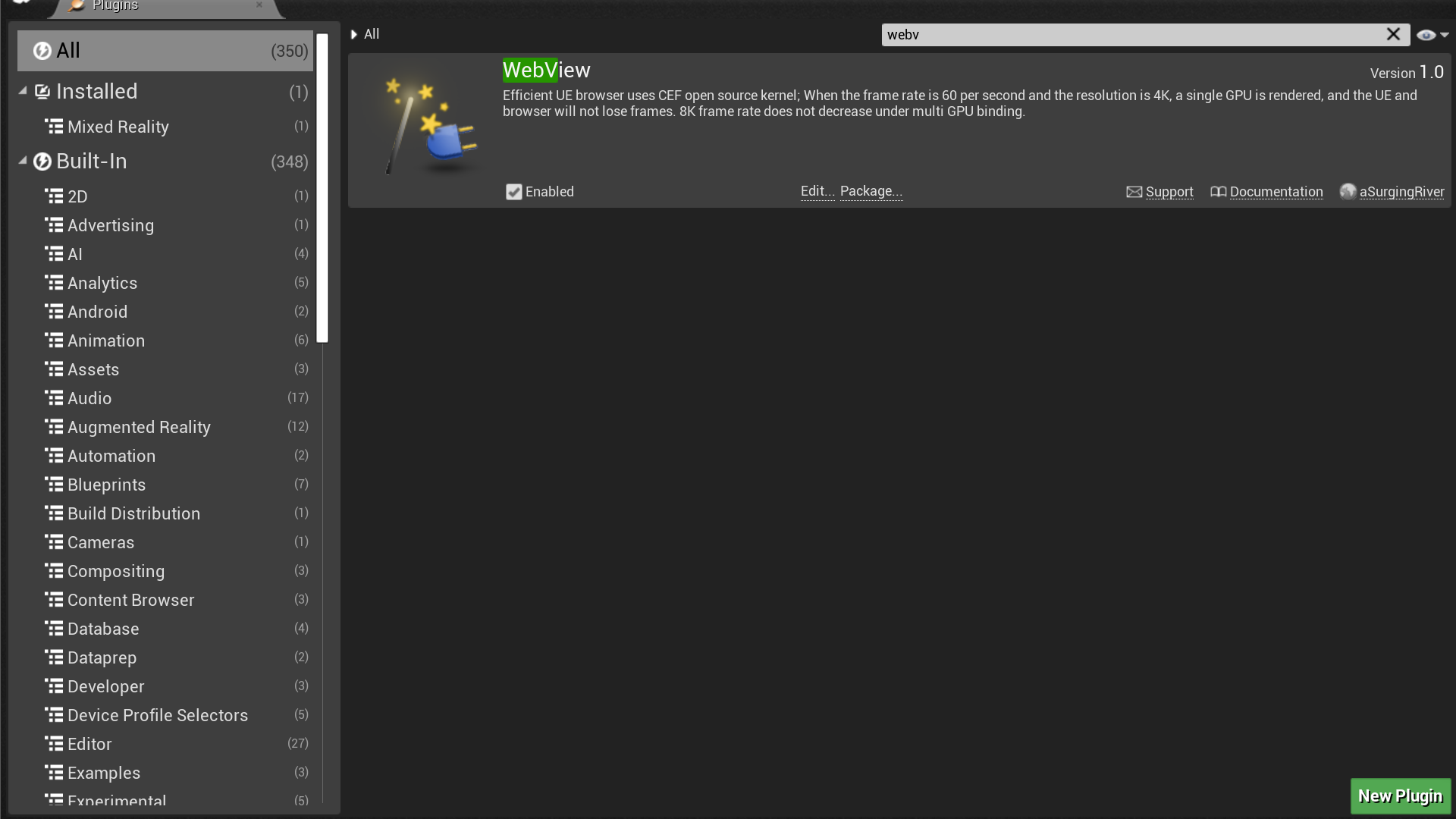
**WebView Plug in description**

WebView is an enterprise browser plug-in designed for the unreal engine. It uses many advanced technologies. You won't get stuck when browsing the web or watching videos. The kernel adopts the latest CEF version and is optimized. It has GPU binding function, supports the separation of game and browser rendering, increases rendering cache inside the plug-in, and the rendering effect is smooth. In terms of communication with web pages, V8 technology is used to make the illusion engine interact with web pages in real time. It is recommended to use the unreal engine to render the scene and JavaScript for UI interaction, which can greatly improve the efficiency of system development.

