集中式和分布式的版本控制系统（VCS）

集中式：安全、必须连接到服务器才能提交(Perforce)

分布式：github的方式，用户克隆仓库到本地（git）

Plastic SCM既支持集中式，也支持分布式

命名标准

Standard	Example
Use descriptive names, and don't abbreviate. Use names that you will remember several months from now. Consider whether another person will understand your notation, and choose names that you can pronounce and remember. Be aware that abbreviations and spelling mistakes can create confusion.	largeButton, LargeButton, or leftButton NOT: lButton
Use Camel case/Pascal case. Avoid spaces in your object names. Camel case or Pascal case improve readability (and typing accuracy according to this study).	OutOfMemoryException, dateTimeFormat, NOT: Outofmemoryexception, datetimeformat
Use underscores (or hyphens) sparingly. Avoid underscores and hyphens in general. However, they can be useful in certain circumstances. Prefixing a name with an underscore puts it alphabetically first. You can also use underscores to denote variants of a specific object.	Active States: EnterButton_Active, EnterButton_Inactive Texture Maps: Foliage_Diffuse, Foliage_Normalmap Level of Detail:Building_LOD1, Building_LOD0
Use number suffixes to denote a sequence. Likewise, don't suffix with a number if it's not part of a list.	For a path, name the nodes: Node0, Node1, Node2, etc.
Follow the design document naming.	If your design document names locations like HighSpellTower or RedDragonLair, use those exact spellings.

文件夹结构

- 内置资源与第三方资源分开放，第三方包括asset store和插件
- 文件夹结构，示例如下

Eample 1

```
Assets
+---Art
| +---Materials
| +---Models
| +---Textures
+---Audio
| +---Music
| \---Sound
+---Code
| +---Scripts  # C# scripts
| \---Shaders  # Shader files and shader graphs
+---Docs   # Wiki, concept art, marketing material
+---Level   # Anything related to game design in Unity
| +---Prefabs
| +---Scenes
| \---UI
```

Eample 2

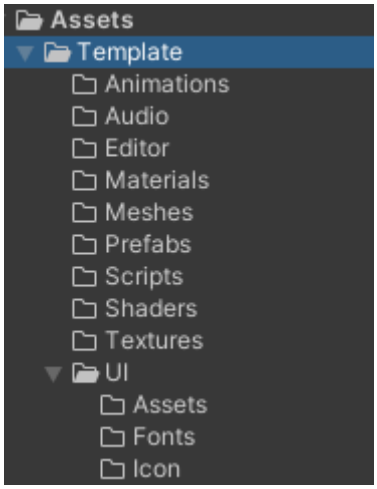
```
+---Art
| +---Materials
| +---Models
| +---Music
| +---Prefabs
| +---Sound
| +---Textures
| +---UI
+---Levels
+---Src
| +---Framework
| \---Shaders
```

- 各种类型的资产

Asset Type	Explanation
Animations	Animations contain animated motion clips and their controller files. These can also contain Timeline assets for in-game cinematics or rigging information for procedural animation.
Audio	Sound assets include audio clips as well as the mixers used for blending the effects and music.
Editor	This contains scripted tools made for use with the Unity Editor but not appearing in a target build.
Fonts	This folder contains the fonts used in the game.
Materials	These assets describe surface shading properties.
Meshes	Store models created in an external digital content creation (DCC) application here.
Particles	The particle simulations in Unity, created either with the Particle System or Visual Effect Graph.
Prefabs	These are reusable GameObjects with prebuilt Components. Add them to a scene to build.

Asset Type	Explanation
Scripts	All user-developed code for gameplay appears here.
Settings	These assets store render pipeline settings, such as for the High Definition Render Pipeline (HDRP) and Universal Render Pipeline (URP).
Shaders	These programs run on the GPU as part of the graphics pipeline.
Scenes	Unity stores small, functional portions of your project into Scene assets. They often correspond to game levels or part of a level.
Textures	Image files can consist of texture files for materials and surfacing, UI overlay elements for user interface, and lightmaps to store lighting information.
ThirdParty	If you have assets from an external source like the Asset Store, keep them separated from the rest of your project here. This makes updating your third-party assets and scripts easier. Third-party assets may have a set structure that cannot be altered.

