

读者再在此应该积极地思考各种 mv 源码运行原理。

（三）通论

drill：简单来说 rmmv 就是一个封装好的盒子，在 rmmv 里面可以向调用外面的 dom 结构，但是在 dom 层，是没法接触到 rmmv 的内部结构的。

（四）PluginManager.loadScript 方法的原理解释

PluginManager.loadScript 方法是插件管理者的本质。这解释了，我们的插件代码是怎么加入到 index.html 内部的。

核心原理：

将.js 文件变成.html 文件内部的一个标签内容，相当于添加类似于如下形式的标签：<script type="text/javascript" src="js/rpg_windows.js"></script> 。

主要是运用了 var script = document.createElement('script'); 和 document.body.appendChild(script); 这两个核心代码。

（五）DataManager.loadDataFile 方法原理解释

我们的 json 文件是怎么加入到项目中的？其本质是运用了 XMLHttpRequest 类[52]的方法。CSDN 大多数的文章都简单地介绍了 XMLHttpRequest 类的使用方式，DataManager.loadDataFile 方法的流程也几乎如出一辙。基本上的流程是：

用 XMLHttpRequest.open 方法来打开 json 文件。

用 XMLHttpRequest.overrideMimeType 方法指定导入文件的类型为 json。

用 XMLHttpRequest.onload 方法来显性地写一个函数，并接受 XMLHttpRequest.responseText 属性的返回值。

用 XMLHttpRequest.send 方法来收尾。

值得说明的是，上述的全部流程都是严格的，先写 open，在写 send；使用 onload 方法时，还运用了 window 全局变量，mv 很多的全局变量都被保存与此；json 文件的本质是字符串形式的 JavaScript 对象，要用 JSON.parse[53] 方法来将字符串转换成具体的对象。

（六）SceneManager 类常用方法原理的简要解释

（本部分要填写一些核心的工作方法，还有一些特别的变量。但不要写成 SceneManager 类的 API 接口。）

1. SceneManager 类的主要功能、大概工作原理及核心执行方法：

SceneManager 类是一个静态类，实现各个场景的“增删改查”操作。其中包括：新建、储存、切换、刷新、删除、截屏、退出、等基本操作。

还有另一种理解是，该类的本质是一个用数组这个“数据结构”实现的一个“栈”，对场景实现栈的“先进后出”操作。最典型的是其 pop 和 push 方法的实现。

[52] MDN. XMLHttpRequest https://developer.mozilla.org/zh-CN/docs/Web/API/XMLHttpRequest

[53] 菜鸟教程.JavaScript JSON.parse() https://www.runoob.com/js/javascript-json-parse.html

2. SceneManager._stack

_stack 变量的本质是一个数组。是最能体现该类本质的一个变量。

3. SceneManager.goto

该方法主要功能是进入到一个场景。一般来说，用这个方法的前提是该场景已经存在。绝大多数情况下，我们使用该方法的目的仅仅只是想进入到另外一个场景内部而已。至于该场景的删除，则不是这个方法所考虑的事情。

该方法切勿和 C 语言的 goto 语句联系，二者毫无关系。

4. SceneManager.push

该方法的主要功也能实现场景的跳转，但是其本质是包含了 SceneManager.goto 方法。一般认为该方法是给数组栈_stack 进行赋值。该方法先完成对数组栈的入栈，再实现场景的跳转。

5. SceneManager.snap

该方法的主要功能是实现

6. SceneManager.update

7. SceneManager.updateMain

8. SceneManager.changeScene

9. SceneManager.updateScene

（七）菜单场景类为什么可以直接退回到地图

给出几个猜测：输入流、场景管理者、场景基类

_previousClass

（八）针对 SceneManager 类的转场解释

SceneManager.goto(Scene_Map);语句可以实现转换场景到指定的场景内部，这个功能可以用来控制场景转换。

（九）解释为什么每次打开 Scene_Menu 时，其背景图都是当前的游戏界面以及半透明效果

我们现在进行源码分析，因为 Scene_Menu 调用了 Scene_MenuBase 的 create 方法。源码如下：

```
Scene_Menu.prototype.create = function() {
    Scene_MenuBase.prototype.create.call(this);
    this.createCommandWindow();
    this.createGoldWindow();
    this.createStatusWindow();
};
```

