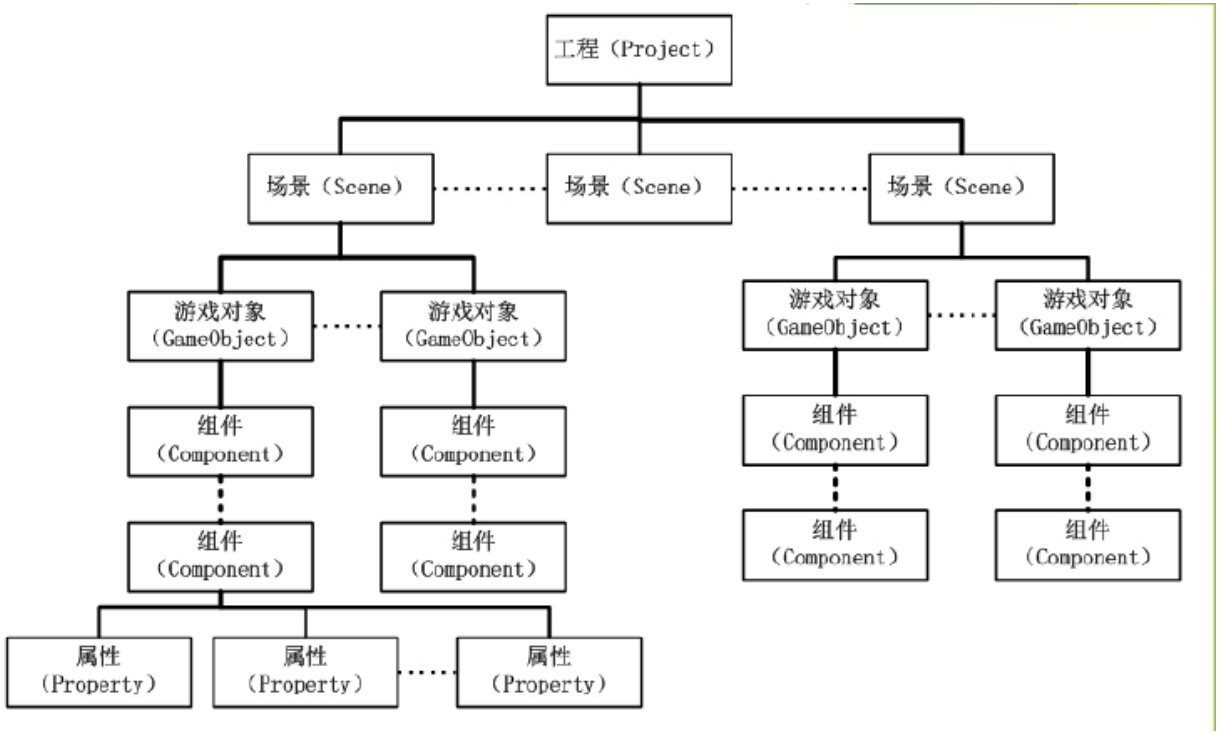
第2章 Unity3D脚本程序介绍
——C#语言基础

回顾—— Unity3D的重要概念

Project工程：一个游戏便是一个工程；
Scene场景：一个游戏包括一个或者多个游戏场景/关卡。
GameObject游戏对象：构成游戏的各种对象，例如场景模型、角色以及其他抽象(如某种控制模块)的但是确实存在于场景中的对象。
Component组件：是Unity的核心架构。
每个游戏对象由一个或者多个组件构成，每种组件赋予游戏对象某种功能。
Script脚本：是实现Component的脚本程序，现在Unity支持C#语言（一种与Java语言类似的语言，只要会Java，上手C#非常简单）。

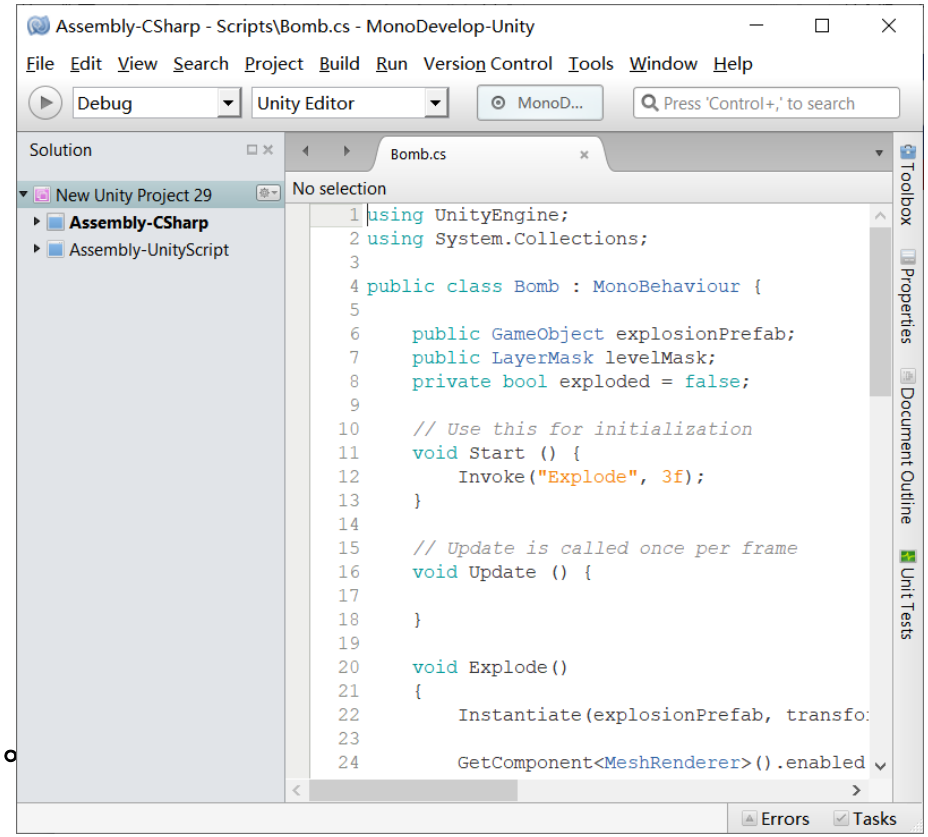


回顾——Unity3D的重要概念——脚本

脚本（**Scripts**）：我们知道，游戏与其他娱乐方式（电影、图书、电视、广播等等）的最大区别在于可互动性。
互动性是游戏的最基本特征之一，而程序脚本便是实现可互动性的最有利的工具。

