WebGAL 1.0 版本 开发教程

# WebGAL

### 一次编写，处处运行，无需网页开发基础，3分钟即可学会所有的语法，只要你有灵感，就可以立刻开始开始创作你自己的Galgame！

Demo: <http://msfasr.com/WebGAL_Demo_Page/>

## 如果你是想要开发属于自己的 Galgame 的开发者，请下载发行版：

<https://github.com/MakinoharaShoko/WebGAL/releases/>

## 如何在本地运行WebGAL？

#### Windows用户可以使用以下方式快速打开调试界面:

现在，你可以直接运行WebGAL-win.exe即可开始调试你的视觉小说。如果遇到杀毒软件拦截或防火墙拦截等情况，请放行以允许该程序运行。

#### 其他用户：

WebGAL理论上可以在任何浏览器上运行，但是由于浏览器的**限制跨域访问**机制，本地 JavaScript 脚本可能无法在浏览器上运行，因此，你需要在本地建立一个http服务器。你可以选择任何你喜爱的的http服务器，并通过本地http服务器访问index.html来开始游戏。如果你没有本地http服务器，你可以使用Node.js简单地搭建一个。

我在**WebGAL发行版**的根目录放置了一个app.js作为 Node.js 服务器脚本，在你安装了 Node.js 后，只需要通过以下命令：

npm install express  
npm install open  
node app.js

即可开始调试你的游戏。

# 快速上手教程：

现在，你应该已经运行起来了一个调试服务器，接下来让我来教你如何编写你自己的剧本

你应该能够发现，在解压后的文件夹下，有一个 WebGAL 文件夹，在 WebGAL 文件夹下，有一个 game 文件夹，那就是你的游戏资源应该存放的地方。

你的所有游戏剧本、图片、立绘都应该在放解压后目录的 WebGAL/game 文件夹下，目录对应的资源说明如下：

| **文件夹** | **存放的资源** |
| --- | --- |
| background | 用于存放背景图片、标题页背景 |
| figure | 用于存放人物立绘 |
| scene | 用于存放用户剧本 |
| bgm | 用于存放背景音乐 |
| vocal | 用于存放配音文件 |

## 用户剧本的编写语法：

首先，程序会从**初始脚本**start.txt开始运行，在后续的脚本编写中，我会告诉你如何跳转章节或设置分支选项。

**编写剧本的方式非常简单，且与自然语言几乎无异。**

### 首先，让我们来学习编写基本的人物对话：

首先，使用记事本或VS Code 、sublime 等文本编辑器，打开游戏资源目录下 scene 文件夹中的 start.txt，然后你就可以开始编写你的第一条对话了。

编写人物对话的方法非常简单，你只需要输入：

**注意，每条对话的冒号、分号应当为英文字符！**

人物:对话;

示例：

雪之下雪乃:请用茶;  
由比滨:啊，谢谢;  
小町:谢谢雪乃姐！;  
一色:谢谢学姐。;

在每条对话/剧本后，使用分号作为结束。（如果不写分号有时候也行，但是作者还是建议加上分号以防止bug）。

#### 连续对话

如果你的多条对话没有改变人物名称，你可以使用连续对话，而可以省略人物名称，再在你需要的时候使用：

雪之下雪乃:你到得真早;  
对不起，等很久了吗？;//此时，对话的人物名称仍然是“雪之下雪乃”。  
比企谷八幡:刚到而已;

### 改变背景/人物立绘：

要让 WebGAL 能够读取背景图片和人物立绘，你的背景图片应该被放在background文件夹内，而立绘图片则应该放在figure文件夹中。

接下来，你可以通过以下简单的语句来改变当前显示的背景图片和人物立绘：

changeBG:testBG03.jpg;//改变背景  
changeP:testFigure02.png;//改变人物立绘  
changeBG:none;//关闭背景  
changeP:none;//关闭人物立绘

你有可能会发现，在你改变背景图片或人物立绘后，你需要再点击一下鼠标才能显示下一条对话，如果你希望在改变背景图片/立绘后立即执行下一条语句的内容，请使用：

changeBG\_next:testBG03.jpg;  
changeP\_next:testFigure02.png;//改变人物立绘  
一色:谢谢学姐！;

如果你这样做，那么在背景图片/立绘替换后，程序会立刻执行下一条语句。

### 将立绘放在不同的位置

现在，你可以在页面的三个不同位置放置不同的立绘，只需要在放置立绘的语句处加上你要放置的位置就可以了，示例如下：

changeP\_left:testFigure03.png;  
changeP:testFigure04.png;  
changeP\_right:testFigure03.png;

以上三行分别对应着左、中、右三个不同的位置。三个不同位置的立绘是相互独立的，所以如果你需要清除立绘，必须分别独立清除：

changeP\_left:none;  
changeP:none;  
changeP\_right:none;

