## Rmmv串行与并行

由于公共事件游离于战斗界面和地图界面，并且事件分为并行和串行，这里很容易混淆实际概念，这里需要反复阅读才能理解。

### 串行与并行

**串行**：是指指令必须按照顺序依次执行，如果途中出现了等待的情况，那么指令都会被阻塞，必须等到等待结束才能执行下一条指令。

（比如地图事件中的 自动执行、确定键 的执行过程。）



**并行：**是指指令可以在同一个时间中，分别执行。指令之间相互独立，互不干扰。

（比如地图事件中的 并行处理 的执行过程。）



注意，串行和并行是对于 **事件页** 而言的，事件页与事件页之间可以并行或串行。

事件页自身中的所有指令，都是必须从上往下按顺序依次执行的。

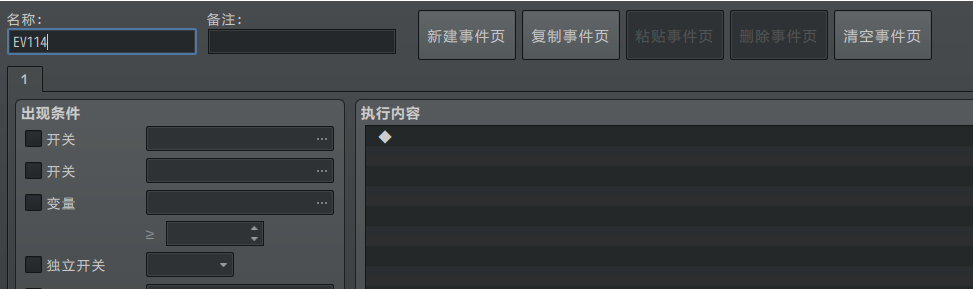
另外，**对话框**事件指令是特殊的指令体，只要执行对话框，就会强制串行，阻塞其他所有事件的线程，这里的 并行串行 不包含该指令体造成的影响。

