|  |
| --- |
| C:\Users\apple\Pictures\logo.png |
| Unity开发规范文档 |
|  |

|  |
| --- |
| 创建日期：2015/05/29  最后修改：2015/06/08 |

目录

[一、编写目的 3](#_Toc422129516)

[二、Unity工程规范 3](#_Toc422129517)

[1、工程命名规范 3](#_Toc422129518)

[2、资源文件夹命名规范 3](#_Toc422129519)

[3、脚本命名规范 4](#_Toc422129520)

[4、 Scene命名规范 4](#_Toc422129521)

[5、 Hierarchy层次视图物体命名规范 4](#_Toc422129522)

[6、 Inspector检视面板Tag、Layer命名规范 4](#_Toc422129523)

[7、 Animator动画系统参数命名规范 4](#_Toc422129524)

[8、 工程打包命名规范 5](#_Toc422129525)

[三、脚本内部规范 5](#_Toc422129526)

[1、注释 5](#_Toc422129527)

[1.1摘要注释规范 5](#_Toc422129528)

[1.2代码注释规范 5](#_Toc422129529)

[1.3注释内容规范 6](#_Toc422129530)

[2、变量 6](#_Toc422129531)

[2.1定义变量位置规范 6](#_Toc422129532)

[2.2变量命名规范 7](#_Toc422129533)

[3、符号 7](#_Toc422129534)

[3.1运算符使用规范 7](#_Toc422129535)

[3.2分隔符使用规范 7](#_Toc422129536)

[4、类、方法、枚举、结构体 8](#_Toc422129537)

[4.1类、方法、枚举、结构体命名规范 8](#_Toc422129538)

[4.2类、方法、枚举、结构体的大括号位置规范 8](#_Toc422129539)

# 一、编写目的

由于Unity组成立时间还不长，还没有形成一个统一的开发规范，这对我们互相学习帮助是有一定影响的，且每位Unity讲师对学生的开发规范要求不统一，每一位讲师都有自己的开发风格和代码习惯，这会大大影响教学质量和统一性。为了使每位讲师和每位讲师所带的学生都形成统一的开发规范，特制定本规范文档，在今后的开发和授课过程中，请严格遵守此文档的开发规范。通过此规范，希望可以起到相互间沟通桥梁的作用，并可以传授给自己的学生统一的开发规范。

在今后的使用过程中，如遇到本文档中没有的并且需要进行规范的问题，待商议后，对此文档进行更新填补。

# 二、Unity工程规范

## 1、工程命名规范

工程的命名规范需根据项目类型命名，如用于教学演示的工程应以“Demo\_”开头，如工程作为研发成果上线，以“Proj\_”开头，工程名称不可以有中文字符，并在文档中对分类进行解释。

例如：Demo\_Piano Proj\_WorldWar

## 2、资源文件夹命名规范

资源文件分两类：第一类是外部导入资源；第二类是开发者自创的。在资源文件夹的名字前加下划线区分，有下划线的表示是开发者自创资源。请注意文件夹的命名不可以以‘S’结尾，命名规范如下所示：

