## 概述

### 相关插件

核心插件：

◆Drill\_CoreOfNumberArray 系统 - 变量数组核心

类似插件：

◆Drill\_CoreOfString 系统 - 字符串核心

这类插件用于扩展可以用的数据类型，用法和 变量、开关相似；

可以通过 脚本调用，也可以通过 插件指令调用。

（但到后期，还是建议大家学习一下脚本，底层知识了解越多，设计思路就越明朗）

|  |
| --- |
| 该文档可以与文档“28.物体管理 > 关于事件的缓存数据.docx”结合来看。 |

### 名词索引

以下你可以按住ctrl键点击下面的词，可以直接定位到想了解的名词：

|  |  |
| --- | --- |
| 定义 | [变量数组](#_变量数组)  [Debug查看](#_1）Debug查看) [用字符串查看](#_2）用字符串查看) |
| 变量数组操作 | [索引取值](#_1）索引取值) [获取长度](#_2）获取长度) [遍历数组写法](#_3）遍历数组_-_标准写法)  [复制/切分数组](#_1）复制/切分数组) [统计值](#_2）统计值) [脚本调用](#_3）脚本调用)  [同步删除](#_同步删除) [插件指令中的批量关系](#_插件指令中的批量关系) |
| 变量数组应用 | [解谜设计-顺序记忆](#_解谜设计-顺序记忆)  [物体设计-复制事件的批量管理](#_物体设计-复制事件的批量管理) |

## 变量数组

### 定义

**变量数组：**指多个值按顺序划分为一个组的数据结构。

每个数字都有一个索引值对应，通过索引可以找到那个数字。



|  |
| --- |
| 变量数组 是一维数组，且只能装 **整数数字**（含负整数），索引从1开始计数。 |

游戏编辑器中的 变量列表 就是一个变量数组。

如下图，数组的最大值为80，也就是说这是一个长度为80的变量数组。

根据索引值21，能够获取到 “缓冲变量A” 的值。



由于游戏编辑器中 变量 经常不够用，

所以这里可以使用 变量数组，以节省 变量、开关、独立开关。

注意，一个单独的变量 与 一个长度为1的数组，不相等。

因为它们不是同类数据。



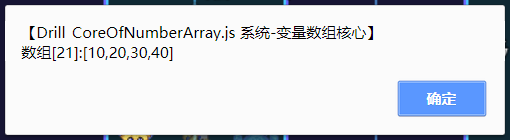
|  |
| --- |
| 在实际编程语言中，数组的定义极为宽泛，  包含：二维数组、多维数组、元数组……这里都不考虑，只考虑简单的一维数字数组。 |

