Wallist 项目文档及晋级中技术学习点总结

# 项目简介

开发工具：[Sublime Text3](http://www.sublimetext.com/)

项目管理工具：GitHub for Windows

语言：HTML，JavaScript

主要技术点：HTML5，JavaScript 前端框架 Knockout.js

项目GitHub地址：<https://github.com/Wayou/Wallist/>

项目网页：<http://wayou.github.io/Wallist/>

# 使用文档

## 开始使用

### 本地

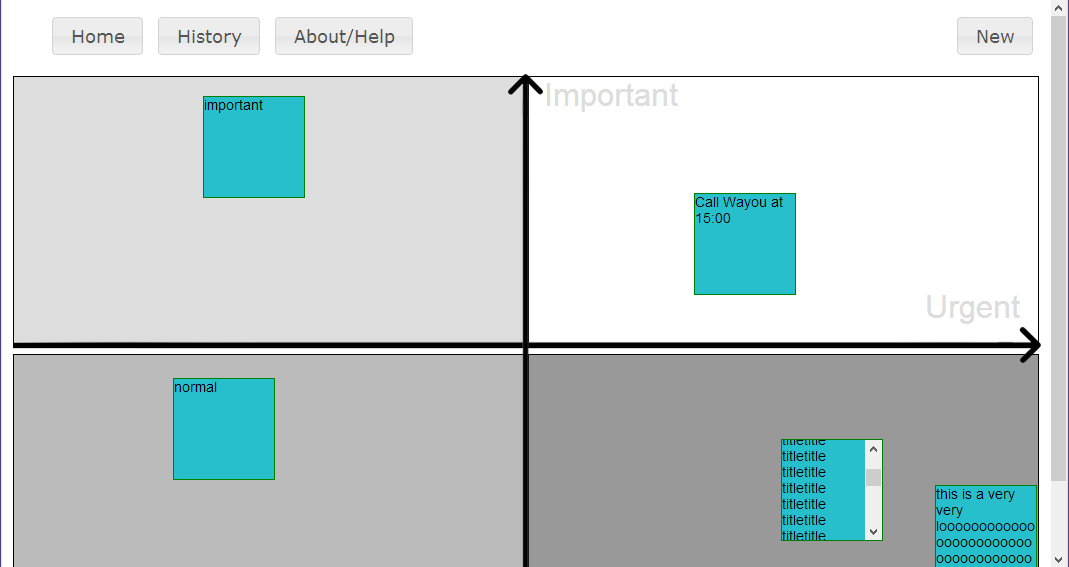
1. 测试支持的浏览器：Chrome，FireFox，暂不支持IE本地运行
2. 已解决了Chrome本地运行报错的问题，现在无需以flags方式启动Chrome了。
3. 本地直接运行index.html文件 。
4. 初始无数据，在页面加载时进行了判断，当用户创建了一个事件项后，下次再打开网页则数据一直存在，直到用户删除并且在History页面清除。

### 服务器端

1. 如果将程序挂在服务器上运行，则不存在本地安全性问题，所以所有浏览器都可正常使用
2. 如果是Windows环境+IIS，直接将项目文件夹复制到“C:\inetpub\wwwroot”，在IIS中确保默认网站处于运行状态。
3. 在浏览器中直接访问localhost/wallist即可打开程序

## 主体界面

主界面采用象限将任务划分为不同优先级。



* 其中第一象限代表既重要又紧急的事项
* 二象限为重要事项
* 三象限为正常优先级的事项
* 四象限为紧急事项

在主体界面，一共有四个按扭：

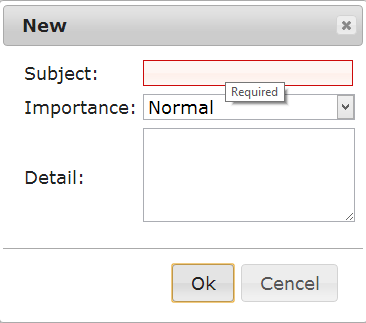
* 主页：刷新或显示主页
* 历史页面：转到历史记录页面，这里会显示已经从主页面删除被标记为完成的事项
* 关于页面：这里提供一些关于程序的相关信息及使用帮助
* 创建按扭：创建一个新的事件项