Scene_MenuBase.prototype.create 方法用了 Scene_MenuBase 对象自带的一个 createBackground 创建背景方法。

```
Scene_MenuBase.prototype.create = function() {
    Scene_Base.prototype.create.call(this);
```

```
    this.createBackground();
    this.updateActor();
    this.createWindowLayer();
};
```

createBackground 方法新建了一个 Sprite 精灵类对象，并用专门的 SceneManager.backgroundBitmap 方法来截取当前的游戏画面，并作为背景。

```
Scene_MenuBase.prototype.createBackground = function() {
    this._backgroundSprite = new Sprite();
    this._backgroundSprite.bitmap = SceneManager.backgroundBitmap();
    this.addChild(this._backgroundSprite);
};
```

SceneManager.backgroundBitmap 静态类方法返回自己的成员变量_backgroundBitmap。

```
SceneManager.backgroundBitmap = function() {
    return this._backgroundBitmap;
};
```

_backgroundBitmap 成员变量是一个新设定的值，这个值被 SceneManager.snap 方法赋初值。其中，SceneManager.snapForBackground 方法还对这个位图进行了模糊效果，即调用了 Bitmap.prototype.blur 方法。Bitmap.prototype.blur 方法就实现了每次开启菜单栏时，菜单背景都是半透明的效果。

```
SceneManager.snapForBackground = function() {
    this._backgroundBitmap = this.snap();
    this._backgroundBitmap.blur();
};
```

调用底层的 Bitmap.snap 方法来截屏。

```
SceneManager.snap = function() {
    return Bitmap.snap(this._scene);
};
```

（十）对 addChild()方法的理解以及与 addWindow()方法的联系，addWindow()方法的必要性说明

在 Stage 基类和 Window 基类中，他们都是对 PIXI.Container 类的实现，在 PIXI.Container 中，都具有 addChild()方法。所以绝大部分的用 Stage 类和 Window 类作为超类的子类都可以使用 addChild()方法。

事实上，addWindow()方法其本质也是实现 addChild()方法。因此，我们可以尝试全部使用 addChild 方法来代替使用 addWindow 方法，减少使用新的方法，减少记忆。源码如下：

```
Scene_Base.prototype.addWindow = function(window) {
    this._windowLayer.addChild(window);
};
```

但是我们最好还是使用 addWindow 方法。因为 addWindow 方法是建立在 Scene_Base 类的_windowLayer 成员变量，只要控制了_windowLayer 变量，就可以对一个场景内部的全体窗口进行统一的控制了。这就是 addWindow 的必要性。

（十一）场景转换原理的解释

SceneManager.goto 方法的原理是在形参内填写一个类名，然后在手动新建一个新的对象，即完成了场景转换。

（十二）可选窗口的“确定点击窗口行为”的工作原理解释——关于 ok 字符串的来龙去脉

Window_Selectable.prototype.callOkHandler 方法调用了 this.callHandler('ok'); 注意 ok 这个字符串。这个字符串更多是一种指示变量，用来引导一个确定行为。追根溯源，callHandler 方法会执行 ok 字符串所对应的一个句柄函数方法。

那么 ok 字符串对应的函数句柄在哪里设置的呢？大部分是在场景类中被设置的。

（十三）人物对话的消息窗口是怎么控制的

Game_Interpreter.prototype.command101 方法设置了人物对话的脸图、和说话内容。

（十四）对项目中出现的 canvas 标签的理解

（学习方式：当前部分的内容以探究为主，尝试实现对不同位置的，已知的 canvas 标签进行修改。）

1. Graphics 图像处理静态类中的 canvas

主要是其成员变量_upperCanvas 和_canvas。_upperCanvas 的本质是上层画布，**而_canvas 暂时理解不了**。

_upperCanvas 上层画布的值要用 style 的方式来调用。值得注意的是，其本质是 HTMLElement.style 属性并返回一个 CSSStyleDeclaration 对象，不是一个可以修改的 canvas。[54]。