### 总体介绍

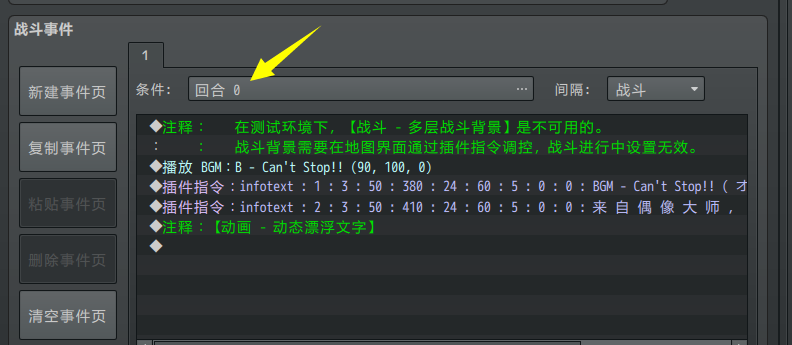
Rmmv的公共事件执行与串行并行如下图所示，具体使用细节可以看看后面章节。



**地图事件**：即地图界面中的事件，在地图中双击即可放置一个事件。地图事件有行走图、坐标、自主移动、触发条件等各个属性。



**战斗事件**：即战斗界面中的事件，在敌群中进行设置。注意，战斗事件没有并行的能力，并且触发条件都是根据战斗阻塞串行的流程进行的。

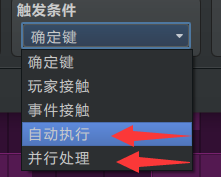


## Rmmv公共事件

### 事件指令执行

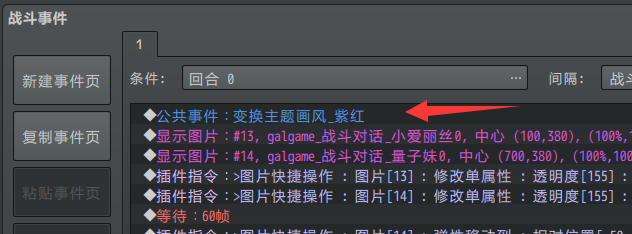
**1）添加方法（地图事件）**

你可以在rmmv地图事件中，添加新的公共事件运行，串行并行与事件设置相关。



**2）添加方法（战斗事件）**

你可以在rmmv战斗事件中，添加新的公共事件运行，战斗界面只能串行执行。



**3）细节说明**

简单来说，公共事件在这里相当于一个便捷的函数封装库。能够封装一系列事件操作。

如果公共事件中有开启独立开关操作，则在事件指令执行中是有效的，能够控制当前事件的独立开关。而其它执行条件下，由于没有事件本体执行，所以独立开关指令是无效的。

如果你在公共事件中使用插件指令“事件[110]”，那么对应的事件是你当前所在地图中，id为110的事件。

**3）串行与并行**

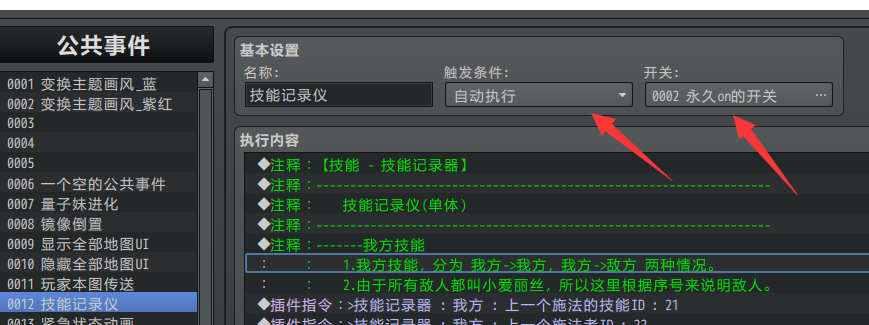
如果事件的设置是自动执行(串行)或其它功能，那么公共事件会阻塞，依次运行。

如果事件的设置是并行处理，那么公共事件与事件指令一起并行运行。

### 自身触发条件执行

**1）添加方法**

你可以在rmmv公共事件自身中，添加触发开关或者并行触发条件。

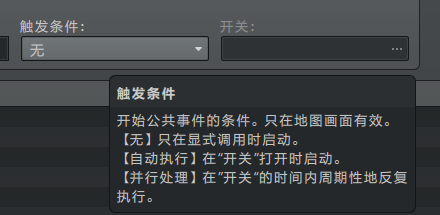


**2）细节说明**

必须要设置触发条件，才会执行，如果触发条件为无，则只能通过 事件指令或技能 来执行。

**3）串行与并行**

注意，自动执行(串行)和并行处理的功能，只能在地图界面中有效。



### 技能执行

**1）添加方法**

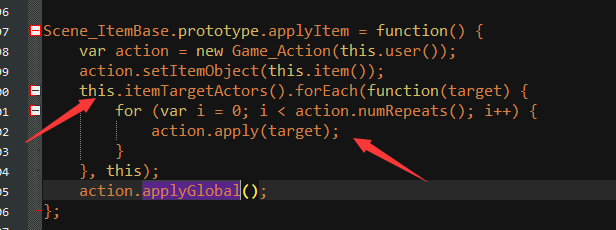
你可以在技能效果中，添加公共事件执行。



**2）细节说明**

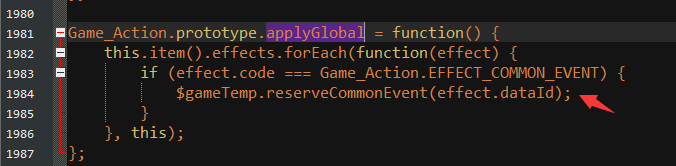
技能中如果写了多个效果和公共事件，那么先顺序执行效果，再顺序执行公共事件。

这里需要从脚本层面上解析，因为公共事件和一般的hp增减功能不一样：



技能范围确定了多个使用者，所以必须先让使用者顺序执行其它效果，再执行公共事件。

图中的ApplyGlobal即执行公共事件函数。



如果有多个公共事件，公共事件之间根据顺序排序执行

**3）串行与并行**

**通过技能执行的公共事件无法并行**。

## 插件功能

### 地图 - 多线程

插件本体：

◆Drill\_LayerCommandThread 地图 - 多线程

子插件：

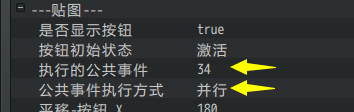
◆Drill\_OperateHud 互动 - 鼠标辅助操作面板

◆Drill\_GaugeButton 地图UI - 地图按钮集

◆Drill\_WindowMenuButton 控件 - 主菜单选项按钮管理器

◆Drill\_SecretCode 系统 - 秘籍输入器

插件都只作用于 地图界面 。可以在地图界面中 串行/并行 执行公共事件。



从原理上看，插件的本质是走 [事件指令执行](#_事件指令执行) 的流程，在地图中创建一个”自动执行”或”并行处理”的事件，并使其执行对应的公共事件。



另外，**菜单界面中不能执行公共事件**。主菜单选项按钮管理器 的虽然是作用于菜单界面，但是在主菜单的按钮执行后，会立即离开菜单界面，进入地图界面来执行相关公共事件。（在物品界面使用物品公共事件也是一样的原理，会立即回到地图界面执行公共事件。）

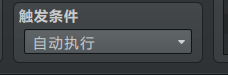
## 技巧说明

### 地图界面的串行

地图中的事件，使用串行是最不容易出错的方法，**因为串行会阻塞其他指令执行，具有先后顺序**。

进入地图前，程序会从id=1的事件开始，如果这个事件是自动执行的，则加入串行的队列，如果这个事件是并行的，则 并行执行指令 。





串行与并行存在 **多个事件** 之间的 **时间差** 的问题。

如果初始化的事件都为串行(自动执行)，那么所有串行的事件指令都能够根据id顺序先后执行事件指令。

而并行执行就不一定了。等待指令需要时间、角色移动需要时间、角色转向需要时间、地图淡入淡出也需要时间。如果这些指令使用并行来执行，那么你无法确定哪个事件最先执行完毕，哪个最后执行。