《游戏引擎架构》简要分析

深互动设计硕 20 班 沈琪川 2020214425

一、什么是游戏引擎

1、定义

在分析游戏引擎架构之前，先给一个游戏引擎的定义。游戏引擎又被称为游戏架构、游戏框架，是为了人们制作电子游戏而设计的软件开发环境。

图 1-1 用游戏引擎渲染的画面

2、目的

游戏引擎的一个重要的目的就是提高游戏开发的效率。

很多游戏引擎除了基本软件组件外，还提供了一套可视化开发工具。这些工具通常在集成开发环境中提供，以实现快速的游戏开发。大多数游戏引擎都提供简化开发的功能，例如图形，声音，物理和人工智能（AI）功能。它们提供了一个灵活且可重用的软件平台，开箱即用，降低了成本。

二、结合 Unity 的游戏引擎架构分析

1、游戏引擎运行时的架构

分别为：

- 系统、驱动、硬件层
- 独立平台层、第三方软件开发包
- 核心系统、资源管理系统
- 主要引擎模块
- 游戏专用子系统

如图 2-1 所示。因为内容比较多我会找一部分来分析一下，下面结合 Unity，讲讲它的主要引擎模块。

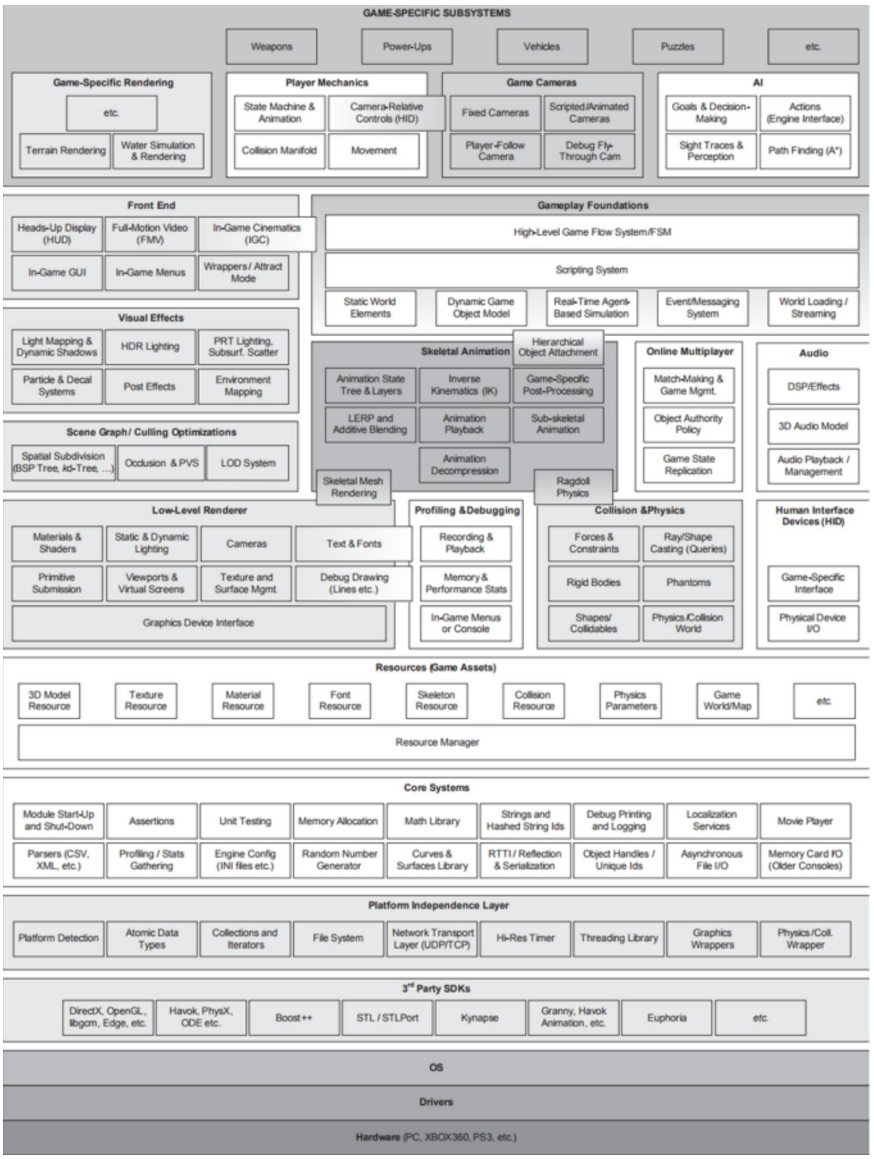

图 2-1 游戏引擎运行时的架构

2、Unity 的主要引擎模块

- 渲染引擎

渲染引擎是最大最复杂的组件之一。渲染器有很多不同的架构方式，但大多数现在的渲染引擎都有一些通用的基本设计哲学和模块，由底层的三维图形硬件驱动形成。低阶渲染器将高速渲染丰富的几何图元，而并不考虑场景是否可见；而在渲染管道方面，就拿 Unity 来说，进行剔除、场景图、后期处理等操作，将你想要的视觉效果显示在屏幕上。

不同的渲染管道具有不同的功能和性能特征，并且适用于不同的游戏，应用程序和平台。我了解到，Unity 提供以下渲染管道：

- 统一的默认渲染管线。它是通用的渲染管道，自定义选项有限。
- 通用渲染管道（URP）允许创建跨多种平台优化的图形。