通过编写程序可以控制游戏中的每一个游戏对象，我们可以让他们根据我们的需要改变他们的状态和行为。

最新版本的**Unity**支持**C#**程序语言。**Unity**支持使用**C++**、**Java**等其他语言编写的插件。



本章提要

- 2.1 C#程序基本语法
  - 基本结构、概要
  - 变量、数组、函数、运算符
  - 控制语句
- 2.2 关键事件回调机制
- 2.3 推荐的学习方法
- 2.4 线性代数（向量知识）
- 2.5 访问GameObject组件

2.1 C#程序基本语法——基本结构

- 在Unity中编写程序一般使用C#语言（“#”读作“sharp”）
- C#脚本
  - 用来编写C#语言的文本文件

- 新建C#脚本：循环语句
  - 在资源管理面板中执行“Create”→“C# Script”命令

2.1 C#程序基本语法——基本结构

代码演示：
Example3_1.cs

2.1 C#程序基本语法——概要

- 变量声明（变量名规定首字母为小写，使用驼峰命名法）：
  - **[public/private] [int/float/bool/double/string/GameObect] ValName = [defaultVal];**
  - **如 public int score = 0; private float speedVal = 20f;**
- 函数声明（函数名规定首字母为大写，使用驼峰命名法）：
  - **[public / private] returnVal FunctionName(ParametersList){ Function Block}**
  - **如 public int HurtCharacter(int damageVal){ damage -= damageVal; }**
- 类的声明（类名规定首字母为大写，使用驼峰命名法）：
  - **[public/private] ClassName : [SuperClass]{}**

学习要点：

语法类似于Java，在一些表达方式有所区别，如继承——java中是“extends”，C#中用的是冒号“：”。

在编程的学习中，不要刻意追求某种语言的好与坏，只有是否合适，没有最好的。
同时，掌握了基本的编程思想（如面向对象编程、面向过程编程、数据结构以及相关算法）后，基本上可以做到语言无关。

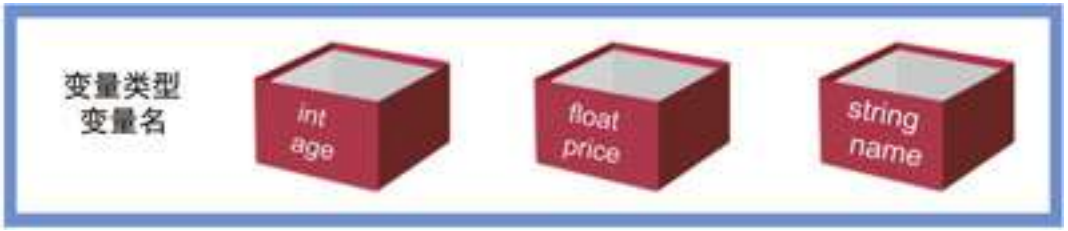
使用Unity中的C#进行程序编程，除了面向对象的基本语法之外，还需要了解它所提供的API，同时要熟悉一些常用的API。

Unity手册： https://docs.unity3d.com/Manual/index.html

Unity脚本API Documentation： https://docs.unity3d.com/ScriptReference/index.html

2.1.1 变量（1/2）

在所有程序设计里都会用到变量，Unity也不例外。变量是用于容纳一个值的存储位置，可以把计算机内存中的变量当作箱子。在这些箱子放入一些东西，然后把它取出来，或者只是看看盒子里是否有东西。



- 变量的命名规则
  - 变量名称是由英文字母、数字以及下划线组成的，不能包含空格、标点符号等其他符号
  - 变量名不能以数字为开头，必须以字母或下划线为开头
  - 变量名不能与C#中的关键字名称相同
- 声明变量的方法
  - 变量声明方法：类型标识符 变量名；要使用变量，需要声明它们，否则无法编译并会报出错误，所以需要在一个声明语句中声明变量的类型和名称
  - 变量赋值方法：变量名＝值
- 注意：变量必须先定义后使用

2.1.1 变量（2/2）——基本类型

类型标识符	类型名	数值范围
int	整型	-21474863648 ~ 2147483647
float	浮点型	-3.402823E+38 ~ 3.402428E+38
double	双精度型	-1.79769313486232E+308 ~ 1.79769313486232E+308
bool	布尔型	true、false
string	字符串型	文本
GameObject	游戏对象型	游戏对象