### 事件项创建

点击创建，会弹出一个新窗口，这里填写新事件项相关信息。

其中包括：

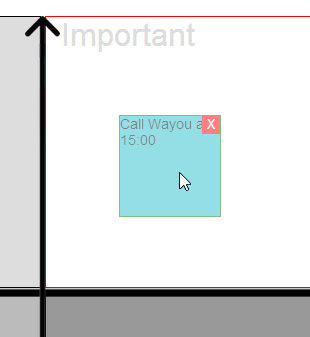
* 事件项的主题，该字段为必需
* 事件优先级，必需
* 事件项详细描述，将会以tooltip的形式在主页面展示，可选



### 事件项移动修改优先极

本项目最大一个特色是支持对事件项进行拖拽进行优先级的更改。

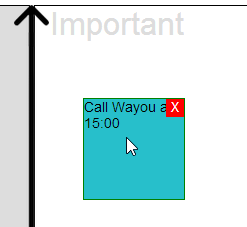
当一个事件项从一个象限移动到另一个象限时，该事件项的优先级随之更改，并且会自动保存，下次打开时该事件项会显示在新的象限里。



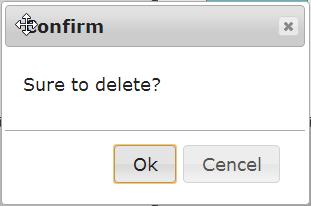
移动时进入的象限边框会以红色显示。

### 事件项的删除

将光标移动到一个事件项上时，删除按扭会出现。

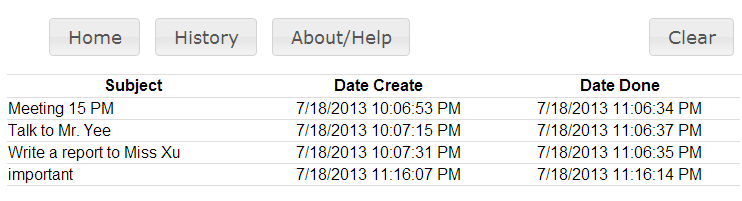


单击会出现删除有确认的窗口，单击“确定” 则该事件项会被标记为已完成而被移动到历史页面。