- 通过脚本创建相同的项目文件结构，创建脚本后，顶部Asset框下多出Create Default Folders选项，自定义根目录文件夹的名字(例如下图中的Template)后创建出模板文件夹




```
using UnityEditor;
using UnityEngine;
using System.Collections.Generic;
using System.IO;
public class CreateFolders : EditorWindow
{
    private static string projectName = "PROJECT_NAME";
    [MenuItem("Assets/Create Default Folders")]
    private static void SetUpFolders()
    {
        CreateFolders window = ScriptableObject.CreateInstance<CreateFolders>();
        window.position = new Rect(Screen.width / 2, Screen.height / 2, 400,
150);
        window.ShowPopup();
    }
    private static void CreateAllFolders()
    {
        List<string> folders = new List<string>
 {
 "Animations",
 "Audio",
 "Editor",
 "Materials",
 "Meshes",
 "Prefabs",
 "Scripts",
 "Scenes",
 "Shaders",
 "Textures",
 "UI"
 };
        foreach (string folder in folders)
        {
            if (!Directory.Exists("Assets/" + folder))
            {
                Directory.CreateDirectory("Assets/" + projectName + "/" +
folder);
            }
        }
        List<string> uiFolders = new List<string>
 {
 "Assets",
 "Fonts",
```

```
 "Icon"
 };
        foreach (string subfolder in uiFolders)
        {
            if (!Directory.Exists("Assets/" + projectName + "/UI/" + subfolder))
            {
                Directory.CreateDirectory("Assets/" + projectName + "/UI/" +
subfolder);
            }
        }
        AssetDatabase.Refresh();
    }
    void OnGUI()
    {
        EditorGUILayout.LabelField("Insert the Project name used as the root
folder");
        projectName = EditorGUILayout.TextField("Project Name: ", projectName);
        this.Repaint();
        GUILayout.Space(70);
        if (GUILayout.Button("Generate!"))
        {
            CreateAllFolders();
            this.Close();
        }
    }
}
```

meta文件

- .meta文件是自动生成的，但它还保存了许多关于与其关联的文件。当更改任何导入的资产如纹理、网格、音频片段等上的设置，更改将写入.meta文件，而不是资产文件。这就是为什么要将.meta文件提交到你的repo，这样每个人使用相同的文件设置

> 空文件夹的.meta说明：
>
> - Plastic SCM can handle empty folders. Directories are treated as entities by Plastic SCM and have a version history associated with them.
> - Unity generates a .meta file for every file in the project, including folders.
> - With Git and Perforce, a user can easily commit the .meta file for an empty folder, but the folder itself won't end up under version control. When another user gets the latest changes, there will be a .meta file for a folder that doesn't exist on their machine, and Unity will then delete the .meta file. Plastic SCM avoids this issue by including empty folders under version control

工作流程优化

- **UnityYAMLMerge**解决 scene和 prefab 间的冲突：Unity - Manual: Smart merge (unity3d.com)
- **Presets**文件夹：Unity - Manual: Presets (unity3d.com)

代码模板

`C:\Program Files\Unity\Hub\Editor\2021.3.22f1c1\Editor\Data\Resources\ScriptTemplates`

创建c#脚本或shader脚本时，Unity使用ScriptTemplates中的模板生成脚本文件

可以在项目中通过实现 `OnWillCreateAsset` 方法，对ScriptTemplate做修改

```
using UnityEngine;
using UnityEditor;
public class KeywordReplace : UnityEditor.AssetModificationProcessor
{
    public static void OnWillCreateAsset(string path)
    {
        path = path.Replace(".meta", "");
        int index = path.LastIndexOf(".");
        if (index < 0)
            return;
        string file = path.Substring(index);
        if (file != ".cs" && file != ".js" && file != ".boo")
            return;
        index = Application.dataPath.LastIndexOf("Assets");
        path = Application.dataPath.Substring(0, index) + path;
        if (!System.IO.File.Exists(path))
            return;
        string fileContent = System.IO.File.ReadAllText(path);
        fileContent = fileContent.Replace("#CREATIONDATE#",
System.DateTime.Today.
       ToString("dd/MM/yy") + "");
        fileContent = fileContent.Replace("#PROJECTNAME#",
PlayerSettings.productName);
        fileContent = fileContent.Replace("#DEVELOPER#",
System.Environment.UserName);

        System.IO.File.WriteAllText(path, fileContent);
        AssetDatabase.Refresh();
    }
}
```

81-C# Script-NewBehaviourScript.cs.txt 模板如下，加入头部信息

```
// /*-------------------------------------------
// -------------------------------------------
// Creation Date: #CREATIONDATE#
// Author: #DEVELOPER#
// Description: #PROJECTNAME#
// -------------------------------------------
// -------------------------------------------*/

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
#ROOTNAMESPACEBEGIN#
public class #SCRIPTNAME# : MonoBehaviour
{
 // Start is called before the first frame update
 void Start()
 {
```

```
 #NOTRIM#
 }
 // Update is called once per frame
 void Update()
 {
 #NOTRIM#
 }
}
#ROOTNAMESPACEEND#
```

VCS工具介绍

Git

- Unity 插件，可导入git工具：GitHub for Unity | Bring Git and GitHub into Unity

Perforce (Helix Core)

- 企业级，EA和育碧使用
- 对大文件的处理比较好，适合Unity
- 使用说明：How to Use Unity + Version Control | Perforce

SVN

- Apache Subversion (known as SVN)
- 可以处理大文件
- 开源、免费、集中式
- merge很痛苦
- 需要第三方GUI插件：How to Use Unity + Version Control | Perforce

