**引擎消息中心和有限状态机优化-初稿**

* By awen

**一 什么是有限状态机**

**FSM （**finite-state machine），又称**有限状态自动机**，简称**状态机**，是表示有限个状态以及在这些状态之间的转移和动作等行为的数学模型。他对于逻辑以及时序的控制能起到非常重要的作用。

代码主要看的是什么？逻辑！所有的设计模式无非是让程序逻辑变得更加利与维护，利于优化，利于升级而已。

那么能不能把业务逻辑整理出来统一操作呢？

你可以理解状态机就是干这个的（当然不仅仅是这样）。这样一来只要将一个业务看成一个状态机，有N中状态，然后统一来控制这些状态时的操作。

大家都用过dom事件绑定，是不是某种状态时候就会触发某种事件？这明显也是状态机的概念能解释的。还有观察者模式或者说是订阅发布模式，抑或是jquery的deffered和pomise，甚至是各种同步编程模块（windjs）等等等等。仔细想想也基本上都是同样的原理：即根据不同的消息，诱发注册的相应回调。区别仅仅是怎么注册或者说生成支持各自机制的对象而已。

将各种业务状态写在一起集中管理的好处是什么？就是你整个流程会变得无比的清晰，一目了然！而不再需要将业务逻辑写的到处都是。方便了整体业务逻辑的管理。（后期会抽象出消息机制概念，彻底解放模块之间的耦合，敬请期待）

**二 状态机的一些概念**

1 状态转换

从一个状态切换到另外一个状态被称为状态转换

2 触发事件

状态改变而引起它的事件称为触发事件.，而触发时间完全可以集中在一起定义，方便管理.业务流程一目了然。

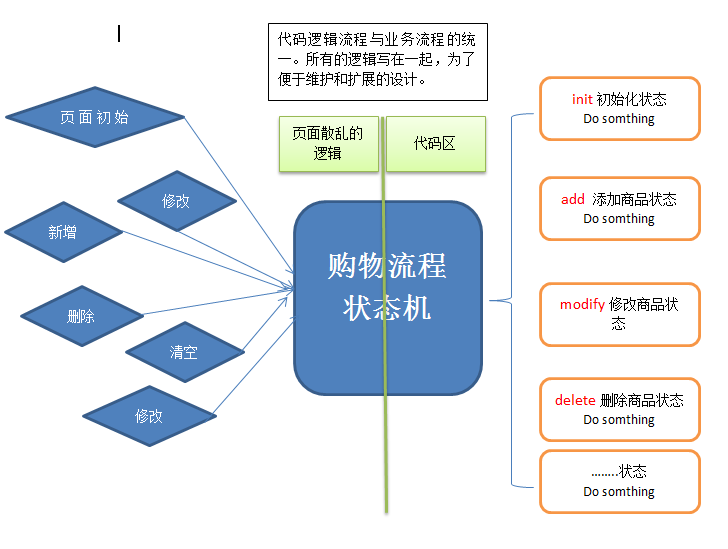
**三 会用在哪些场景**

状态机可以用在任何能抽象出业务流程的地方，大到页面加载，引擎加载，购物流程，小到tab标签，显示隐藏控件，等等地方。

会不会经常被一大堆的回调函数，异步函数回调之间的逻辑高的头昏眼花？抽象出一个状态机来，外部的各种操作（异步，同步，回调）只需要改变这个状态机的状态就可以执行相应的操作了。

**四 举个例子**

1 购物流程

****

2 比如我们引擎的业务流程：

--初始化引擎（initcore）

--初始化模板（inittpl）

--获取数据（initdata）

--初始化页面（initpage）

--初始化页面成功（pageok）

我们可以将这个业务流程生成一个状态机，比如叫“app”，那么我们需要做的是什么呢？

我们只需要将这些状态赋予这个叫做“app”的状态机对象，同时生成一个状态改变时的触发事件管理器。就可以方便并集中地来管理整个业务流程了。逻辑代码展现大概如下：

var fsm = FSM.create([’initcore’,’inittpl’,’initdata’,’initpage’,’pageok’],

function(from,to,data){

switch（to）{

case ‘initcore‘:

break;

case ‘coreok’:

break;

….

case ‘pageok’:

break;

}

});

这样一来整个app加载流程就会一目了然，而不管是在异步中还是页面回调中需要业务变更时，只需要改变对应状态机的状态就会执行相应的逻辑。比如：

fsm.change(‘pageok’);

我们可以称之为面向业务的状态模型，也可以称之为状态模式。

**下面，我们再来看看消息中心的想法。**

**五 关于状态机为核心的消息中心抽离**

**5.1让我们看看以前的模块式开发**

以前引擎和模块，模块和模块之间，一般都是互相直接调用接口，也就是api模式。这就是面向对象基本模式了，每一个模块都是一个对象，对外开放了很多接口，其他对象只需要调用目标对象的相应接口就可以执行相应的逻辑。目前来说，基本上都是这样的设计模式，这种方式为我们带来了很大的便利（图）。

接口

接口

接口

接口

接口

模块

模块

接口

**5.2提出问题**

但是这样没法避免的一个问题就是，在一个模块内部需要直接调用另外一个模块的api，你中有我，我中有你。试想一下，如果另外一个模块不存在，接口不存在呢，或者接口返回错误的情况下该怎么办呢？不至于每个调用其它模块的地方全部都写上容错吧，那么代码基本没法看了。

那么我们就该想想了，是不是应该有这样一套机制呢，暂时叫做调度中心,我们简单设想一下它应该有一下功能：

1 调度中心能统一的管理引擎以及各个模块之间的相互调用，以方便记录当前应用运行的的各种实时操作。

2 所有的模块不会直接调用其他模块的接口，而是告诉调度中心，我要调哪个模块的哪个方法，并提供必要的参数,调度中心会解析消息并调用相应的模块，当然必要的错误处理机制是必须的。

3 每个模块必须有相应的消息处理机制，有调度中心统一方法“吃入”模块，当然必要的配置文件是必须的。

**这就是我们说的消息机制。O(∩\_∩)O~**

**5.3什么是消息机制**

每个引擎都会有很多很多的模块，好的引擎模块间的耦合会比较松，但是大多数引擎模块之间都是结合的很紧密的。

消息，很多情况下都跟通信相关。那么消息机制应用到引擎中是什么样子呢？（图）

3 将消息处理结果返还消息中心,可能包含错误信息。

4 将结果以消息发回模块

模块

模块

消息中心

错误？返回

2 解析消息，发送到相应模块（如果有）

1 发送消息到消息中心

**5.4要注意的问题**

1. 流程上增加了复杂度，开发一个高效的消息中心很重要
2. 每个模块需要注册进消息中心
3. 消息错误时 要有统一处理机制

**总结：**

**想一下，状态机作为消息中心的核心引擎来实现消息中心思想，是不是很给力呢？**