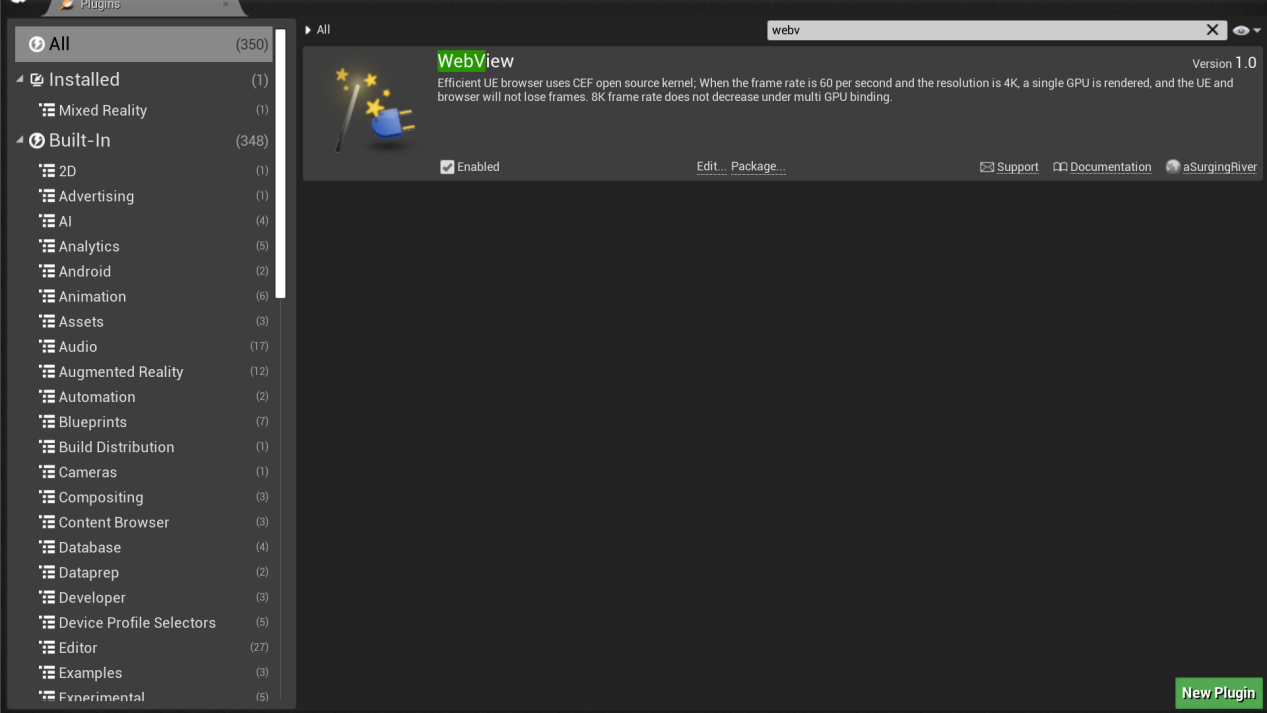
**WebView 插件说明**

WebView是一个为虚幻引擎设计的企业浏览器插件。它采用了许多先进技术。浏览网页或观看视频时不会卡顿。内核采用最新CEF版本并进行优化。具有GPU绑定功能，支持游戏和浏览器渲染分离，插件内部增加渲染缓存，渲染效果流畅。在与网页的通信方面，采用V8技术，使虚幻引擎与网页进行实时交互。推荐使用虚幻引擎渲染场景和JavaScript进行UI交互，可以大大提高系统开发效率。

