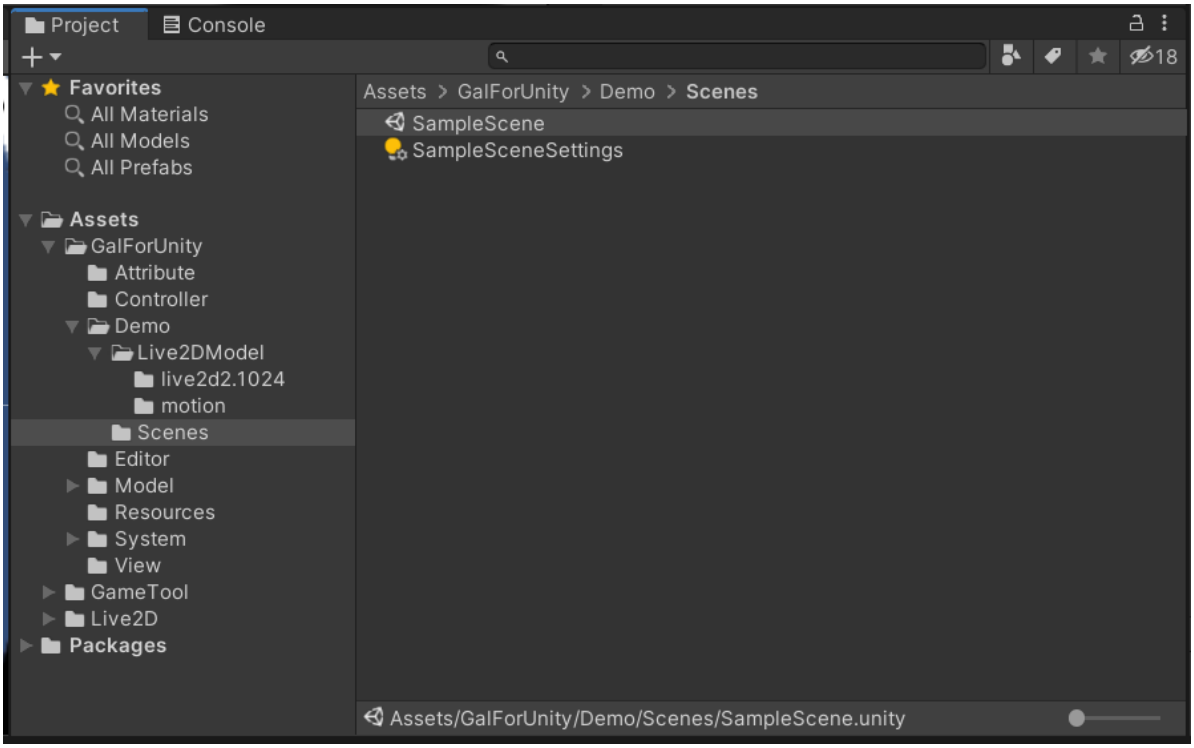
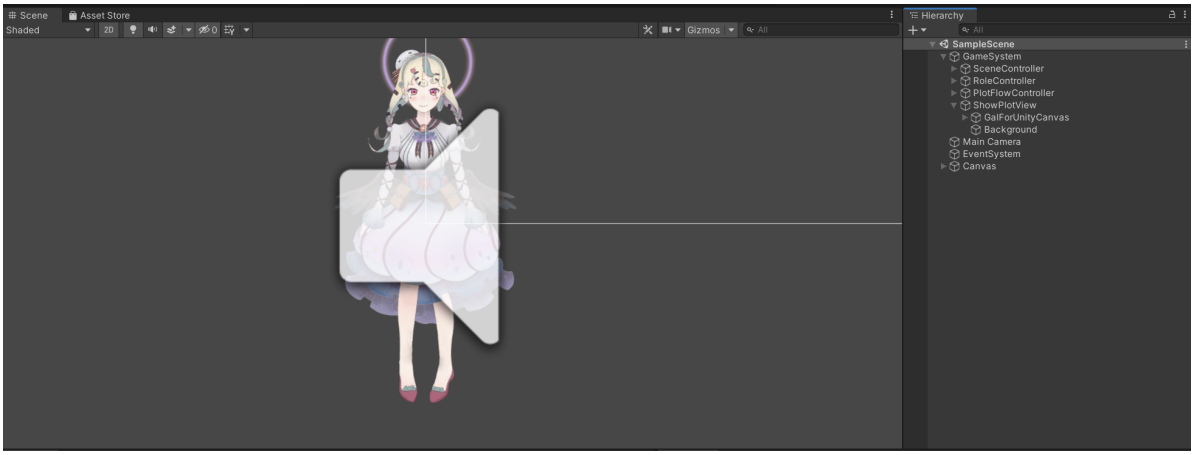
Introduction to

1. Find GalForUnity/Demo/Scenes/SampleScene scenarios, and open the scene (/ scenario is what? (https://docs.unity3d.com/Manual/CreatingScenes.html)),

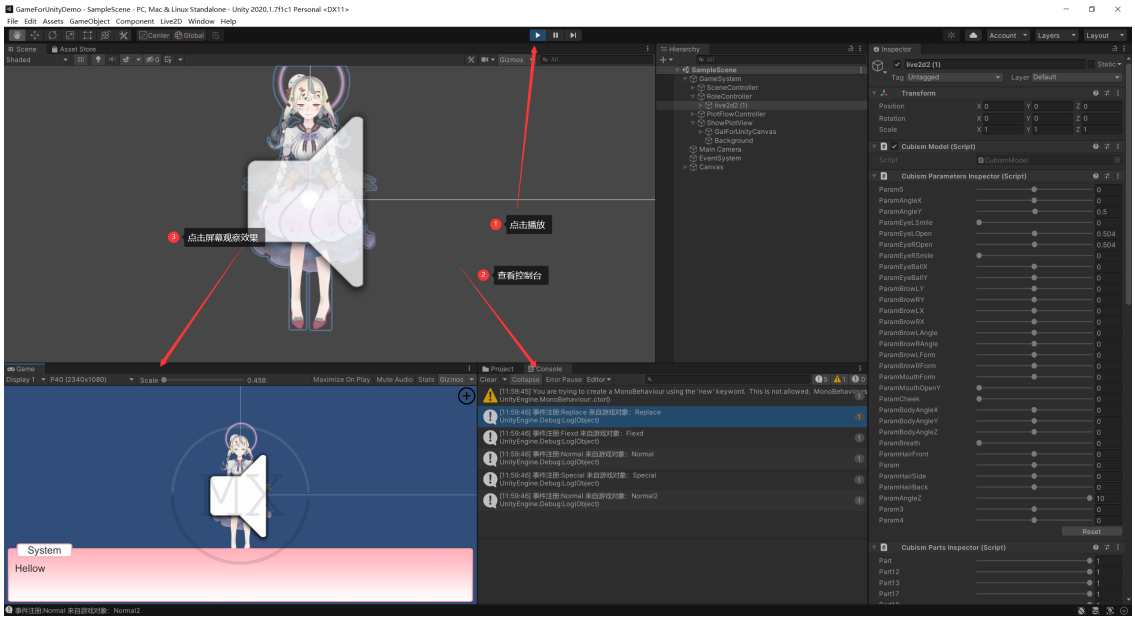


After opening, as shown in the panel [Hierarchy] (https://docs.unity3d.com/cn/current/Manual/Hierarchy.html). For a GalForUnity use case



> ***If the character is not displayed, double-click /RoleController/Live2d2 in Hierarchy and drag the left scroll bar***

2. Click to play directly, and then [Game view (Game)] (https://docs.unity3d.com/Manual/GameView.html) in the click of a mouse can be observed the effect.