Plastic SCM

- 擅长处理大文件、大仓库
- 可以只下载单个文件
- 免费5G的云存储
- SCM的特性和介绍：Version control solution for artists | Unity
- 既可以分布式、也可以集中式
- 其他资料：
  - How to get started with Unity Plastic SCM | Unity
  - Breakpoint 2: Create without Compromise with Unity Plastic SCM - YouTube
  - 关于Unity和Plastic SCM协作的坑 - 知乎 (zhihu.com)
  - https://docs.plasticscm.com/book/

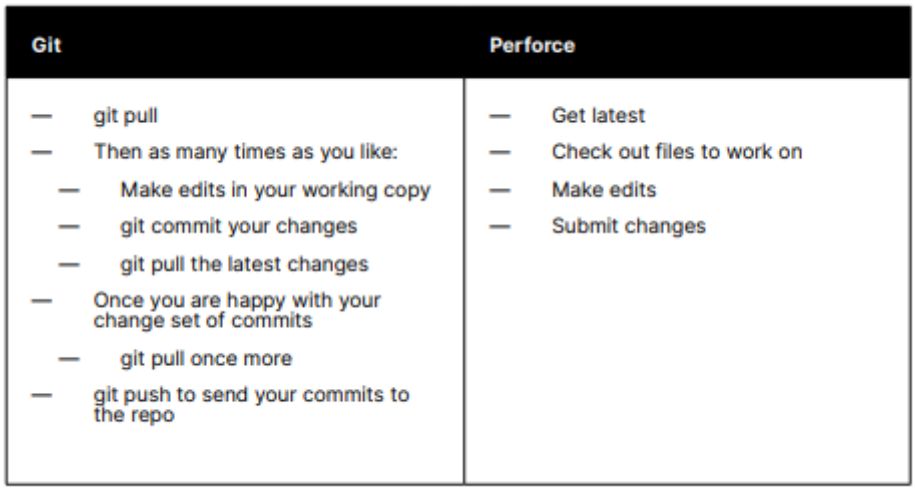
各类VCS的对比

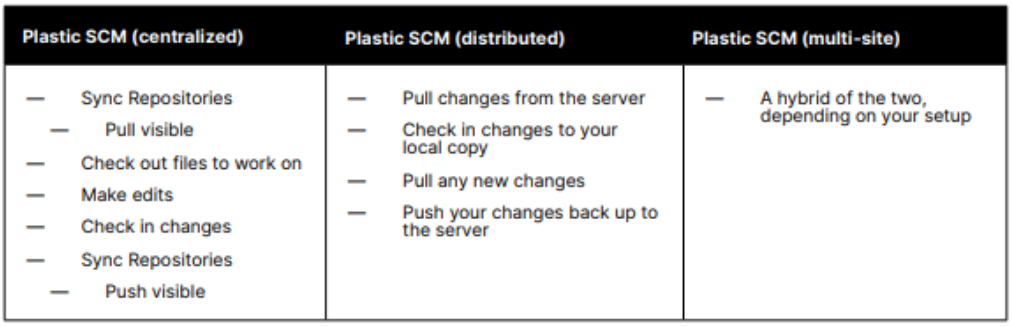
Comparison ● Fully Supported ● Partially Supported

		Plastic	Git	Preforce	Subversion
Flexibility	Good to work centralized Just checkin, no push/pull	●		●	●
	Good to work distributed Push/pull + local repo	●	●		
Binaries	Good with huge repos	●		●	
	Good with huge files	●	●	●	●
	Can lock files to avoid merging	●		●	●
GUI	Visualizes your repos (so you don't need a PhD in branching)	●	●		
	Comes with great GUIs	●	●	●	
	Special GUI and workflow for artists and non-coders	●			
Workflow	Creates effective task branches	●	●		
Merge	Very good detecting merges between branches	●	●		
	Comes with great diff and three-way merge tools	●		●	
	Tools help you understand the merge	●			
	Good merging renames, moved files, directories, refactors	●			
Cloud	Can host repos in the cloud	●	●	●	●
	Cloud hosting is good with huge repos	●			
DIFF	Can diff code moved across files	●			
	Can show you the history of a method	●	●		
	Enterprise Support	●		●	

使用VCS

- Window > Unity Version Control
- Edit > Project Settings > Version Control
- 忽略文件：
  - Plastic SCM：[Plastic SCM blog: Definitive ignore.conf for Unity projects]

    - Git：gitignore/Unity.gitignore at main · github/gitignore
- 提交文件：只提交不能生成的文件，即 `Assets` 和 `Project Settings` 以及 `Packages` 目录下的文件
- 处理大文件：
    - SCM和Perforce都可以用集中式保证大文件只在云端有一份
    - Git需要下载Git LFS并指定大文件，保证大文件在云端，本地使用指针指向该大文件
- 工作流程：
    - 

    - 

总结

- 个人学习的话，只是想要保留平常练手的项目的话，使用Unity官方推荐的Plastic SCM就可以了，相比上传github没有网络困扰