2.1.2 数组

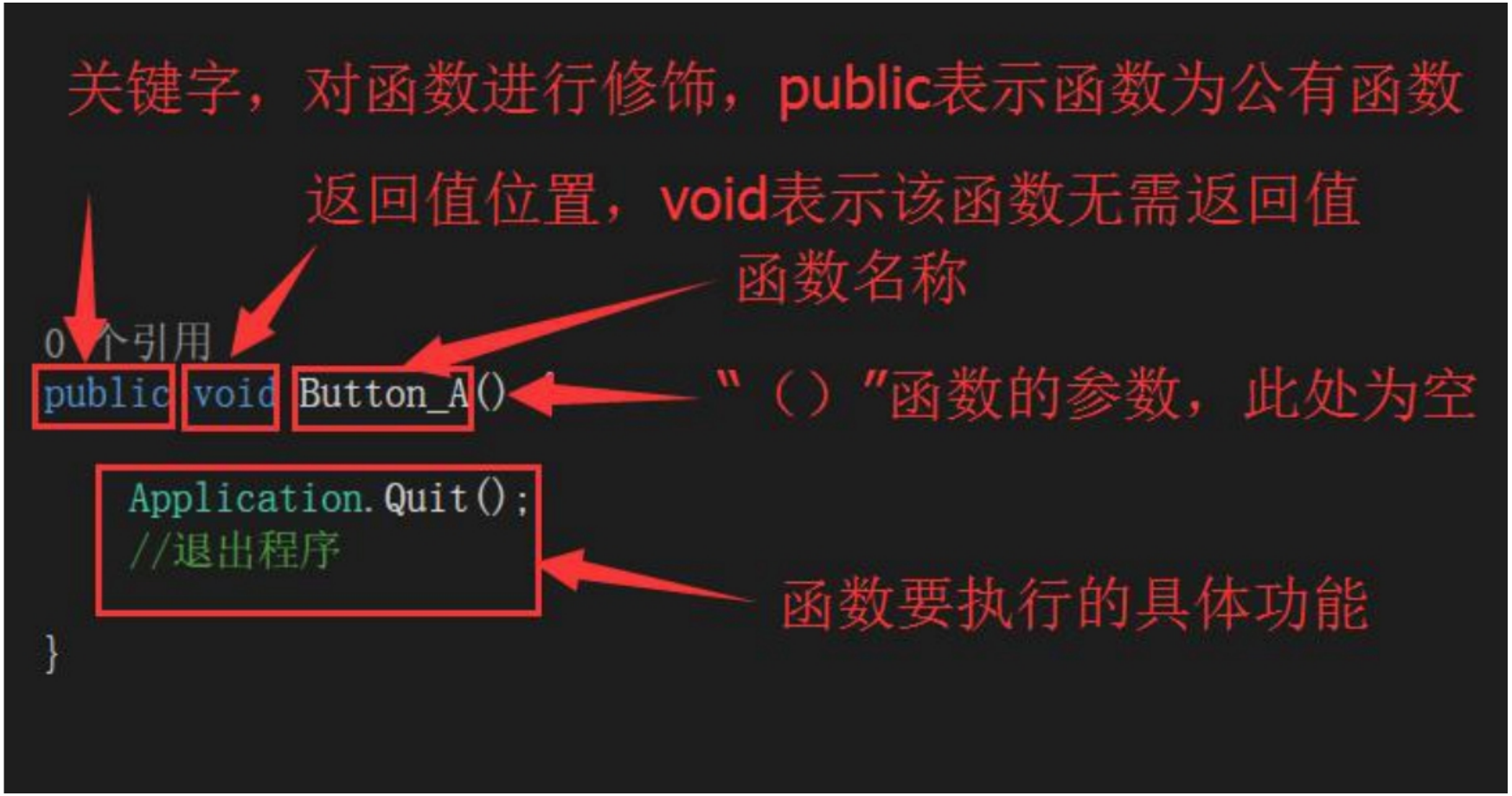
- 数组是具有相同类型的一组数据，如教师信息、学生信息、课程信息等都可以看成一个数组
- 数组就是同一数据类型的组值
- 在C#中只能使用内建数组