值得区别的是：

Graphics._upperCanvas 是 canvas；Graphics._upperCanvas.style 是 CSSStyleDeclaration 对象。

主要的修改方式示例：Graphics._upperCanvas.style.backgroundColor = 'rgb(255,0,0)';

backgroundColor 在 CSS 中的写法是：background-color，CSS 和 JavaScript 之间的样式更改是有区别的[55]。

2. Bitmap 位图类的 canvas

相关的只有_canvas 变量。Bitmap 类的本质是 canvas 标签。canvas 便签的写法几乎都被封装了[56]。

UP 主：赤瞳大白猫：

“有些 bitmap 的 canvas 属性是 null

bitmap 是这样的，如果你提供图片的文件地址，他只是一个浏览器的 img 标签元素

但如果你一旦准备在上面绘图，他就会自动创建 canvas”

关于 bitmap 本质的东西暂时跳过。有争议。〇

3. Sprite 精灵类、WindowLayer 类的 canvas

本质是还是运用了 bitmap 自带的 canvas。基本上没有太大的操作空间。

（十五）bitmap.x 的写法误区原理解释

比如 this._windowBackSprite.bitmap.y 的写法事实上是错误的，在断点调试时，总是查到其变量未定义。bitmap 的本质是一个 canvas 标签。这个标签尚未说明其具体位置。bitmap 通常是依赖于 sprite 的，精灵 sprite 有坐标值，而 bitmap 本身没有坐标值。

（十六）makeCommandList 方法所设置的“命令名”和“命令关键字”的保存位置 ？

首先，makeCommandList 方法来自于 Window_Command 类，起核心作用的是 Window_Command 的 addCommand 方法。

---

[54] MDN. HTMLElement.style https://developer.mozilla.org/zh-CN/docs/Web/API/HTMLElement/style

[55] MDN. CSS Properties Reference https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Properties_Reference

[56] CSDN.学习 HTML5 Canvas 这一篇文章就够了 https://blog.csdn.net/u012468376/article/details/73350998

（十七）window 系的 opacity 变量的本质

这里指的是 window 基类定义的 opacity 变量，opacity 变量的本质是 _windowSpriteContainer.alpha，“窗口精灵容器”的透明度。只要修改了 opacity 的值，就修改了窗口的框和其背景的透明度[57]。

其中，窗口精灵容器_windowSpriteContainer 包括了：_windowBackSprite 和_windowFrameSprite ，即窗口背景精灵和窗口框精灵。

读者可以探究以下的问题来加深对此内容的理解：为什么 Window_MapName 窗口可以让其窗口框及背景为空？使得我们每次看到地图名称时，都看不到窗口的边框和背景？

（十八）window 系的 contentsOpacity 和 contents 变量的本质

这两个变量都和_windowContentsSprite，“窗口内容精灵”相关。contentsOpacity 是精灵的透明度、contents 是精灵的位图 bitmap。

Window_MapName 窗口的背景绘图就是在 contents 内部绘制的，其渐变的刷新效果也是控制 contentsOpacity 变量。

（十九）Window_Base._dimmerSprite 变量设计的意义

这个变量设计的意义是，让窗口的背景和其框统一变成由_dimmerSprite 控制的灰色渐变样式。，让 dimmerSprite 的层级覆盖过 this._windowContentsSprite 窗口内容精灵，进而实现“替换”窗口背景和窗口边框。

我们可以探究以下这个问题：

Window_MapName.prototype.drawBackground 和 Window_Base.prototype.refreshDimmerBitmap 的效果区别问题？

首先，Window_Base.prototype.refreshDimmerBitmap 方法修改的是 this._dimmerSprite.bitmap，

而 Window_MapName.prototype.drawBackground 方法修改的是 this.contents 即 this._windowContentsSprite。由这二者的区别，我们可以提出：为什么要设置 Window_Base._dimmerSprite 这个变量？绘制阴影为什么要专门设置该变量？

