THE NVL MAKER

新手教程

（三）

选择的制作

一、选项按钮外观设定

上一教程提到了脚本文件与标签，这在电子小说里用得最多的就是“选择”了。THE NVL Maker 里提供了比较简单的选择按钮的制作方法。

首先我们需要对选择按钮的外观等等进行设定。
在工程设定（或界面设定）栏目下，可以看到“选项设定”。

在这里用的默认设定，表现在游戏里是这样的按钮。
当然，你可以随意修改图片。


我要选择A
我要选择B

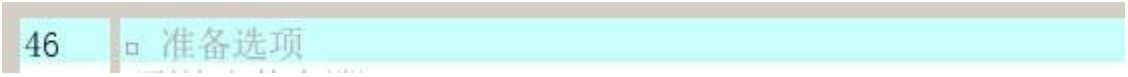
二、显示选项按钮

设定完之后我们返回“脚本编辑->打开脚本”，开始制作选择~
总之在之前的脚本的基础上继续吧。

（1）准备选项

新建一行，在指令窗口里选择“准备选项”。弹出了下面这样的窗口。

由于很可能是对话一阵之后，出现了可以分歧的路线，才让玩家选择的，因此对话框和系统按钮可能都在画面上。如果你喜欢在选择的时候让画面空白一点，可以勾选这两个选项，如果你希望让玩家在选择的时候也能做存档等等操作，就这样不改设定也没问题。
点下“确认”。



（2）设定位置

现在可以开始显示选项按钮了。

不过，按钮显示之前，当然需要设定一下位置。

假设我们需要显示两个按钮，在画面上的位置分别是（0,200）和（0,300）。
那么就可以这样做。
继续新建一行，选择“按钮位置”，输入第一个按钮的位置，点下确认。

（3）设定按钮

再次新建一行，选择“选项按钮”。

因为大部分的设定，例如“一般选中按下”时的按钮样式，我们都设定了默认值，所以这里只需要填写少量内容。

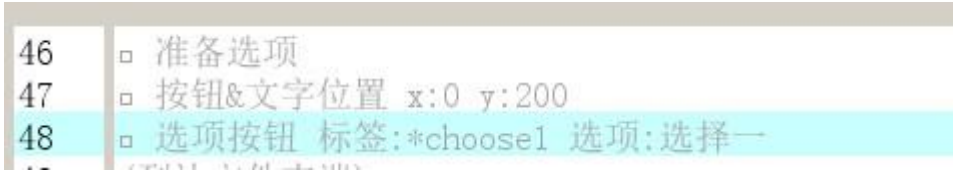
第一是【选项文字】，也就是要描绘在选项按钮上的，玩家可以看到的文字。

第二是【标签】和【文件】，用来指定当这个选择按钮点下后，会继续执行的脚本段。

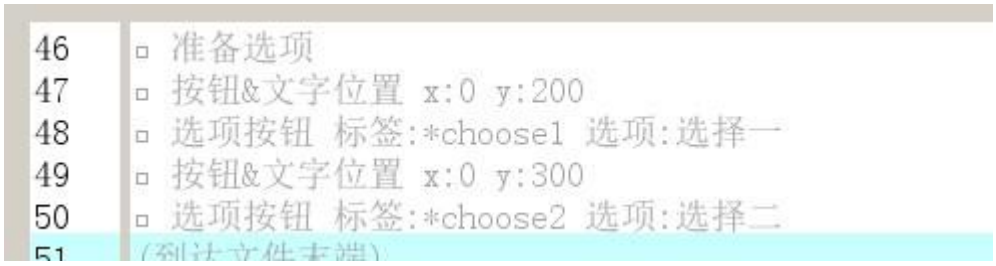
注意：对于选择按钮，除了选项文字以外，标签也是必须填写的，而文件名是可选的，在不填写文件名的情况下，默认会在同一个文件内寻找对应的标签名。