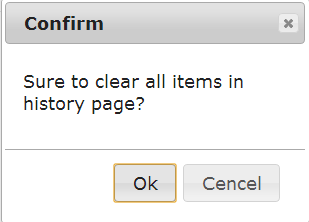
## 历史记录页面

可在历史页面中查看已经完成的所有事项。



本页面包含与主页面相同的前三个按扭，最右边包含一个清除按扭，用以清空历史页面。

单面会出现确认窗口。



点击确认则历史页面中的数据会被清空，更改也会保存进数据，操作无法撤消。

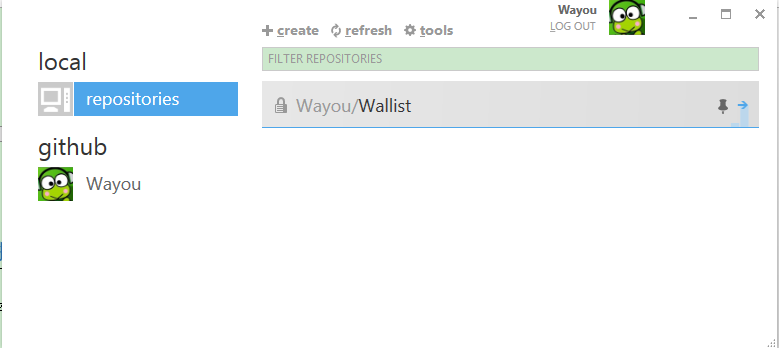
# 技术介绍

本项目中体现晋级计划中关于技术学习方面的技术点。

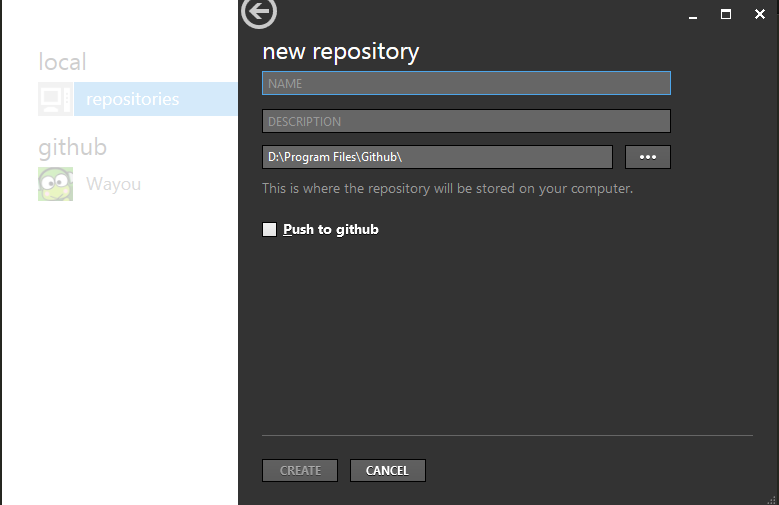
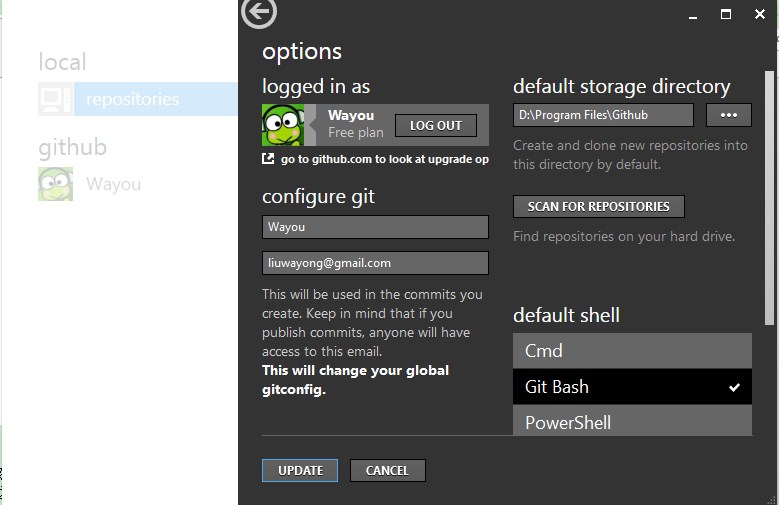
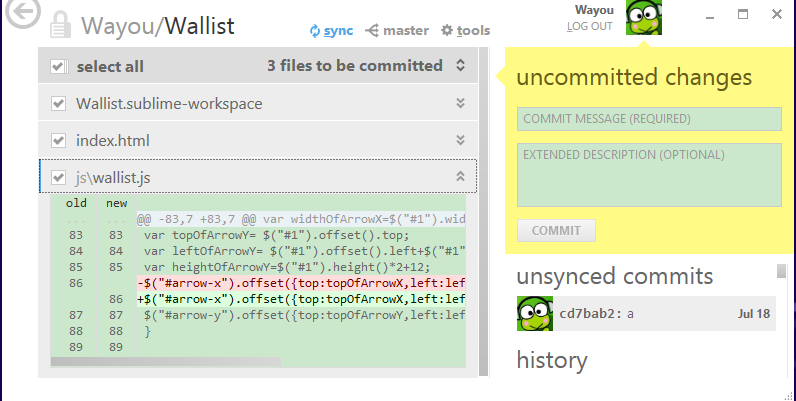
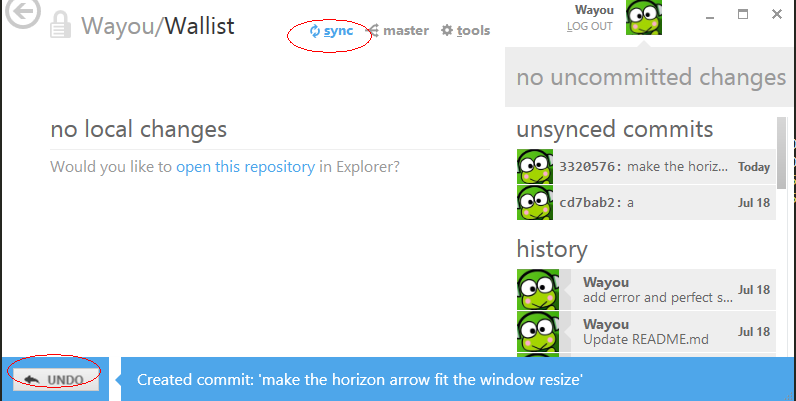
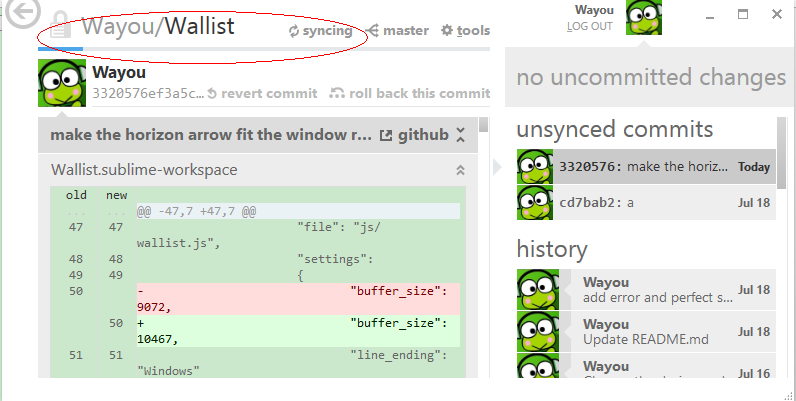
## GitHub项目管理