代码演示：
Example3_1.cs

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class Sample12_2_2 : MonoBehaviour {
5
6     private int[] array = new int[5];
7
8     void Start () {
9
10        for(int i = 0;i<array.Length;i++)
11        {
12            array[i] = i;
13            print(i);
14        }
15
16    }
17
```

2.1.3 函数

■ 函数也叫作方法，可以看作脚本中的一个模块，能够实现特定的功能。



2.1.4 运算符

算术运算符	说　明	例子（a=4）	结　果
+	左边数值加上右边数值	a=a+2	6
-	左边数值减去右边数值	a=a-2	2
*	左边数值乘以右边数值	a=a*2	8
/	左边数值除以右边数值	a=a/2	2
%	左边数值除以右边数值的余数	a=a%3	1
++	左边数值加1	a++	5
--	左边数值减1	a--	3

比较运算符	说　明	例子（a=4）	结　果
==	左边数值等于右边数值吗?	a==5	False
! =	左边数值不等于右边数值吗?	a ! =3	True
<	左边数值小于右边数值吗?	a<6	True
<=	左边数值不大于右边数值吗?	a<=7	True
>	左边数值大于右边数值吗?	a>9	False
>=	左边数值不小于右边数值吗?	a>=4	True

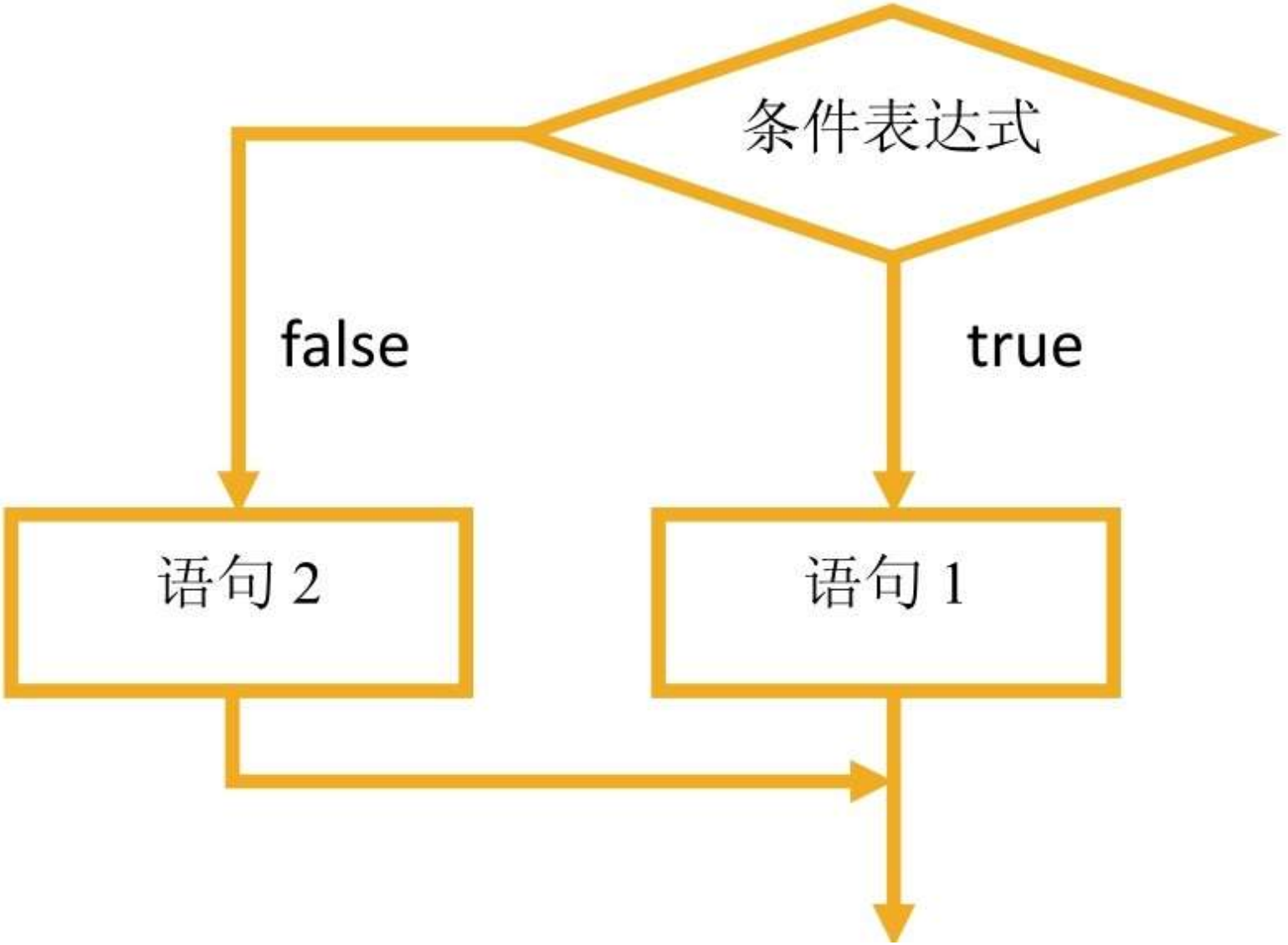
逻辑运算符	说　明	例子（a=4 b=1）	结　果
&&	左边与右边进行与运算	a==5 && b==1	False
||	左边与右边进行或运算	a==4 || b==2	True
!	与左边进行非运算	！（a==5）	True

2.1.5 控制语句

- 条件语句
  - if
  - switch
- 循环语句
  - for
  - foreach
  - while
  - do while

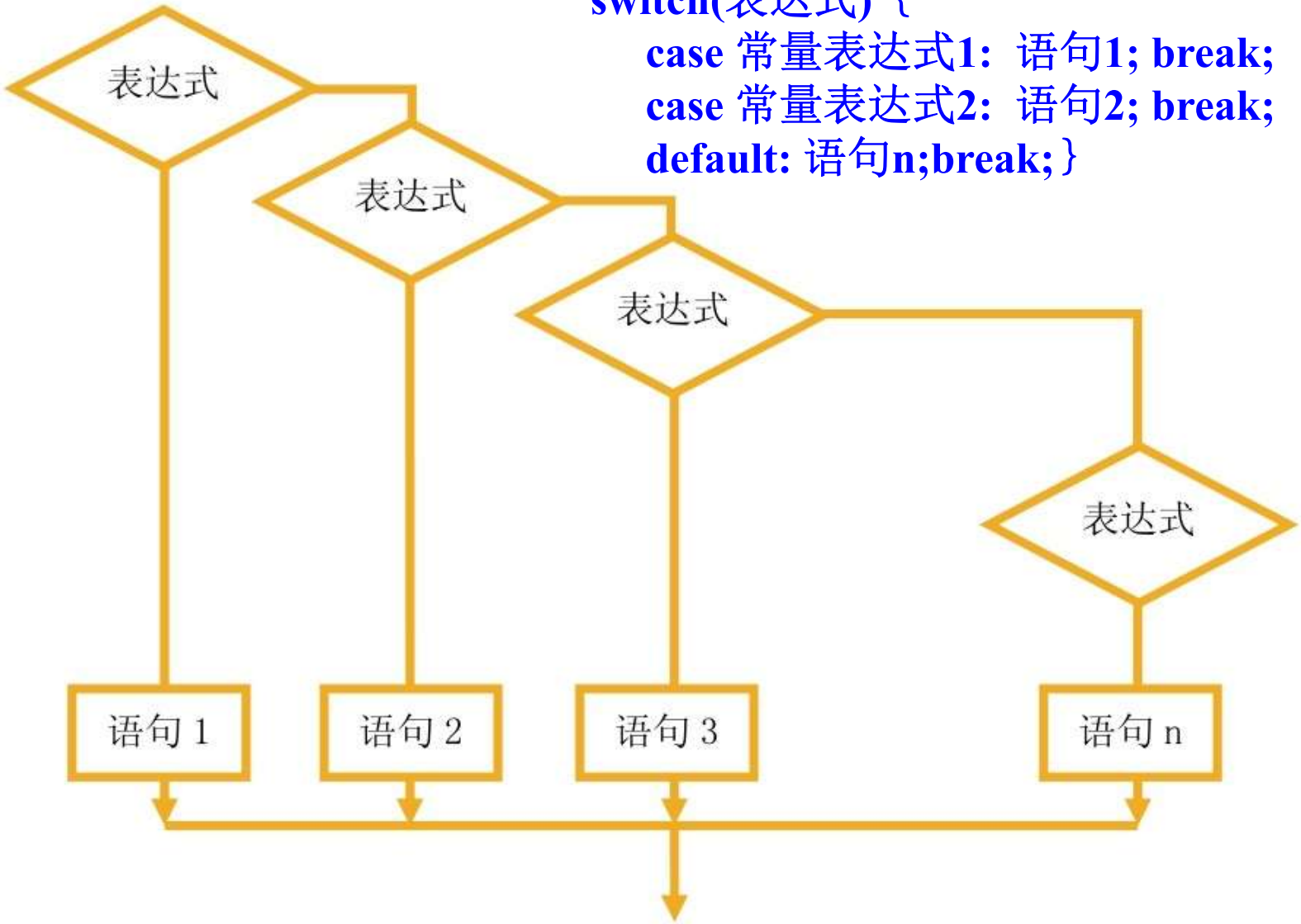
2.1.5 控制语句——条件（if）

if(条件表达式){　　表达式1;}else{　　表达式2;}



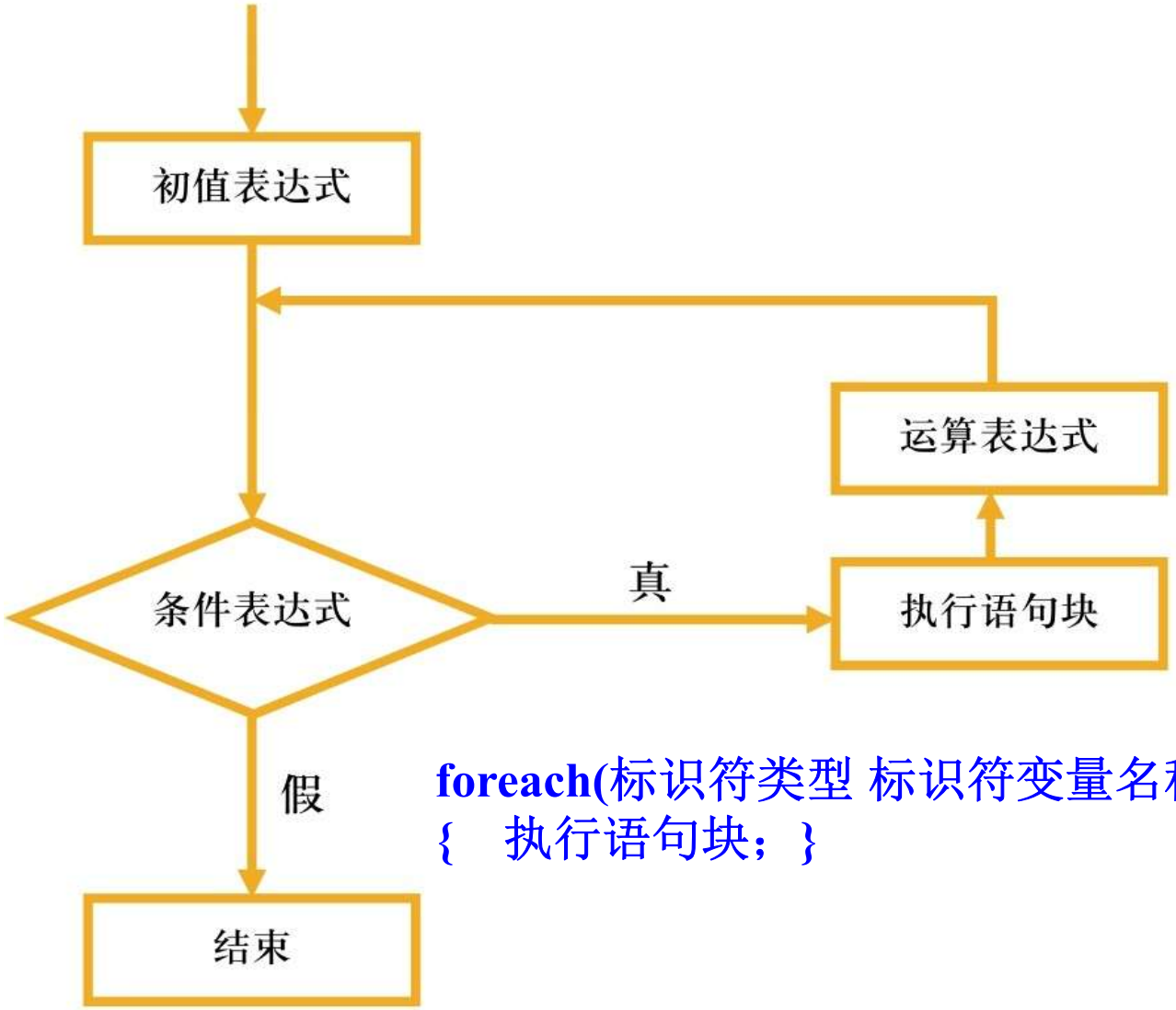
2.1.5 控制语句——条件（switch）