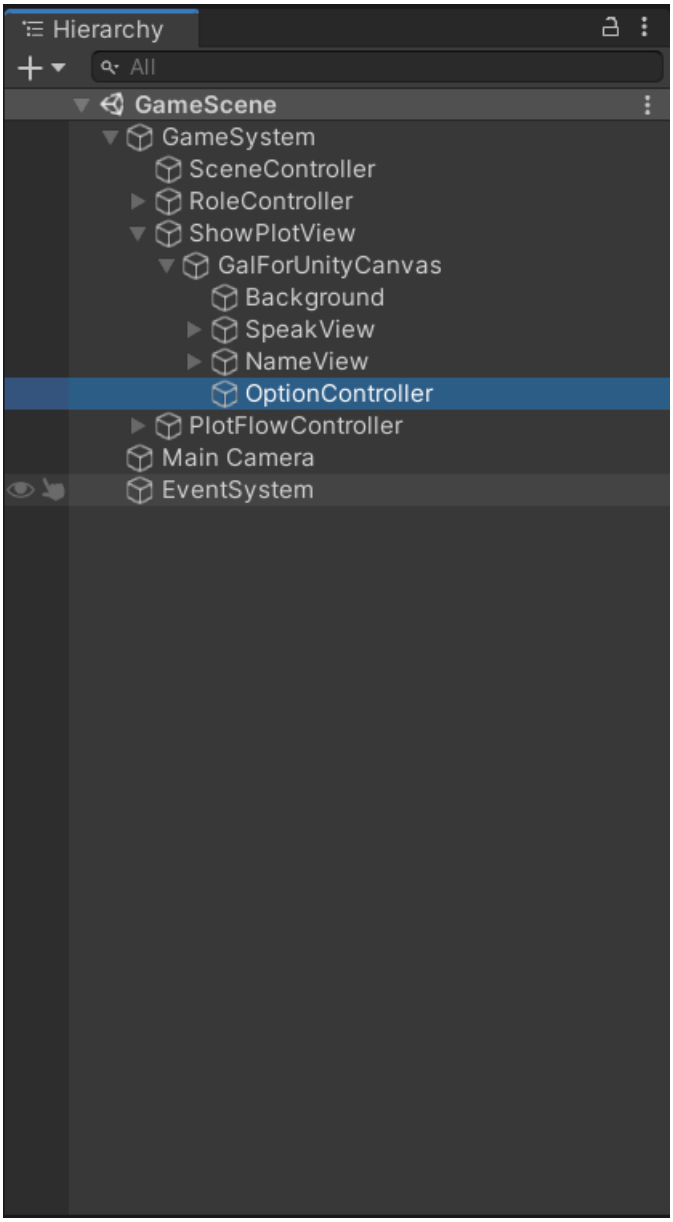
2. 

Frameworks and their use

> The first thing that stands out in the hierarchy view is the GameSystem object. This is the management class of the game system, managing the game from beginning to end

> **SceneController**, this object is the controller of the scene, at the start of the game, all the scene models (SceneModel) will be attached to the controller
>
> **RoleController**, this object is the controller of the character to which all the roleModels are attached at the start of the game
>
> **PlotFlowController**, this object is the controller of the story flow, and at the start of the game, all the plotModels are attached to the controller
>
> **ShowPlotView**, this object is the story view display object, can display the story view on the screen
>
> **OptionController**, which is the OptionController for the story and manages the options that need to be selected by the player

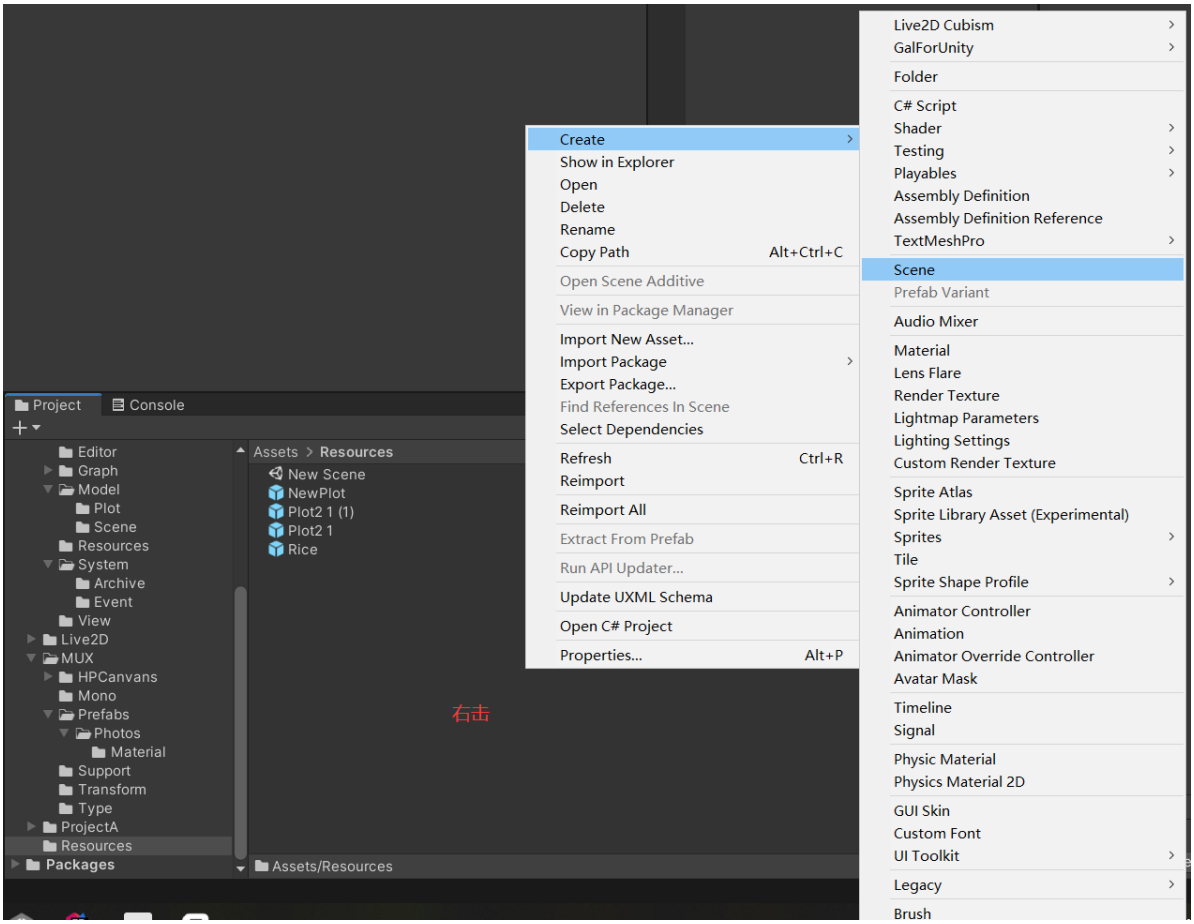
All of the above controllers are coordinated and managed through GameSystem, and are accessible via $GameSystem.Data$. Although the entire game system is automatically initialized at the start of the game, there is still an initialization button in the Inspector view, so that developers can get a good framework for game maintenance and development