_dimmerSprite 和其他常见的窗口精灵不同，他被首次定义在 Window_Base.initialize 方法内部，是 Window_Base 定义了它。

```
Window_Base.prototype.initialize = function(x, y, width, height) {
    Window.prototype.initialize.call(this);
    ......
    //模糊精灵 = null
    this._dimmerSprite = null;
};
```

真正使其加载进入窗口的是 Window_Base.showBackgroundDimmer 方法。这个方法不仅加载_dimmerSprite，而且还实质性地设置了具体的 bitmap 值，使得窗口出现竖直的灰色渐变。

```
Window_Base.prototype.showBackgroundDimmer = function() {
    if (!this._dimmerSprite) {
        this._dimmerSprite = new Sprite();
```

---

[57]

```
Window.prototype._createAllParts = function() {
    ......
    this._windowSpriteContainer.addChild(this._windowBackSprite);
    this._windowSpriteContainer.addChild(this._windowFrameSprite);
    ......
}
```

```
        this._dimmerSprite.bitmap = new Bitmap(0, 0);
        this.addChildToBack(this._dimmerSprite);
    }
    var bitmap = this._dimmerSprite.bitmap;
    if (bitmap.width !== this.width || bitmap.height !== this.height) {
        this.refreshDimmerBitmap();
    }
    this._dimmerSprite.visible = true;
    this.updateBackgroundDimmer();
};
```

Window.addChildToBack 方法才将_dimmerSprite 正式并归进入窗口内部。他并归的是一个 window 的_windowSpriteContainer，即“窗口精灵容器”的后面一位。这个代码让_dimmerSprite 盖过了_windowSpriteContainer，所以我们才看不到_windowSpriteContainer 所包含的窗口背景及窗口框。

```
Window.prototype.addChildToBack = function(child) {
    var containerIndex = this.children.indexOf(this._windowSpriteContainer);
    return this.addChildAt(child, containerIndex + 1);
};
```

（二十）待整理的原理

今天写代码的时候遇到了很多前所未有的挑战，懂得了一下的经验：

Graphics 和 SceneManager 的初始化问题：

你只有执行了 main.js，你才可以使用 Graphics 的方法，而且只要是用了 Graphics，SceneManager 也一并启动了。因为 Graphics 的初始化就是用 SceneManager 来完成的。因此在写代码报错时，要注意此时是否已经执行到 main.js 了，如果没有运行到 main.js 的 SceneManager，就不能使用 Graphics 系列的静态方法。

SceneManager.run 方法无条件获取错误的特点：

只要是被 SceneManager 运行过的代码，自己单独写的 throw error 就一定会被其 catchException 方法所捕捉，然后后续输出的 e.name 是无法被更改的，这就意味着，我们做不了自定义错误类的 name 属性。

try-catch 语句的特性：

真正使程序彻底停下来的，不是 throw error，而是 try-catch 语句。当程序有多个错误时，mv 提示界面只显示了一个错误的原因是，try-catch 语句会立刻让程序停下来。

（二十一）其他人的一些随笔说明

Drill_up：

“窗口和贴图都是同源的，窗口就是一种组合式贴图，只不过窗口被 rmmv 封装了一层，把特殊文字、窗口皮肤、下一步指针封装到一起了，而且还不能拆出来。这里比较麻烦的是，如果要显示简单文字，贴图可以直接 drawtext，但是如果要显示特殊文字，包括多行文本、不同颜色不同大小的字符，都必须贴个窗口，这就显得比较臃肿，虽然不会用上……”