如果不填写标签的话，在测试时会默认跳至对应脚本的“*start”这个标签。假如这个标签不存在，那自然就是出错了（喂）。

总之先填成这样吧，点击“确认”。



现在，重复一的操作，建立一个位置在（0,300），选项文字为“选择二”，标签为*choose2的按钮。



（5）等待选项

现在所有的准备工作都做好了，只需要再加上一条“等待选项”，就可以看到成果了。
所谓“等待选项”，意思就是显示出已经设定好的选项按钮，然后等待玩家选择。
在玩家做出选择之前，游戏会一直静止在那里。

可以设定各种各样的显示效果，不过这里就全部用默认值好了。点下“确认”。

```
46  准备选项
47  按钮&文字位置 x:0 y:200
48  选项按钮 标签:*choose1 选项:选择一
49  按钮&文字位置 x:0 y:300
50  选项按钮 标签:*choose2 选项:选择二
51  等待选项
52  (到达文件末端)
```

三、设定选择后的处理

（1）添加标签和内容

现在可以开始测……呃等一下，虽然选项设定了标签“*choose1”和“*choose2”，可这两个标签好像还没有加？
先加上好了。

```
46    ▫ 准备选项
47    ▫ 按钮&文字位置 x:0 y:200
48    ▫ 选项按钮 标签:*choose1 选项:选择一
49    ▫ 按钮&文字位置 x:0 y:300
50    ▫ 选项按钮 标签:*choose2 选项:选择二
51    ▫ 等待选项
52    ★标签 标签名:*choose1
53    选择了一。[w]
54    ★标签 标签名:*choose2
55    选择了二。[w]
56    (到达文件末端)
```

不知道怎么加这些内容的，回去看前面一讲哦~

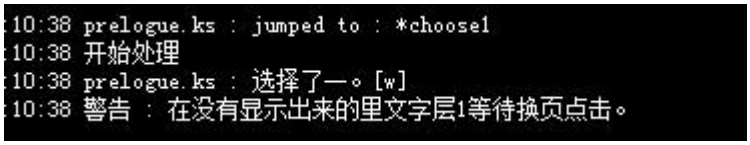
现在测试一下：

好像很简单，随便选一个吧……

（2）好像有哪里不对

现在问题来了，不管点了哪个按钮，理论上说都会出现“选择了一。”“选择了二。”之类的对话啊。
对话哪里去了？为什么游戏还是静止的？

不过如果你点开游戏菜单栏上面的“调试->控制台”，打开黑色的窗口，就可以看到这样的警告：

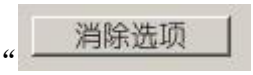


因为 THE NVL Maker 里选择按钮显示时候用到的是“里文字层 1”，而平时用的对话显示是在“表文字层 0”，这个设置现在还没改过来呢。
有些时候不知道出了什么问题，可以打开这个窗口看看。

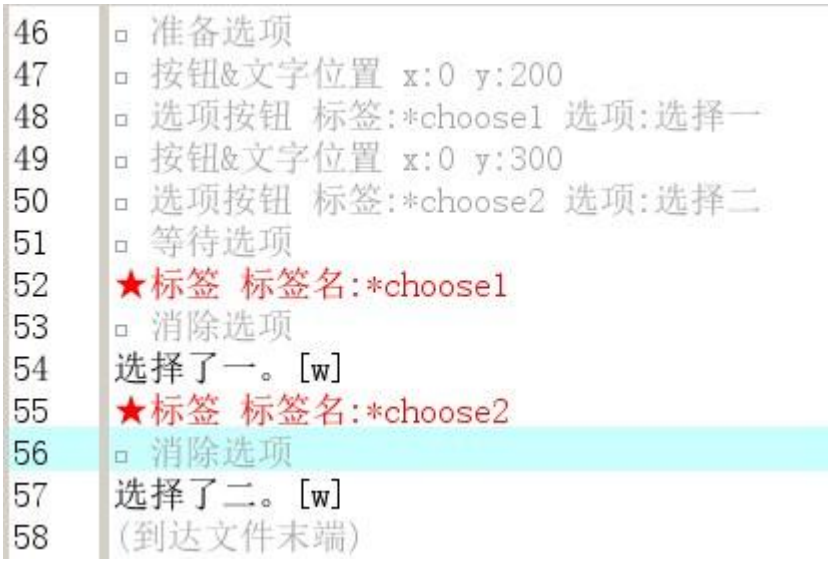
文字和图片，设定在“表页”的时候，可以立即显示。
而设定在“里页”的时候，可以通过“执行切换”和“等待切换”来做出各种各样的效果。（比如刚刚的选择按钮，就是渐入显示的。）
页和层的概念，后面也会不时涉及到。请先大概留个印象。