```
switch(表达式) {
    case 常量表达式1:  语句1; break;
    case 常量表达式2:  语句2; break;     ...
    default: 语句n;break;}
```



2.1.5 控制语句——循环（for，foreach）

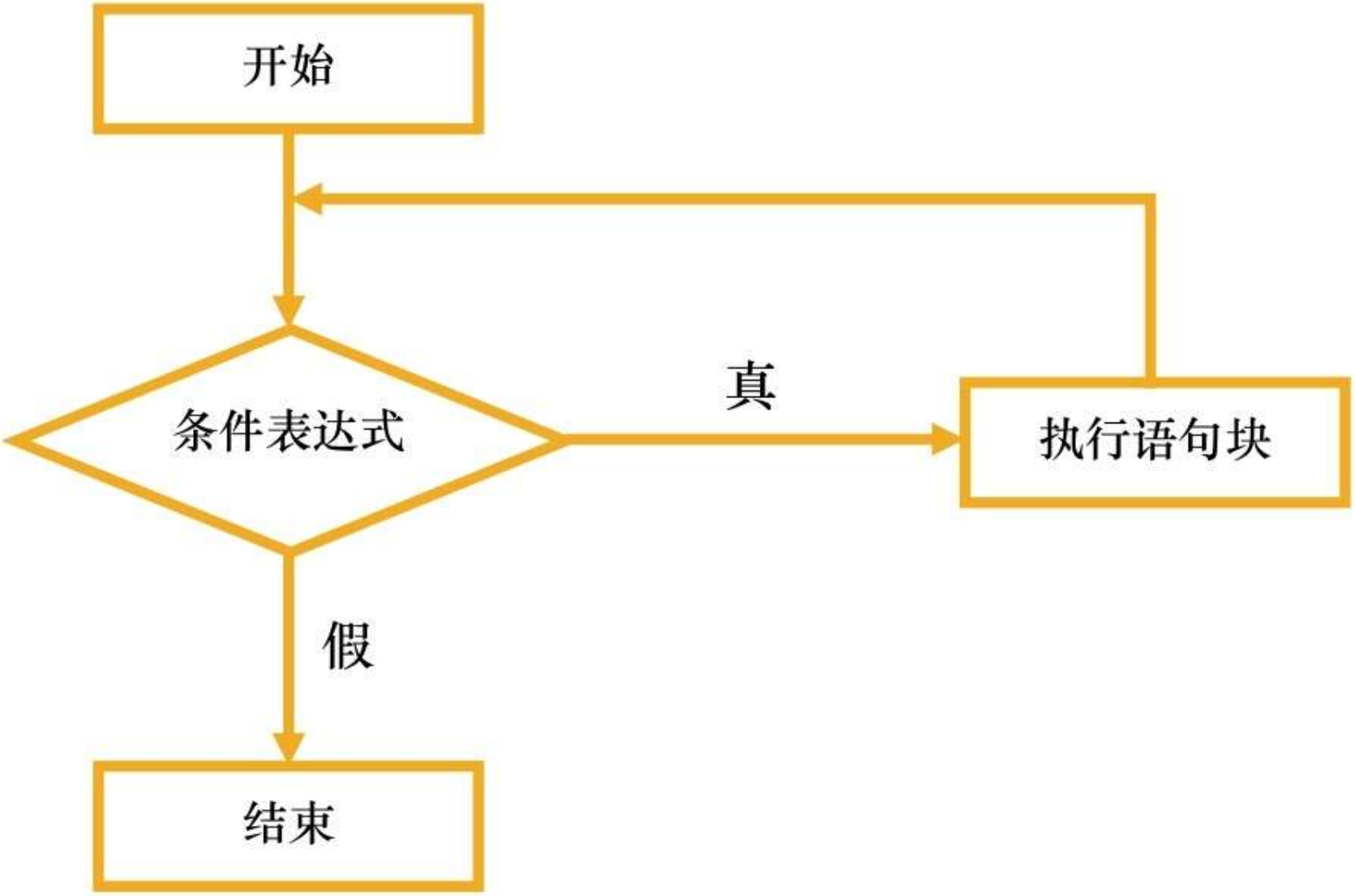
for(初值表达式；条件表达式；运算表达式){　执行语句块；}



foreach(标识符类型 标识符变量名称 in 表达式)
{　执行语句块；}

2.1.5 控制语句——循环（while）

while(表达式){　执行语句块；}

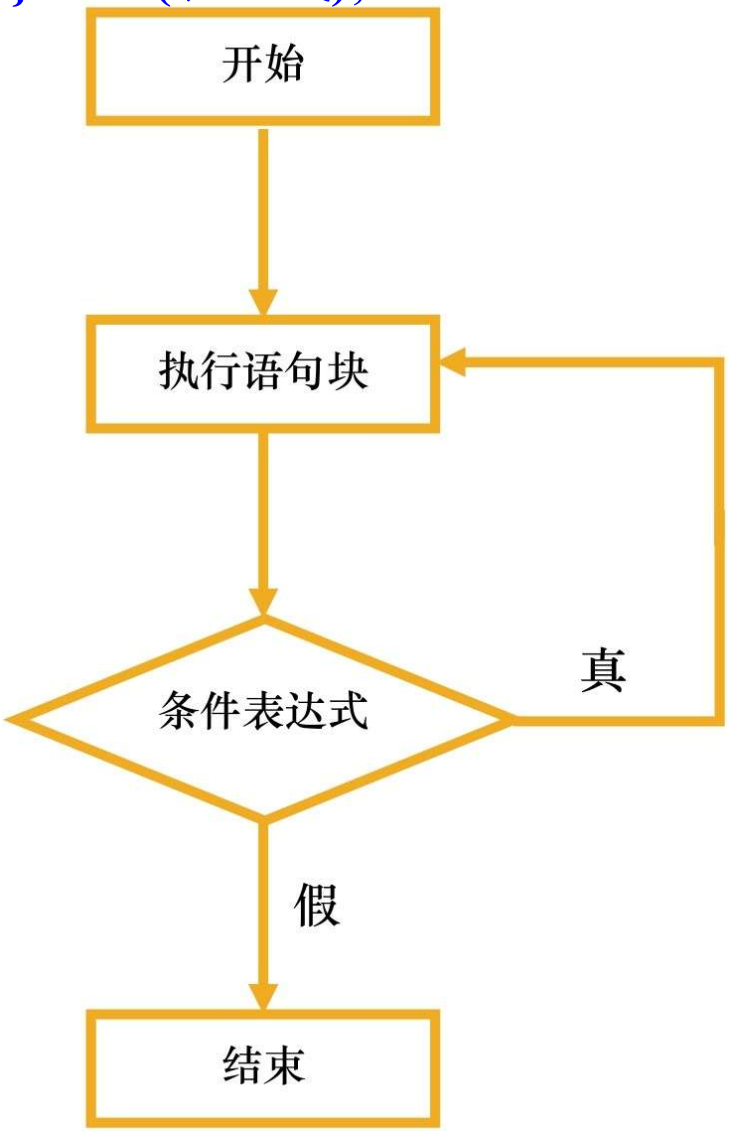


2.1.5 控制语句——循环（do-while）

do{　执行语句块；}while(表达式);

代码演示：

Example3_2.cs



2.2 关键事件回调机制——主循环

在游戏的整个运行机制中，最关键的一个架构是游戏大循环，如果使用C++或者java等实现一个游戏时，基本上都要使用到大（主）循环。但是在游戏引擎中，该大循环已经封装好了，我们只需要调用相关接口即可，所以需要熟悉这些接口的调用时机和作用。