这是在进行项目过程中额外学到的，对于以后进行开源项目的贡献会有帮助。

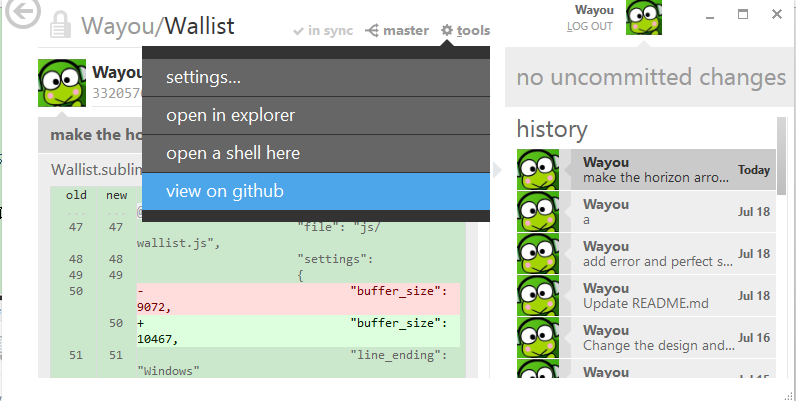
关于Git的使用，以后会写一篇详细的教程分享出来。这里简要介绍下项目过程中所学到的。

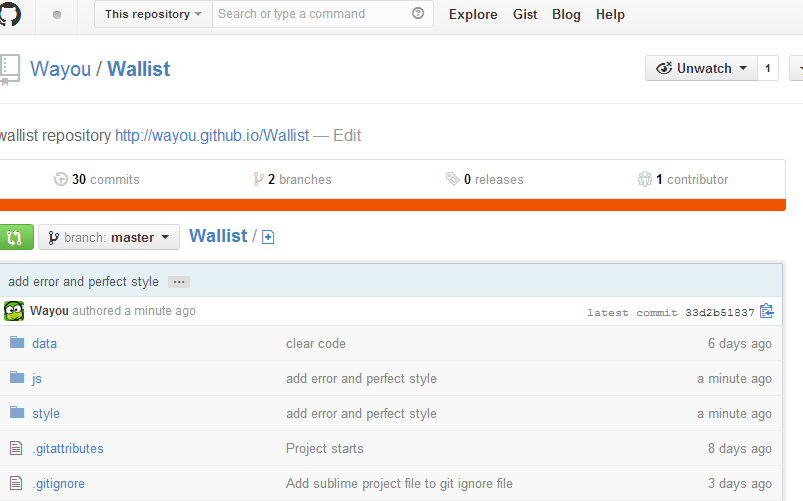


### 本地

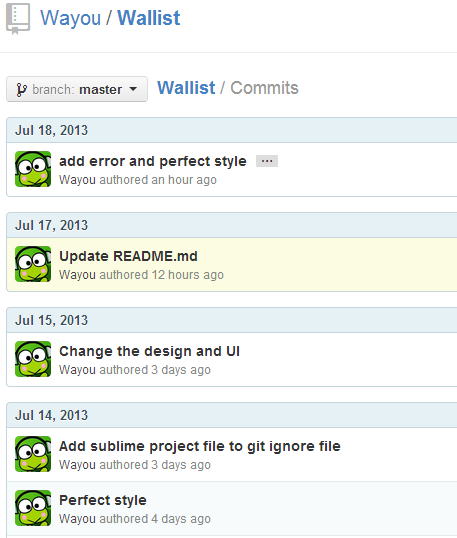
1. 首页需要在本地创建Repository. 最快捷的方法是将项目文件夹拖动入GigHub for Windows中。  
   