（3）清理一下画面，让一切恢复正常

不过现在不用管那么多基本原理，只要知道如何改掉这个设置就行了。

用到的指令“ 

”，它会帮你把文字层复位，并且清理掉选项按钮。
注意，跳转到的每个标签后，都要加上这个指令。



现在再测试一下。没有问题了吧？

接下来记得在选择 1 和选择 2 对话完毕的时候，加上“事件跳转”，让两个选择最后能够合并到一条线上来继续剧情。（可以参考 Template 里的范例写法。）

如果不这么做的话，在显示“选择了一”之后，还会继续显示“选择了二”的对话。

四、好感度？

你一定会遇到这样的剧情设计，就是不管选择了哪一项，台词都是一样的，但是根据选择不同，游戏里可攻略人物的好感度会上升或下降。
这是剧本偷懒的经典方式（喂）。

可是总不能因为剧本偷懒，脚本就写两段完全一模一样的内容，然后再合并吧？

这个时候就要用到“让两个选择都跳到同个段落，但是数值根据选择变化”的功能了。

在吉里吉里中，数值变化的操作是用一些简单的代码来执行的，被称为“TJS 表达式”。

……喂，也不要听到“代码”两个字就点叉啊，我保证这个简单到你会加减乘除就一定会写啊。

（1）重新设定按钮

```
46    ▫ 准备选项
47    ▫ 按钮&文字位置 x:0 y:200
48    ▫ 选项按钮 标签:*继续 选项:选择一
49    ▫ 按钮&文字位置 x:0 y:300
50    ▫ 选项按钮 标签:*继续 选项:选择二
51    ▫ 等待选项
52    ★标签 标签名:*继续
53    ▫ 消除选项
54    不管选择哪一项，剧情都会这样继续的。[w]
55    (到达文件末端)
```

稍微修改一点点设置，现在脚本变成了这样。

（2）表达式（TJS 式）

然后假设“选择一的时候好感度+5”，“选择二的时候好感度-5”。
设定第一个按钮是这样的。

表达式一栏，填入【f.好感度+=5】。

那么第二个按钮的表达式，当然是填上【f.好感度-=5】了。

```
46  ▫ 准备选项
47  ▫ 按钮&文字位置 x:0 y:200
48  ▫ 选项按钮 TJS:f.好感度+=5 标签:*继续 选项:选择一
49  ▫ 按钮&文字位置 x:0 y:300
50  ▫ 选项按钮 TJS:f.好感度-=5 标签:*继续 选项:选择二
51  ▫ 等待选项
52  ★标签 标签名:*继续
53  ▫ 消除选项
54  不管选择哪一项，剧情都会这样继续的。[w]
```

现在变成了这样。
保存测试一下，一切正常。

（3）变数

每次写教程都要讲这个还真是麻烦啊，把 Nscripter 全攻略的变数介绍拿来一下吧（喂）。

可以加加减减的数值，在游戏里就叫做“变量”或者“变数”——因为是可以变化的嘛。
变量本身需要有一个名字，例如“某人的好感度”，“某人的 BT 度”“HP”，“昨天的台风等级”等，而变量的内容，就是一个数字。（有些时候不只数字，不过这里先知道可以存储数字就行了。）

