Introduction to the Entity Component System and C# Job System

---

Unity 做游戏的经典方式

- MonoBehaviours (使用GameObjects和MonoBehaviour)
- Data / Processing Coupled (数据和对数据的处理 高度耦合，即都放在MonoBehaviours中)
- Largely dependant on reference types (很大程度上依赖引用类型)

这种经典做法会造成数据分散(Data is scattered)。代码中的GameObject相当于一个容器，其中存放的是指向别的内存块(memory)的引用(references)，然后，这些内存块中存放的又可能是其它内存块的引用。这样做会造成从内存(memory)向缓存(cache)加载数据会很慢。另外，每次加载数据也会加载一些当前用不到的数据，比如，你需要用到一个GameObject的Postion信息，你就需要加载transform组件，然而transform中还有许多用不到的数据。

Job System 概述

Job System是一种把数据和功能分开，利用多核处理器的方式，可用来安全、简单地写多线程程序。

- Separate data from function
- Multi-core processing
- Save multi-threading

使用Job System 的好处

使用多线程时会遇到以下问题:

- Thread safe code is difficult (线程安全，如：不能两个线程同时修改一个变量的值)
- Race conditions (条件竞争)
- Context switching is expensive (环境切换代价昂贵)

Job System可以为你管理这些issues，而你只需要专注于你的游戏代码。你想让某个线程发生一些事时，直接让Job System去做就行了。用Job System写线程安全的代码很容易。