目录

背景介绍

春节前的前端应用代码，主要存在以下问题：

1， 代码耦合度高，基本上每个控件间都是互相调用，一旦某个控件存在问题，就会引起连锁反应，导致整个页面不可用
2， 每个组件的实现方式都不一样，而且因为没有采用一定的模式，组件的代码量都比较大，导致维护成本较高
3， 全局变量使用混乱，没有统一的命名，不好记，也很容易被污染
4， 键盘响应都写在了组件里，一旦键盘响应键值变化，带来的维护成本巨大
5， CSS 使用较少，样式控制都写在了代码里

解决思路

1， 引入统一的全局变量类，每个需要的参数都是他的一个属性
2， 用 jquery 的风格编写组件，降低维护成本
3， 在组件内引入订阅机制，对组件的每个原子功能解耦
4， 也在组件间引入订阅机制，实现组件间的解耦
5， 将应用代码独立编写，有效的将组件代码和应用代码分离

代码解读

以 /views/client/main.gsp 为例

阅者要求

1， 熟悉 jquery 开发及其事件机制
2， 熟悉 jquery 组件编写风格
3， 熟悉 jquery 的 function 特性及闭包
4， 熟悉开发模式中的订阅/发布

引入基础库：

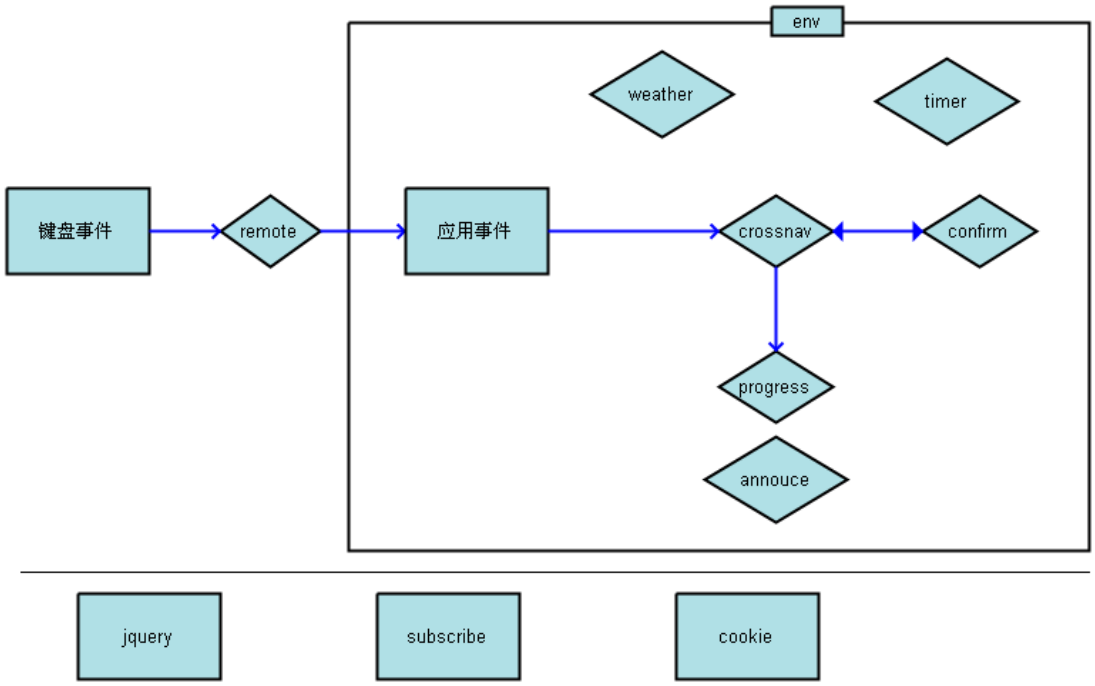
jquery-1.4.2.js	框架	最核心的库,不解释
jquery.hotkeys-0.7.9.js	插件	实现键+组合键绑定事件,目前没任何代码用到 唯一好处就是从源码里面抓了键值和键名的对应 MAP 目前是保留中 以便组合键方向的扩展
jquery.subscribe.1.1.js	插件	实现了订阅/发布功能
jquery.cookie.js	插件	实现了 cookie 功能,主要是在记录主菜单访问历史和付费提示上

编写应用代码

client.env.js	核心类	应用的全局变量,全局事件,对 js 功能扩展,一些工具代码
client.remote.js	组件类	实现了键盘事件和应用事件的转换,实现解耦
client.progress.js	组件类	实现了等待条的显示/隐藏/超时功能
client.crossnav.js	组件类	实现菜单的浏览功能
client.confirm.js	组件类	付费确认提示框的实现

client.weather.js	组件类	结合后台功能实现天气预报功能
client.timer.js	组件类	显示日期/时间
client.billing.js	组件类	显示计费信息,暂无实现代码
client.annouce.js	组件类	在屏幕底部显示帮助信息/酒店公告等
application.desktop.js	应用类	初始化组件,并在组件间实现事件订阅/发布
style.css	样式类	所有组件的样式都在一个文件里,通过 JS 来选用不同的状态

