

搜索版前端架构设计介绍

since:2014-03-05

author:Will Joe

WLJ(walk lonely javascript)采用纯 js 脚本设计,依托于 Ext 免费版本 3.3.1 为基础,并结合 YC.CRM 产品 v4.5 对于 Ext 的扩展、补丁研发,采用面向对象的理论,对前端对象进行了大量的对象化封装。搜索版前端架构设计主要包括:瓷贴版首页框架设计以及业务逻辑开发框架设计两个部分。

首页框架结合 METRO-UI 理念,对于 UI 体验上进行的重新设计,并在此过程中,加强对于性能、易用性等方面的改造;

功能开发框架中,主要对开发人员逻辑开发场景进行分析,结合开发人员中存在的对面向对象的理解不深、对浏览器运行机制的不了解,以及代码调试困难等问题,进行了专项封装,如:代码过程化、片段化、扁平化处理;提供一个可见易用的控制台,以及相应的日志 API 等。

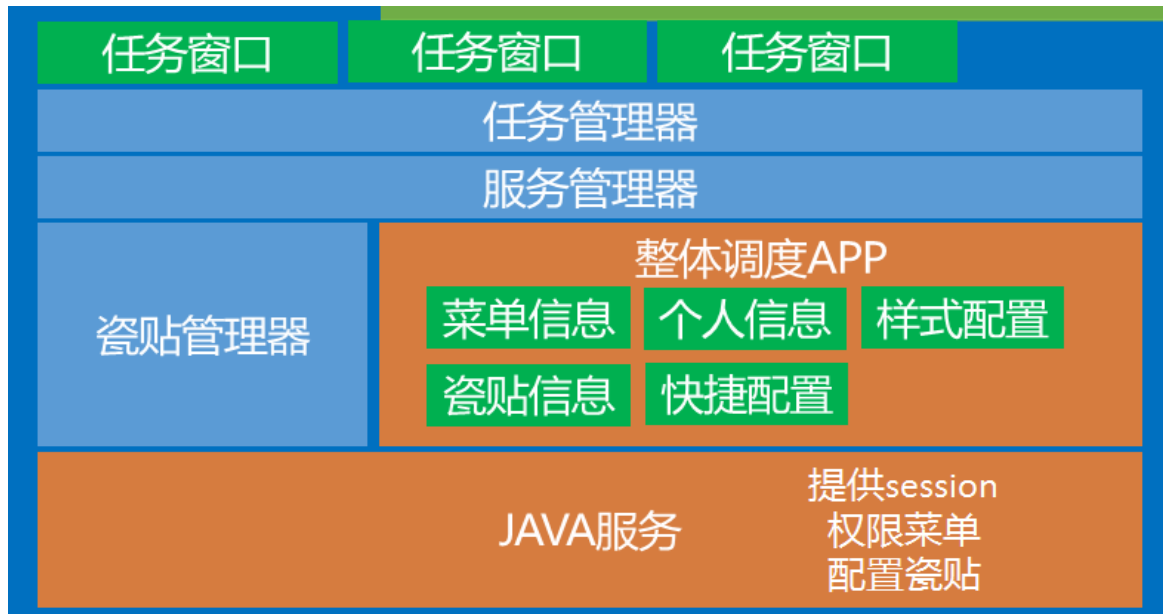
首页框架侧重点在于交互体验、运行性能、用户操作接口等方面的设计;开发框架的侧重点在于开发开发场景的归纳与抽象,并在此基础上形成构筑开发所必须的基础设施,以降低代码量、代码风险、以及代码调试成本。