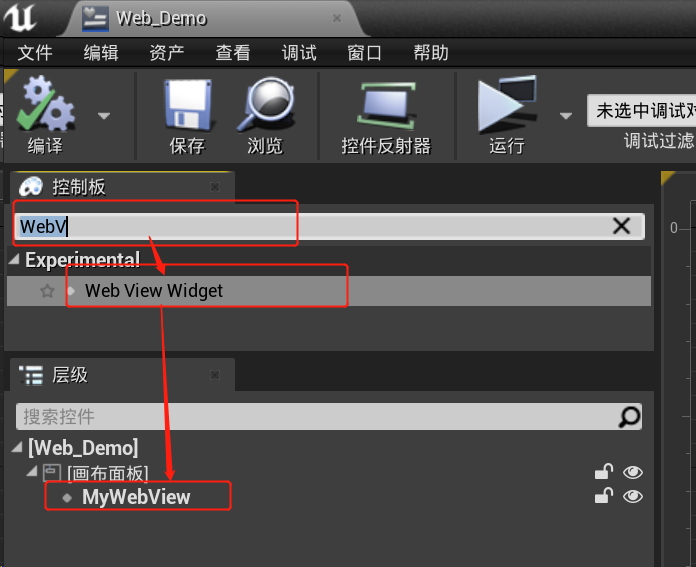
1. Import plug-in

Put the WebView plug-in package into the project plugins directory. Compile the project and ensure that the project is compiled successfully. After running the project, edit - > plug-in, and enter cefweb in the search box to see that the plug-in has been loaded successfully. As shown in the figure below:

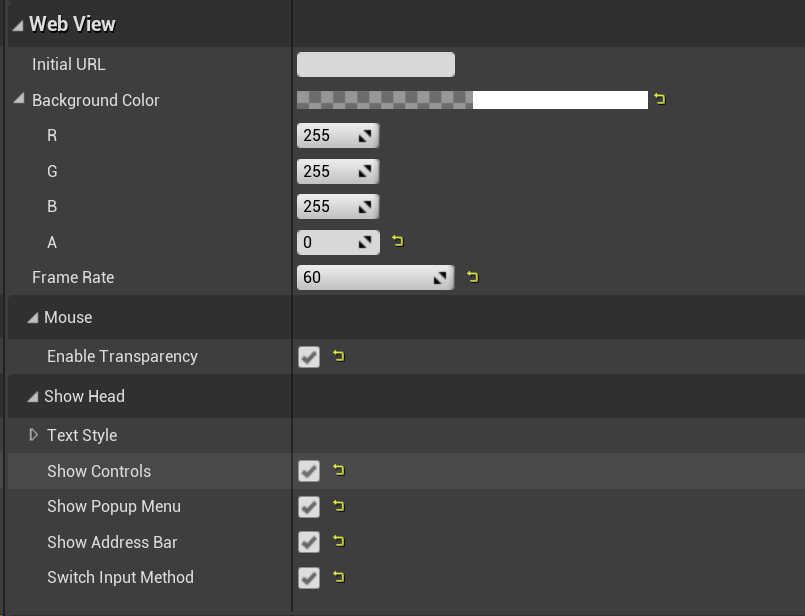


1. Using plug-ins

Create a new UMG blueprint, enter WebV in the palette to see the WebView widget widget widget, drag the widget into the canvas, and adjust the full screen display of the widget in the canvas. As shown in the figure below:



Select the WebView widget part in the canvas and find the Web View configuration item in details. As shown in the figure below:



Configuration Item Description:

Initial URL : the URL where the browser starts to load the web page, such as www.baidu.com

Background Color: controls the background display color of transparent web pages. When alpha is not 255, transparent web page display is supported. The smaller the value, the smaller the penetration of the mouse. If you want most transparency to penetrate, you can increase the value.

Frame Rate: set the frame rate output by the browser between 30 and 60. The larger the value, the smoother the browser display. If playing video is recommended to be set to 60, ordinary web browsing is recommended to be 30. This value is independent of the UE frame rate.

Enable Transparency: controls whether to allow mouse penetration when the web page is transparent, that is, clicking the transparent web page will penetrate the UI or scene below the web page. It is often used in projects where JS is used as UI and UE is used as scene rendering.

Text Style: configure the subsequent address bar, and control the font size and color displayed by the button.

Show Controls: controls whether web pages display control buttons.

Show Popup Menu: controls whether the right-click page displays the navigation menu.