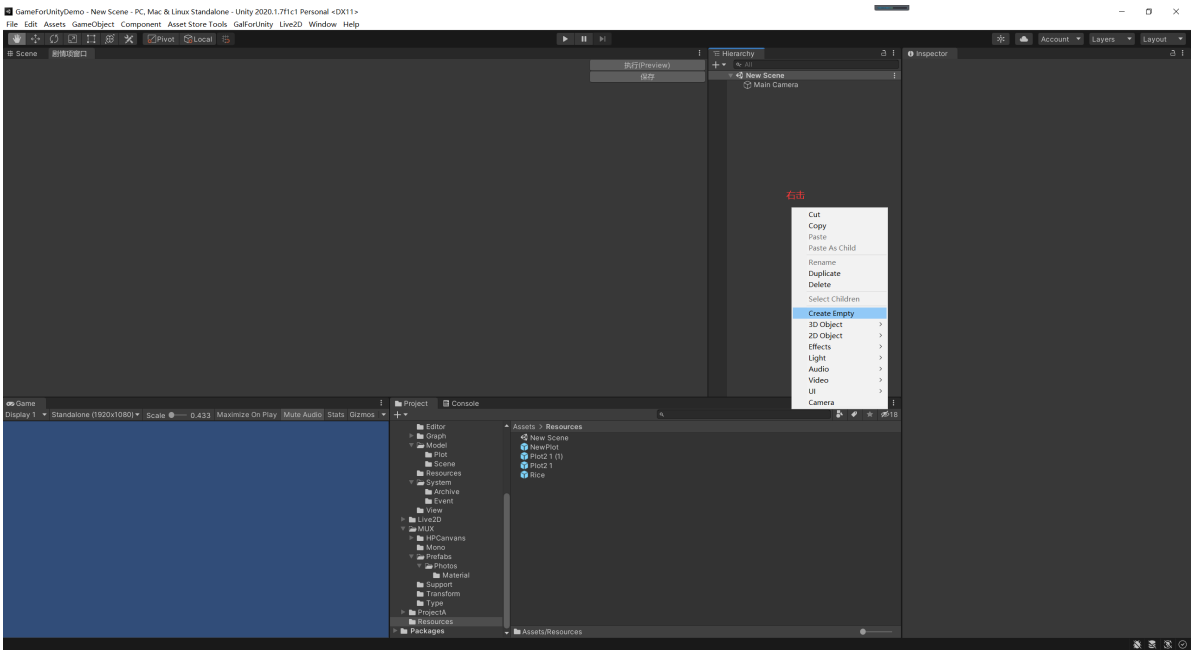
For an empty GameSystem, clicking on the GameSystem initialization button automatically creates and references the corresponding controller, and if the scene already has one, the controller is grabbed by the last GameSystem initialized. But all GameSystem will hold the same controller reference. ~

Do not try to create additional controllers, even though only one of them will be referenced in the end, but a large number of controllers attached to the hierarchical view will cross execute, resulting in unpredictable results