（1）脚本资源文件夹命名为：外部导入Script 自创\_Script

（2）预制资源文件夹命名为：外部导入Prefab 自创\_Prefab

（3）音频资源文件夹命名为：外部导入Audio 自创\_Audio

（4）动画资源文件夹命名为：外部导入Animation 自创\_Animation

（5）场景资源文件夹命名为：外部导入Scene 自创\_Scene

（6）图片资源文件夹命名为：外部导入Image 自创\_Image

（7）材质资源文件夹命名为：外部导入Material 自创\_Material

（8）贴图资源文件夹命名为：外部导入Texture 自创\_Texture

（9）NGUI图集文件夹命名为：外部导入Atlas 自创\_Atlas

（10）粒子系统文件夹命名为：外部导入Particle 自创\_Particle

（11）编辑器扩展脚本文件夹：Editor

（12）动态加载资源文件夹：Resources（必须放在上述资源文件夹的根目录下）

## 3、脚本命名规范

（1）见名知意。

（2）可以由多个单词组成，但每个单词的首字母都要大写（驼峰法命名）。

例：PlayerControllerScript

## Scene命名规范

命名统一使用小写英文。若该工程仅有一个Scene，那么以工程名命名即可。例如工程Demo\_Piano的场景名为“piano”。

多场景命名范例如下：

（1）登陆注册界面：register

（2）战斗界面：fight

（3）游戏框架界面：main

## Hierarchy层次视图物体命名规范

1. 见名知意。
2. 命名时首字母必须大写，如导入模型，只需修改root层级命名即可。
3. 不可使用缩写。如BackGround代表背景，不可以缩写为BG。

## Inspector检视面板Tag、Layer命名规范

1. 见名知意。
2. 命名首字母必须大写。例如：Monster、Player

## Animator动画系统参数命名规范

（1）见名知意。

（2）参数命名首字母必须大写。例如：Attack、Speed

## 工程打包命名规范

压缩文件名称前部分与工程名字相同,后部分需要加入版本号。如有需要，可以在压缩工程里附带一个文本文档进行工程说明。

例如：Demo\_Piano\_1.0.zip 附件：Demo\_Piano\_1.0说明.txt

# 三、脚本内部规范

## 1、注释

### 1.1摘要注释规范

脚本最上方需要加summary注释，其中包括：脚本功能、添加对象、知识要点、创建日期。这样方便查找。如下所示：

1. using UnityEngine;
2. using UnityEngine.UI;
3. UsingSystem.Collections;
4. ///<*summary*>
5. /// *脚本功能：用脚本代码给UGUI  Button控件添加事件*
6. /// *添加对象：Button对象（有UGUI Button组件）*
7. ///*创建日期：2015.5.19 duzixi.com*
8. ///*知识要点：委托 delegate///*
9. ///<*summary*>
10. public class PlayAudioScript : MonoBehaviour{

### 1.2代码注释规范

（1）注释能使用中文翻译及中文的标点符号尽量不用英文。

（2）注释与注释分隔符用一个空格分开。

例如：

public float playerMovespeed; // *声明玩家移动速度。*

（3）变量注释和变量在同一行，与变量分开两个Tab键。

且与距离左侧最远的注释符对齐。

例如：

public float playerMovespeed; // *声明玩家移动速度。*

public intplayerHp; // *声明玩家生命值。*

（4）算法必须有注释。

例如：

Change（inta,intb） // *交换算法*

{

intc;

c = a, a = b,c = b;

return a;

}

（5）在循环和逻辑分支的地方必须写上注释。

例如：

int sum = 0;

For(int a= 1; a <= 10; a++) // *使用for循环求从1加到10的和。*

{

sum += a;

}

（6）程序段或语句的注释在程序段或语句的上一行。

例如：

// 玩家属性

public float playerMovespeed; // *声明玩家移动速度。*

public intplayerHp; // *声明玩家血量。*

（7）在代码交付之前，必须删掉临时的或无关的注释。

### 1.3注释内容规范

注释的书写要通俗易懂，更接近于口头语。

## 2、变量

### 2.1定义变量位置规范

定义变量时，应该以功能模块划分来摆放位置，比如在PlayerControllerScript脚本中有关主角属性的变量放在一起，有关主角状态的变量放在一起，杂项放在一起，中间用一行空格隔开。如下所示：

// *玩家属性*

public float playerMovespeed; // *声明玩家移动速度。*

public intplayerHp; // *声明玩家生命值。*

public Vector3 playerPosition; // *声明玩家坐标。*

public AudioSourceplayerAttackSound; // *声明玩家移动速。*

private boolplayerAttackDelay; // *声明玩家攻击延迟。*

private Animation anim; // *声明玩家动画。*

// *怪物属性*

privateGameObject monster // *声明怪物物体*

public intmonsterHp; // *声明怪物生命值。*

public Vector3 monsterPosition; // *声明怪物坐标。*

public boolmonsterAttackDelay; // *声明怪物攻击延迟。*

//*其他变量*

private float Distance //*声明玩家与怪物距离*

### 2.2变量命名规范

（1）由数字、字母、下划线“\_”或“@”组成，并且不能以数字开头。

（2）不能与系统关键字重名

（3）不能使用重复的变量名（C#大小写敏感）

（4）见名知意

注：变量的命名一般只使用单个小写单词，如果多个单词使用驼峰命名法。

## 3、符号

### 3.1运算符使用规范

运算符前后加空格。

例如：

//*计时结束，输出胜利所用时间*

win.text = timer +""\";

tBest.text = "历史最高：" + save1 + "\"";

### 3.2分隔符使用规范

（1）逗号“，”后加空格，如图1所示：

（2）分号“；”后加空格，如图2所示：

// *边界*

move.transform.localPosition = newVector (0, 1, 0);

图1

//*for循环*

for(inti = 9; i<buttons.Length; i++);

图2

## 4、类、方法、枚举、结构体

### 4.1类、方法、枚举、结构体命名规范

类、方法、枚举、结构体在命名时，需见名知意，并且每个单词首字母必须大写，（驼峰命名法）。如：GameController、ScoreScript、CameraController、Season、Dog等。见图1：

class MyClass

{

}

enum Season

{

Sprite = 1，

Summer = 2，

Autumn = 3，

Winter = 0

}

### 4.2类、方法、枚举、结构体的大括号位置规范

在类、方法里，第一个大括号“｛”要放在下一行,见图2：

if(firstClick)

**{**

timer +=Time.time;

firstClick = false;

}

图2