# 事件系统

配表 data/event\_config.lua

事件系统是用于给策划使用的通过游戏内事件触发行为

### 事件

每个**事件**有数个**子事件**,每个**子事件**,可以触发多个**动作**

比如一个接受任务的事件,有n个**子事件**每个**子事件**表示接受了那个任务,然后每个**子事件**可以触发多个**动作**

['acceptQuest'] =

{

[7] = {

{ action = 'movie', params = { 'act5' } }

},

[12] = {

{ action = 'movie', params = { 'act6' } }

},

},

上面的例子定一了接受任务的**事件**，然后任务编号7 为一个**子事件，**触发了**动作，**播放剧情动画**act5**

#### 当前支持的事件

|  |  |
| --- | --- |
| **loadingEnd** | 读取完毕，系统默认事件，默认在表里面，无须修改  只有一个**子事件**，id = 1**,** 触发**动作**，***finishLoading*** |

|  |  |
| --- | --- |
| **newGame** | 玩家创建新号，并进入游戏，在第一次进入游戏，并游戏读取完成时触发  只有一个**子事件**，id = 1**,** 触发**动作**，填表者自行设置 |
| **例子**  ['newGame'] =  {  [1] = {  { action='movie', params = { 'act1' } }  },  }, |

|  |  |
| --- | --- |
| **endOfMovie** | 剧情动画播放完毕事件  **子事件**，为剧情动画id |
| **例子**  ['endOfMovie'] =  {  ['act1'] = {  { action = 'showUI', params = { 'welcome\_win' } }  }  }, |

|  |  |
| --- | --- |
| **acceptQuest** | 接受任务事件  **子事件**，为任务id |
| **例子**  ['acceptQuest'] =  {  --任务接收    [7] = {  { action = 'movie', params = { 'act5' } }  },  [12] = {  { action = 'movie', params = { 'act6' } }  },  }, |

|  |  |
| --- | --- |
| **finishQuest** | 完成任务事件  **子事件**，为任务id |
| **例子**  ['finishQuest'] =  {  --任务完成  [3] = {  { action = 'movie', params = { 'act2' } }  },  [4] = {  { action = 'movie', params = { 'act3' } }  },  [5] = {  { action = 'movie', params = { 'act4' } }  },    }, |

#### 当前支持的行为

|  |  |  |
| --- | --- | --- |
| **行为** | **movie** | |
| 播放剧情动画 | |
| **参数** | params[1] | 动画id |

|  |  |  |
| --- | --- | --- |
| **行为** | **showUI** | |
| 显示UI窗口 | |
| **参数** | params[1] | 窗口名字 |

|  |  |  |
| --- | --- | --- |
| **行为** | **doQuest** | |
| 执行任务 | |
| **参数** | params[1] | 任务id |
| params[2] | 已接任务，quest\_type 2= 可接任务 |