Cha3-认识Unity3d的好基友C# 01

在上一章的内容中，我们已经使用Unity3d开发了自己人生中的第一款游戏，但是在最后总结的时候也提到了，这款游戏并不完美，甚至一点都不完善。

当然，在这个过程中，我们最大的收获就是熟悉了Unity编辑器的构成和各个部分的大概作用。

但是仅仅使用Unity编辑器的图形用户界面，至少目前还无法开发出一款令人满意的完整游戏。实际上即便在《绝地逃生》这款小游戏里面，也涉及到了跟代码相关的地方，只是我们故意忽略不去看罢了~

在这一章的内容中，我们将进入代码的世界，来一起认识Unity3d的好基友C#编程语言，以及如何在Unity3d的游戏开发中使用C#。

还等什么呢？让我们开始全新的旅程吧~

个人微信号：iseedo

微信公众号：vrlife

01 认识Unity3d的好基友C#-游戏脚本语言概述

在这一课的内容之中，我们主要回答两个问题：

1.游戏脚本语言是干嘛用的？（为什么会有脚本语言这样的物种诞生？）

2.Unity中使用的脚本语言是什么？

游戏脚本语言的诞生

在第一章的内容中，我们介绍过游戏引擎的诞生和发展，这里再重温一下上古时代游戏开发者是如何开发游戏的。

在电子游戏行业的“石器时代”，游戏开发者的日子非常不好过，对游戏开发者的能力要求相比其它类型软件的开发也更高。以早期的Atari 2600游戏机为例，在这个平台上耕耘的游戏开发者必须

精通硬件的底层，包括如何合理使用和显示相关的硬件，并熟悉相关操作系统的内核。苹果公司创始人乔布斯的第一份工作就是给 Atari 公司开发游戏。当然，为其它设备平台开发游戏似乎相对容易些，但即便开发者可以无视显示部分的底层架构，内存的限制和调用又成了开发者头上的金箍圈。

一个游戏开发工程师需要全面了解硬件底层的知识，操作系统的内核，3D 数学，C 或者 C++语言。总之，在游戏引擎出现之前的远古时代，游戏开发绝 B 是一个超级硬核的工作，其难度绝对不亚于如今的 AI 人工智能算法工程师。

进入二十世纪九十年代，一代游戏大神 John Carmack 创立了 id Software，并把目标投向 3D 游戏的研发。在他的带领下，id Software 接连推出了《德军总部 3D》,《毁灭战士》,《雷神之锤》等惊世之作。在开发这些游戏的过程中，卡马克史无前例的采用了自己独创的 3D 游戏引擎。

以《毁灭战士》为例，其软件架构可以清晰的分为三部分：核心游戏组件（如 3D 图形渲染系统、碰撞检测系统、音效系统等）、美术资源和游戏场景，以及和玩家游戏体验紧密相连的游戏规则。通过使用游戏引擎，游戏开发者得以在游戏核心架构不变的情况下设计自己的游戏画面、人物角色、武器和关卡，也就是所谓的“游戏内容”或“游戏资源”。游戏引擎把和游戏内容本身无关的碰撞检测机制、渲染等独立出来，从而让开发者可以专注于游戏内容和机制的设计。

很多游戏引擎都提供了一整套的可视化开发工具，以及可重用的软件功能。这些开发工具通常以一种集成开发环境（IDE）的形式提供，让开发者不必从零开始“造轮子”。游戏引擎有时候又被称作“中间件”，给开发者提供了极大的便利和灵活性，可以有效的提高开发效率，降低开发的复杂度，缩减开发成本。

简单来说，在游戏引擎出现之前，游戏开发者需要消耗极其大量的时间在与游戏体验无关的底层开发工作上，比如图形渲染，音效播放，动画的生成，网络的实现等等。

随着商业游戏引擎的出现，开始出现了所谓 GPP(GamePlayProgrammer)的职位。与此同时，游戏策划设计与美术设计人员也不再被排除在游戏开发的流程之外。

现在的游戏开发分工已经有了很大的进化，底层的渲染、物理机制、音效系统实现等统统由引擎搞定，开发团队的工作重心是如何通过自己的创意和想象力来实现独一无二的“游戏内容”。开发团队中每个成员的工作都以游戏引擎为中心，有了共同的沟通和协作中心。游戏策划设计人员（包括关卡设计）可以使用游戏引擎来设计简单的 demo 原型，美术设计人员可以使用游戏引擎来搭建游戏场景，打造视觉效果。游戏开发人员则把重心放到如何实现游戏的核心逻辑和玩法上。