```
const int FRAMES_PER_SECOND = 25;
const int SKIP_TICKS = 1000 / FRAMES_PER_SECOND;
DWORD next_game_tick = GetTickCount(); // 返回当前的系统已经运行的毫秒数
int sleep_time = 0;
bool game_is_running = true;
while( game_is_running )
{
    update_game();
    display_game();
    next_game_tick += SKIP_TICKS;
    sleep_time = next_game_tick - GetTickCount();
    if( sleep_time >= 0 )
     {
         Sleep( sleep_time );
    }
}
```

游戏主循环

2.2 关键事件回调机制——MonoBehavior

Awake()：一个游戏对象被创建之前会被调用一次，主要用于初始化一些参数。

Start()：一个游戏对象被激活时调用一次，也是用于初始化一些参数。

Update()：游戏大循环的入口，每一帧会被调用一次，直到游戏结束。该函数主要用于监听和更新各种事件和参数。

比如监听键盘输入，使游戏对象改变位置等等。

OnEnable()：当一个游戏对象被激活时调用一次。

OnDisable()：当一个游戏对象失效时，但未销毁时被调用一次。

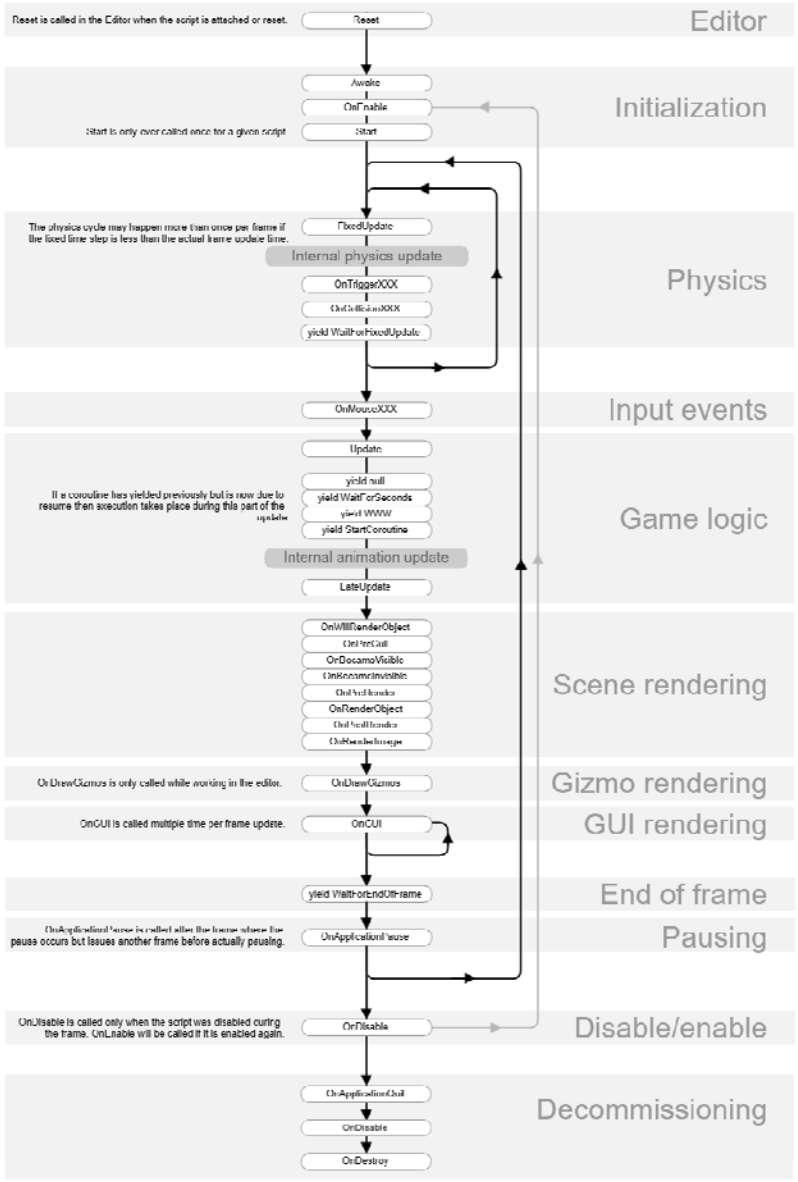
其他回调函数：在Unity中，如果函数前面带有On介词，那么便表示为回调函数。

要使得脚本有以上回调函数功能，该类需要继承自Unity的MonoBehavior类。

其他回调函数可以参考文档：

https://docs.unity3d.com/Manual/ExecutionOrder.html

2.2 关键事件回调机制——MonoBehavior



2.2 关键事件回调机制——MonoBehavior

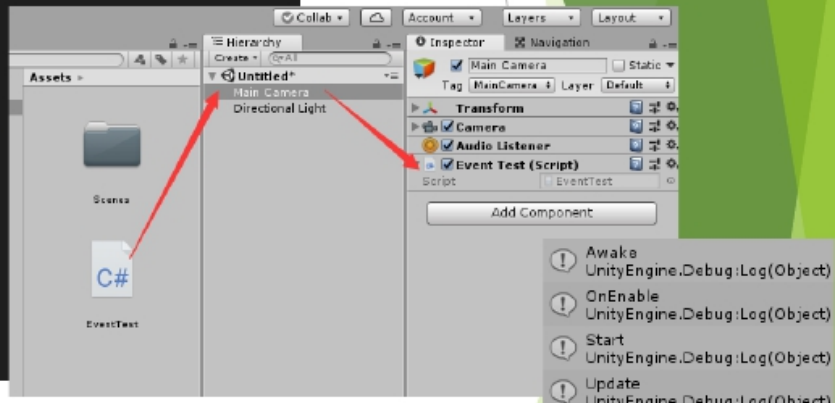
代码演示：

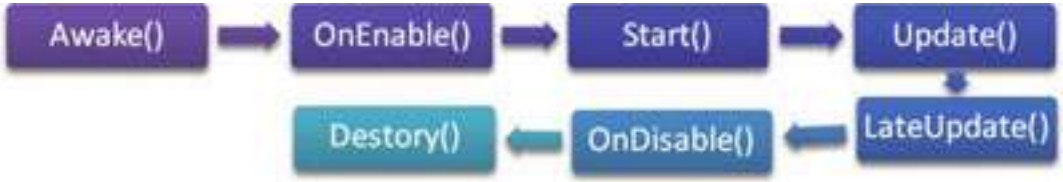
Example3_5.cs

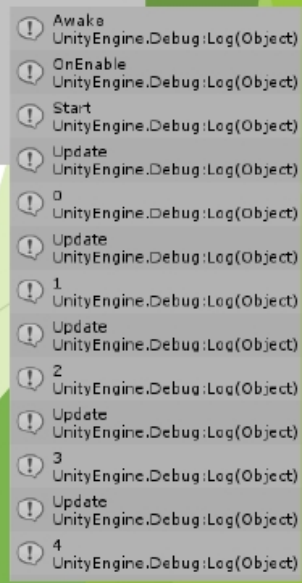
输入代码，保存后把代码挂在场景中的**MainCamera**对象上。

点击运行，观察控制台。