一、 首页功能设计

1 结构及运行

首页功能框架以 Ext3.3.1 以及 YC.CRM 4.5 前端为基础进行研发。采用完全对象化的设计理念设计研发。

1) 框架结构



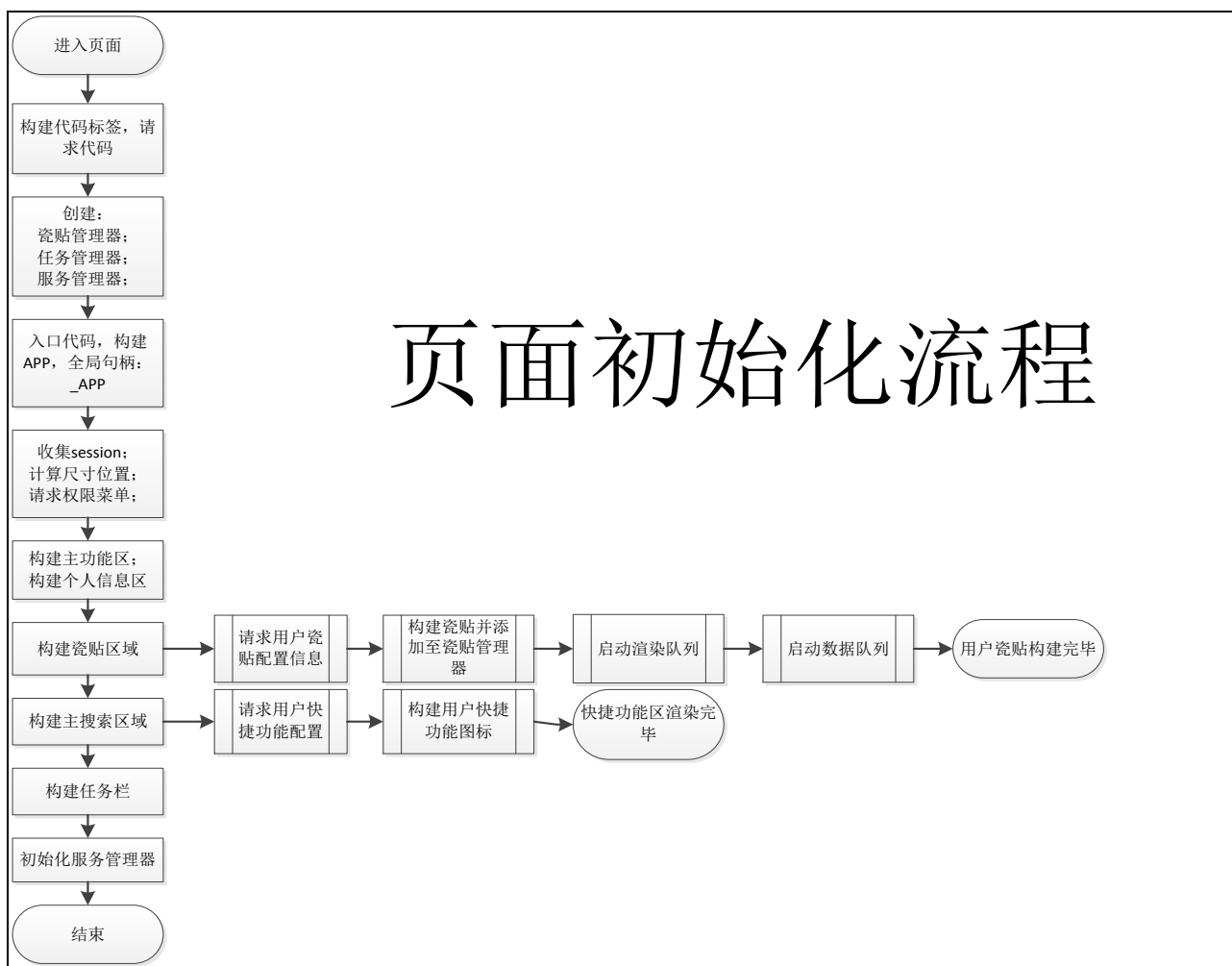
2) 底层运行

运行情况见以下几点:

- 1、首页框架代码构建器 (WljAPPBooter.js) 以标签方式构建整体首页运行必须的所有 js、css 代码;
- 2、Js 代码请求完成后, 会自动构建空的瓷贴管理器 (Wlj.TileMgr), 任务管理器 (Wlj.TaskMgr), 服务管理器 (Wlj.ServiceMgr);
- 3、首页入口代码 (/contents/wljFrontFrame/searchFace.js) 在所有类库文件加载完毕之后, 创建 APP 对象, 并构建全局句柄 _APP; 代码如下:

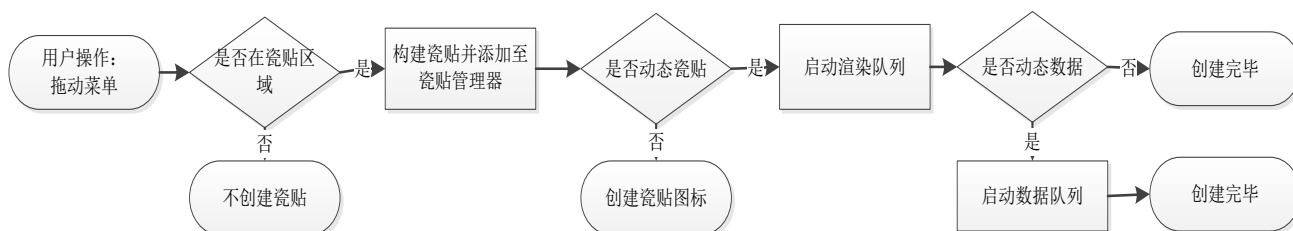
```
Ext.onReady(function() {  
    window._APP = new Wlj.search.App();  
});
```

- 4、APP 对象被创建出来后, 首先收集 session 信息, 以及计算页面以及各个工作区域的尺寸和位置参数;
- 5、APP 对象发起请求权限菜单数据;
- 6、权限菜单数据请求完毕后, 构建各个工作区域对象;
- 7、根据 session 信息构建主功能区域以及个人信息展示区;
- 8、请求用户瓷贴配置信息并构建首页瓷贴展示区域, 创建所有用户瓷贴, 并添加到瓷贴管理器, 启动瓷贴管理器渲染队列; 渲染队列完成后, 将触发瓷贴数据队列;
- 9、构建主搜索区域对象, 并在搜索区域渲染之后请求用户快捷操作配置, 并快捷操作配置;
- 10、构建任务栏对象;
- 11、根据菜单数据中可提供功能模块的菜单数据, 初始化服务管理器。

