# **Unity游戏框架概述**

作者：[vian](http://gad.qq.com/user/index?id=55567)

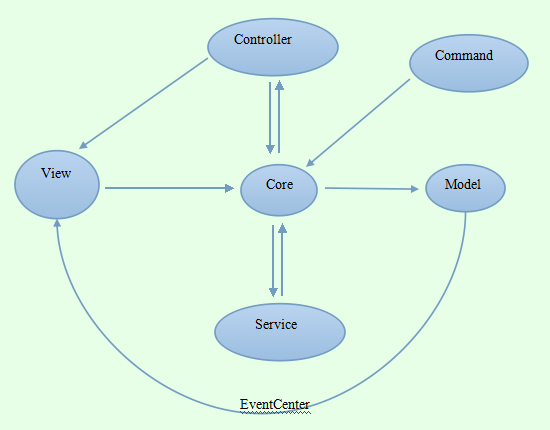
链接：<http://gad.qq.com/article/detail/287367>

**前言：**

设计框架就好像盖房子打地基，说到框架我们可能一下子会想到好多特征要求，稳定、安全、健硕、简单、灵活、易用等等，的确这些都是一款好的框架就应该具备。这里不大谈设计模式、不大谈老的mvc框架等等，先给大家分享一款游戏框架，一起来讨论分析优缺点，希望能在实际运用中给大家带来帮助。

**1.框架图解**

整个框架核心如下图所示：



**2.框架详细介绍**

通过上文图解，我们能比较清晰的看到各个层次之间的关系。可能不一定能看明白整个框架要表达的意思，框架是怎么运作的？下面来进一步介绍。

2.1）Core，我们可以往Facade设计模式来理解，它提供了底层核心接口，通过其接口我们可访问到Controller、Model、Service。通常其中各接口可设计为静态供外部直接使用。

2.2）Service，主要处理网络相关，用来发收协议。可通过Core可获得所需Model，然后去更新Model里面的数据。一般不需要通过Core去获得Controller。

2.3）Model，主要负责数据记录，数据变化可用事件派发出去，谁监听谁处理。

2.4）Controller，主要负责控制视图还有其他逻辑处理，以及接受相关Command对应要做的逻辑。它也能通过Core访问到所需Model、Service。

2.5）View，即视图。比较独立，设计时我们要求视图里面尽量不参杂逻辑代码，一般逻辑都可在Controller去做，如把方法作为参数传给View，这样View只需要回调就可以。View还可以在打开状态时监听所需事件，数据更新由对应Model派发事件来驱动更新。为了使用更方便，我们让View能通过Core去获取所需，但这样带来的问题是View能通过接口获取更多资源。

2.6）Command，它里面不访问任何接口不做具体逻辑仅由Core去执行，最终逻辑处理在Command对应的Controller里面。所以我们可以在不同的地方通过Call不同Command去做一些我们想要的事情。如背包模块里背包View有个商店快捷按钮，我们点商店按钮要做具体逻辑可不用关心，只需要找到并Call商店模块打开商店View的Command即可。Command模式使用非常方便，给各模块之间通信带来很大便利，且逻辑分离省心省事。

2.7）EventCenter，事件中心用来监听、移除、派发事件，可设计为单例模式。其使用很方便。Command也可以用来消息通讯，实际项目中可综合考虑按需使用。有关EventCenter详细介绍在http://gad.qq.com/article/detail/48139。

以上，通过图解和详细介绍，希望大家理解了整个框架层次和运作。

**3.优缺点讨论**

优点：

设计符合MVC理念，数据视图逻辑的分离；

框架层次清晰、实用简便易用。

缺点：

核心Core提供的接口开放性太高，一定要严格注意规范使用；

Command需要找到对应Controller，在找到Controller里面对应的Method，注意消耗，注意缓存Method，注意Command命名避免有相同。

等等，欢迎补充。

补充下Command和EventCenter对比：

EventCenter使用前后要注意先监听、在派发、不用后要移除；

Command随时Call即可。

**4.总结**

没有一套万能的、一劳永逸的框架，但框架大体设计理念都一样，只有能满足实际开发所需、适合自己项目的框架那就是好框架。