这里强烈安利一本十几年前的老书《游戏脚本高级编程》(英文名是 Game Scriping Mastery)，在这本书中，详细讲述了如何游戏脚本语言的作用，以及从零开始设计一个游戏脚本编程语言。

和之前推荐的《游戏引擎架构》类似，虽然如今的游戏引擎已经足够强大，我们无需从零开始打造一款自己的游戏引擎。但是了解游戏引擎底层的一些知识，对于我们更好的使用引擎有着极大的帮助。

当然，需要提醒大家的是，《游戏脚本高级编程》和《游戏引擎架构》这两本书都是属于很 NB，但是不太容易看懂的。大家在初学阶段先不要深入到其中的细节，可以当做参考书慢慢看，哪怕你是一个在游戏开发领域有着十几二十几年经验的老鸟，依然可以把这两本书放在自己的案头，以便让容易失眠的自己早一分钟入睡~

所以说，游戏脚本语言到底是干嘛用的？为什么会诞生这个新的物种呢？

简单总结一下吧：

1.随着游戏引擎的诞生和发展，游戏核心架构和底层实现的工作交给了游戏引擎。开发者更加关注的是如何基于游戏引擎实现独一无二的游戏内容。

在早期开发游戏的时候，游戏开发者往往一个人身兼多职，除了实现基本的游戏逻辑机制，还得想办法实现图形的渲染，声音的播放，碰撞检测等等。

而在开发《魔兽世界》、《大话西游》、《王者荣耀》、《绝地求生》等游戏的时候，开发者需要做的是基于策划的新 idea 和美术人员的新场景和人物，把新的游戏内容填充到游戏之中，而不用花费大量的精力来写代码实现如何将角色和场景渲染到设备屏幕上来。

2.几乎所有主流的游戏引擎都是使用 C/C++实现的，包括一些大厂的自有游戏引擎也是如此。但是 C/C++语言属于难以精通（不是难以入门~）的语言，相应的开发人才难以培养。在基于游戏引擎开发成为主流之前，为了保证游戏的高性能，不得不全部使用 C/C++语言开发。但是在游戏引擎兴起之后，跟性能和运行效率相关的工作基本上都由游戏引擎搞定了，开发团队更关注的是游戏内容和游戏逻辑机制的实现。而实现这些东西可以使用相对开发者更友好的语言，如 Lua/Python/javascript/Java/C#等等。

Unity 所使用的脚本语言

首先要告诉大家一个简单的事实，Unity3d 引擎也是用 C/C++语言写的。但是我们使用 Unity3d 引擎来开发游戏却不需要怎么学 C/C++语言，而是用到 Unity3d 所支持的脚本语言。是的，Unity3d 的好基友 C#终于有机会登场亮相了~

首先来看一下 C#在兵器排行榜上的排名吧~

Apr 2019	Apr 2018	Change	Programming Language	Ratings	Change
1	1		Java	15.035%	-0.74%
2	2		C	14.076%	+0.49%
3	3		C++	8.838%	+1.62%
4	4		Python	8.166%	+2.36%
5	6	^	Visual Basic .NET	5.795%	+0.85%
6	5	v	C#	3.515%	-1.75%
7	8	^	JavaScript	2.507%	-0.99%
8	9	^	SQL	2.272%	-0.38%
9	7	v	PHP	2.239%	-1.98%
10	14	^^	Assembly language	1.710%	+0.05%
11	18	^^	Objective-C	1.505%	+0.25%
12	17	^^	MATLAB	1.285%	-0.17%
13	10	v	Ruby	1.277%	-0.74%
14	16	^	Perl	1.269%	-0.26%
15	11	vv	Delphi/Object Pascal	1.264%	-0.70%
16	12	vv	R	1.181%	-0.63%
17	13	vv	Visual Basic	1.060%	-0.74%
18	19	^	Go	1.009%	-0.17%
19	15	vv	Swift	0.978%	-0.56%
20	68	^^	Groovy	0.932%	+0.82%

可以看到，江湖排行榜前三由 Java/C/C++长期霸占，而 Python 凭借在数据统计分析和人工智能领域的异军突起也牢牢把控着第 4 的未知。Unity3d 的好基友 C#则在第 5 和第 6 之间徘徊。

