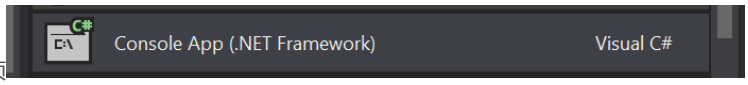
Unity 3D游戏编程自学#1——C#基本语法

原创 Zerg_Wang 于 2019-02-19 18:49:32 发布 4672 收藏 26 编辑 版权

分类专栏： Game Programing

1.如何开始

打开Visual Studio，左上角：文件—新建—项目，选择以下项



创建即可。

创建该项目的同时会创建一个解决方案，一个解决方案下可以有多个项目，每个项目下可以有多个类。

要打开一个解决方案，打开.sln后缀名的文件即可。

要打开一个项目，打开.csproj后缀名的文件即可。

一些规则：

驼峰命名法：组成变量名的所有单词首字母大写（第一个除外），该法则一般用于变量的命名。

例如：intArray、userName

帕斯卡命名法：所有单词首字母都要大写，一般用于给函数和类命名。

例如：GetNumber、ReadKey

对于常量，一般全大写，单词之间用下划线分开，例如：SERVER_IP

2.float类型赋值问题

```
float f = 1.1;
```

错误！1.1默认为double型，正确的应该在末尾加一个字母f：

```
float f = 1.1f;
```

double型直接赋值即可。

3.在C#中char类型可以放一个中文单字

4.C#中“+”的特殊用法

若char和整数、小数相加，则char会转为其对应的ASCII码的值。

若“+”左右有字符串string，则分两种情况：

```
1 int b = 22;
2 string s = "BBB" + b;
3 Console.WriteLine(s);
4 Console.WriteLine(s + b);
```

第一种情况下，两者相连，b被转换为字符串加入到s中，输出为BBB22

第二种情况，输出时，先输出s然后直接输出b，中间无空格无换行，输出为BBB2222

5.占位符

```
string name = "Zerg";
int age = 20;
string gender = "男";
Console.WriteLine("我的名字是{0},我今年{1}岁,我的性别是{2}",name,age,gender);
```

占位符为{0}、{1}、{2}，从0开始。占位符与变量一一对应。少了无影响。多了会报错。

实际上可以调换顺序。

```
Console.WriteLine("我的名字是{1},我今年{2}岁,我的性别是{0}", name, age, gender);
```

输出为：我的名字是20,我今年男岁,我的性别是Zerg

6.转义字符

回车：\n　英文双引号：\"　Tab键：\t　退格键：\b　一个\：\\

若要使字符串中的转义符号全部失效（即字符串里面是什么就输出什么，常用于路径的处理），则在字符串前加“@”

```
string s = @"C:\Program Files\Adobe";
```

7.输入输出

Console.Read()：

返回读入值的ASCII码，返回值是int型。举例：

```
int a = Console.Read();
Console.Write(a+" ");
```

输入0和回车，输出：48 13 10

因为0的ASCII为48，回车符的为13，换行符的为10。

同理，输入AB和回车，输出：65 66 13 10

Console.ReadLine()：

读取一行用户的读入，返回的是字符串。

String s = Console.ReadLine();

Console.ReadKey()：

读取任意一个值后返回，常用在程序末尾起“按任意键退出”的作用。

Console.Write()和Console.WriteLine()：

输出内容，后者比前者多输出一个换行。

前者必须要有输出内容，后者括号内可以为空（就输出一个回车）。

8.类型转换

int型的值可以直接转换为float或者double型，即int型变量可直接赋值给float和double型，float型变量也可直接赋值给double型。

相反则不行，需要强制转换，要额外添加代码：

```
float a = 1.7f;
int b = (int)a;
```

结果是截去小数，只留整数部分。同理，double转为float要加：(float)

此外，若要将string类型转为int或double，则：

```
string s1 = "12.34";
string s2 = "23";
double m1 = Convert.ToDouble(s1);
int m2 = Convert.ToInt32(s2);
```

convert函数中没有ToFloat

拓展：数字转字符串：

```
string s;
int a = 10;
s = "" + a;
```

9.数组

与C++不同，以C++中int intArray[100];为例

```
int[] intArray = new int[6];
```

C#中的这个“6”只的是可以取0到5一共6个数组下标。

在函数中，若数组为形参，写法：

```
static void Work(int[] intArray)
```

初始化：

和一般的变量不一样，数组中的默认值均为0（无论是int、float、double还是bool），可以直接使用（变量则不能）。若要初始化：

```
int[] intArray = new int[6] { 1,2,3,4,5,6 };
```