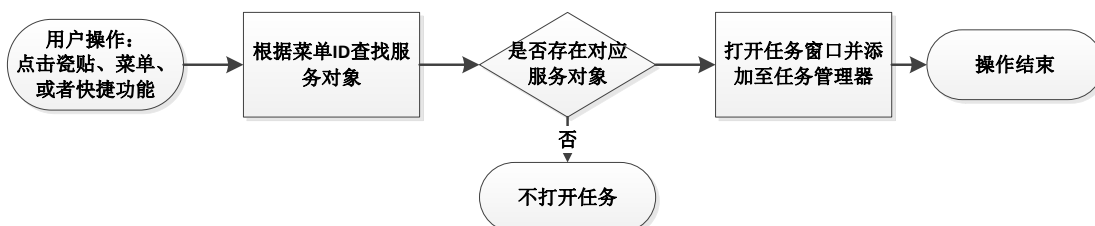


3) 操作入口

1、拖动菜单至瓷贴区，会自动创建对应功能点配置的动态瓷贴；



2、点击瓷贴、菜单，或者快捷功能图标；



3、选择搜索类型，输入搜索条件，点击搜索按钮；



2 目录及对象

1) 文件目录规划

1、WEBRoot/contents/frameControllers 目录下包含：

- /crmnewui/WebContent/contents/frameControllers/Wlj-frame-base.js: 包含整体 WLJ 版本信息、基础工具方法等；
- /crmnewui/WebContent/contents/frameControllers/wlj-function.jsp: 开发框架默认入口 JSP 页面；
- /crmnewui/WebContent/contents/frameControllers/Wlj-header-base.js: 首页头文件，构建全局性质 API；
- /crmnewui/WebContent/contents/frameControllers/Wlj-memorise-base.js: 首页内存控制模块，主要包括：瓷贴管理器、任务管理器、服务管理器；
- /crmnewui/WebContent/contents/frameControllers/Wlj-search-APP-cfg.js: 首页 APP 静态配置信息项；
- /crmnewui/WebContent/contents/frameControllers/Wlj-search-APP.js: 首页 APP 对象；
- /crmnewui/WebContent/contents/frameControllers/WljAPPBooter.js: 首页代码部署器，构建首页逻辑运行所必须的代码标签；
- /crmnewui/WebContent/contents/frameControllers/WljFunctionBooter.js: 业务模块代码构建起，构建业务逻辑开发框架部署所必须的代码标签；

2、WEBRoot/contents/frameControllers/widgets/search 目录：

- /crmnewui/WebContent/contents/frameControllers/widgets/search/header.js: 首页主功能区、个人信息区域等类原型；
- /crmnewui/WebContent/contents/frameControllers/widgets/search/menu.js: 首页菜单类原型；
- /crmnewui/WebContent/contents/frameControllers/widgets/search/search.js: 首页搜索框、快捷功能区域类原型；
- /crmnewui/WebContent/contents/frameControllers/widgets/search/service.js: 服务对象原型；
- /crmnewui/WebContent/contents/frameControllers/widgets/search/tiles.js: 瓷贴、瓷贴容器等类原型；
- /crmnewui/WebContent/contents/frameControllers/widgets/search/window.js: 任务框架口类原型；

3、WEBRoot/contents/frameControllers/widgets/views/index 目录：

- 本目录下存放首页瓷贴可用于个性化瓷贴开发的类原型。

2) 类对象列表

Wlj-frame-base.js:

类名	Wlj (单态对象)		
说明	Wlj 框架基础对象，Wlj 框架下所有类均在此命名空间下		
父类		Xtype	
事件方法属性	Wlj.version: 版本信息; Wlj.tools: 基础方法;		

Wlj-header-base.js:

无类型定义;

Wlj-memorise-base.js:

类名	Wlj.memorise.Tile (单态对象句柄: Wlj.TileMgr)		
说明	瓷贴管理器对象, 管理所有瓷贴对象。页面初始化之前构建。全局作用。		
父类	Ext.util.Observable	Xtype	
事件方法属性	属性: tiles: 瓷贴集合; tilesData: 动态瓷贴数据配置集合; 方法: addTile: public, 添加一个受控瓷贴; addDataCfg: public, 添加一个瓷贴的异步数据配置;		

类名	Wlj.memorise.Task (单态对象句柄: Wlj.TaskMgr)		
说明	任务窗口管理器对象, 管理所有被打开的任务窗口。提供一个可视化的当前任务列表窗口, 可关闭任务。全局作用。		
父类	Ext.util.Observable	Xtype	
事件方法属性	属性: tasks: 任务集合; 方法: addTask: public, 添加一个任务窗口; removeTask: public, 移除一个任务窗口; getTask: public, 根据菜单 ID 获取已打开任务对象;		

	showTasks: public, 打开任务管理器窗口;