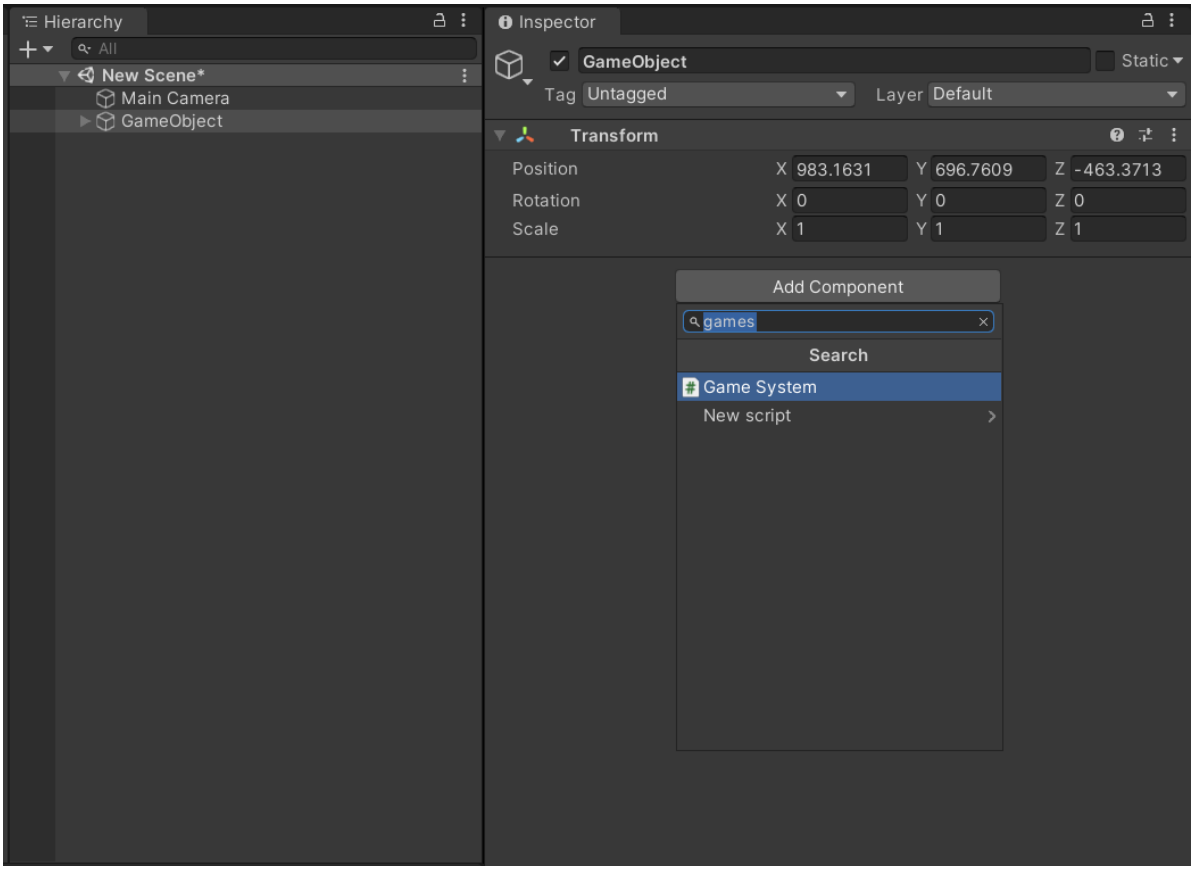
创建新场景



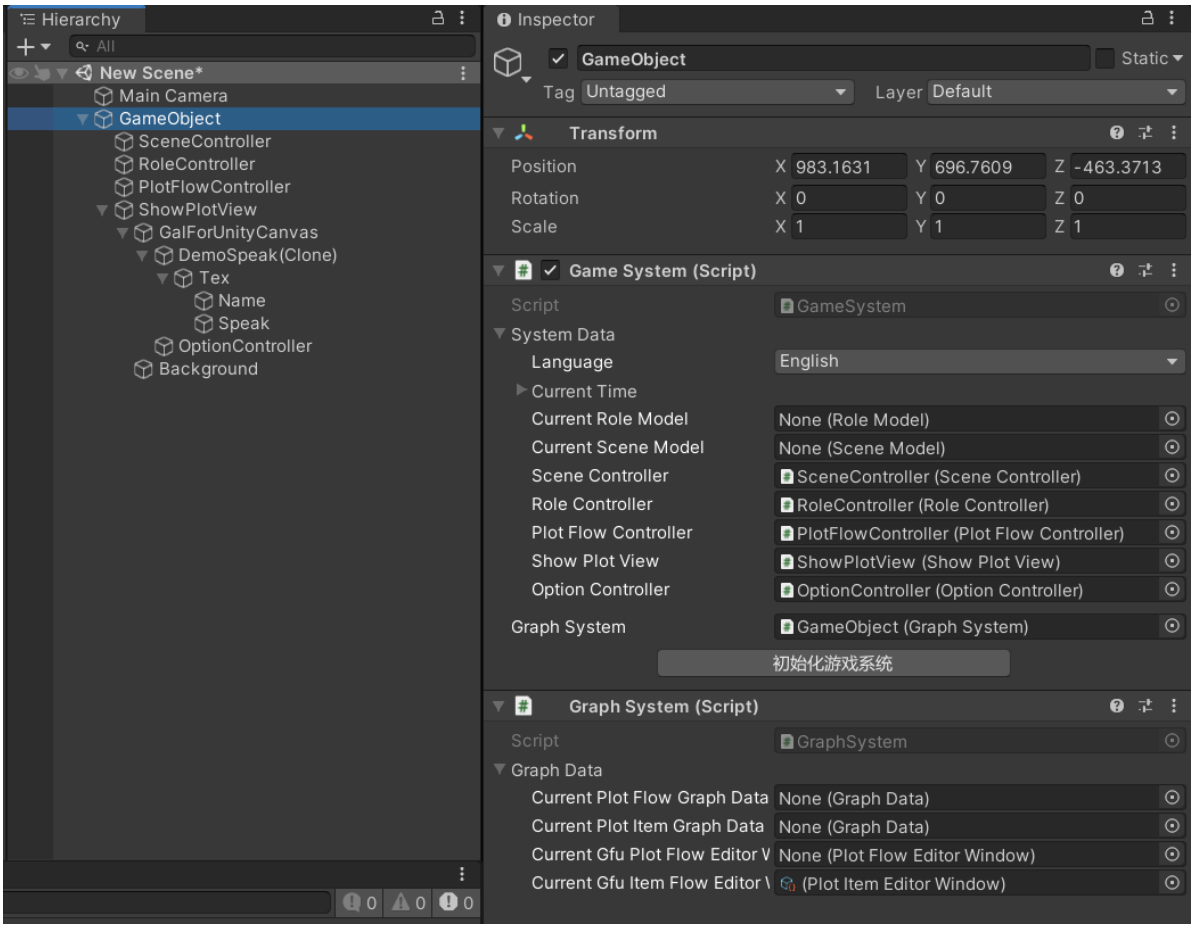
Start by creating a new Scene in the Project directory and double-click to open it.



Right - click in Hierarchy to create a new empty object.



Select the newly created object, click AddComponent, type GameeSystem, click script and add.

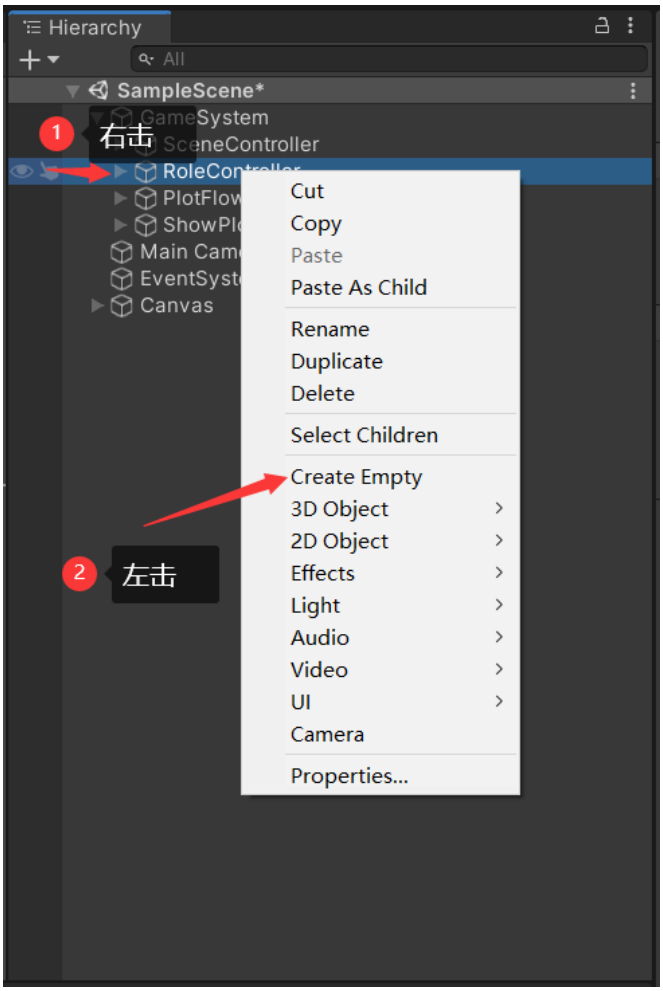


> 1. Select the newly created object, click AddComponent, type GameeSystem, click script and add.

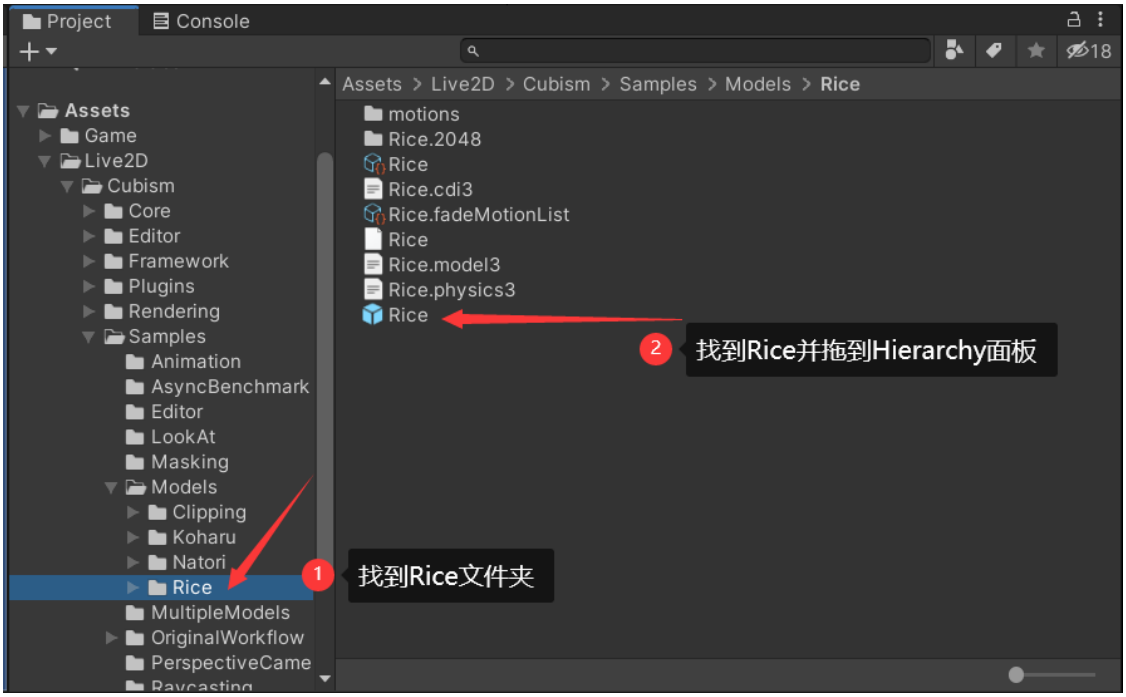
GameSystem automatically initializes the entire system, so you don't have to do this manually. When you have a problem in the future with a messy project, you can make it clean by clicking on the initialization game system.

