批量注释：/\*开头 \*/结尾

日志输出：print();只能在组件中使用（MonoBehaviour）

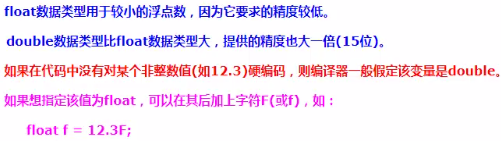
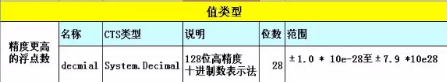
Debug.Log();可以在任意方法中使用

Debug.LogWarning();黄色警告

Debug.LogError();报错

变量： int 命名 = 数值;

小数和整数数据类型：选择不同的类型节约内存



2021_01_18_14_53.45

字符串：string 命名 = “”;

四则运算：例：int a = 90; int b = 10;

Int res = a + b;

Print(res);

整除小数部分会被舍去

多个数据类型运算，返回的结果类型是数据类型大的

两个字符串也可以进行相加；

比较运算符：<,>,<=,>=,==,!=; 操作数1 运算符 操作数2；

例：int hp = 100;

bool res = hp>0;

Print(res);

运算符：<https://wenku.baidu.com/view/93c32317a76e58fafab00341.html>

自增：命名++;自身加一，自减：命名--;自身减一

If语句：例：int 命名 = 数值;

If (命名 比较运算符 数值)

{

控制是否执行;

}

If else语句：例：int 命名 = 数值;

If (命名 比较运算符 数值)

{

执行一;

}

else

{

执行二;

}

数组声明和使用：例：int[] 命名 = { 数值, 数值, 数值, 数值, 数值, };

数组长度为5，通过索引来访问数据 从0开始

命名[索引] n-1 n=数组个数

数组的其他声明方式：例：int[] 命名 = new int[数值];

Int[] 命名 = null;

Int[] 命名 = { };

Int[] 命名 = new int[数组个数]{数值}；

new代表构造

for循环(创建多个目标)：例：for(int i = 起始数; i <= 个数; i++){

循环体

}

for循环遍历数组：例：for(int i = 数组个数-1; i < 命名.Length; i++){

循环体

}

方法的定义和调用：void 方法体(参数){

方法体

}

枚举类型(可在public class前)：例：enum 命名{

类型

} //定义

命名 rt = 命名.类型; //调用

方法中参数的作用：给方法传递参数

方法中的返回值：例：void add(int a, int b){

Int res = a + b;

return = res;

}

类的创建，声明和构造：例：class 类名{

字段，方法;

} //创建声明

类名 命名 = new 类名(); //构造

类中的字段的使用：在创建的类中字段，方法前加上public才可以调用

脚本中变量的定义：public显示在面板，private或无则不显示

逻辑运算符：&&(逻辑与操作)，||(逻辑或操作)，!(取反)

&&满足两个条件还会输出，||满足其中一个条件久会输出

If语句和switch语句的区别和联系：例：switch(变量){

case 1:

Break;

case 1:

Break;

case 1:

Break;

Default:

Print(“default”)

Break;

}

for,while和dowhile循环：例：int i = 0;

While( i<数组 ){

If(变量[i] != transform){

执行命令；

}

i++；

}

do{

If(变量[i] != transform){

执行命令；

}

i++；

} while( i<数组 )；

foreach遍历：例：foreach(Transform 命名 in 数组){

If(命名 != transform)

执行命令;

}

组件获取：GetComponent<Transform>(); //访问内部组件

public 脚本名 命名; //拖拽赋值

组件的禁用和激活：例：BoxCollidder 命名 = GetComponent<BoxCollider>();

Collider.enabled = false; //禁用脚本但方法还可以使用

游戏物体获取：public GameObject 组件名;

Transform.Find(“GameObject (1)/ GameObject”);

GameObject 命名 = GameObject.Find(“组件名”);

GameObject player = GameObject.FindWithTag(“标签名”);

//第四种先指定标签