KAG 系统中有三种变数。
以“f.”开头的，被称为一般变数，游戏里最常使用到的也是这种。会保存在游戏存档里。
以“sf.”开头的，被称为全局变数（系统变数），会保存在独立的记录里。
以“tf.”开头的，叫做临时变数，游戏里随时可以用，但是不会被保存，一旦关闭游戏下次再打开，就和没存在过一样。

刚刚说到的“f.好感度”就是一个变数的名字。
为变数命名，在吉里吉里中还是比较简单的，只要是“f.”开头，支持英语，中文，克林贡语，多斯拉克语，或者随便什么语，以及以上语种的混排（只要开头不是数字就行）。根据你的习惯起名吧。
当你写下一个新的变数名的时候，系统会自动送给它一个空的值，可以被视为 0。
而接下来你通过 TJS 式对这个变数的操作，就会被一步步记录下来。
在玩家存档的时候，他们也会被保存。

如果觉得以上太复杂，我们先回到“f.好感度”这个变数上。

在游戏刚开始它是不存在的，只有在玩家点下任何一个按钮以后，它才被创建出来。

所以一开始它的默认值是 0。

在“f.好感度+=5”以后，它的值是 5。
在“f.好感度-=5”以后，它的值是-5。

这个很好理解吧？

（4）让变数显示出来

有时候为了方便调试，或者剧情需要（好感查询系统？），变数的值需要显示在游戏里。这时候就要用到新的 TJS 式了。而且，我们可以把它插入到对话里。

比如说这样：

在 KAG 里，以@开头的，或者[]括起来的内容就是一行指令。
包括“指令名”和“属性”两个部分。格式是这样的。
[指令名 属性 1=值 1 属性 2=值 2 属性 3=值 3]
@指令名 属性 1=值 1 属性 2=值 2 属性 3=值 3

这两种格式都可以使用，但@要求每行只能书写一个指令。所以在对话里插入时，用[]的形式会比较容易。

EMB 这个指令就是用来显示变数的。它有一个叫 EXP 的属性。EXP 的值则是你填写的 TJS 表达式。

测试一下。

嗯，完全没有问题。

顺便一说，EMB 在脚本编辑器里，对应的是这个指令“显示变数（显示 TJS 式）”。

假如对话末尾没有分页等待之类的符号，在两段对话中插入这么一行也可以达到如上文的效果。


```
46  现在的值是
47  ◇显示TJS式 表达式:f.好感度
48  哦。[w]
49  (到达文件末端)
```