十四、对 MV 界著名开源框架——Drill，的一系列理解

十五、对 Galv 系列插件的理解

（一）Galv_QuestLog.js 任务插件 的理解

理解其前台和后台的制作流程。

十六、对 Pixi.js 的学习与研究，探索 pixi 与 mv 代码之间的联系

（一）初始化与本地服务器搭建（Live Server）

RPGmv 源码运行的原理是在一个.html 网页文件上运行的。现在我们尝试做一个.html 文件的方式来导入相关的 js 脚本文件，进而实现整个游戏[58]。

我们在直接打开 RPGmv 项目自带的 index.html 文件时，会出现文件导入失败。这是因为“文件协议”和“html 协议”是有区别的。解决方式是两种：1：用 mv 软件来打开游戏文件。2：在自己的电脑里面，搭建本地服务器。我们肯定要选择后者。建立本地服务器。通过查找大量的资料得知，我们要建立的是“本地 http://localhost”。

localhost 的建立方法有很多，有的是用下载 wamp 来配置[59]，有的是直接在电脑配置 IIS[60]，还有一种是在 VScode 中下载插件“Live Server”[61]。在安装 Live Server 插件时，我并没有找到对应要求的修改.json 文件，因为找不到。

在使用 Live Server 时，要注意用 VScode 的打开文件夹的方式来打开对应的 index.html 文件。Live Server 默认打开名为 index 的文件。至此，我们才算是搭建好了服务器。

我们现在的目的不仅仅是为了打开 RPGmv 项目自带的 index.html 文件，而是写出另外一份导入了 pixi 框架的 html 文件。接下来的问题是导入 pixi 框架。教程里面默认是用外部导入的方式来实现的。我的方式是，导入现成的 pixi 文件。在这里，我们暂时不下载最新版本的 pixi 来作为我们的 lib[62]。

（二）创建画布、渲染器、与舞台

在编写对应的代码时，发现了两种不同的写法，一种需要写 PIXI.Application，另一种需要写渲染器 (renderer)(另有说法称“画布”)PIXI.autoDetectRenderer。通过查找 mv 源码，并没有发现任何代码调用了 PIXI.Application，但是有代码调用了 PIXI.autoDetectRenderer。查找 pixi，发现 Application 会调用到 autoDetectRenderer。换句话说，autoDetectRenderer 更加底层。我们选择 autoDetectRenderer 作为写 renderer 的方式。

（三）导入图片

接下来是导入图片。导入图片会涉及到几个复杂的概念。图片都是“材质/纹理 texture”，材质不能直接体现，要用“精灵”来显示出来。接下来是创建精灵对象。1：用材质来创建精灵。2：用图片直接创建精灵。这些创建精灵的方式都是可以理解为导入图片。导图的方式是用 PIXI.loader 来导入，再用 setup 函数，来统一地导入图片。

但是我发现了更加简单清晰的导入图片方式：

```
var sprite = new PIXI.Sprite.fromImage('assets/image.png');//直接由图片创建
```
[63]

这个方式有点问题。现在不得不学习如何用材质的方式、.json 的方式来导入图片。但是这个方式决定了我们必须使用图块组的方式来进行导图。

接下来的工作，应该是把时间花在了解 mv 源码上。

plugin.name = YEP_CoreEngine

plugin.parameters 本质上是一个数组。就是插件参数。

---

[58] 使用 Pixi.js 创建 HTML5 游戏 https://learnku.com/articles/3838/use-pixijs-to-create-html5-games

[59] 360 新知.用自己的电脑做服务器搭建网站并让互联网访问 http://xinzhi.wenda.so.com/a/1542122927208741

[60] 360 新知.怎样打开 win7 的 IIS 功能 http://xinzhi.wenda.so.com/a/1518018465610079

[61] CSDN.vscode 下关于 Live Server 的使用 https://blog.csdn.net/zhouwei_doris/article/details/80604604

[62] GitHub.pixi 最新版下载地址 https://github.com/pixijs/pixi.js/releases

[63] pixi.js 教程中文版--基础篇 https://www.cnblogs.com/songchunmin/p/7789611.html

十七、把 mv 源码从 ES5 版本调整到 ES6 版本

在阅读源码时，我们注意到 ES5 版本的 JavaScript 在编写类的语法相当繁琐，我们考虑用一些比较特别的方式让 mv 变成 ES6。

这里使用（）的工具来实现转写。

十八、基于 ES2015 的 mv 插件开发

首先，这里说的“ES2015”指的是“ES6”，即现代 JavaScript，并不是前文多次提到的“ES5”。在这一章节中，我们会广泛地使用 ES6 的新特性和语法糖来开发 mv 插件。我建议先了解一下 ES6 的新特性，再来阅读此部分[64]。