Content目录包含插件使用模板Web\_Demo, Resources目录Sample1.html为JS调用UE的模板。

1. 导入插件

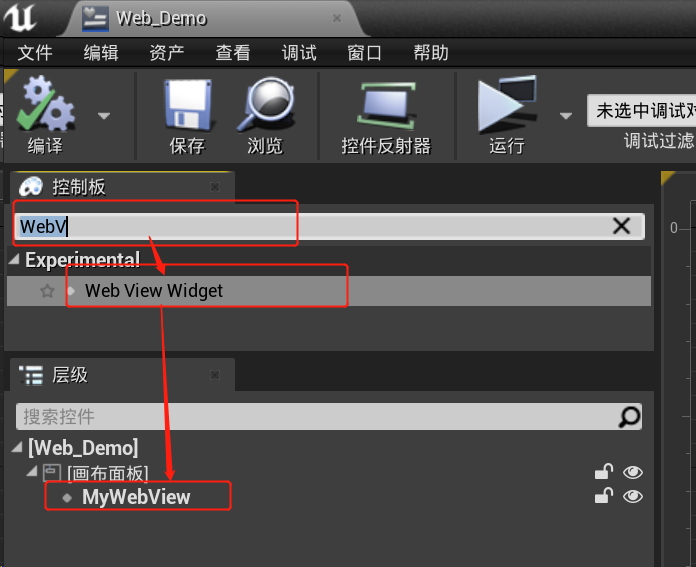
将WebView插件包放入项目plugins目录下。编译项目并确保项目编译成功。运行项目后，编辑->插件，在搜索框中输入WebView可以看到插件已经加载成功。如下图所示：



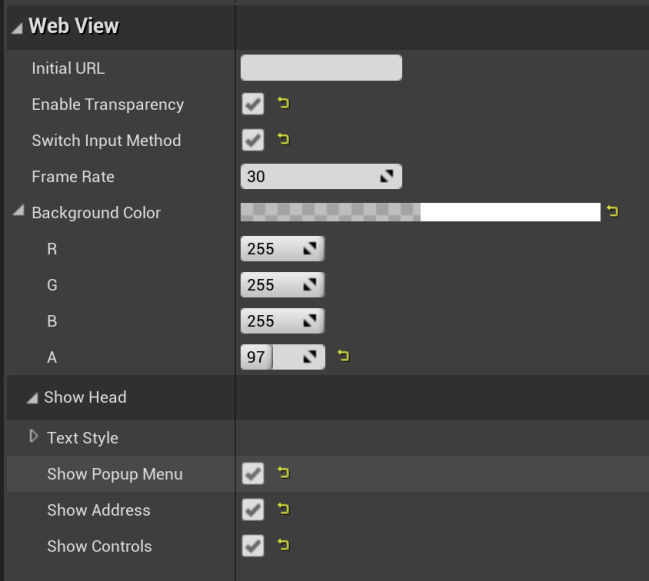
WebView插件和WebBrowser插件一样内核使用CEF，只是他们的版本不同。在使用过程中这两个插件不能同时使用，如果同时开启则会出现不正常。

1. 使用插件

新建UMG蓝图，在Palette中输入Cef即可看到Cef Web Widget窗口部件并将部件拖入画布中，并调整部件在画布中全屏显示。如下图所示：



在画布中选中Web View Widget部件，在Details中找到Web View配置项。如下图所示：



配置项目说明:

Initial URL : 浏览器启动加载网页的URL如 [www.baidu.com](http://www.baidu.com)

Background Color: 控制透明网页背景显示颜色。其中Alpha不为255时则支持透明网页显示，该值越小鼠标的穿透度越大，如果希望绝大多数透明能穿透可将该值调小。

Frame Rate: 设置浏览器输出的帧率，在30与60之间，值越大浏览器显示越流畅，如果播放视频建议设置为60，普通网页浏览建议30。该值与UE帧率无关。

Enable Transparency: 在网页为透明的情况下，控制是否允许鼠标穿透，即点击透明网页鼠标事件将穿透到网页下面的UI或场景中。常用于JS作UI，UE作场景渲染的项目中。

