<http://www.xuanyusong.com/archives/3229>

<http://www.cnblogs.com/lexiaoyao-jun/p/5208249.html>

**Assets**

**与脚本编译顺序相关的文件夹**

在unity开发中，我们可以为自己的项目随意命名文件夹以管理游戏资源，包括代码资源。但unity保留了一些特殊文件夹用来做特殊用途，例如编译顺序。  
unity的脚本编译有4个阶段（phase），脚本处在哪个编译阶段取决于脚本所在的文件夹。如果你的一些脚本需要引用一些别的文件夹中定义的类，则需要关心他们的编译顺序。你引用的类需要先于你的当前类编译。或者当你需要引用其他语言的脚本时，那么该脚本必须处于更早的编译阶段。  
unity中的4个编译阶段如下：  
1.处于 **Standard Assets**, **Pro Standard Assets** 和 **Plugins** 文件夹中的运行时脚本（Standard Assets 文件夹需是Assets的一级子文件夹.）  
2.处于 **Standard Assets**, **Pro Standard Assets** 和 **Plugins** 文件夹下的以 **Editor** 命名的一级子文件夹中的脚本。  
3.顶级 **Editor** 文件夹中的脚本。  
4.其他 **Editor** 文件夹中的脚本（例如其他文件夹下的以 **Editor** 命名的子文件夹）  
另外在 **Assets** 文件夹中以 **WebPlayerTemplates** 命名的顶级子文件（即Assets/WebPlayerTemplates）将不会编译。若是处于别的文件夹下的 **WebPlayerTemplates** （如Assets/Scripts/WebPlayerTemplates）将不会防止编译。

## 注意：

**Editor** 文件夹中的脚本主要用来扩展unity编辑器的功能方便开发。这些脚本将不会打包进最终发布的游戏中。项目中可以使用多个 **Editor** 文件夹，但是该文件夹中的脚本不允许用当GameObject对象的组件（Component）。

**Plugins** 文件夹中存放用于扩展unity功能的插件（多为C/C++写成的原生动态链接库(DLLs)）。这些插件可以访问第三方代码库，系统API以及其他超出Unity功能的模块。

# Editor Default Resources

我们使用 **Editor** 文件夹中的脚本扩展unity编辑器的功能时，可以使用函数[EditorGUIUtility](http://docs.unity3d.com/ScriptReference/EditorGUIUtility.html).Load 加载资源。该函数将优先加载Assets下的以 **Editor Default Resources** 命名的一级子目录。如果没有找到将尝试通过名字查找内置于编辑器中的资源。

1. 隐藏文件夹  
   以.开头的文件夹会被Unity忽略。在这种文件夹中的资源不会被导入，脚本不会被编译。也不会出现在Project视图中。

2.Standard Assets  
在这个文件夹中的脚本最先被编译。  
这个文件夹中的脚本会被导出到Assembly-CSharp-firstpass, Assembly-UnityScript-firstpass 或 Assembly-Boo-firstpass项目中，依语言而定。参考[http://docs.unity3d.com/Documentation/Manual/ScriptCompileOrderFolders.html](http://jump.bdimg.com/safecheck/index?url=x+Z5mMbGPAvciY82xyiNlAjeBmjilxKnjHUZ8hq2uu7RrsEvKtQRij5w5KTeMCk5rRR0mWV3IgAKLhWqRFWDVegh4H43SKViWSZL/IaEUF/nHkQnVaxvbF9sQA2TXYlBq70iNrV/lENKIJqFbS9kcrBYKb7ILcJVXCW9gg1D84gzH2pu4wBbbdJvKYZ2lcp/1nrRRGB9If12Pah4egTNMg==) 。在这个文件夹中的脚本比其他脚本都要先编译。将脚本放在这个文件夹里，就可以用C#脚本来访问js脚本或其他语言的脚本。

3.Pro Standard Assets  
跟Standard Assets相同，只不过里面的文件是给Pro版本的Unity使用的。

4.Editor  
以Editor命名的文件夹允许其中的脚本访问Unity Editor的API。如果脚本中使用了在UnityEditor命名空间中的类或方法，它必须被放在名为Editor的文件夹中。Editor文件夹中的脚本不会在build时被包含。  
在项目中可以有多个Editor文件夹。  
注意：如果在普通的文件夹下，Editor文件夹可以处于目录的任何层级。如果在特殊文件夹下，那Editor文件夹必须是特殊文件夹的直接子目录。

5.Plugins  
Plugins文件夹用来放native插件。它们会被自动包含进build中去。注意这个文件夹只能是Assets文件夹的直接子目录。  
在Windows平台下，native 插件是dll文件；Mac OS X下，是bundle文件；Linux下，是.so文件。  
跟Standard Assets一样，这里的脚本会更早的编译，允许它们被之外的脚本访问。  
5.1.Plugins/x86  
如果为32bit或64bit平台创建游戏，那么这个文件夹下的native plugin文件会被自动的包含在游戏build中。如果这个文件夹不存在，则Unity会查找Plugins文件夹下的native pluglins。  
5.2.Plugins/x86\_64  
如果为32bit或64bit平台创建游戏，那么这个文件夹下的native plugin文件会被自动的包含在游戏build中。如果这个文件夹不存在，则Unity会查找Plugins文件夹下的native pluglins。  
  