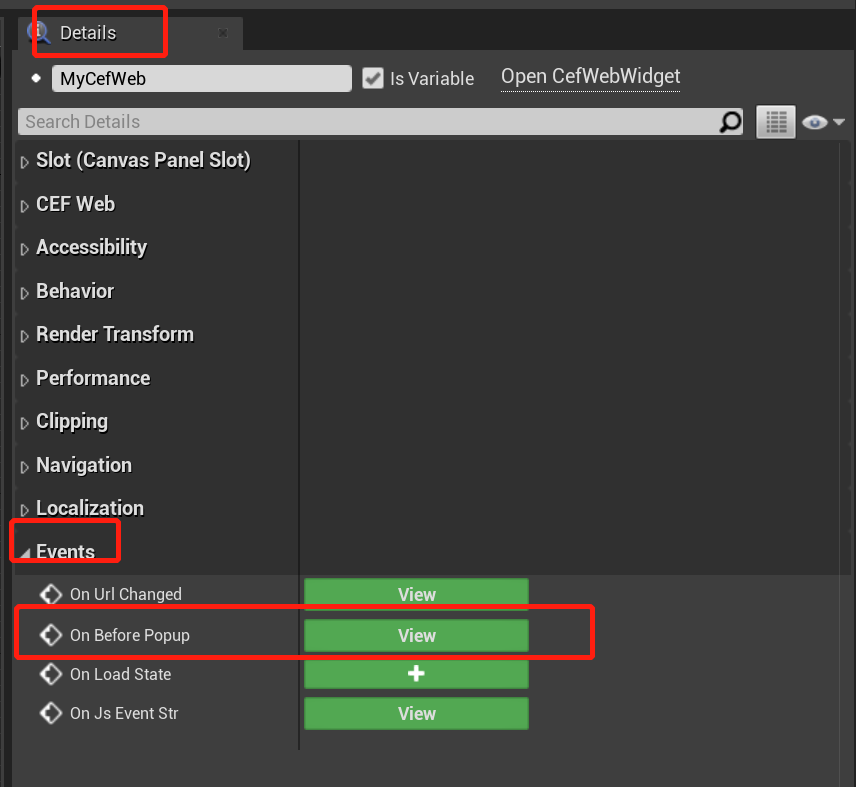
Show Address Bar: controls whether the web page displays the address bar.

Switch Input Method: controls whether the web page supports input method switching.

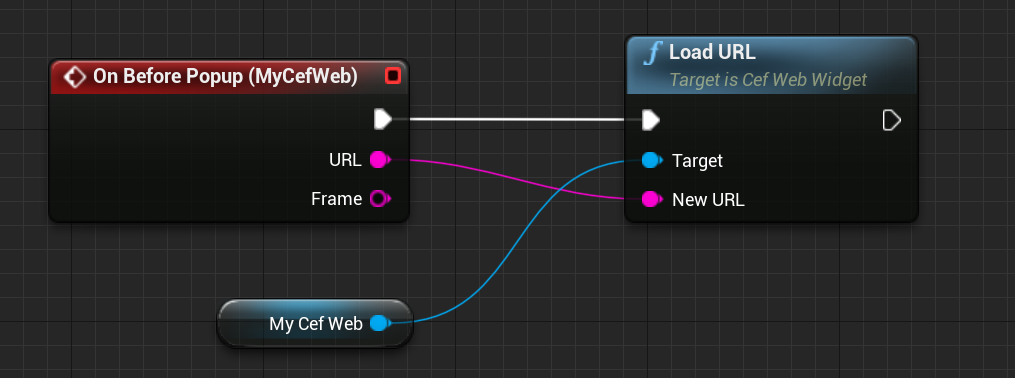
1. Web page Jump

When browsing web pages or Baidu search, jump to new web pages.

Select the Web View widget part in the canvas panel. Details - > events bind the on before popup event.

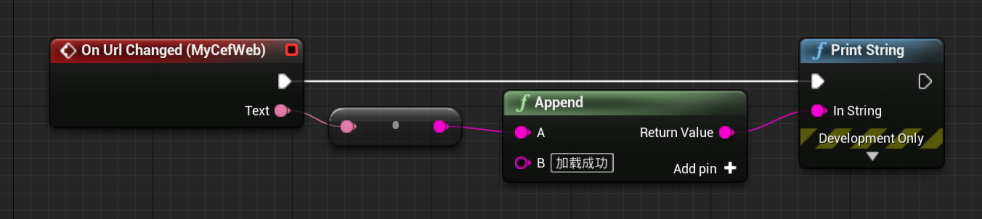


In the implementation of on before popup event, just load a new URL with Web View widget. As shown in the figure below:



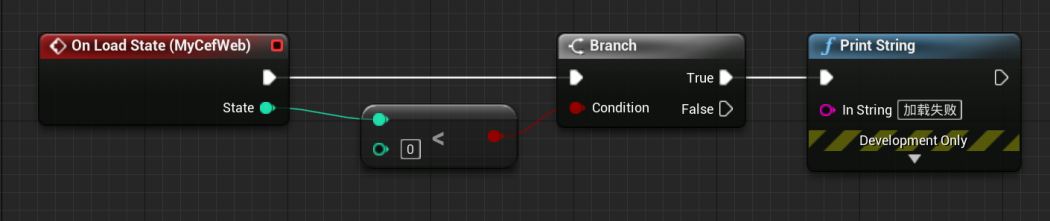
1. Page loading completed

Select the Web View widget part in the canvas panel. Details - > events bind the on URL changed event. And implement the event. As shown in the figure below:



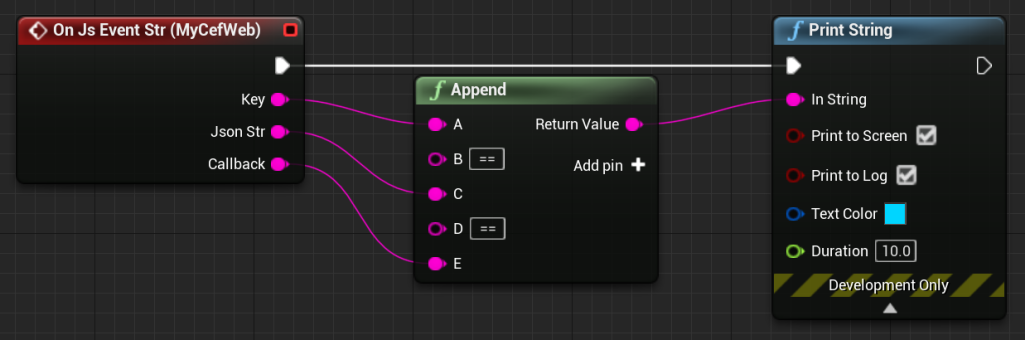
1. Page load failed

Select the Web View widget part in the canvas panel. Details - > events bind the on load state event. And implement the event. As shown in the figure below:



1. JS calls UE

Select the Web View widget part in the canvas panel. Details - > events bind on JS event STR event. And implement the event. As shown in the figure below:



Introduce the resources / jstoue.js file into the JS script. And call UE4. The function type is as follows:

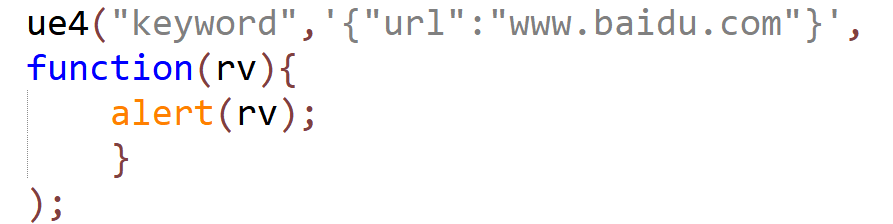
**ue4(type,content,callback)**

Type: To identify the type, it is convenient for ue to distinguish calls.

Content: The parameters passed from JS to UE are generally in JSON format. It is convenient for multi parameter transmission.

Callback: Asynchronous callback function after UE processing

Examples are as follows:



1. UE calls JS
2. Register UE calling function in JS file

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

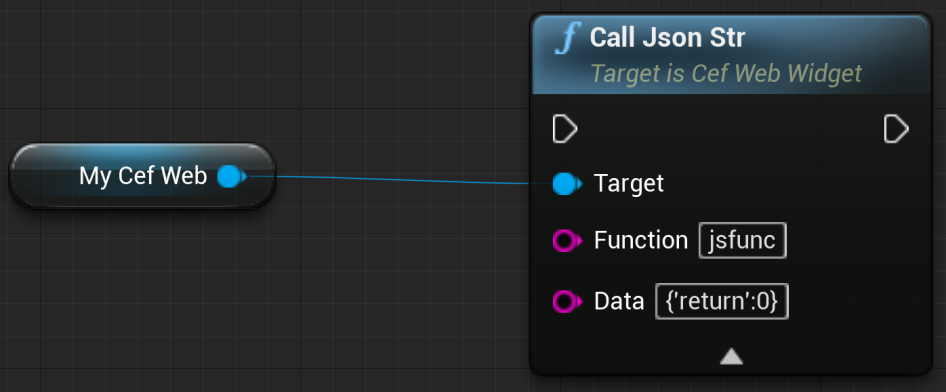
Jsfunc : Function name

Content: Receive the parameters of UE, generally JSON.

Examples are as follows:



1. UE initiates call



1. JS debugging

During the development process, you need to check the execution of JS pages. JS debugging function is required. The plug-in itself provides JS debugging function, but there is no debugging interface. You need to use the debugging interface of chrome. The operation steps are as follows:

1. Add parameters on the project command line cefdebug=18080
2. Or add the following content to the game.ini configuration file

[WebView]

cefdebug=18080

1. Start UE program
2. Enter in the chrome address field http://localhost:18080 , select the JS page to be debugged from the page.
3. Plug in configuration file

Configure the following in the defaultgame.ini or game.ini file

[WebView]

DeleteLog**=**true # Automatically delete CEF logs

cefdebug=18080 # CEF debug JS port

ClearCache=true # Automatically clean cache when plug-in restarts

gpuid=0 # When using multiple graphics cards, select GPU 0 for browser rendering