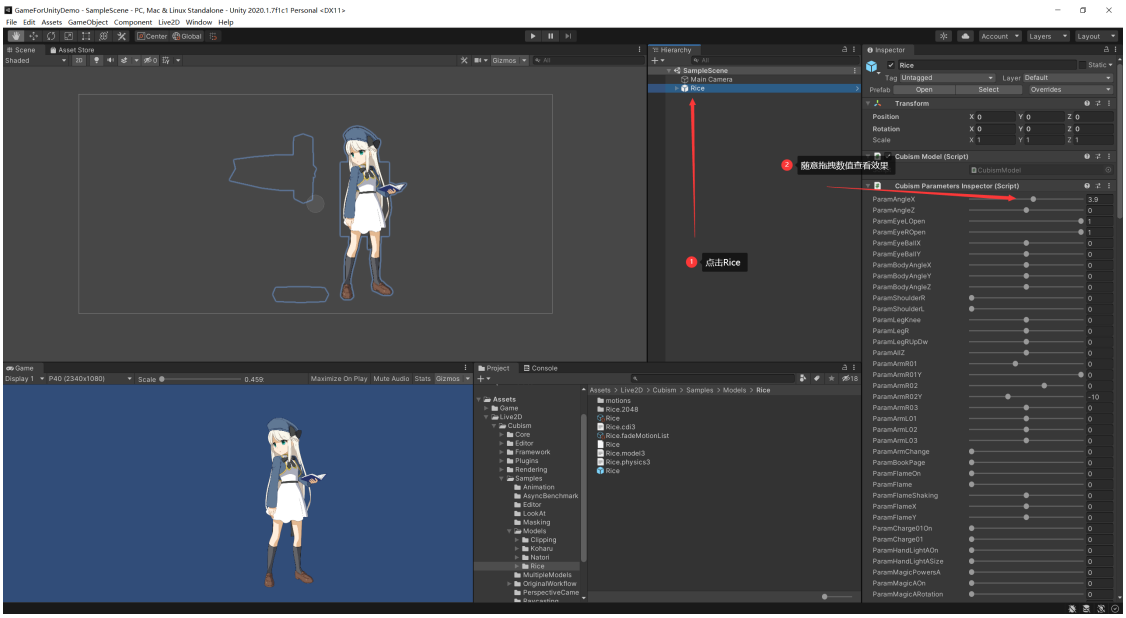
For an empty **GameSystem**, you can either click *initializeGameSystem*and create a new GameObject under the RoleController controller, or drag and drop the Live2D model into the hierarchy view, The RoleModel class is attached

1. Right-click the RoleController in Hierarchy and click Create Empty to Create a New GameObject.

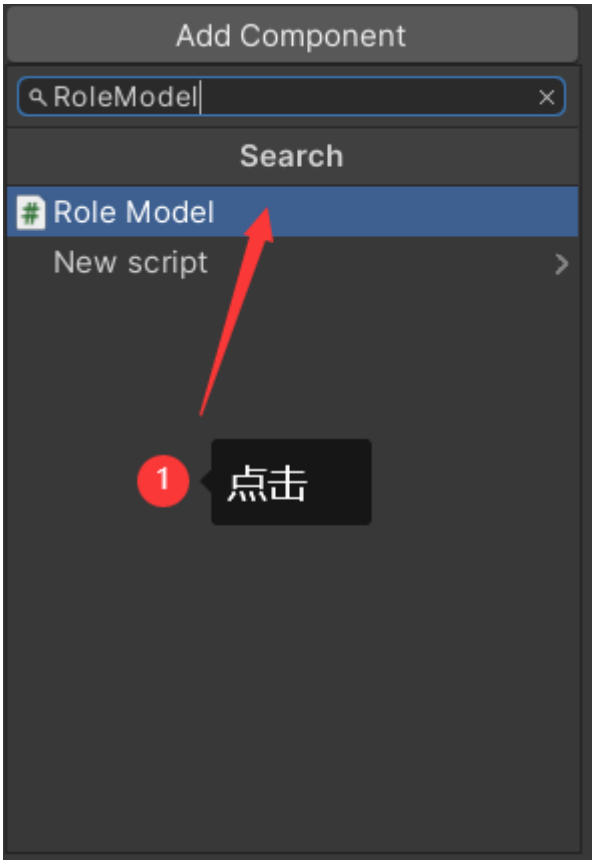


2. It is recommended that you drag and drop from your own model into the view to reduce the number of steps (the Rice model is the default model provided by the Live2D plug-in, GalforUnity does not)