类名	Wlj.memorise.Service (单态对象句柄: Wlj.ServiceMgr)		
说明	服务管理器对象, 管理所有可用服务。服务对象由可提供页面 URL 的菜单数据生成, 一个服务代表一个业务功能模块。提供一个全部服务的可视列表窗口。全局作用。		
父类	Ext.util.Observable	Xtype	
事件方法属性	属性: services: 服务集合; 方法: addService: public, 添加一个服务对象; showServiceWindow: public, 打开服务列表窗口; findServiceByID: public, 根据菜单 ID 获取服务对象;		

Wlj-search-APP-cfg.js:

存放 APP 对象静态配置;

类名	Wlj.search.App.HEADERBUTTONS (配置对象)		
说明	首页主功能区图标配置, 做为全局主功能区入口。		
父类	Array	Xtype	
事件方法属性	属性: 目前包含主功能列表, 主要包括: “开始”、“返回”、“菜单”、“配置”、“个性化”、“模式”、“任务”, “服务”等功能。		

类名	Wlj.search.App.TILECOLOR (配置对象)		
说明	瓷贴系统背景颜色列表。		
父类	Array	Xtype	
事件方法属性			

类名	Wlj.search.App.BACKGROUND (配置对象)		
说明	系统背景图片列表。		
父类	Array	Xtype	
事件方法属性			

--	--

Wlj-search-APP.js:

类名	Wlj.search.App		
说明	首页系统 APP 对象类型。主要作用包括：1、请求权限、配置等各类初始化信息；2、根据数据、配置等构建各个工作区域对象；3、对外提供部分公共 API。		
父类	Object	Xtype	
事件方法属性	API: openWindow: 打开一个任务窗口; ShowSearchArea: 转到主查询面板; HideSearchArea: 返回瓷贴面板; getMenuDataByProperty: 根据指定字段值, 获取菜单数据, 生成新的内存单元, 用于页面对象创建; findMenuDataByProperty: 根据指定字段值, 获取菜单数据, 不生成新的内存单元, 用于标识位判断; createRootMenuCfg: 根据菜单菜单数据生成一级菜单配置列表; createSubMenuCfg: 根据菜单 ID, 生成下级菜单列表; createGroup: 根据 index 属性生成此贴组对象, 如已存在则返回该对象; createTile: 创建一个瓷贴; createTileCfg: 构造瓷贴配置; translateSize: 根据编码返回瓷贴尺寸对象; translateSizeCode: 根据瓷贴尺寸对象, 返回尺寸编码;		

WljAPPBooter.js:

无类型定义, 构建所有首页代码运行所必须的 css 文件标签以及 js 文件标签。

WljFunctionBooter.js:

无类型定义, 构建业务模块代码运行所必须的 css 文件标签以及 js 文件标签。

/widgets/search/header.js:

文件中定义了搜索模式中顶部主功能栏对象原型。

类名	Wlj.widgets.search.header.HeadBar		
说明	首页主功能栏和用户信息区域容器。		
父类	Ext.Container	Xtype	headbar
事件方法属性	属性： backgroundTpl: headBar 背景区域模版。		

类名	Wlj.widgets.search.header.MainFunction		
说明	首页主功能区域原型。		
父类	Ext.BoxComponent	Xtype	mainfunction
事件方法属性	属性： functionTpl: 功能图标对象模版； mainTpl: 整体功能区域模版； items: 功能配置数组； headerFun: 主功能对象集合； 方法： add : public, 添加一个主功能按钮；		

类名	Wlj.widgets.search.header.Function		
说明	首页主功能按钮原型。		
父类	Object	Xtype	
事件方法属性	属性： el: DOM 对象； cfg: 配置信息； id: id； 方法： show : public, 显示图标； hide : public, 隐藏图标。		

类名	Wlj.widgets.search.header.UserInfo		
说明	用户信息对象。		
父类	Ext.BoxComponent	Xtype	userinfo
事件方法属性	属性： userImg: 用户头像； userName: 用户名称； userRole: 角色名称； userOrg: 所属机构；		

	headTpl: 头像模版对象; logoutTpl: 登出按钮模版对象; infoTpl: 用户信息模版对象;

/widgets/search/menu.js:

定义首页菜单组件原型;

类名	Wlj.widgets.search.menu.MenuItem		
说明	首页菜单项原型。		
父类	Ext.Container	Xtype	wljmenuitem
事件方法属性	属性: id: 根据菜单 ID 生成; name: 菜单名称; isLeaf: 是否叶子节点; enableTile: 可否被拖出成为瓷贴; defaultType: 默认子菜单类型; displayTemplate: 菜单项展示模版; displayTemplate: 被拖动时瓷贴类型代理模版; subTemplate: 子菜单容器模版; subTrigger: 当存在子菜单时, 右侧箭头标志; 方法: showSubs: 显示子菜单; hideSubs: 隐藏子菜单; hideAll: 递归隐藏所有下级菜单; getSubDatas: 获取子菜单数据;		

类名	Wlj.widgets.search.menu.MenuItem.DD		
说明	菜单拖动对象。当菜单被拖动时, 负责处理菜单代理的所有触发事件。		
父类	Object	Xtype	
事件方法属性			