2. 然后进行好相关设置，以及连接GitHub服务器的帐号等。  
   
3. 之后便可以对本地文件进行添加，修改，删除等操作。比如在Sublime中进行代码的编写。
4. 然后系统会检测到这些改动，显示可以提交的文件，并提示你提交。  
     
   
5. 提交前需要简单填写本次提交的标题，描述为可选项。其中标题应该简要说明本次提交所做的改动，方便以后回滚。  
     
   
6. 最后 一步就是点击同步，将修改提交到服务器。之前的提交只是在本地操作的。  
     
   

### 网页端

在网页端也可以进行在本地一样的大部分操作。  
  




同样在这里也可以看到整个项目所有的代码提交记录。



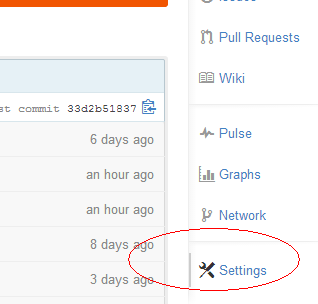
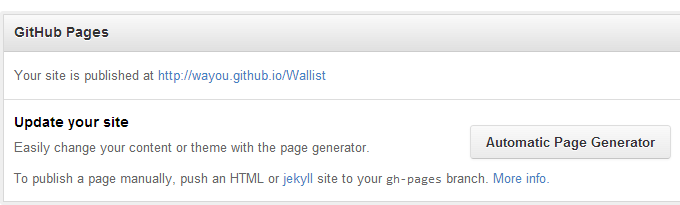
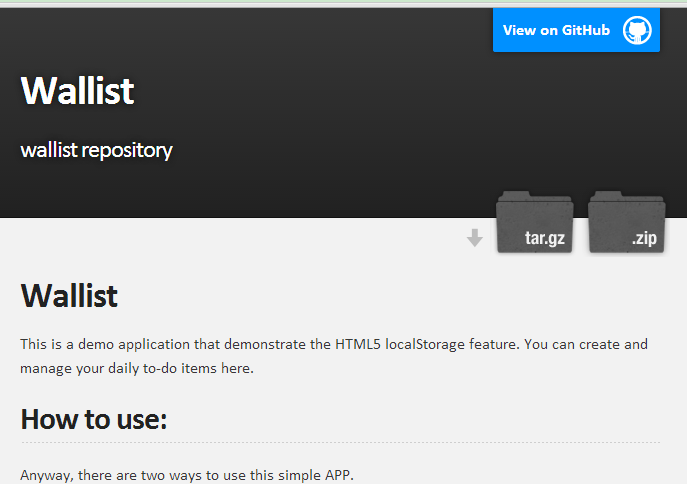
之前从本地提交了项目过来后，在网页中就能看到所有最新的文件。

在这里可以进行文件的增删改查，项目管理等操作。

具体不再赘述。

### 创建项目网页

值得一提的是GitHub为每一个项目提供了创建独立介绍页面的功能，类似于一个关于该项目的独立网站。

1. 进入 项目的设置页面  
   
2. 在GitHub Pages节点内，有创建项目页面相关的操作  
   
3. 在项目页面里，可以展示更多关于项目相关的文档，比在项目中的README文件会更丰富些。  
   

## HTML5

本项目主要涉及HTML5中关于本地存储的API的使用。

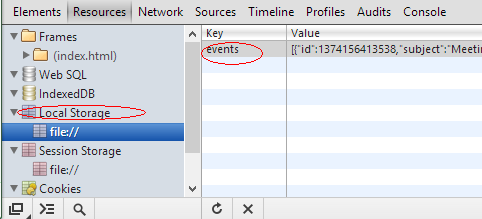
由最开始考虑的是用XML来实现数据的存储，在发现不可能后，才改由使用HTML5的localStorage特性来实现。