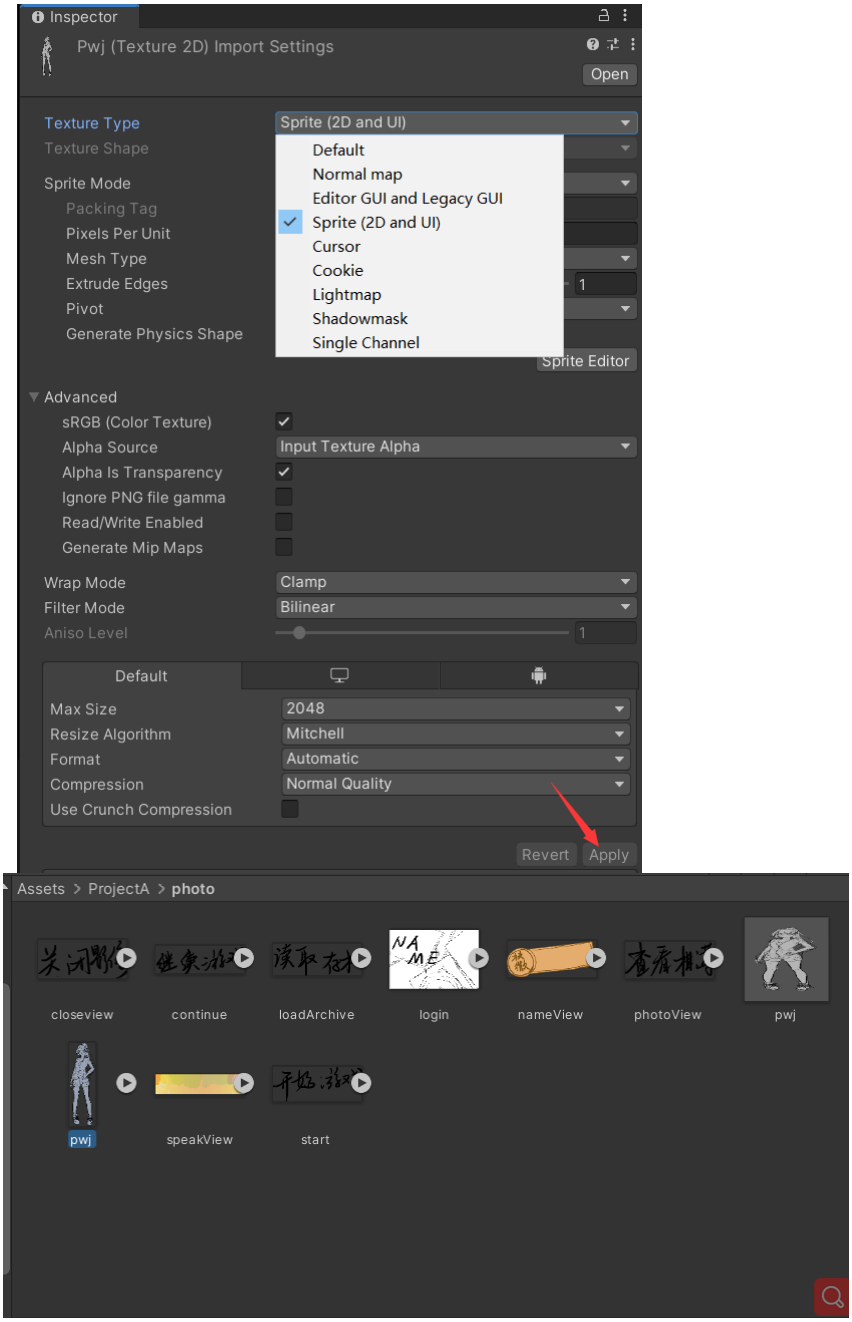


3. Click new GameObject, and at the bottom of the [Inspector view] (https://docs.unity3d.com/Manual/UsingTheInspector.html) click AddComponent input RoleModel add components.

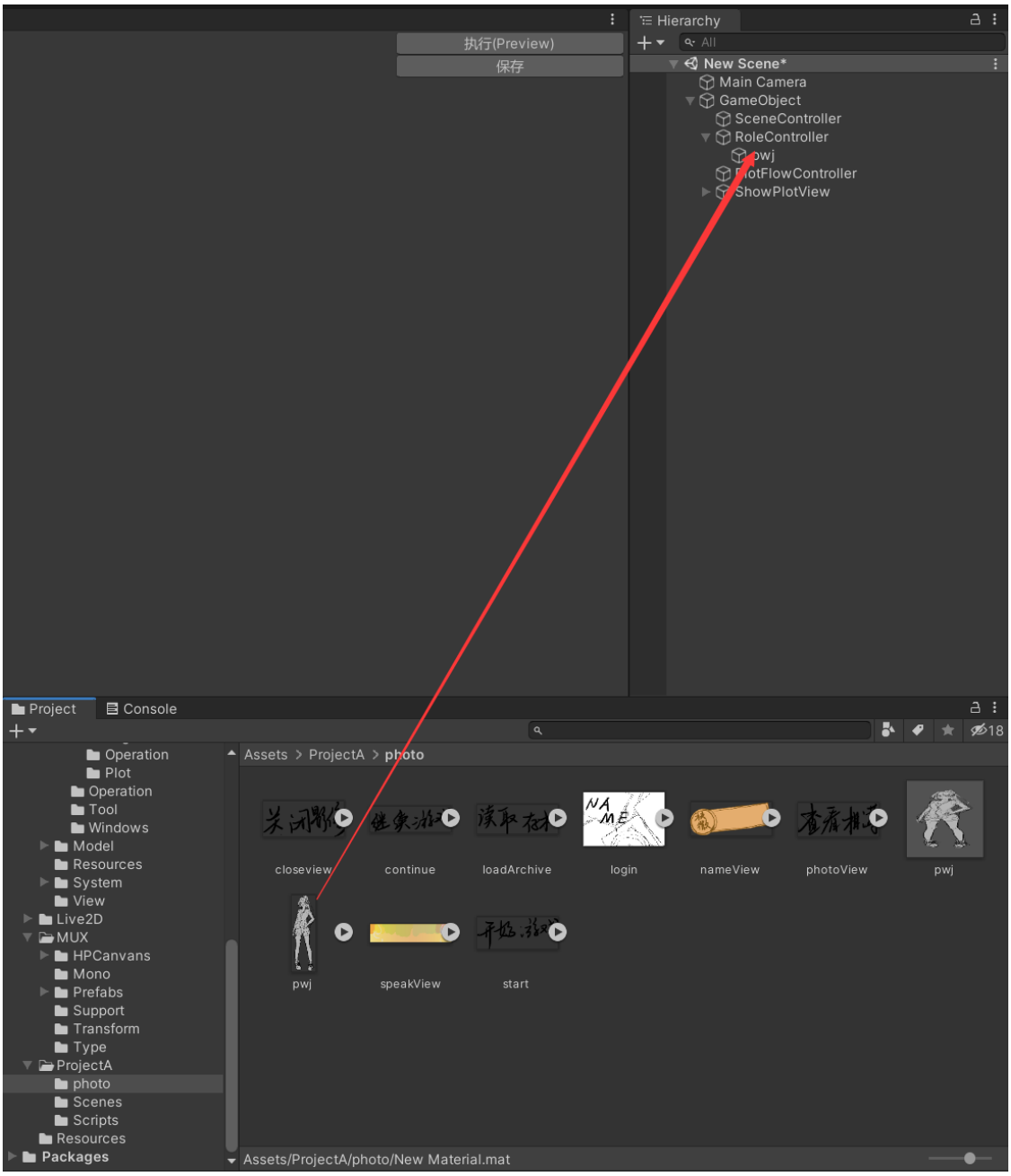


1. If you do not have a Live2D license, you can also use Unity Sprites, or grid renderers. For grid renderers and Unity sprites, you can refer to the Unity manual. This document describes the procedure for adding data.

1. Click Project to select your favorite image. You can drag and drop the image from outside and change TextureType to Sprite in the Inspector view. Click on the application



2. Drag the image under RoleController to see your Sprite



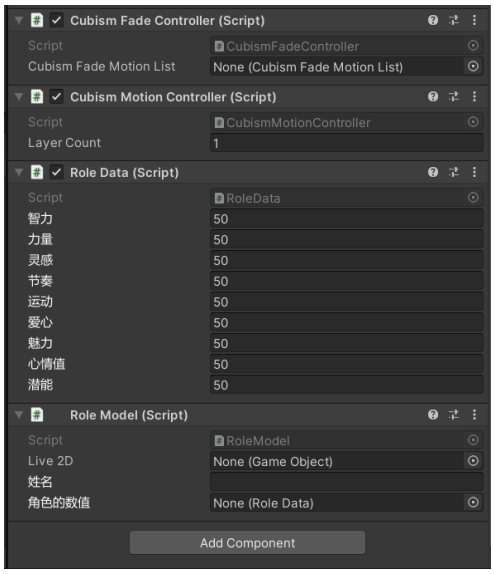
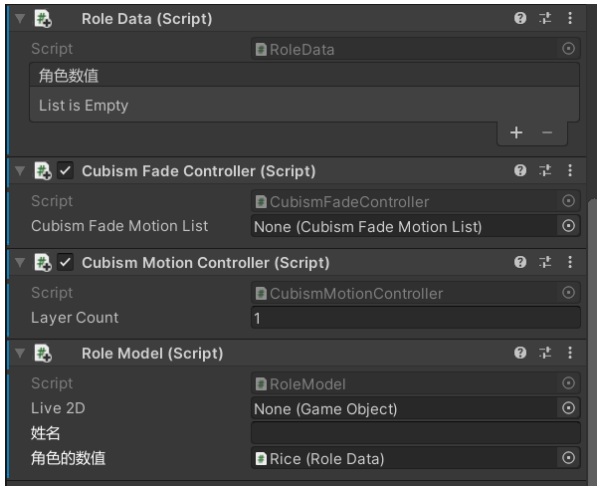
3. Remember to add a RoleModel for your character
4. If the image is too large. Change the size by modifying the Scale property of the Transform component

A reasonable suggestion is to create a new empty object, move the role under the empty object, and change the Scale of the empty object. Controlling the size through the parent component can ensure that the runtime is not affected.