Text Style: 配置后续地址栏，控制按钮显示字体大小及颜色。

Show Controls: 控制网页是否显示控制按钮。

Show Popup Menu: 控制鼠标右击页面是否显示导航菜单。

Show Address Bar: 控制网页是否显示地址栏。

Switch Input Method: 控制网页是否支持输入法切换。

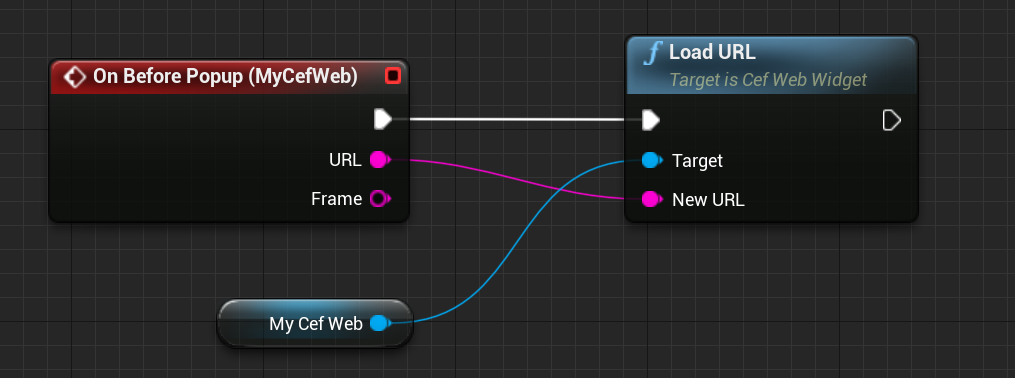
1. 网页跳转

在浏览网页或者百度搜索时，对新网页实现跳转功能。

画布面板中选中WebView Widget部件。Details->Events绑定On Before Popup事件。

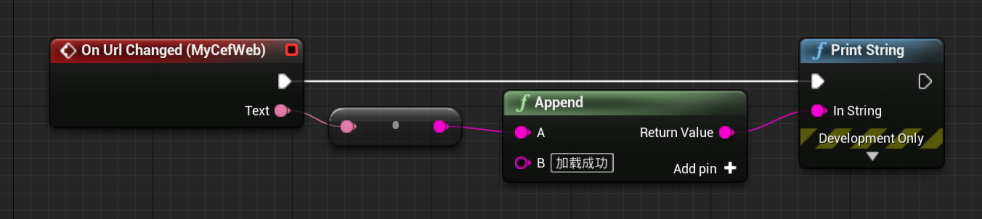


在On Before Popup事件实现中用WebView Widget部件加载新的URL即可。如下图所示：



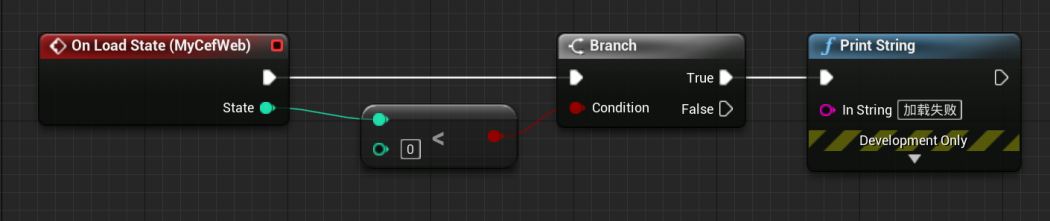
1. 网页加载完毕

画布面板中选中WebView Widget部件。Details->Events绑定On Url Changed事件。并实现该事件。如下图所示：



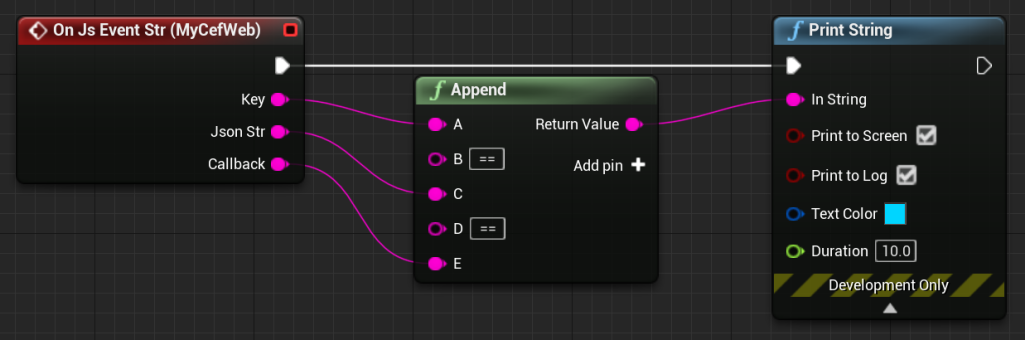
1. 网页加载失败

画布面板中选中WebView Widget部件。Details->Events绑定On Load State事件。并实现该事件。如下图所示：



1. JS调用UE

画布面板中选中WebView Widget部件。Details->Events绑定On Js Event Str事件。并实现该事件。如下图所示：



在JS脚本中引入Resources/jsToUe.js文件。并调用ue4，该函数类型如下：

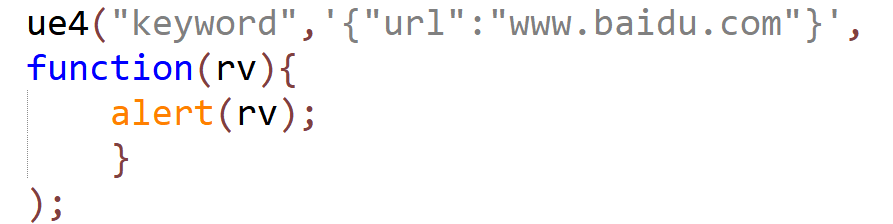
ue4(type,content,callback)