若要初始化，则要一一对应，不能多也不能少。

在初始化时，可以：

```
int[] intArray = new int[] { 1,2,3,4,5,6 };
```

系统会自动计算初始化的数量以确定数组范围。

数组长度：数组名.Length。以以上为例是：intArray.Length

二维数组的定义、初始化及其调用：

```
int[,] intArray = new int[2, 2] { { 1, 2 }, { 3, 4 } };
intArray[1, 1] = 100;
```

一种新的数组遍历方法：

```
int[] intArray = new int[3] { 0, 1, 2 };
foreach(int i in intArray)
{
    Console.WriteLine(i);
}
```

10.引用传递

```
1 int[] intA = new int[] { 0, 1, 2, 3 };
2 int[] intB = intA;
3 intB[2] = 100;
4 Console.WriteLine(intA[2]);
```

如上述代码，对于所有的数组类型（包括string）在执行将数组intA赋值intB时同样将intB的内存地址覆写为intA的地址，因此更改intB的值，intA也相应改变，反之亦然。

11.函数

写在class Program里面，static void Main之后（都在void Main后面即可，具体顺序无需操心）。

```
class Program
{
    static void Main(string[] args)
    {
```

除了前面多个static，其他与C++一样

文档注释：

连续写三个“/”，这种注释方式一般用于给类和函数注释。

```
/// <summary>
/// 加和函数
/// </summary>
/// <param name="x">参数1</param>
/// <param name="y">参数2</param>
/// <returns>两数之和</returns>
static int Add(int x,int y)
{
    return x + y;
}
```


“加和函数”的位置一般描述函数功能，参数1、2的位置描述参数，“两数之和”位置描述返回值。完成注释后，调用该函数时会有以下效果：

```
init();
int c = Add()
Console.
Console.
```

```
int Program.Add(int x, int y)
加和函数
x: 参数1
```

函数重载：

多个相互之间参数都不同的函数，其函数名可以相同。

例如将两个double相加的函数，和将两个int相加的函数，函数名都可以为Add。

注意：

若使用函数重载，区分是由参数区分的，同名的函数中，不能有参数相同的（参数数量相同，类型不同，或者类型相同，数量不同）。

返回值与函数重载无关，不能以返回值的不同来区分同名函数。

函数高级参数：

ref：在讲实参传入形参的时候，不采用值传递，而采用引用传递。

也就是说，函数内形参的值发生改变，外面的实参也发生改变。

编写参数表和调用时都需要加上关键字“ref”。

```
//参数表
static void Add(ref int x,int y)
//调用
Add(ref a,b);
```

注意：若使用到ref，则实参在传入函数前必须要有值。

out：若函数要返回多个值，则可以将所需返回的值写于参数表中，之前加上out关键字，在调用的时候也需要加上out关键字。

```
static void Main(string[] args)
{
    int a, b, c;
    c = Add(out a, out b);
}

static int Add(out int a,out int b)
{
    a = 1;
    b = 2;
    return a + b;
}
```


无论函数本身是否有返回值，都可以使用out。

注意：使用out的变量在函数中必须要有赋值。

12. 字符串操作

大小写转换

```
string s = "Zerg";
s = s.ToUpper();
s = s.ToLower();
```

将字符串中所有小写字母转为大写：ToUpper

将所有大写转为小写：ToLower

字符串中若有非字母字符，一样可以用，不会报错。

字符串分割、截取

```
char[] c = new char[] { '@', '#' };
string s1 = "aab#bb3#65@2";
string[] stringArray = s1.Split(c);
```