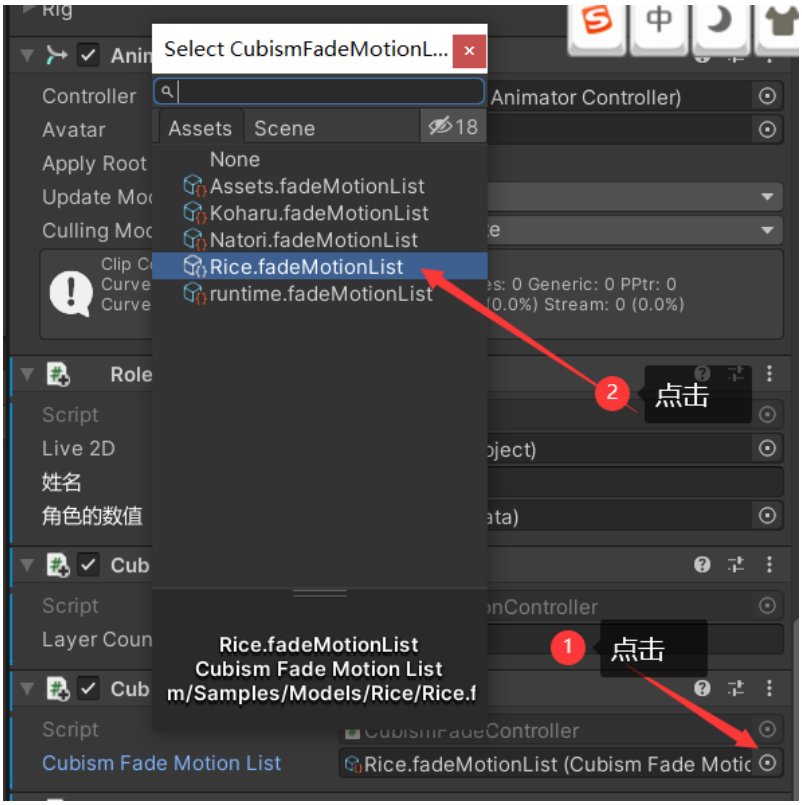
The RoleModel class automatically binds and adds the RoleData class, and when added to the Live2D model automatically adds the CubismFadeController and CubismMotionController (editor mode only). You can change the initial value of the role in this RoleData, and fill in the role name in the RoleModel. If you haven't added the RoleModel to your Live2D model or your Sprite, you

also need to manually attach the model properties ([how to attach? (https://docs.unity3d.com/cn/current/Manual/EditingValueProperties.html))

And in the 1.0R version, we have rewritten RoleData. Compared with the previous fixed data, the latest version of RoleData can support a large number of custom data, just click "+" in the lower right corner of RoleData

优化前	优化后
	

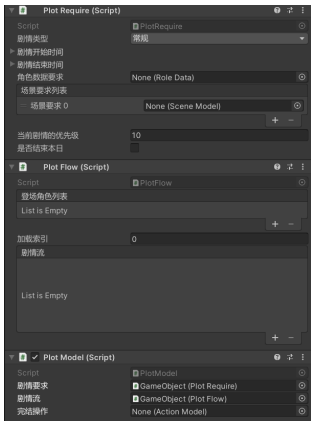
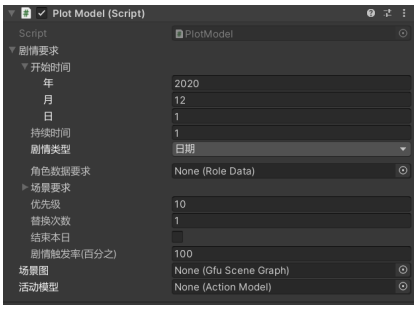
> ***Here we give him a name, for example: Seven Yue; Enter*** in the name property
>
> And click the CubismFadeMotionList button to attach the action resource



We then create a new GameObject in the PlotFlowController and attach the **PlotModel**

PlotModel will automatically append class components from three in 1.0B to two in 1.0R, and will not append scripts in 1.1b, except for invisible GfuInstance, but this does not mean that PlotModel no longer supports control over character data and the like. Instead, This gives developers more freedom to manually control attachments.

PlotFlow and PlotRequire have been rewritten in the latest 1.1r release to optimize the interface, and the **ActionMotion** functionality that was added will soon be overwritten by PlotScript in order to be deprecated

1.0b	1.0r	1.1b
		

Note that PlotRequire has been integrated into PlotModel in the latest release, with some modifications

- **Note that PlotRequire has been integrated into PlotModel in the latest release, with some modifications**

> **RoleData(character data) :** is responsible for recording the story's requirements for character values, all values meet the requirements to trigger the story bound to the RoleData. Textcolor {RedOrange}{it is recommended to define an object group to manage all character requirements. You can also attach character requirements to the story model by referring to}*$

> **PlotFlow:** As the name suggests, a PlotFlow is the plot itself, recording the content of the plot, characters' names, actions, voices, words spoken, background, etc

> **PlotRequire** record the type and requirements of the story, including when the story is allowed to detect the trigger, when to end the detection, and the reference of the management scene requirements (SceneModel) and role requirements (RoleData), etc. If the RoleData and scene requirements are met, the story will be triggered, and the story stream will be executed one by one

> **PlotModel:** holds references to plot requirements and plot flow, and manages plot requirements and plot flow. Generally, no operation is required.