（一）类编写的规范

注意这一段代码：

```
this.name = name || "";
```

严格来说，这并不是 JavaScript 的要求，但这几乎是开发者的惯用写法。在对类的属性初始化时，我们用这种手段保证代码一定会初始化而不会出现 undefined 的情况。

（二）如何用 ES6 的类语法糖来继承 prototype 的“函数类”？

想要使用 ES6 开发插件，必须会遇到如何使用 ES6 来继承源码的 prototype 函数类，MDN 的 class 参考给了我们一个具体的示例[65]：

```
function Animal (name) {
 this.name = name;
}
Animal.prototype.speak = function () {
 console.log(this.name + ' makes a noise.');}
class Dog extends Animal {
 speak() {
   super.speak();
   console.log(this.name + ' barks.');
 }}
var d = new Dog('Mitzie');
d.speak();//Mitzie makes a noise.  Mitzie barks.
```

我们可以给出这样的准则：

① 在 class 继承的子类中，在构造函数内都使用 super 来调用父类的属性。否则子类无法使用 this 来调用父类的属性。

② 如果重写的方法在功能上和父类相同，就使用 super，否则不使用 super。这种写法相当于.call(this)。

（三）ES6 类的继承写法

https://developer.mozilla.org/zh-CN/docs/Learn/JavaScript/Objects/Inheritance#inheritance_with_class_syntax

请阅读上面的英文解释即可。

十九、问题堆栈与 sundry：插件开发的几个实际问题？

（一）插件开发问题

[64] https://babeljs.io/docs/en/learn

[65] https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Classes

那这样的话，就可以去看事件源码，看人物和队伍类了

比如说，怎么用代码显示当前的战斗队伍
怎么用代码给角色的最大生命增加 50
怎么用代码给背包增加一件装备，或者减少
写装备强化啊，装备套装啊，装备镶嵌啊插件

WindowLayer {_events: Events, _eventsCount: 1, tempDisplayObjectParent: null, transform: TransformStatic, alpha: 1, …}
children

_paramPlus

项目是怎么导入 window.png 的？又是如何运用内部图片的？懂了。

1. canvas 标签 id 查找问题

当前的观点是，每一个 mv 对象都是 canvas 标签，怎么用 JavaScript 的方法准确的找到当前 canvas 标签的 id？并让其执行指定的 CSS 样式？

答案：Graphics._upperCanvas 的标签 id 是 upperCanvas。

2. 精灵和其附属位图的宽高值是那个代码负责的？

this._windowBackSprite.width 与 this._windowBackSprite.bitmap.width 的值是一样的。是哪个代码产生这样的结果的？

答案：是 Sprite.prototype._onBitmapLoad 方法。

```
Sprite.prototype._onBitmapLoad = function() {
  if (this._frame.width === 0 && this._frame.height === 0) {
    this._frame.width = this._bitmap.width;
    this._frame.height = this._bitmap.height;
  }
  this._refresh();
};
```

这个方法让精灵和其附属位图的宽高值相等，在导入位图的确认时，框定好这个精灵容器的框架大小。

drill：这个问题其实你没答到关键点，js 是用的浅拷贝，也就是指针，width 指向了 bitmap 的 width，所以是同一个值

有待完善〇

3. 为什么 Window_MapName 窗口可以设置其 content（bitmap）的颜色渐变并让其窗口框及背景为空？（开始探究）

查询的内容：

open 方法。

4. 为什么 Window_MapName 窗口的底层灰色内容不会遮挡到地图的名称？

5. 不同 mv 启动方式对存档调用的问题

用 VScode+Chrome、VScode+360 浏览器、rmmv 这三种启动方式都遇到了不同形式存档问题。

目前唯独 rmmv 的启动方式可以调用全部的存档。

问题初步排查：目前怀疑的是 Window_TitleCommand.prototype.makeCommandList 方法。因为这个方法会让“继续游戏”这个字段变成灰色。

```
Window_TitleCommand.prototype.makeCommandList = function() {
    this.addCommand(TextManager.newGame,   'newGame');
    this.addCommand(TextManager.continue_, 'continue', this.isContinueEnabled());
    this.addCommand(TextManager.options,   'options');
};
```