```
1 reference
private int updateCount = 0;
0 references
void  OnEnable()
{
    //Debug.Log函数为控制台输出函数，跟Java中的System.out.print()，类似
    Debug.Log("OnEnable");
}
0 references
void Awake(){
    Debug.Log("Awake");
}

// Use this for initialization
0 references
void Start () {
    Debug.Log("Start");
}

// Update is called once per frame
0 references
void Update () {
    Debug.Log("Update");
    Debug.Log(updateCount++);
}

0 references
void  OnDisable()
{
    Debug.Log("OnDisable");
}
```







2.3 推荐的学习方法

- 在线Unity手册
  - https://docs.unity3d.com/Manual/index.html
- 在线Unity脚本API Documentation
  - https://docs.unity3d.com/ScriptReference/index.html
- 离线文档
  - https://storage.googleapis.com/docscloudstorage/5.5/UnityDocumentation.zip

代码演示：
Example3_3.cs

2.4 线性代数（向量知识）

代码演示：

Example3_4.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Example3_4 : MonoBehaviour {
        private Vector3 a;//向量a
        private Vector3 b;//向量b
        // Use this for initialization
        void Start () {
                a = new Vector3 (1, 2, 1);
                b = new Vector3 (5, 6, 0);
                float c = Vector3.Dot (a, b);
                //向量a,b的夹角,得到的值为弧度，我们将其转
换为角度，便于查看！
                float angle = Mathf.Acos (Vector3.Dot
(a.normalized, b.normalized)) * Mathf.Rad2Deg;
                float distance = Vector3.Distance (a, b);
                Debug.Log ("向量a，b的点积为：" + c);
                Debug.Log ("向量a，b的夹角为：" + angle);
                Debug.Log ("向量a，b之间的距离为：" +
distance);
        }
        // Update is called once per frame
        void Update () {}
}
```

Vector3.Dot(a,b)=0时？

2.5 访问GameObject组件(1/4)

- 在Unity中组件属于游戏对象，组件（Component）其实是用来绑定到游戏对象（Game Object）上的一组相关属性。本质上每个组件是一个类的实例。常见的组件有：MeshFilter、MeshCollider、Renderer、Animation等等。

组件名称	变量名称	组件名称	变量名称
Transform	transform	Rigidbody	rigidbody
Renderer	renderer	Camera	Camera （只在摄像机对象有效）
Light	Light （只在光源对象有效）	Animation	animation
Collider	collider		

2.5 访问GameObject组件(2/4)

- 访问游戏对象组件
- using UnityEngine;
- using System.Collections; //引入系统包
- public class BNUComponent : MonoBehaviour {//声明类
- void Update(){ //重写Update方法
- transform.Translate(1, 0, 0); //沿x轴移动一个单位
- GetComponent<Transform>().Translate(1, 0, 0);
  //沿x轴移动一个单位
- }}

曾经见过：

this.transform.Translate(1,0,0)

2.5 访问GameObject组件(3/4)

■访问其它游戏对象

- 通过属性查看器指定参数
- 确定对象的层次关系
- 通过名字或标签获取游戏对象
- 通过组件名称获取游戏对象

应用情形：区分游戏角色中的队友 **or** 对手

2.5 访问GameObject组件(4/4)

代码演示：

Example3_6.cs

```
...
        public GameObject SomeSphere;

        // Use this for initialization
        void Start () {
                SomeSphere = GameObject.Find ("Sphere1");
                SomeSphere.GetComponent<Renderer>
().material.color = Color.red;
                SomeSphere.GetComponent<Transform>
().position = new Vector3(0,10,0);
...
```

作业3 Unity3D-C#脚本程序之语言基础

- **1.使用print()函数或Debug.Log()函数打印阶梯式的九九乘法表。**
- **2.阅读“Unity 5.x 完全自学手册 - P262 to P266.pdf”，完成：**
  - **（1）重现该游戏实例的效果；**
  - **（2）从网上下载FBX人物或卡通角色模型到场景中，点击运行时实现Cube和模型的同时旋转。**
- 要求：
  - 图文并茂记录整个过程，粘贴完整的代码。
  - 通过网络教学综合平台（**http://jxpt.cuc.edu.cn**）提交作业，截止时间：**2020-04-20、24**时前。

参考文献

- 张帆, 范义娜. Unity5.X游戏开发基础. 浙江工商大学出版社，2017.
- 吴亚峰, 于复兴, 索依娜. Unity3D游戏开发标准教程. 北京：人民邮电出版社，2016.
- 吴雁涛. Unity3D平台AR与VR开发快速上手. 北京：人民邮电出版社，2017.
- Unity公司 主编 史明 刘杨 编著. Unity 5.X/2017标准教材. 北京：人民邮电出版社，2018.
- 商宇浩，李一帆，张吉祥 编. Unity 5.x 完全自学手册. 北京：电子工业出版社，2016.