- 高清渲染管道（HDRP）是一个脚本渲染管道，可以让你创建前沿的高保真图形。

- 物理引擎

物理引擎往往和碰撞紧密结合，在现在很少有游戏公司会自己编写碰撞及物理引擎，取而代之的是第三方的物理 SDK，比如 Havok 或者 PhysX 等等。

在 Unity 中，它提供了不同的物理引擎，用户可以根据自己的项目需求使用它们：3D，2D，面向对象或面向数据。

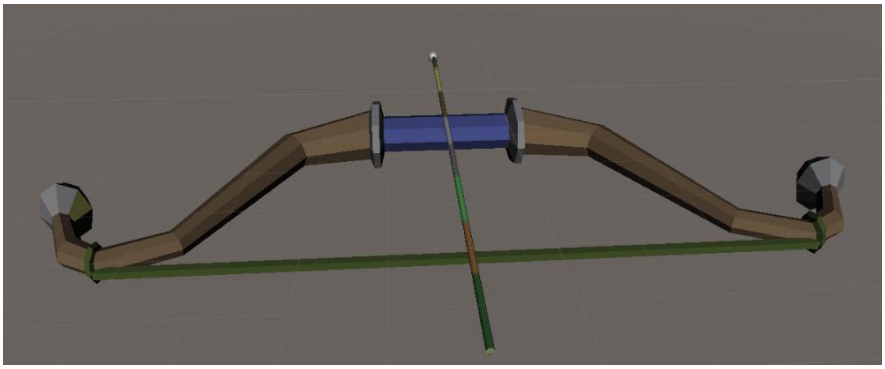

图 2-2 运用了物理碰撞的弓箭 demo

面向对象的内置物理引擎：

- 内置 3D 物理（Nvidia PhysX 引擎集成）
- 内置 2D 物理（Box2D 引擎集成）

面向数据的物理引擎软件包：

- Unity Physics 软件包：默认情况下，需要安装 DOTS 物理引擎才能在任何面向数据的项目中模拟物理。
- Havok Physics for Unity 软件包：用于 Unity 的 Havok 物理引擎的实现，用作 Unity Physics 软件包的扩展。

- 人体学接口设备

很多游戏引擎都可以外接设备，Unity 也是一样。我的毕业设计在 Unity 中运用了 MR 硬件设备 Microsoft Hololens,是因为 Unity 里有输入/输出的接口。当然还有许多最简单的设备比如鼠标、键盘、游戏手柄等等。

当然还有其余的引擎模块，这里就不一一赘述了。

- 音频引擎
- 动画引擎
- 网络系统
- 游戏性基础系统

三、在 Unity 中进行操作

在 Unity2019.4 版本中，我用 Tilemap 自己制作了一张竖版的 2D 地图，并给砖块元素加入了物理碰撞控制。而后导入了一个小女孩的模型资源文件，编写脚本控制她的跑跳，实际运用了一下 Unity 中的物理引擎。

图 3-1 小女孩站立的地面上

图 3-2 小女孩跳跃