类名	Wlj.widgets.search.menu.MenuProxy		
说明	菜单拖动代理。当菜单被拖动时, 为菜单创建一个代理 DOM 节点用于拖动。如菜单不能被拖动, 则不会创建此对象。		
父类	Object	Xtype	

事件方法属性	

/widgets/search/search.js:

定义查询界面中的查询框、快捷功能等类原型。

类名	Wlj.widgets.search.search.SearchField		
说明	主查询区域输入框对象。与查询类型控件配合使用。		
父类	Ext.form.TextField	Xtype	searchfield
事件方法属性	triggerWidth: 按钮宽度; defaultlWidth: 默认宽度;		

类名	Wlj.widgets.search.search.SearchType		
说明	查询类型控件。		
父类	Ext.BoxComponent	Xtype	searchtype
事件方法属性	属性: searchUlTemplate: 查询类型展示模版对象; items: 类型数组; 方法: selectFirst: 选择第一个类型选项; setCurrentType: 选择一个类型; getSearchType: 获取当前选择类型;		

类名	Wlj.widgets.search.search.SearchI		
说明	单个查询类型原型。		
父类	Object	Xtype	
事件方法属性	属性: template: 查询类型展示模版对象; el: DOM 对象; 方法: clearSelect: 取消选择; click: 点击方法; getSearchType: 获取类型;		

类名	Wlj.widgets.search.search.SearchComponent		
----	---	--	--

说明	主查询面板，内部加载查询输入框及查询类型控件。		
父类	Ext.Container	Xtype	searchcomponent
事件方法属性	属性： searchTypeSelect：是否需要类型选择控件； comfex：是否需要高级查询； comfexFn：高级查询逻辑； 方法： doSearch：执行查询； getValue：获取值； setValue：设置查询值；		

类名	Wlj.widgets.search.search.ModeShort		
说明	快捷操作入口对象；		
父类	Ext.BoxComponent	Xtype	modeshort
事件方法属性	属性： icon：图标； name：快捷操作名称； 方法： click：点击操作；		

类名	Wlj.widgets.search.search.ShortTitle		
说明	快捷操作入口类型标签对象；		
父类	Wlj.widgets.search.search.ModeShort	Xtype	shorttitle
事件方法属性	属性： icon：图标； name：快捷操作名称； 方法： click：点击操作；		

类名	Wlj.widgets.search.search.ShortContainer		
说明	快捷操作入口类区域容器		
父类	Ext.Container	Xtype	shortcontainer
事件方法属性	属性： layoutElTemplate：容器布局对象面板；		

类名	Wlj.widgets.search.search.SearchMainContainer		
说明	主查询区域整体容器		
父类	Ext.Container	Xtype	searchmaincontainer
事件方法属性			

/widgets/search/service.js:

定义服务员行以及服务瓷贴、容器等；

类名	Wlj.widgets.search.service.Service		
说明	服务对象原型，以一条菜单数据做为参数构建，受控于服务管理器。该对象能够打开一个任务。		
父类	Ext.Action	Xtype	
事件方法属性	属性： menuData: 菜单数据； id: 服务 ID； 方法： execute: 执行服务，打开任务。		

类名	Wlj.widgets.search.service.ServiceItem		
说明	服务对象在服务列表中的展现图标		
父类	Wlj.widgets.search.tile.Tile	Xtype	
事件方法属性	属性： serviceObject: 对应服务对象		

类名	Wlj.widgets.search.service.ServiceContainer		
说明	服务列表容器		
父类	Ext.Container	Xtype	
事件方法属性			

/widgets/search/tiles.js:

定义了瓷贴、瓷贴容器、瓷贴组、瓷贴代理、拖拽等对象；

类名	Wlj.widgets.search.tile.TileContainer		
说明	瓷贴容器，用于展示瓷贴组列表。		
父类	Ext.Container	Xtype	tilecontainer
事件方法属性	属性： innerTpl：瓷贴组容器模版对象； 方法： scroll：瓷贴容器滚动，根据滚轮方向，判断向前或者向后滚动一屏。		

类名	Wlj.widgets.search.tile.TileGroup		
说明	瓷贴组对象，用于展示瓷贴对象。其内部瓷贴布局		
父类	Ext.Panel	Xtype	tilegroup
事件方法属性	属性： baseCls：基础样式； width：一个瓷贴组宽度； height：瓷贴组高度； dd：瓷贴组的 dropZone 对象。用于瓷贴、菜单拖动时的目标的容器标识。		

类名	Wlj.widgets.search.tile.GroupDropZone		
说明	瓷贴组的 dropZone 对象。用于瓷贴、菜单拖动时的目标的容器标识。		
父类	Ext.dd.DropTarget	Xtype	
事件方法属性			