五、变数的应用

（1）第三种选择

除了之前说的两种情况，还有一种常见的剧情设定，就是在选择之后剧情并没有什么特别的变化，但是到了某个分歧点，突然之前埋下的“地雷”就炸了。

例如说，在第一个事件里让角色的好感度下降了，
在出现第二个事件的时候，本来有三个选项可以选，但好感度不够，于是只显示了两个。

这种设定的话，就需要用到条件分歧了。

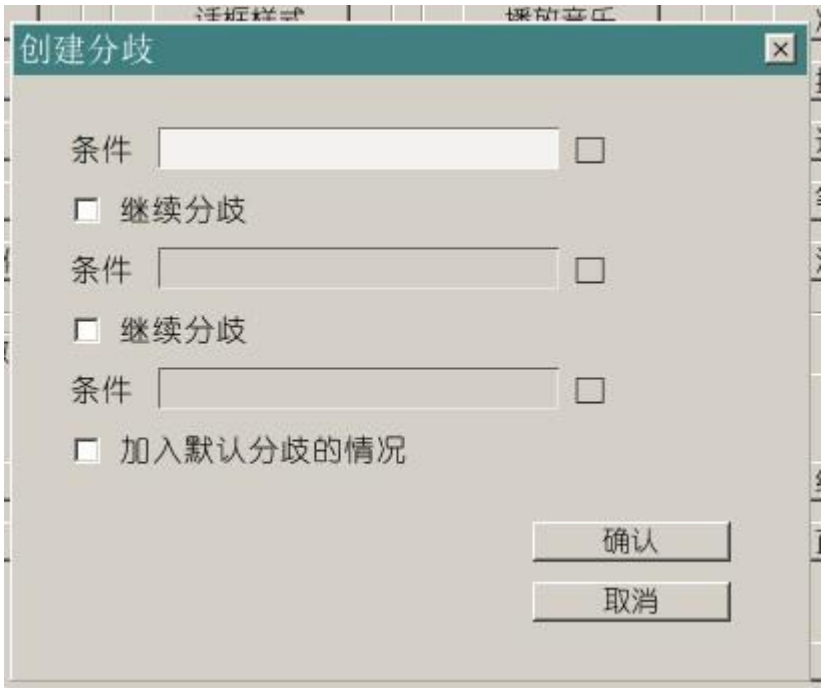
假设一下接下来就是这种情况。如果 f.好感度的值>=5，那么，你可以多一个选择。

```
57  ▫ 准备选项
58  ▫ 按钮&文字位置 x:0 y:200
59  ▫ 选项按钮 标签:*planA 选项:方案A
60  ▫ 按钮&文字位置 x:0 y:300
61  ▫ 选项按钮 标签:*planB 选项:方案B
62  ▫ 等待选项
63  (到达文件末端)
```

现在来加这个隐藏的“方案 C”。

（2）创建条件分歧

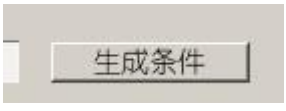
在第二个按钮下面新建一行，戳一下“变数系统-创建分歧”。



这里的条件要填什么呢。虽然可以猜得出来是“f.好感度>=5”，不过也不能这么蒙吧。

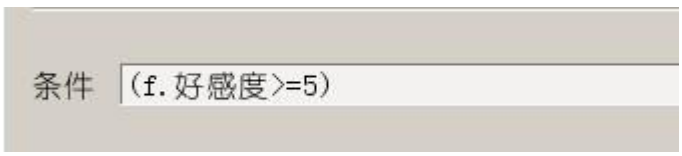
这个文本框后面有个□。在 THE NVL Maker 里，这就表示“你基本可以用鼠标来解决问题”。

填写这样的东西，应该没问题吧。

先别急着点确认，这里需要多一步： 

。

点一下这个按钮，查看一下系统自动生成的 TJS 条件式是不是对的。



没有问题，点下“确认”。

因为没有其他的分歧情况，也没有“都不符合”的时候的默认分歧情况，所以就这样保持原样，点下确认就好了。

```
44  ◇条件分歧 表达式:(f.好感度>=5)
45  ◇条件分歧结束
```

接下来要插入的就是第三个选项了。
在“条件分歧”和“条件分歧结束”中间插入新行，填写按钮位置和选项按钮。

```
57  ▫ 准备选项
58  ▫ 按钮&文字位置 x:0 y:200
59  ▫ 选项按钮 标签:*planA 选项:方案A
60  ▫ 按钮&文字位置 x:0 y:300
61  ▫ 选项按钮 标签:*planB 选项:方案B
62  ◇条件分歧 表达式:(f.好感度>=5)
63  ▫ 按钮&文字位置 x:0 y:400
64  ▫ 选项按钮 标签:*planC 选项:方案C
65  ◇条件分歧结束
66  ▫ 等待选项
67  (到达文件末端)
```

不过，记得加入标签*planA，*planB 和*planC，以及后续的内容~

（3）测试

现在就只有在之前选择了一，f.好感度为 5 的情况下，选项“方案 C”才会显示。

六、扩展知识

（1）“等于”和“等于等于”不一样

为什么给 TJS 条件式增加了特殊的生成器，而普通的 TJS 式就要自己填呢。
因为 TJS 条件式虽然大部分情况下也挺好写的，但是总会不小心出现各种各样的错误。
比如说，“当 f.好感度等于 5”和“设定 f.好感度的值为 5”，这两个式子一个是条件，一个是执行的内容，当然不会一样。

