# iOS重构方案

针对之前的问题文档，并且结合现有代码，指定这个重构方案，主要还是参考现有代码，包括素材以及一些细节。

## 浏览器环境选择

依然是webkit不会变，Chrome的速度很快，硬件加速支持很好，对HTML5以及CSS3的支持也不错，safari上的效果也完全不输chrome，并且主流智能手机内置webkit内核，或许有一天真的可以转移到移动平台上去。

## 开发语言

有些偏向于用CoffeeScript开发，但是学习新语言以及新的语法都需要一定的时间，并且ruby式语言也许并不适合我，习惯于c/java语法转移过去可能并不好适应，不过CoffeeScript语法精简干练，并且内置面向对象支持，而且还能较好地与jQuery共存，需要好好考虑。

## MVC

已经确定将采用MVC来写，接下来就是选择MVC框架的问题，目前有如下选择：

1. Ext内置MVC。
2. JavascriptMVC。
3. 自己管理各模块。

目前使用的面向对象模块就是从Ext里剥离出来的，若采用Ext的Mvc，则可以直接采用整个Ext模块，也可以使用其中一些已经开发好的组件，省去很多事情。但是Ext本身非常庞大，未压缩版本高达10万行代码，即使压缩版本也超过了1M，对于一个js文件来说超过1M是很恐怖的事情，下载它也许会花很多时间，即使在本地加载可能影响也会比较大。不过由于对Ext较熟悉，并且Ext一直在更新，目前还是暂时先采用Ext的Mvc，并且动用Ext全部组件，在写下第一个modul之前，若看到比较好的mvc框架，也会考虑替换。

加了一个第三选项，自己还是比较喜欢控制所有的代码，各个框架虽然将模块间的关系处理得很好，但是或许有些东西和自己选择的别的类库重复了，到时候也许会变得比较麻烦，总之这也作为一个待定项吧。

Mvc目前还是不用框架了，自己进行管理也还是可以的，而且各框架都会做一些自己不想他做的事情，自己还是比较喜欢掌控一切东西，高于浏览器层面的东西其实都想自己来控制，以达到最高的效率，出问题的时候也可以自己快速找到解决办法。

加入store，专门用来与后台进行通信，用来存取数据。

## 架构

之前因为架构设计不好，而在后期制作的时候对架构改动比较大，特别是在设计cardView的切换逻辑时，花费了很长的时间，并且现在也并没有达到很好的效果。Sencha Touch有一个demo对iOS进行比较好的模拟，可以参考一下那个demo的逻辑设计。

## 面向对象模块

面向对象模块还是会采用原先Ext的模块。毕竟对这个模块比较属性，而且使用了这么久下来也没遇到什么问题，关键现在已经较好地剥离了下来，代码量也还很客观。

## 代码托管平台

Google Code使用很方便，也很喜欢Google向来的风格，但是无奈天朝过于V5，致使Google code连接老是成问题，有时候连提交和同步都是一个问题，虽然在sae上也有备份，以后准备采用Github为主，Google code为辅，而使用sae完全做备份的方法，最大限度保证随时都可以更新和同步代码。

## 模块化

Seajs还是不错的，国产，api简单，使用方便，而且目前没有与自己期望的冲突的地方。

## 动画类

将所有动画展示效果封装起来还是有必要的，Ext内置的不错，就是有些效果没有满足自己的要求，或者尝试修改Ext源码，或者自己写一个，有了以前的经验，应该不是会很难。

新增一个iOS.Animation的全局动画效果，包括拖拽等的一系列效果。

## 事件类

这次开始把移动设备上的事件也加进去，封装个事件类，iOS.Event，判断下客户端类型，自动在移动设备和桌面设备上进行切换。

## 后台

后台很纠结，没有后台的话，数据存储是一个问题，localStorage还是觉得不靠谱，如果加后台的话，又太依赖环境了，html5的本地文件I/O貌似也只能在一个沙箱里面工作。或者就只能用nodejs，把nodejs文件目录一起提交，然后在宿主机上装一个nodejs就可以了，把数据存在json文件里面，利用文件夹来做层级目录，如此看来，后台这一块的工作量也不少。

## 属性与方法的访问

**属性：**

所有对象不对外暴露任何属性，统一按照如下原则来存放属性：

public ： 使用get+属性名的方法来获取共有属性，

**（修改日期：2012年10月27日星期六）**

**~~protected ： 存放在inheritableStatics中，~~**

**~~private ： 存放在statics中~~**

Ext的类中，若在实例对象中动态改变statics的值，则会对所有该类实例化的对象甚至包括类本身都产生影响，困扰了一天的一个问题，那么，各种值的存储方式如下：

Protected ：inheritableStatics,和之前一样

Private(**变量**) **:** values

Private(**直接量**) ：statics

**Statics属性仍然继续使用，只作为直接量的存储地方，this.self从此就作为只读属性，不可以进行动态修改。**

**方法：**

本来想把方法也写在那里面，但是一想肯定不行，对象还是按照如下规则：

public ： 没有修饰符，

protected ： 方法名字前加”\_”，

private ： 方法名字前加”\_\_”

## 目前待定事项

数据存储方式：localStorage 或者别的形式，数据类的设计，图片的缓存。