在这个方法中，就开始牵扯到数据管理者的方法了。

```
Window_TitleCommand.prototype.isContinueEnabled = function() {
    return DataManager.isAnySavefileExists();
};
```

显示灰色的主要原因是下面的 isAnySavefileExists 方法返回了假值。可能是 globalInfo 数组根本就什么内容都没有。

```
DataManager.isAnySavefileExists = function() {
    var globalInfo = this.loadGlobalInfo();
    if (globalInfo) {
        for (var i = 1; i < globalInfo.length; i++) {
            if (this.isThisGameFile(i)) {
                return true;
            }
        }
    }
    return false;
};
```

通过深入的断点调试，目前到了这个可以的方法：

在存档问题中，其他人的观点：

奋斗小乐：

使用 NW.js 的时候就使用它的接口保存在本地上。一般来说，浏览器就没有本地文件的操作权限。你网页不保存 localStorage，那还想保存哪里。操作本地文件的权限很可怕的，可以直接让你电脑中毒。mz 使用了 localforage 库，你也可以用这个库

赤瞳大白猫：

localforage 会先检查浏览器是否支持 indexDB。不支持就用 localstorage

在 StorageManager 类中被广泛使用的 localStorage 变量，在 localforage.js 重写了。localStorage 变量用于存档，将文件变成 json 格式。

出现在 StorageManager 类的字符串压缩算法方法。

LZString.compressToBase64

从源码中，我们发现 decompressFromBase64 方法似乎是一个对应的解压方法。这个方法是否运用到了 mv 中？是的。

现在的解决思路是，自己搭建一个数据储存的过程，将网络数据储存到本地中。

6. 使用被 JSDoc 的@private 标签修饰过的方法，可能降低运行速度的问题

7. 14.1.2 小优任务插件，代码尚未运行

报一个 bug

VScode+Chrome 浏览器的方式打开游戏，会出现$gameSystem.appendListener 方法不存在的错误。通过断点调试，发现程序不会去运行以下的程序段：

```
Game_System.prototype.initialize = function() {
    this.Lagomoro_Mission_initialize();
    this._Lagomoro_Mission_Data = {};
};
```

但是使用 rmmv 来启动程序时，则不会出现上述的问题。

我搞不懂，为什么 mv 就不运行被重写的 Game_System.prototype.initialize 方法呢。

8. 待添加的问题

（二）sarange-project-code-database 萨兰奇项目注意事项

1. JSDoc 命令：

在 E:\GitText\sarange-project-code-database\js 文件下启动 cmd，并执行以下代码。

```
jsdoc CodeManager.js -r plugins
jsdoc -r plugins
```

代码解释：对 js 文件夹内的 CodeManager.js 和其目录下的 plugins 文件夹的全部内容，进行 JSDoc 文档的输出。

2. nodejs 环境变量配置：

可能是每次 Git 自动更新，导致环境变量丢失，现在将阮中楠本机的配置环境变量的常用步骤记录于此：

自主检查配置是否成功的 cmd 指令：

```
npm -v
node -v
npm config get prefix
npm config get cache
```

先输入 node，再输入 require('cluster')

```
npm install axios
npm install jsdoc
```

用户变量 NODE_PATH 添加：

E:\NodeJS_install

系统变量 path 添加：
E:\NodeJS_install

在 E:\NodeJS_install 目录下创建两个目录，分别是 node_cache 和 node_global

用户变量 NODE_PATH 添加：
E:\NodeJS_install\node_global\node_modules
在用户变量 path 中，将 npm 的值改成：
E:\NodeJS_install\node_global\