将字符串s1按c中的字符为分隔符分开，因此函数返回值是一个字符串数组。

参数可以为字符数组，也可以为单个字符，但不能为字符串。

```
string s1 = "1ze#Rg";
string s2 = s1.Substring(2, 2);
string s3 = s1.Substring(3);
```

s2截取到了s1从第2位开始的连续2个字符，s3截取到s1从第3位直到最后的全部字符。（字符串起始位下标为0）

字符串查找

```
string s1 = "rbgjkrbgo";
int a = s1.IndexOf("rbg");
int b = s1.LastIndexOf("rbg");
```

IndexOf：在s1中查找“rbg”第一次出现时所处的位置，上述程序返回0。

LastIndexOf：在s1中查找"rbg"最后一次出现的位置，上述程序返回5。

若不存在所查字符串，返回-1。

还有两个函数：

s1.StartsWith(s2)，如果是s1的开头是s2，返回true，否则false。

s1.EndsWith(s2)，类似。

s1.Contains(s2)，如果s1中有s2，返回true，否则false。

字符串替换

```
string s2 = s1.Replace("aa", "bb");
```

将s1中出现的所有"aa"替换为"bb"，将新字符串赋给s2。

字符串修改

s1.Trim()：返回去除s1开头空格和结尾空格的字符串，字符串中间的空格不删。

s1.TrimStart()：只去掉开头空格。

s1.TrimEnd()：只去掉结尾空格。

字符串判断

判断字符串是否为空：

```
1 string s1 = null;
2 bool f = string.IsNullOrEmpty(s1);
```

当s1为""或者null时，都返回True，但当s1为""时是占据内存空间的， null则不占。

判断两个字符串是否相等：s1.Equals(s2)，相等返回True。

（ Equals和==不一样。对于值类型两者相同，对于除字符串的引用类型两者不同。 ）

字符串特点

字符串是引用类型，其数据存储在堆空间，在栈空间中存储该数据引用地址。

每当给字符串赋以新值时，旧值并未销毁，而是开辟了一块空间存储新值。

13.StringBuilder类

与字符串类似，但比string效率高很多，而且节省空间。

定义与向里面追加数据：

```
1 StringBuilder sb = new StringBuilder();
2 sb.Append(12);
3 sb.Append(12.34);
4 sb.Append('a');
5 sb.Append("as");
```

sb.Append()可追加的数据类型极多，包括但不限于整数、小数、字符、字符串。

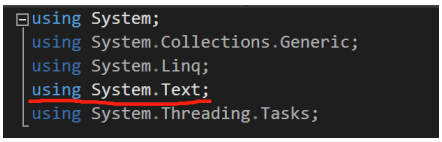
将sb中所有数据转换为字符串：

```
string s = sb.ToString();
```

输出s，会得到1212.34aas

清空sb中数据：sb.Clear()　　无返回值

注意：要使用stringBuilder类，需要在开头调用System.Text，一般程序创建时已经写好了。



14.Stopwatch类

用于计算程序运行时间。若要使用，开头需要调用：

```
using System.Diagnostics;
```

创建：

```
Stopwatch sw = new Stopwatch();
```

在需要开始计时的程序代码处键入：sw.Start();

结束处键入：sw.Stop();

输出程序段运行时间：Console.WriteLine(sw.Elapsed);

输出的格式：

00.00.00.0000028　分别是时、分、秒及更小。

15.枚举类型

定义（一般写于namespace之下，方便所有的类都可用到）

元素名的规范与变量名相同（除此之外，元素名还能是中文，也就是说，元素名需以英文、中文或下划线开头），元素之间用逗号隔开，最后一个元素之后不用逗号。

调用：

```
1 Gender play1 = Gender.男;
2 Console.WriteLine(play1);
```

输出为：男

使用枚举类型是为了在同一项目中统一一些描述性词语的用法。例如在形容玩家性别时，直接调用Gender，则所用的描述性词语就只能是Gender之中的元素了。

16.结构体类型

一种值类型，定义于namespace之下（例子：创建一个结构体存储玩家信息）：

```
public struct Player
{
    public string name;
    public int level;
    public Gender gender;
    public string career;
}
```

注意：都要有public修饰符。

调用与赋值：

```
Player player1 = new Player();
player1.name = "Zerg";
player1.level = 100;
player1.career = "mage";
player1.gender = Gender.male;
```

本文部分内容来自擅码网（http://www.mkcode.net）Unity 3D课程，经本人学习、整理得来，若有错漏，欢迎指正！