类名	Wlj.widgets.search.tile.Tile		
说明	瓷贴核心基础对象类。此类为瓷贴模式的核心类型，瓷贴对象可以自己的控制自己的尺寸，以为在父容器中的座标为之等信息。瓷贴对象具有自己的动画对象、拖动代理、拖动逻辑对象，同时瓷贴对象托管于瓷贴管理器，对于动态瓷贴，瓷贴在初始化完成后，会由瓷贴管理器排队渲染、请求数据。 在瓷贴对象的基础上，派生出滚屏瓷贴、可变大小瓷贴、首页动态瓷贴等类型。		
父类	Ext.Container	Xtype	tile
事件方法属性	属性： tileName：瓷贴名称；		

imgpath: 1*1 时显示图标时的图标文件路径;
 Animate: 动画逻辑对象, 在瓷贴对象被渲染之后, 会自动创建此对象 (Wlj.widgets.search.tile.Tile.Animate), 此对象负责处理瓷贴对象的移动、渐隐、渐现等动画效果;
 jsUrl: 动态瓷贴标识。默认为 false, 动态瓷贴此属性为一段 js 的 url 属性。瓷贴托管于瓷贴管理器后, 瓷贴管理器会根据此属性请求渲染代码, 并执行;
 draggable: 瓷贴可否被拖动;
 removeable: 瓷贴可否被从父容器中移除, 其主要控制右上角的移除按钮是否显示;
 enforceHeight: 瓷贴是否根据 pos_size 中的尺寸属性重定义高度;
 enforceWidth: 瓷贴是否根据 pos_size 中的尺寸属性重定义宽度;
 position: 默认为"absolute",
 float: 默认为"false", 为 left 或者 right 时, position 属性失效, 采用浮动方式定位瓷贴位置;
 insertProxy: 瓷贴拖动时是否插入一个站位的 DIV 来保留原来的位置;
 ownerW: 父容器中一行瓷贴的最大座标, 本瓷贴的最大座标为 ownerW-pos_size.TW;
 ownerH: 父容器中一列瓷贴的最大座标, 笨瓷贴的最大座标为 ownerH-pos_size.TH;
 ownerWI: 默认为 0, 父容器中一行瓷贴的最小座标;
 ownerHI: 默认为 0, 父容器中一列瓷贴的最小座标;
 baseSize: 瓷贴单位座标的尺寸;
 baseWidth: 单位横座标的尺寸;
 baseHeight: 单位纵座标的尺寸;
 baseMargin: 瓷贴边缘的 margin 属性;
 defaultDDGroup: 瓷贴被拖动生效的默认目标区域;
 tileManaged: 是否受控于瓷贴管理器;
 autoEl: 瓷贴渲染默认属性配置;
 layoutTpl: 瓷贴装载子对象的默认的标签模版对象;
 toolTemplate: 瓷贴移除按钮的模版对象;
 logoTemplate: 动态瓷贴尺寸为 1*1 的时候, 显示图表的 DOM 模版对象;
 pos_size: 瓷贴的位置座标、尺寸等信息, 包含: TW、TH、TX、TY, 分别代表瓷贴宽度、高度、横坐标、纵坐标属性, 均为 int 类型;
 tileSize: 瓷贴的尺寸编码, 如为 false 或未定义, 则代表 1*1; 转换逻辑见于 APP 对象 translateSize 方法;
 JsLoader: Ext.ScriptLoader 的引用;
 方法:
 moveToPoint: 将瓷贴移动到指定座标;
 getXYOrder: 根据页面座标转换为瓷贴在瓷贴组中的座标;
 buildTilePosition: 定位瓷贴;
 buildTileSize: 修复瓷贴尺寸;
 getSize: 获取瓷贴的像素尺寸;
 setSize: 重设瓷贴尺寸, 参数为基于瓷贴的尺寸信息;
 initPosition: 获取瓷贴位置;

	setPosition: 重设瓷贴位置信息, 参数为基于瓷贴的位置信息; removeThis: public, 在父容器中移除此瓷贴; clickFire: public, 调用瓷贴的点击逻辑;

类名	Wlj.widgets.search.tile.NegaLayout		
说明	布局对象, 引用标记: "nega"。定义一个多屏滚动的布局, 该布局将容器内的子对象进行分屏显示处理, 显示当前活动子对象; 当被活动对象被重设时, 显示对象被滚动到活动对象界面。		
父类	Ext.layout.CardLayout	Xtype	
事件方法属性	方法: setActiveItem: 根据子对象顺序属性, 滚动到的对应子对象显示区域。		

类名	Wlj.widgets.search.tile.NegativeTile		
说明	多屏滚动瓷贴。根据一定的时间间隔, 滚动展示各个子对象。		
父类	Wlj.widgets.search.tile.Tile	Xtype	negatile
事件方法属性	属性: layout: "nega", 滚屏布局; interval: 默认 1000, 滚动间隔, 以毫秒为单位; intervalId: 过程变量, 用于清除滚动线程; 方法: scrollToNext: 滚动至下一个子对象; clearInterval: 停止滚屏; setInterval: 滚品逻辑; reRunInterval: 重新滚动;		

类名	Wlj.widgets.search.tile.ResizeTile		
说明	可拖动改变大小的瓷贴;		
父类	Wlj.widgets.search.tile.NegativeTile	Xtype	resizetile
事件方法属性	属性: enableResize: 可否被拖动改变大小; RESIZEABLE: 尺寸大小代理对象;		

类名	Wlj.widgets.search.tile.ResizeTile.Resizeable		
说明	尺寸变动代理对象;		
父类	Ext.Resizable	Xtype	
事件方法	属性:		