HTML5 localStorage 将数据存在浏览器中，直到用户手动清除为止，所以使用localStorage不必担心不够保险容易丢失等危险。

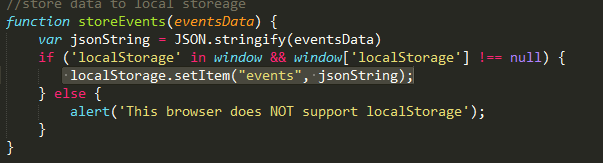
而在HTML5 localStorage之前传统Web程序使用cookie来进行少量的数据存储。大小一般受到4kb的限制，当然不两只浏览器有不同的标准。HTML5 localStorage将这个限制提高到了5M。而5M对于一个Web程序来说，远远足够了。

本项目中，使用一个“events”变量来存储所有事件项数据。

HTML5 localStorage中将数据以字符串的形式进行存储。：



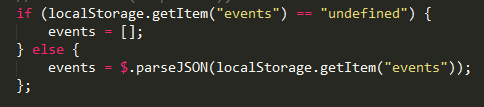
将数据存储到localStorage是通过调用localStorage.setItem(“events”)来实现的，当然还可以通过localStorage[“events”]=blah 来实现。但前者是更推荐的做法。



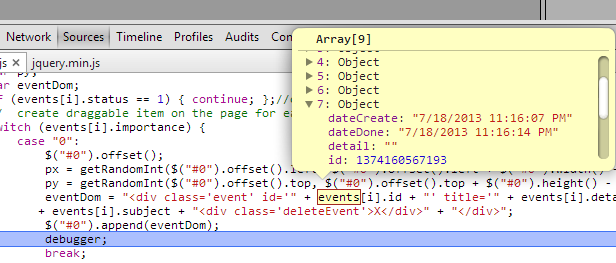
实现使用时我们使用的是JSON格式的数据，只有在存储前将其序列化为字符串。

var jsonString = JSON.stringify(eventsData)

所以在取出来时需要反过来将其转化为JSON格式 。



因为JSON格式才方便我们在程序中访问事件项的属性，获取事件项的id来进行对特定事件项进行操作。



本项目中没有采取将每个事件项分开独立在localStorage设置 一个变量的方式，一来感觉不容易管理，二来数据条目多了会使整个键值对变得非常冗长。

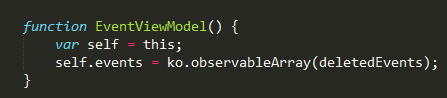
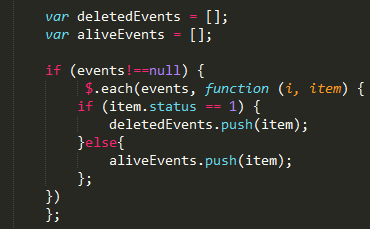
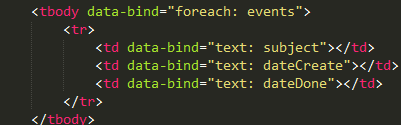
本项目中将所有事件项保存为JSON再转为字符串存入一个localStorage中的events变量中。在需要的时候取出来进行遍历，根据id对具体事件项进行操作。

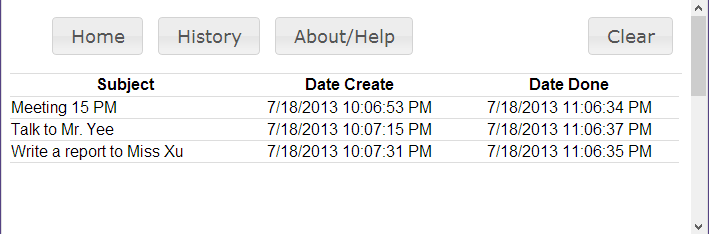
## JavaScript MVVM框架Knockout.js

对其官网的[教程](http://learn.knockoutjs.com/)进行了认真学习, 对于用其进行构建商业项目的复杂用法还需要进一步进行深究。

在本项目中，主要使用于History页面。

History页面的表格使用Knockout页面模型的遍历属性进行自动产生。

1. 使用Knockout.js是从建立页面模型开始的。  
     
   
2. 本项目的History页面的功能十分简单，只是简单的数据呈现。这决定的页面模型不会太复杂。从代码中可以看出，页面模型中仅包含一个可遍历类型数据源用于提供数据展示时的数据来源。我们事先把所有events从localStorage中取出来准备好，再赋值给knockout的这个页面模型。  
     