如果你想要在立绘改变后立刻执行下一条语句，操作方法与之前一样，即在命令后加上 \_next :

changeP\_left\_next:testFigure03.png;  
changeP\_next:testFigure04.png;  
changeP\_right\_next:testFigure03.png;

注意命令的书写顺序，如果你这样写（先写next后写位置），那么将是无效的：

changeP\_next\_left:testFigure03.png;  
changeP\_next\_right:testFigure03.png;

### 跳转场景与分支选择：

在 Galgame 中，跳转章节、场景与分支选择是不可或缺的，因此，本游戏模组也支持场景跳转与分支选择。

你可以将你的剧本拆分成多个 txt 文档，并使用一个简单的语句来切换当前运行的剧本。

假如你现在写了两个章节的剧本，分别是 Chapter-1.txt 与 Chapter-2.txt ，在Chapter-1.txt 运行结束后，你希望跳转到 Chapter-2.txt 对应的场景，请使用以下语句：

#### 场景跳转：

语句：

changeScene:Chapter-2.txt;

示例：

(Chapter-1.txt)  
......  
......  
changeScene:Chapter-2.txt;  
接下来执行的就是Chapter-2.txt的内容了。  
......  
......  
(Chapter-2.txt)

通过执行这条命令，你将切换游戏场景，并使接下来的剧情发展按照新的场景剧本运行。新的剧本会在下一次鼠标点击后运行。

#### 分支选择：

如果你的剧本存在分支选项，你希望通过选择不同的选项进入不同的章节，请使用以下语句：

choose:{叫住她:Chapter-2.txt,回家:Chapter-3.txt};

你只需要将选项的文本与选择选项后要进入的剧本名称一一对应起来，就可以实现分支选择的功能了。

### 引入背景音乐（BGM）

引入背景音乐的方法极其简单，你只需要将背景音乐放置在 /bgm 目录下，然后便可以简单地在需要的时候令其播放：

bgm:夏影.mp3;

### 在显示对话时同时播放语音

众所周知，Galgame 最吸引人的地方往往在于其有优秀的配音。在此版本中，你可以为对话引入配音了，将你的配音放入vocal文件夹，然后在对话的前面，人物名称的后面加入 vocal-语音文件名,即可引入，语法示例如下：

比企谷八幡:vocal-V3.ogg,刚到而已;

在连续对话时，语音的引入方式也是一样的：

雪之下雪乃:vocal-V1.ogg,你到得真早;  
vocal-V2.ogg,对不起，等很久了吗？;

## 进阶教程：

### 在同一个场景（TXT文件）内实现语句跳转、分支跳转

如果你想要创建一个分支，但是却觉得为此新建一个TXT文件太麻烦，你可以尝试使用以下方式在同一文件内实现创建分支和跳转语句。

**注意：如果你的分支很长，我不建议你使用这种方式，因为一个TXT的行数不适宜太长，否则可能会导致加载慢、响应迟钝等性能问题。**

#### 首先，你必须理解如何使用label：

......  
jump\_label:label\_1;  
//以下这些行会被忽略:  
......  
......  
......  
//然后，你需要创建label:  
label:label\_1;  
然后，程序会在这里继续执行。;  
......  
......

简而言之， jump\_label 类似于 goto 语句，能够立刻让你的剧本跳到场景（TXT文件）中的任意一个位置，而这个位置需要你用label创建。

如果把jump\_label 比作任意门，那么这个任意门的终点就是 label 所在的位置。

#### 有了上面的基础，你就可以通过 choose\_label 的方式来实现用分支来跳转到 label 所在的位置了：

WebGAL:让我们来测试分支跳转到label！;  
choose\_label:{测试1:label\_1,测试2:label\_2};  
label:label\_1;  
现在应该是1号分支;  
jump\_label:end;  
label:label\_2;  
现在应该是2号分支;  
jump\_label:end;  
label:end;  
现在是统一的结束;

注意，在每个分支的结尾，你应该用 jump\_label 来跳转到你希望的位置。由于程序是线性执行的，所以如果你没有在分支的结束跳转，那么程序会继续往下运行，比如说：

WebGAL:让我们来测试分支跳转到label！;  
choose\_label:{测试1:label\_1,测试2:label\_2};  
label:label\_1;  
现在应该是1号分支;  
label:label\_2;  
现在应该是2号分支;  
现在是统一的结束;

在这个剧本中，如果你选择了分支2，那么一切看起来好像没有问题。但是如果你选择了分支1，你会惊讶地发现，在分支1执行完后，竟然分支2又执行了一遍。那是因为程序按顺序继续执行下一行了，而你没有指定在分支结束后跳到哪里。