组件间设计图



主要代码讲解

ENV 类

EVENTS
局部变量,通过键值对方式定义了应用事件,通过 ClientENV 访问
如:

```
ON_MAINMENU_UP : 'on_mainmenu_up',//主菜单上移
ON_MAINMENU_DOWN : 'on_mainmenu_down',//主菜单下移
ON_MAINMENU_LEFT : 'on_mainmenu_left',//主菜单左移
ON_MAINMENU_RIGHT : 'on_mainmenu_right',//主菜单右移
```

```
ON_MAINMENU_SELECT : 'on_mainmenu_select',//主菜单被选中
ON_MAINMENU_ACTIVED : 'on_mainmenu_actived',//主菜单激活
ON_MAINMENU_BACK : 'on_mainmenu_back',//回退
```

ClientENV

全局变量,通过键值对定义了常用参数,亦可在页面直接添加,不推荐

如：

```
CONTEXT_PATH : '',//web应用路径
DEFAULT_IMAGE_PATH : '',//默认图片路径
LCAPP_PATH : '',//本地应用路径
MENU_LEVEL:0,//菜单级别
MENU_PARENT:0,//父菜单ID
CITY : 'beijing',//所在城市
CITYNAME : '',
EVENTS : EVENTS
```

Remote 类

将键盘事件转换成应用事件,实现耦合,主要实现了【上】【下】【左】【右】【确定】【回退】的响应,这个类需要在每个应用代码里面开头就实现

如：

```
$(document).bind('keydown', function(evt) {
        switch(specialKeys[evt.keyCode]){
            case 'up': //上移
                $.publish(ClientENV.EVENTS.ON_MAINMENU_UP);
                $.publish(ClientENV.EVENTS.ON_EXPLORER_UP);
                $.publish(ClientENV.EVENTS.ON_HOTELSERVICE_UP);
                $.publish(ClientENV.EVENTS.ON_HELP_UP);
            break;
            //略
        }
}
```

Crossnav 类

实现主菜单的浏览功能,并和其他组件通信

主要代码如下：

读取菜单数据.URL 地址由应用类提供：

```
/** 读取数据*/
nav.load = function(config){
    if(ClientUtil.isArray(config.data)){
        nav.data = config.data || [];
```

```
            nav.call('afterload');
        }else if(config.url){
            $.ajax({
                type:'POST',
                url:ClientENV.CONTEXT_PATH+config.url,
                data:{level:ClientENV.MENU_LEVEL,showType:ClientENV
                .SHOW_TYPE},
                dataType:"json",
                cache: false,
                success: function(data){
                    nav.data = data || [];
                    nav.call('afterload');
                },
                error:function(){
                    //TODO:显示数据加载错误
                }
            });
        }
    };
```

内部原子功能解构,其实是一个小型的订阅/发布实现,
但需要和组件间区别,因此命名有所不同

```
    /**内部事件触发*/
    nav.call = function(event, data){
        var handlers = nav[event];
        if(handlers && handlers.length > 0){
            for(var i=0; i<handlers.length; i++){
                handlers[i].handler.call(_this, data);
            }
        }
    };

    /**内部事件监听*/
    nav.listen = function(event, handler, data){
        if(typeof nav[event] === 'undefined')
            nav[event] = [];
        var len = nav[event].length;
        nav[event][len] = {handler:handler,data:data};
    };
```

响应应用事件,实现浏览功能:

```
    /** 监听外部事件*/
    $(_this).subscribe(ClientENV.EVENTS.ON_MAINMENU_UP,function(){
        if(disabled === false) return;
```

```
        nav.up();
    });
```

动画效果的实现

```
/**垂直向滑动一个单位*/
nav.vslide = function(){
    $(_this).animate({top:-48 * (mainPaging.current -1)});
};
/**水平向滑动一个单位*/
nav.hslide = function(){
    $(_this).find('div.category:eq('+mainPaging.current+')
    ul').animate({left:-1 * BLURWIDTH * subPaging.current});
};
```

应用类

这个类（application.client.js）主要是实现组件间的初始化及相互事件通知

初始化

```
//监听遥控器事件
$(document).remote();

//进度条
var progress = $('.j-loading').progress();
```

组件间订阅/发布

```
var nav = $('#j-navs').crossnav();
...
//加载完毕 发送隐藏事件
nav.listen('afterrender',function(){
    $('#j-navs').publish(ClientENV.EVENTS.ON_PROGRESS_HIDE);
});
```

其他组件说明

client.explorer.js	组件类	提供了电影/游戏的浏览功能

client.foreigexchange.js	组件类	提供了外汇信息查看功能
client.help.js	组件类	实现了系统帮助页面的交互功能,主要是上下翻页
client.hotelservice.js	组件类	实现了酒店服务信息页面的交互功能,主要是上下翻页
client.confirm.js	组件类	付费确认提示框的实现
client.news.js	组件类	结合后台功能实现新闻快报功能
client.railway.js	组件类	提供时刻表之类的信息查询,暂未实现
client.stock.js	组件类	提供中国股市个股查看功能,暂未完成
client.unconnect.js	组件类	客户终端和计费服务不通时的交互效果实现,并应用其他错误页面,404 页面等
Client.welcome.js	组件类	欢迎页面的交互效果

未尽事宜

1， 组件在不同分辨率的自适应能力不够强大
2， 部分代码重复使用,可以考虑使用类似 KISSY 的 MIX 设计理念
3， 代码和场景结合的比较厉害 可能有潜在风险
4， 未进行性能优化

参考资料

scalable-javascript-application-architecture
http://www.slideshare.net/nzakas/scalable-javascript-application-architecture