     
     
   当然，获取数据之前还包含一些对数据可靠性的一系列检测等，这是为了保证页面不会报错，即使是在首先打开History页面时。
3. 接下来的工作是准备好页面元素，用于数据的展示。Knockout将页面元素通过一些特定的属性语意化。  
     
   
4. Tbody为Table中呈现数据的主题部分，大部分语意化声明都在这里。基本的数据绑定包括指定绑定类型（这里为“text”）,绑定的数据源中相应的字段名等。
5. 最后一步就是将页面模型与具体的页面或元素进行显示绑定。这相当简单，只是一句代码的调用而以。  
     
   
6. 我们并没有一条一条地具体去写，就是这么几句简单的knockout代码，我们就可以得到整个数据的呈现。



由此可以看到在前端应用这些基于JavaScript的MVVM框架，将会大大减少我们设计，以及代码的书写。

# 解决的难题（Solved problems）

* 本地数据的保存问题
* HTML5 localStorage 在读取本地XML文件
* 一个事件项拖动后新区域的识别
* 主页面初始加载时将事件项限制在各自象限内
* 将当前选中/拖动项置于顶端
* 事件项标题过长后的显示问题

## 本地数据的保存问题

最开始采用当初讨论的设计，使用XML实现数据的本地持久化。但实践后发现行不通。关键点在于JavaScript中无法进行任何关于本地文件的写操作。

这是最开始没有考虑到的，以至于项目进行到中期才开始寻求解决方法。

**解决**：不过还好，借助HTML5中的新特性—localStorage,可以考虑将数据保存至浏览器，由此解决了数据存储的问题。

## HTML5 localStorage 在读取本地XML文件

出于安全策略相关的考虑，浏览器一般不允许JS代码访问本地文件。所以在最开始使用XML保存数据那种方案时，在读取本地XML文件时遇到了问题。

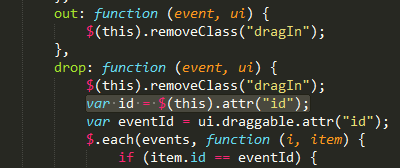
**解决：**后来经过调研，发现以flags标签运行Chrome可以解决此问题。具体做法是右击Chrome图标，选择属性，在目标后面加上： "--enable-file-access-from-files" 然后重启浏览器。

## 一个事件项拖动后新区域的识别

用户拖动一个事件项后，需要判断当前所处的象限从而将改变后的优先极应用于当前事件项。在如何判断当前象限上遇到了问题。

**解决**：于对jQuery UI的drag stop 事件里，会传入当前放入的目标对象，由此想到可以进而获取目标对象的id,从而根据id判断当前放入的象限，然后再相应更改当前事件项的优先极并进行保存。

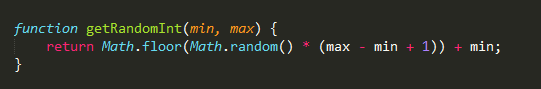
具体实现代码：



## 主页面初始加载时将事件项限制在各自象限内

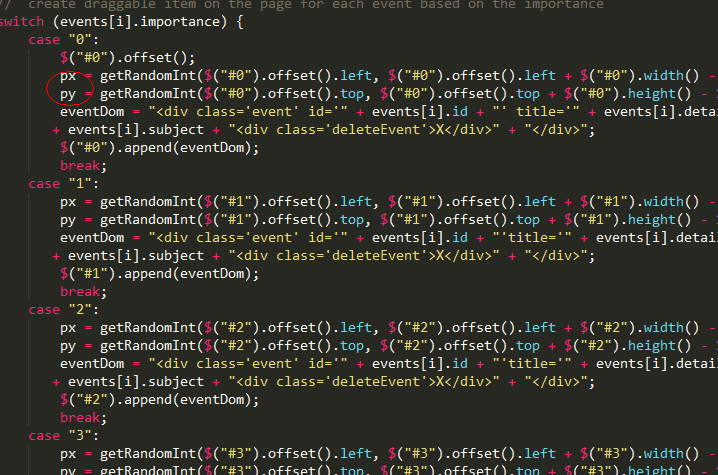
其实在判断各事件项的属于哪个象限并不是问题，问题是如何将这些事件项放入DOM上时将它们限制在各自的象限内。并且实现随机的效果。