Type: 为标识类型方便ue区分调用。

Content: js传递给ue的参数，一般为json格式。便于多参数传递。

Callback: 当UE处理完毕后的异步回调函数

示例如下:



1. UE调用JS
2. 在Js文件中注册UE调用函数

ue.interface.jsfunc = function( content){}

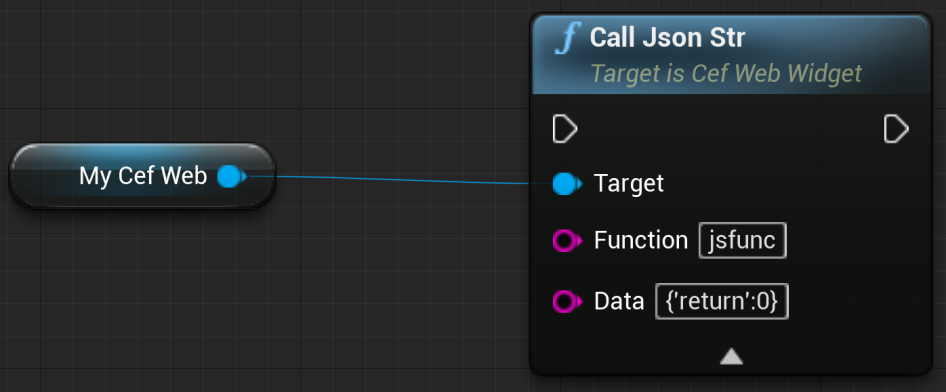
Jsfunc : 为函数名

Content: 接收ue的参数，一般为json。

示例如下:



1. UE发起调用



1. JS调试

开发过程中需要检测js页面的执行情况。需要使用Js调试功能。插件自身提供了Js调试功能，但没有调试界面，需要借助Chrome的调试界面。操作步骤如下:

1. 在项目命令行添加参数 -cefdebug=18080
2. 或者在Game.ini配置文件中增加如下内容

[WebView]

cefdebug=18080

1. 在chrome地址栏中输入<http://localhost:18080>，在出来的页面中选择需要调试的JS页面。
2. 插件配置文件

在defaultgame.ini或者game.ini文件中配置如下内容

[WebView]

DeleteLog**=**true # 自动删除cef日志

cefdebug=18080 # CEF调试js端口

ClearCache=true # 插件重启时自动清理缓存

gpuid=0 # 多显卡时，选择使用GPU 0作为浏览器渲染

cefcore=key=value key2 key3=value3 #浏览器启动参数

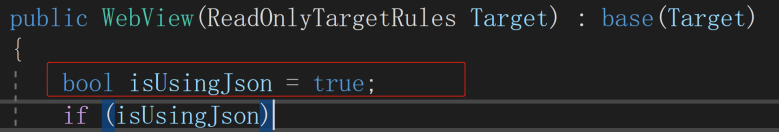
浏览器启动参数

|  |  |  |
| --- | --- | --- |
| 参数 | 参数值 | 说明 |
| enable-webgl |  | 开启webgl渲染 |
| ignore-gpu-blocklist |  | 忽略显卡黑名单 |
| allow-file-access-from-files |  | 允许本地调试webgl |
| unsafely-treat-insecure-origin-as-secure |  | 忽略不安全源 |

1. 兼容JsonLibaray库使用

在Js和UE通信时，可能需要使用JsonLibaray库做为Json解析。考虑到这个库时第三方库，所以代码不方便直接使用，但提供了快速接入JsonLibaray的能力。只需要以下两步即可。

1. 修改WebView.Build.cs文件中 isUsingJson = true;



1. 修改WebView.uplugin 文件中 JsonLibrary插件对应的"Enabled": true