条件判断的时候“等于”是用“==”，两个等号来表示的。因此，

“当 f.好感度等于 5”应该写成“f.好感度==5”。
而“设定 f.好感度的值为 5”，才是“f.好感度=5”。

为了避免写错，就专门给“条件”加上了生成器。

这样也可以写出更复杂的式子，比如说当 f.好感度 等于 5 并且 f.HP 大于 10 并且 f.今天的天空颜色 等于 0x0000FF 的时候，才有方案 C 的选项出现……

（2）“变数”不一定是“数”

之前已经说过了，变数的值除了数字，还可以有其他的形式。
例如说，字符串。

假如你需要让玩家输入自己定义的角色姓名，这个姓名一定也是存储在一个变数里的。

在吉里吉里中，字符串是用双引号""（注意不是“”）括起来的文字。并且也可以保存在变数里。

例如你可以设定选择按钮：
选择“方案 A”的时候，f.方案="方案 A"。
选择“方案 B”的时候，f.方案="方案 B"。
选择“方案 C”的时候，f.方案="方案 C"。
最后在对话里说：
选择了[emb exp="f.方案"]。
写成什么样都没关系，不要忘记双引号哟。

（3）加不加引号？

除了给字符串设置值以外，刚刚用到的 EMB 指令什么的，后面的 exp=也跟着双引号。这是在告诉系统，双引号之内的内容是连续的。
通常来说，除了字符串以外的东西加不加双引号并不是很重要。但是……

（1）属性的内容包含空格的情况

假如你的文件名有空格，例如 图片 abc.png 这种时候，写成
[bg storage=图片 abc]
就会被当成是显示一张名叫“图片”的文件，而不是“图片 abc”。这时候的结果很可能就是报错找不到图片。

（2）值为数字的情况

假如你是在操作或者比较变量。用于数字的时候，应该写成：
@eval exp="f.好感度=5"，或者@if exp="f.好感度==5"这样的形式，数字周围是不能加引号的，否则就会被认为是字符串。

（3）外面已经有一层双引号的情况

在 TJS 式内，也就是@eval exp=""里面的内容时，由于 TJS 式本身是要用双引号括起来的，就不能再重复使用了。
所以如果要给字符串赋值，请写成这样的形式：
@eval exp="f.星期几='星期五'"，用单引号'代替双引号。
或者写成：
@eval exp="f.星期几=\"星期五\""，用\"来代替双引号。

THE NVL Maker 会为所有合适的属性都加上双引号或者单引号，但自己撰写 KAG 脚本和 TJS 式的时候，单、双引号的使用请额外注意。

（4）全局变数的使用：初次打开游戏与二周目

通常情况下，使用 f.变数来应付游戏里的大部分情况就足够了。但是，假如需要制作“整个游戏的公共设置”这样的功能，就要用到全局变数了。

所谓全局变数，就是会被系统保存，但是不会受到存档、读档操作影响的变数。（例如玩家打开游戏，但是没有做任何一次保存，只要脚本里对全局变数进行了操作，依然会被记录下来。）

例如说，我们需要游戏在第一次执行的时候播放片头动画，之后就不再播放。
或者需要给标题画面增加一个按钮，只有在通关以后才出现。
类似这两种情况就要使用到全局变数。

初次执行游戏播放片头动画（以后执行游戏就不再播放）的代码示例：

```
;假如变数 sf.first 为 0，则意味着从来没有播放过动画
[if exp="sf.first==0"]

;播放片头动画的代码
[video width="1024" left="0" visible="true" height="768" top="0"]
[playvideo storage="op.mpg"]
[wv canskip=false]

;执行到这里，说明动画已经播放完了
;接下来操作变数 sf.first 为 1，系统将会记录这个值
 [eval exp="sf.first=1"]
;下次打开游戏的时候，因为 sf.first 已经不再是 0 而是 1 了，因此从上面的 if 到这个 endif 中间的代码都不会再执行
[endif]
```