（三）值得被保留的一些代码写法

```
/**
 * @author 阮中楠
 * @method loadSetting_RuanZhongNan
 * @deprecated
 * @description
 * 导入阮中楠的开发者个人设置。这个方法用于导入阮中楠专属的设置控件。
 *
 * 主调方法：Scene_Boot.prototype.create
 *
 * 算法：
 * 在游戏开始装载时，与 DataManager.loadDatabase()方法并列地导入数据。
 *
 * 使用 DataManager.loadDataFile 的算法来导入。
 *
 * 教程：
 * 主教程：https://blog.csdn.net/ryelqy/article/details/79279273
 *
 * 规范示例：https://xhr.spec.whatwg.org/#interface-xmlhttprequest
 *
 * API 接口：https://developer.mozilla.org/zh-CN/docs/Web/API/XMLHttpRequest
 */
static loadSetting_RuanZhongNan() {
  var xhr = new XMLHttpRequest();
  //网址请求 打开( 'GET' , url 位置) 这里写的是绝对路径。
  xhr.open('GET', 'js/plugins/customDateFile/setting_RuanZhongNan.json', true);
  //设置导入数据的数据类型
  xhr.overrideMimeType('application/json');
  //导入数据 这个写法是固定的。
  xhr.onload = function () {
    //我们的全局变量 $settingRZN 都被 window 全局变量所保存。
    window.$settingRZN = JSON.parse(xhr.responseText);
  };
  //发送请求
  xhr.send(null);
}
```

（四）代码阅读

常见的名词：

Graphviz

CodeViz

doxygen

（截至目前，尚未找到合适的代码阅读工具，只有一个 JSDoc 代码 API 接口文档生成工具正在使用。）

（五）其他可视化编程的工具、技术、引擎、框架

ECharts

https://www.runoob.com/echarts/echarts-tutorial.html

前端游戏框架哪个好

https://blog.csdn.net/valada/article/details/81639708

（六）sundry

这是巴哈姆特的 siakomobi 写的书本：《RPG Maker MV 游戏制作基本外功篇:从操作到完成游戏一镜到底,马上就会!》这个书本可以说是 RPGmv 界内的教材。我现在没有。

Siakomobi 的作品：《七音图腾篇》（UltraKagura）

ok 字段是哪里设置的？

E:\NodeJS_install\node_cache

npm config set prefix “E:\NodeJS_install\node_global”

npm config set cache “E:\NodeJS_install\node_cache”

npm config get prefix

npm config get cache

E:\NodeJS_install\node_global\node_modules

E:\NodeJS_install\node_global\

require('cluster')

DataManager.loadDataFile 的原理与 PluginManager.loadScript 的区别

fetch.then((response) => { })

XMLHttpRequest

（七）临时代码

（八）待学习的打包技术：
QQ：770436947
然后输 npm init
npm install @capacitor/core @capacitor/cli
然后就是 npx cap add 加相应平台。
然后把 add 换成 open 就可以打开
如果在 add 的时候出现了错误因为你没有在那个文件夹中放入 3w 文件夹并且在其中放入 index 点 html。
然后那个工程就已经构建好了，按照那个工具的打包方法就可以打包了
详情参照 as 和 xcoda 的使用教程。

https://www.bilibili.com/read/cv7828113

https://ionicframework.com/
https://capacitorjs.com/

（九）待学习的 socket.io 技术
socket 是网络编程技术。
可以找：
QQ：770436947、1442417954

（十）待研究的光追技术
YEP_GridFreeDoodads.js
FilterController.js
ParticleEmitter.js
https://sigmasuccour.itch.io/false-server
遊戲這裡下載

（十一）待学习的 mv 新版 pixi 更新包
在官方论坛里搜 pixi 就可以找到了，本体在 itch

（十二）待学习的 jsdoc2md 技术
这个技术可以将 jsdoc 转变成 md。
https://github.com/jsdoc2md/jsdoc-to-markdown

https://github.com/jsdoc2md/jsdoc-to-markdown/blob/master/docs/API.md