### 查看方式

#### 1）Debug查看

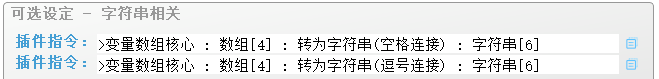
使用插件指令，可以直接查看数组内的数据。





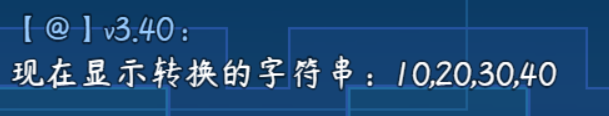
#### 2）用字符串查看

使用插件指令，可以将数组转为字符串，然后在对话框中显示出来。



如下图将 数组[21] 转为字符串，给字符串[21]，字符串使用窗口字符显示。





### 基本操作

#### 1）索引取值

**索引取值：**在指定的数组中，设置索引值，可以通过插件指令获取到相应的变量值。

获取到值后，将把值赋给变量：（如下图，把数组的第1个值，给变量21）

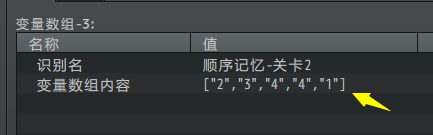




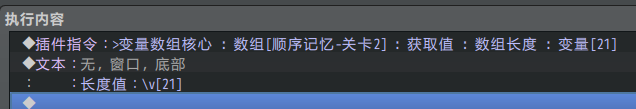
#### 2）获取长度

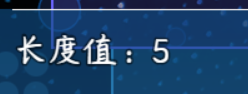
**变量数组长度：**每个数组都具有自己的长度值，具体根据配置、插件指令而定。

比如，核心中配置了五个数字，则长度值为5。



使用插件指令赋值，并显示，可以看到下图效果：





需要注意的是，如果使用索引获取时，索引值超出了数组长度，则获取到默认的0值。

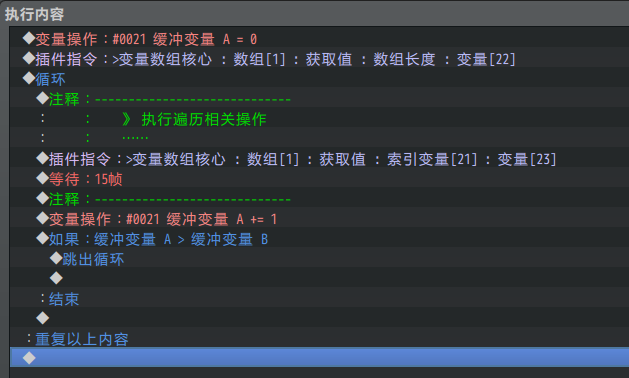
#### 3）遍历数组 - 标准写法

**遍历：**是指一次访问数组的每个元素，并执行相关操作。



一个标准的遍历写法如下，

变量23即每次从数组中获取到的值。



用到了3个变量：

缓冲变量A（id 21）： 用于表示当前的索引。

缓冲变量B（id 22）： 用于表示当前数组的长度。

缓冲变量C（id 23）： 用于表示数组中取到的数字，取到即用。

|  |
| --- |
| 注意，如果此遍历写在了**并行事件中，且有 等待 时间**，  那么你一定要确保同一时间里，变量ABC不会被其他事件调用，  否则会出现赋值错位的情况。 |

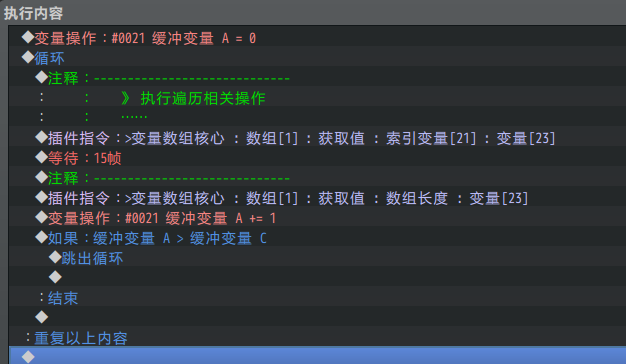
|  |
| --- |
| 如果你会编程，那么脚本写法是下面这样的：  var arr = //...数组[1]  var v22 = 10;  for(var v21 = 0; v21 < v22; v21 += 1 ){  var v23 = arr[v21];  // 》 执行遍历相关操作  }  这就是一个常规的for循环，只不过参数名为v21,v22,v23 。 |

#### 4）遍历数组 - 节约变量写法

由于数组长度可以实时获取，

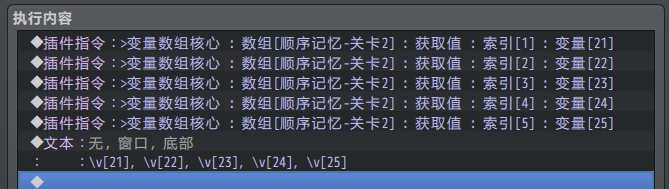
你可以按下图写，节约一个变量B：

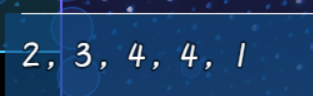
（变量C被用了两次，第一次执行相关操作，第二次用来获取数组长度并判断）



如果你知道数组长度，比如长度为5，且不会变化，

那么直接 索引取值 就可以，只不过相同的指令要连续写5次：





|  |
| --- |
| 注意，这里的节约写法只是节约变量，用起来方便一点点，  而计算机能处理大量的数据，即使你写10000个变量，也能在一秒内执行完毕，  所以放开手去用变量，只要不写成死循环，都没问题。 |