法属性	handles: 代理框位置;

类名	Wlj.widgets.search.tile.IndexTile		
说明	首页特定动态瓷贴对象。		
父类	Wlj.widgets.search.tile.ResizeTile	Xtype	indextile
事件方法属性	属性: enableTransSize: 可否变换尺寸; resizeTemplate: 尺寸变换按钮模版对象; 方法: resize: 尺寸变换, 根据功能模块配置所支持的尺寸类型, 循环变换; reBuiltTile: 重构瓷贴内容; clearTileContent: 移除瓷贴内容;		

类名	Wlj.widgets.search.tile.TileProxy		
说明	瓷贴拖动代理对象; 在瓷贴被拖动时, 创建一个等尺寸无内容的 DIV 用以拖动。		
父类	Object	Xtype	
事件方法属性			

类名	Wlj.widgets.search.tile.Tile.DD		
说明	鼠标拖动代理对象, 负责处理拖动前后、过程中的逻辑计算;		
父类	Ext.dd.DragSource	Xtype	
事件方法属性			

类名	Wlj.widgets.search.tile.Tile.Animate		
说明	瓷贴动画逻辑处理对象; 负责瓷贴的动画逻辑处理		
父类	Ext.util.Observable	Xtype	
事件方法属性	属性: defaultAnim.duration: 默认动画时长; 方法: fadeIn: 渐现方法; fadeOut: 渐隐方法; moveWithRaletive: 相对位置移动效果; moveWithAbsolute: 绝对位置移动效果;		

--	--

/widgets/search/window.js:

定义了任务栏、任务对象、任务窗口、任务快捷方式等类型原型：

类名	Wlj.widgets.search.window.TaskBar		
说明	任务栏对象；存放所有打开的任务句柄以及窗口句柄。		
父类	Ext.Container	Xtype	taskbar
事件方法属性	属性： windowManager：任务窗口组； containerTemplate：任务栏基础样式； 方法： openWindow：打开一个任务； getTaskByTaskId：根据 ID 获取任务对象；		

类名	Wlj.widgets.search.window.TaskItem		
说明	任务对象；包含一个展示于任务栏的标签对象，以及相应任务窗口对象。		
父类	Ext.BoxComponent	Xtype	taskitem
事件方法属性	属性： name：任务名称； action：任务 URL； minisized：可否最小化； serviceObject：相应服务对象引用； taskTemplate：任务栏标签模版对象； 方法： createWindow：创建任务窗口对象； select：选中任务对象； blur：失去任务对象焦点； close：关闭任务； removeFromManager：从任务管理器移除； itemClick：点击任务栏标签事件； windowClick：任务窗口点击事件； minisize：最小化； reload：刷新功能页面；		

类名	Wlj.widgets.search.window.Window		
说明	窗口对象；受控于对应任务对象。		

父类	Ext.Window	Xtype	wljwindow
事件方法属性	方法： fitContainer: private, 重写最大化逻辑;		

类名	Wlj.widgets.search.window.WindowBar		
说明	窗口顶部快捷功能栏，主要包括：系统菜单、相关功能、及一些常规功能按钮;		
父类	Ext.Toolbar	Xtype	wljwindowbar
事件方法属性			

类名	Wlj.widgets.search.window.InnerMenu		
说明	窗口顶部快捷功能栏区的菜单项对象，主要用于系统菜单和相关功能展示;		
父类	Ext.menu.Item	Xtype	wljinnermenuitem
事件方法属性			

3 开发配置入口

首页框架下，客户化开发的主要需求点集中在针对业务模块的个性化业务逻辑开发。主要包括：瓷贴 UI 开发和模块瓷贴配置。

1) 瓷贴 UI 开发

瓷贴 UI 开发，将以适应于首页瓷贴展示情况的类模型为开发基础，开发人员需要编写独立的 js 代码文件，构建瓷贴内部 UI 对象。

以过程 function 为入口，并返回对应句柄数组的方式开发;

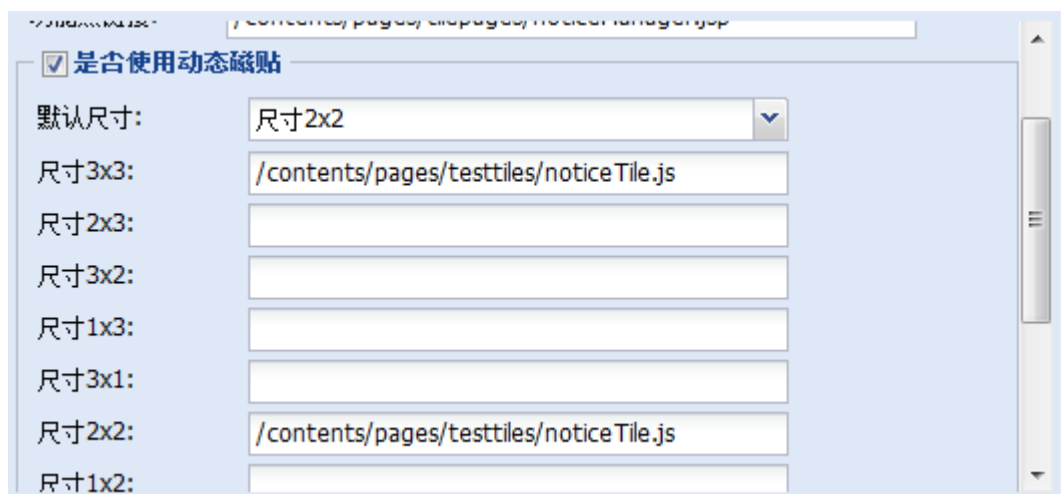
目前，框架中已提供了一个双列列表类，适用于 2*2 的瓷贴内容展示;