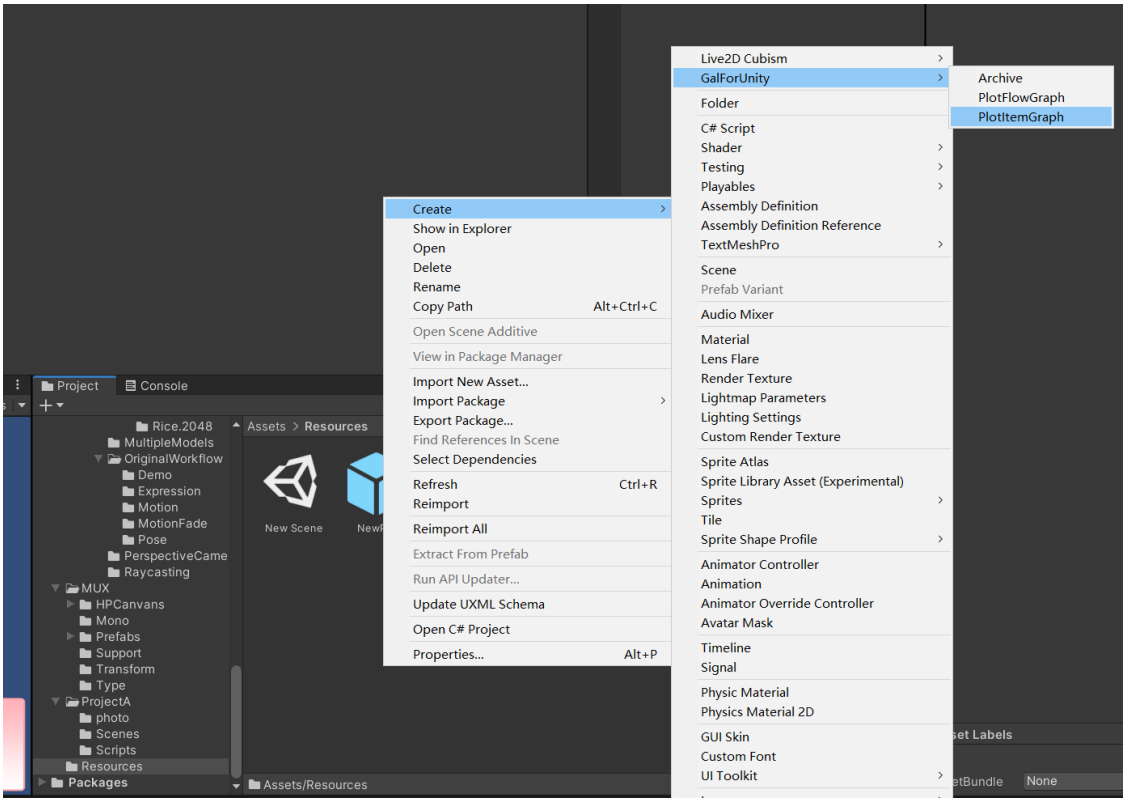
> **ActionModel(ActionModel) :** (obsolete)

Then we manipulate the story flow, attaching Rice to the character, adding the edit story item, and attaching the character animation to play. If you have requirements for scene and role data, you can add them yourself.

Leave everything as default

Now comes the highlight, a look at the new Gal development workflow!

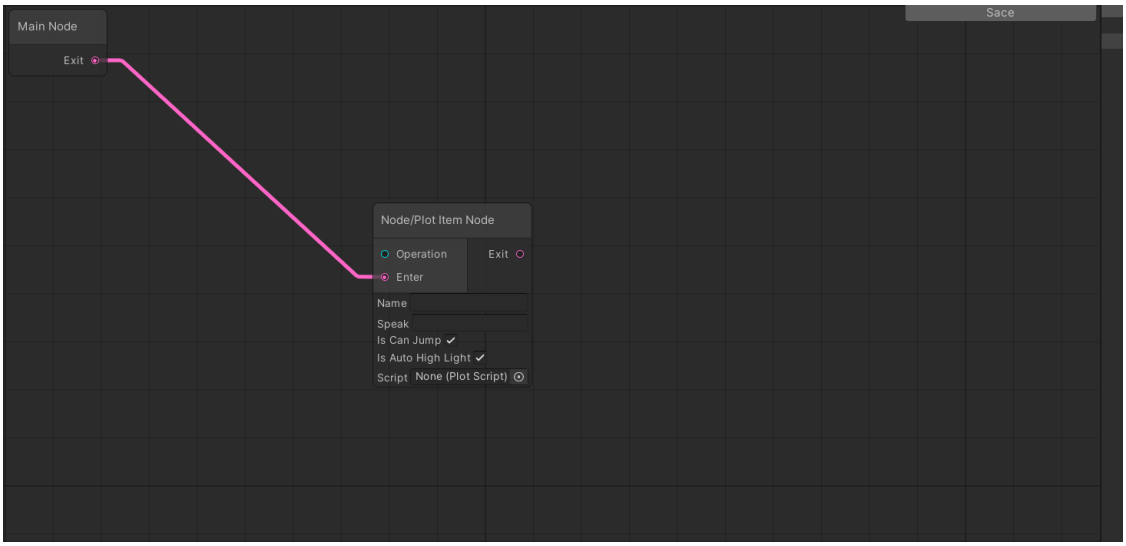
1. Right click on Project and select GalForUnity to create PlotItemGraph



2. Once created, double-click to open it, move the popover to the appropriate TAB or location, and you will see an empty view

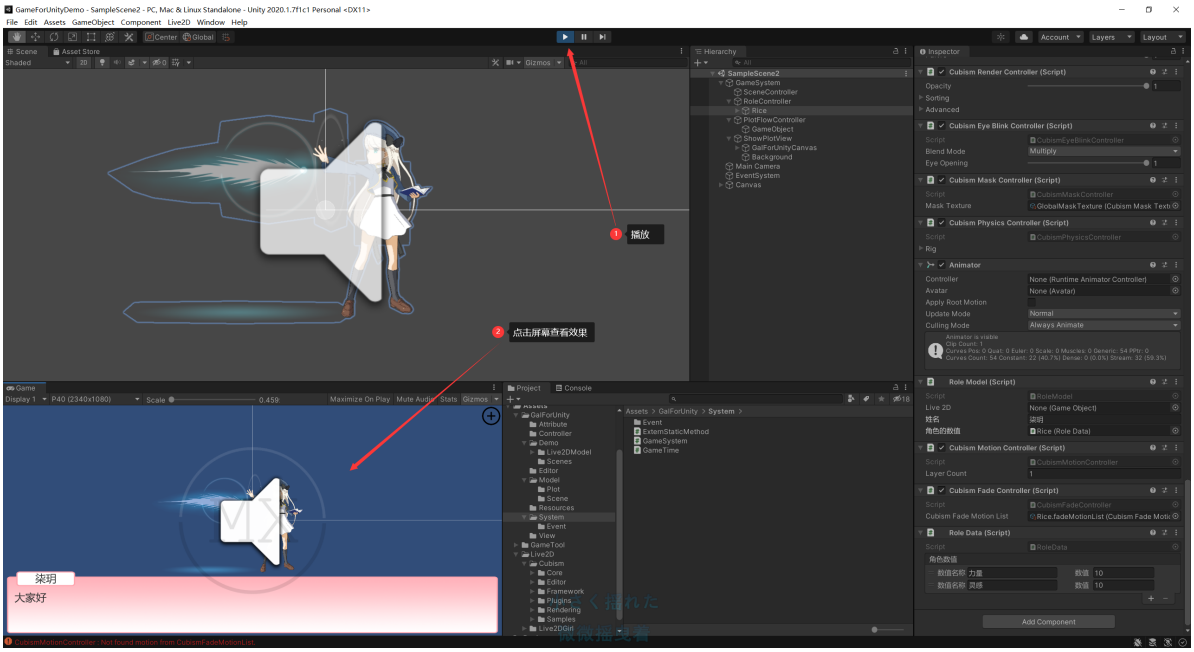
3. ight-click the empty view and click Create Node -> Node -> Plot Item Node to Create a new Node and enter the name and statement to display. Finally, connect the new Node

to the master Node. **Click Save**.



4. Finally, create a GfuSceneGraph from the plot Graph in the PlotModel you created earlier and drag the nodes into the Graph

When everything is ready, click the Play button in the middle of the screen and then click the Game view to see what it looks like.



What? The character disappears? That is because the character has not yet made an entrance....

What? How do you get in? Take a look at the node system! Tip: Operation -> Role Node -> Operation type

[And you can look Vedio]