### 扩展操作

#### 1）复制/切分数组

你可以在核心中，通过复制数组，作为一个临时数组。

复制后操作这个临时数组，用来进行遍历或其他操作。



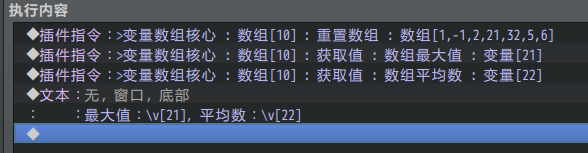
#### 2）统计值

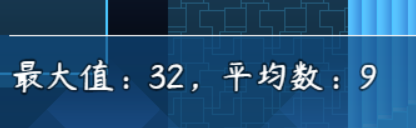
变量数组核心还额外提供了一些查找最大最小值、平均数的方法。

（注意，由于平均数不能存小数，所以计算结果会被四舍五入。）



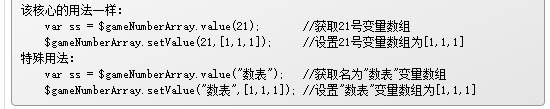
用法如下：





#### 3）脚本调用

如果你需要使用一些脚本来调用到数组，可以用下面的指令：



### 同步删除

**同步删除：**是指将多个数组绑定到一起，删除时影响所有绑定的数组。

注意，所有绑定的数组长度必须一致，才能同步删除。





绑定同步删除后，

任意一个数组执行了删除操作，那么其它数组都会一起删除，如下图。



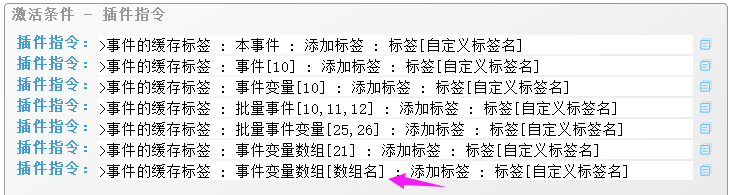
|  |
| --- |
| 同步删除功能，多用于 事件id数组 和 事件属性值数组，  准备两个数组，绑定同步删除，然后根据 事件的属性值 进行条件删除，  属性值会留下不符合删除条件的，事件id数组也会只留下对应的事件id。 |

你可以去看看示例 物体管理管理层，筛选出 缓存数字大于50 的所有小爱丽丝的用法。

### 插件指令中的批量关系

#### 1）事件变量数组

部分子插件支持变量数组的使用，如下图。



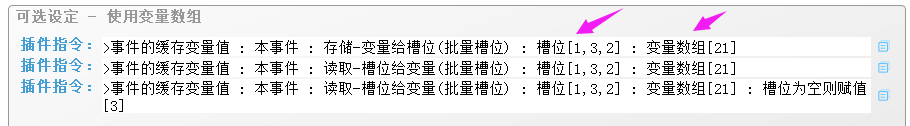
“事件变量数组”即数组中存放的所有事件id，会依次执行相关的插件指令操作。

#### 2）事件的缓存数据交互

部分子插件还能用 变量数组 进行数据交互，如下图。

存储值时，会将变量数组中的值，转移存储到槽位中。

读取值时，槽位中的值会被拿出来，形成新的变量数组。



详细介绍和用法可以去看看：“28.物体管理 > 关于事件的缓存数据.docx”。

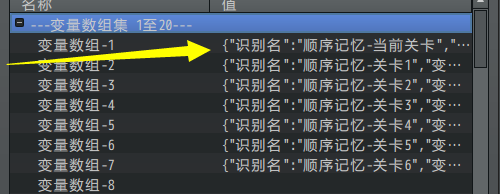
## 变量数组的应用

### 解谜设计-顺序记忆

解谜设计中，顺序记忆关卡用到了变量数组。

通过变量数组，将关卡的 按钮要求顺序 存储了起来。

具体可以去看看“8.物体 > 解谜设计-顺序记忆.docx”。



### 物体设计-复制事件的批量管理

物体管理中，经常会复制大量的事件。

这些新复制的事件，需要将事件id存放到数组中统一管理起来。

具体可以去看看“9.物体触发 > 物体设计-复制事件的批量管理.docx”。