**解决**：通过CSS实现。具体是先写一个产生随机数的函数方便用于每个事件项的position属性。



并且这个随机数要根据目标象限的范围来产生，这样才能将各事件项放入到指定的象限。

然后通过switch语句对所有事件项进行遍历，将产生好的随机position应用到各事件项上。



最后出来的效果令人满意。



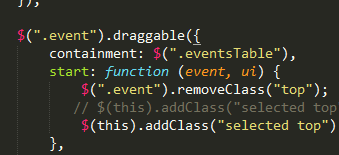
## 将当前选中/拖动项置于顶端

当同一象限有两个以上事件项时，应该将当前选中的置于最上面。同样，当移动一个事件项到其他象限时，它也应该在最上面。实现这个功能时遇到了不小的问题。

因为开始使用的方法是：

1. 对于当前移动的事件项应用一个类“.top”，将其z-index设置为最高，同时使用jQuery的siblings()函数选中其他同类事件项将其身上的”.top”类去掉。
2. 但siblings()方法只能返回当前象限的同类元素，以至于将一个事件项移动到另一个象限后，这一功能就不正常了。

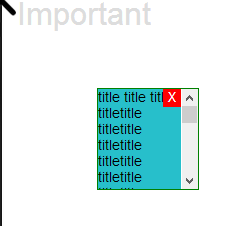
**解决**：改一种实现方式，不使用siblings()函数。使用$(“.event”)对所有事件项进行操作，移除“.top” 类，然后再将当前元素重新应用上“.top”类。



## 标题过长的显示问题

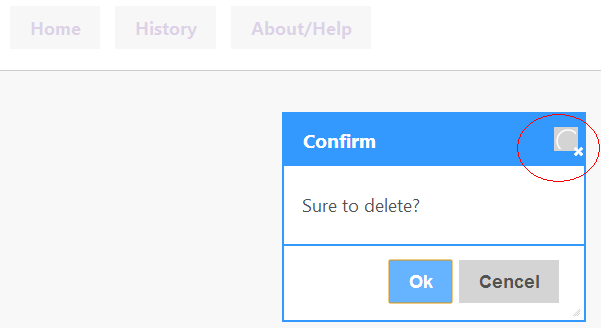
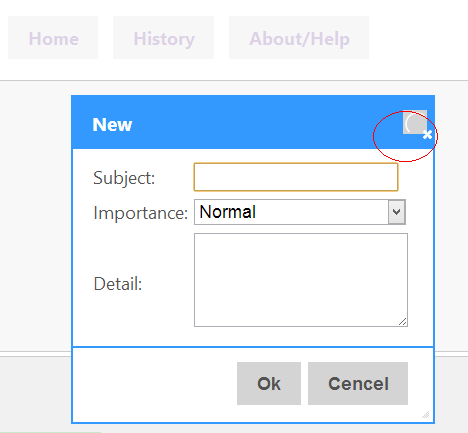
这一问题由徐丹测试时发现，当一个事件项的标题过长时，就需要考虑如何处理。

目前的做法是显示一个scrollbar. 并重写了相关样式。



# TODO

项目还有不足以可以改进的地方，这些将加入到新一批的to do issues中，详见GitHub的[issues页面](https://github.com/Wayou/Wallist/issues?state=open)。

1. 为整个项目的界面换个Metro风格的样式似乎会更好看些，经过调研找到了一款适合的样式[jQUIT (jQuery UI Themes)](http://jquit.com/build-jquery-ui-themes-with-jquit-builder/)。  
     
   偿试在项目中应用，效果还不错。但有些小问题，比如截图中的关闭按扭。  
     
   
2. 增加数据导入导出功能  
   这样可以随时备份恢复数据，防止意外或是浏览器清空本地数据时丢失数据。  
   已加入到[GitHub issues](https://github.com/Wayou/Wallist/issues/3)中。
3. 增加快捷键支持，已加到issues中，完整issues列表请到[这里](https://github.com/Wayou/Wallist/issues?state=open)浏览。