如果要创建universal build，建议你同时使用这两个文件夹。然后将32bit和64bit的native plugins放进相应的文件夹中。  
5.3.Plugins/Android  
在这个文件夹里放入Java.jar文件。用于java语言的plugins。.so文件也会被包含进来。参考[http://docs.unity3d.com/Documentation/Manual/PluginsForAndroid.html](http://jump.bdimg.com/safecheck/index?url=x+Z5mMbGPAvciY82xyiNlAjeBmjilxKnjHUZ8hq2uu7RrsEvKtQRij5w5KTeMCk5rRR0mWV3IgBUVMj6gNF1N/AUVK3RZV/6d1HeoQXiVNBsEmICBz/viPyyEez8/Qo7SiCahW0vZHKwWCm+yC3CVVwlvYINQ/OIMx9qbuMAW23SbymGdpXKf9Z60URgfSH9dj2oeHoEzTI=)  
5.4.Plugins/iOS  
A limited, simple way to automatically add (as symbolic links) any .a, .m, .mm, .c, or .cpp files into the generated Xcode project. See[http://docs.unity3d.com/Documentation/Manual/PluginsForIOS.html](http://jump.bdimg.com/safecheck/index?url=x+Z5mMbGPAvciY82xyiNlAjeBmjilxKnjHUZ8hq2uu7RrsEvKtQRij5w5KTeMCk5rRR0mWV3IgBUVMj6gNF1N/AUVK3RZV/68DbAcQ5sUvervSI2tX+UQ0ogmoVtL2RysFgpvsgtwlVcJb2CDUPziDMfam7jAFtt0m8phnaVyn/WetFEYH0h/XY9qHh6BM0y)  
If you need more control how to automatically add files to the Xcode project, you should make use of the PostprocessBuildPlayer feature. Doing so does not require you to place such files in the Plugins/iOS folder. See[http://docs.unity3d.com/Documentation/Manual/BuildPlayerPipeline.html](http://jump.bdimg.com/safecheck/index?url=x+Z5mMbGPAvciY82xyiNlAjeBmjilxKnjHUZ8hq2uu7RrsEvKtQRij5w5KTeMCk5rRR0mWV3IgDNfaf+9EifWtgS7/nJiyDGrX+l9jJ9vzBEXSLXQqXdXau9Ija1f5RDSiCahW0vZHKwWCm+yC3CVVwlvYINQ/OIMx9qbuMAW23SbymGdpXKf9Z60URgfSH9dj2oeHoEzTI=)  
  
  
6.Resources  
Resources文件夹允许你在脚本中通过文件路径和名称来访问资源。但还是推荐使用直接引用来访问资源。  
放在这一文件夹的资源永远被包含进build中，即使它没有被使用。因为Unity无法判断脚本有没有访问了其中的资源。  
项目中可以有多个Resources文件夹，因此不建议在多个文件夹中放同名的资源。  
一旦build游戏，Resources文件夹中的所有资源被打包进游戏存放资源的archive中。这样在游戏的build中就不存在Resources文件夹了。即使脚本中仍然使用了资源在项目中的路径。参考 [http://docs.unity3d.com/Documentation/Manual/LoadingResourcesatRuntime.html](http://jump.bdimg.com/safecheck/index?url=x+Z5mMbGPAvciY82xyiNlAjeBmjilxKnjHUZ8hq2uu7RrsEvKtQRij5w5KTeMCk5rRR0mWV3IgAS3yRKlqycocZRd+n6Yf84P446pdyig5+THpkTRlkW2FNUCC4vf3fBq70iNrV/lENKIJqFbS9kcrBYKb7ILcJVXCW9gg1D84gzH2pu4wBbbdJvKYZ2lcp/1nrRRGB9If12Pah4egTNMg==)  
注意：当资源作为脚本变量被访问时，这些资源在脚本被实例化后就被加载进内存。如果资源太大，你可能不希望它被这样加载。那么你可以将这些大资源放进Resources文件夹中，通过Resources.Load来加载。当不再使用这些资源了，可以通过Destroy物体，再调用Resources.UnloadUnusedAssets来释放内存。

7.Editor Default Resources  
这是为editor 脚本使用的文件夹。

8.Gizmos  
Gizmos文件夹存放用Gizmos.DrawIcon方法使用的贴图、图标资源。放在Gizmos文件夹中的贴图资源可以直接通过名称使用，可以被Editor作为gizmo画在屏幕上。

9.WebPlayerTemplates  
用来替换web build的默认网页。这个文件夹中的脚本都不会被编译。这个文件夹必须作为Assets文件夹的直接子目录。

10.StreamingAssets  
这里的文件会被拷贝到build文件夹中，不会修改（移动和网页版不同，他们会被嵌入到最终build文件中）。它们的路径会因平台而有差异，但都可以通过Application.streamingAssetsPath来访问。  
参考

[http://docs.unity3d.com/Documentation/Manual/StreamingAssets.html](http://jump.bdimg.com/safecheck/index?url=x+Z5mMbGPAvciY82xyiNlAjeBmjilxKnjHUZ8hq2uu7RrsEvKtQRij5w5KTeMCk5rRR0mWV3IgDPpotONP6odw6vFdw8nnXiaFT9lfxnaZ1B49q5Oh8l1U4OZxGBDw0W1Wzh7cSZH/2lFAYxIe2sYWkEU4+zhN1XULSheTE8wXwrqSrazPl9EZuQye2MG4bWMDxm7iZ2BjQ=)

和

[http://docs.unity3d.com/Documentation/ScriptReference/Application-streamingAssetsPath.html](http://jump.bdimg.com/safecheck/index?url=x+Z5mMbGPAvciY82xyiNlAjeBmjilxKnjHUZ8hq2uu7RrsEvKtQRij5w5KTeMCk59VmCQnX79Up8q/iF4q/IcJTHlkYakSmmPW00btLnu9QSxZVPK2iaH76NOnu15f8jKc80+XVbgfMjGjhXeAe9D26pl+6ZZnYrTg5nEYEPDRbVbOHtxJkf/aUUBjEh7axhaQRTj7OE3VdQtKF5MTzBfCupKtrM+X0Rm5DJ7YwbhtYwPGbuJnYGNA==" \t "_blank)。