那么什么是 C#语言，它主要用在哪些地方？

用官方语言说，”C#是一种安全的、稳定的、简单的、优雅的，由 C 和 C++衍生出来的面向对象的编程语言。它在继承 C 和 C++强大功能的同时去掉了一些它们的复杂特性（例如没有宏以及不允许多重继承）。C#综合了 VB 简单的可视化操作和 C++的高运行效率，以其强大的操作能力、优雅的语法风格、创新的语言特性和便捷的面向组件编程的支持成为.NET 开发的首选语言。“

虽然说起来这么高大上，其实 C#语言只不过是 IT 界江湖恶斗的产物，甚至本来是可能一直默默无闻的~

C#读作 C Sharp，它有个更酷的名字叫 COOL。早在 1998 年 12 月，微软就开启了一个名为 COOL 的业余项目，2000 年 2 月更名为 C#，当然代表它的身上有 C 语言的高贵血统了~

为什么会有 C#的诞生呢？主要原因就是微软跟 SUN 公司（如今已经倒闭了）的撕逼。微软的一员大将基于 Java 语言开发了 Visual J++，并且成为 Visual Studio 全家桶的重要组成部分。SUN 公司认为 Visual J++违背了 Java 平台的中立性，发起了对微软的法律诉讼。就如同如今苹果和高通的专利大战大战一样，苹果相比高通实乃巨人也，但却就是没有自研的基带芯片，只有任人吊打。当年的微软也是如此，虽然实力远超 SUN 公司，但奈何可以跟 Java 抗衡的拿得出手的开发语言。痛定思痛，微软在 2000 年发布了全新的语言 C#，用来取代 Visual J++。因为历史的原因，C#语言深受 Java、C 和 C++语言的影响。

C#语言可以用来编写 Windows 平台应用、服务器端和数据库应用等等。

不过对于游戏开发者来说，则幸运的看到 Unity3d 如今已经把 C#语言作为唯一支持的脚本语言了~

跟汇编语言，C/C++语言相比，C#和 Java 的速度可能没有那么快，但是易上手，也容易掌握。

而跟 Python/Perl/Lua/Javascript 这些解释型动态语言相比，C#的效率更高。

One more thing…

虽然我们的课程是面向小白读者的，但是在谈到 Unity3d 和 C#的时候，不得不提到 Mono 这个词。

实际上，Unity3d 之所以可以支持跨平台开发，就是因为 Mono。

Mono 是一个由 Xamarin 公司（先前是 Novell，最早为 Ximian）所主持的自由开放源代码项目。该项目的目标是创建一系列匹配 ECMA 标准（Ecma-334 和 Ecma-335）的.NET 工具，包括 C#编译器和通用语言架构。与微软的.NET Framework（共通语言运行平台）不同，Mono 项目不仅可以运行于 Windows 系统上，还可以运行于 Linux，FreeBSD，Unix，OS X 和 Solaris，甚至一些游戏平台，例如：Playstation 3，Nintendo Wii 或 XBox 360。

Mono 基于.NET 和 CIL(Common Language Infrastructure，通用语言架构，又被称为 CLR)，允许 Unity3d 引擎和用户的托管代码运行在每一个目标平台上。

Unity 实现跨平台的主要机制是 Mono 或 IL2CPP：

1.编译

在运行前把 C#通过 Mono compiler 编译为 CIL，或由 IL2CPP 再转成 CPP；

2.运行

在运行时加载 CIL 或 CPP，由虚拟机转换成各平台的原生码给 CPU 执行。

参考：

https://baike.baidu.com/item/mono/60423

https://gamedev.stackexchange.com/questions/140493/difference-between-unity-scripting-backend-il2cpp-and-mono2x

关于上面这段话，如果你看不懂，Don’t panic，完全正常~

在后面的内容中，我会对一些重要的概念反复提及，直到你可以完全理解为止。

好了，简单总结一下。

在这一课的内容中，我们主要回答了两个问题：

1.什么是游戏脚本语言

2.Unity3d 中支持的脚本语言是什么

如果你还有不明白的地方，或者想更加深入的探索这两个问题。

那么建议：

1.认真阅读《游戏脚本高级编程》

2.通过 google 或 stack overflow 搜索 Unity3d,mono，IL2CPP 等关键词，来了解 Unity3d 和 C#,MONO,IL2CPP 的底层机制。

但是对于普通的小白读者，建议先到此为止。

从下一课开始，又到了我们的动手时间了~

让我们下一课再见。