类名	Wlj.widgets.views.index.grid.TileGrid		
说明	定义于： /contents/frameControllers/widgets/views/index/grid/grid.js 文件。		
父类	Ext.DataView	Xtype	
事件方法属性	属性： Title: 瓷贴标题，显示于头部; root: 瓷贴数据根节点; url: 数据请求 URL;		

	columns: 列模型定义, columnName, 定义数据列名, key, 是否逐渐字段, show, 是否显示;
	dataSize: 查询多少条数据;

2) 模块瓷贴配置

瓷贴代码编写好后, 需要配置关联至相应模块, 在模块管理中, 为模块指定支持的瓷贴尺寸, 以及各个尺寸下需要调用的业务逻辑代码;



The screenshot shows a configuration window titled 'Module Management' with a sub-header '是否使用动态磁贴' (Use Dynamic Tiles). Below this, there is a list of tile sizes and their corresponding JavaScript files:

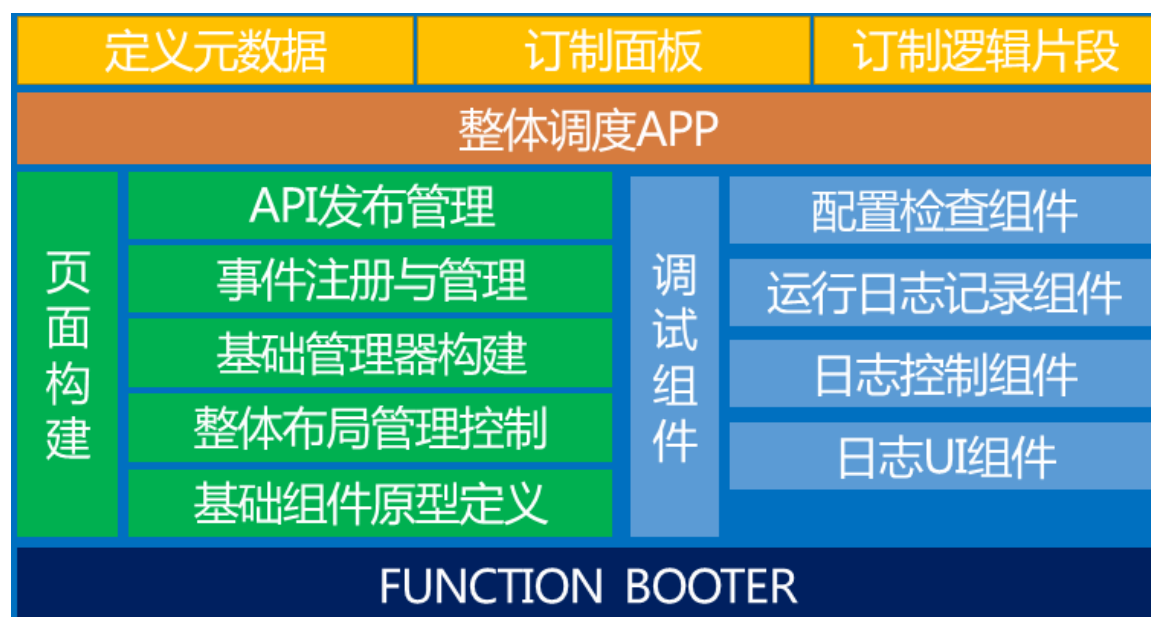
尺寸 (Size)	文件路径 (File Path)
默认尺寸: (Default Size)	尺寸2x2 (Size 2x2)
尺寸3x3:	/contents/pages/testtiles/noticeTile.js
尺寸2x3:	
尺寸3x2:	
尺寸1x3:	
尺寸3x1:	
尺寸2x2:	/contents/pages/testtiles/noticeTile.js
尺寸1x2:	

二、 开发框架设计

1. 面临问题及解决

见 PPT;

2. 整体结构



3. 目录及对象

1) 文件目录规划

- 1、/contents/frameControllers/wlj-function.jsp: 公共 JSP 入口;
- 2、/contents/frameControllers/WljFunctionBooter.js: 业务模块代码构建器;
- 3、/contents/frameControllers/widgets/debug.js: 调试模块, 控制台 UI 对象;
- 4、/crmnewui/WebContent/contents/frameControllers/widgets/app/Wlj-frame-function-api.js: 系统 API 列表及说明;
- 5、/crmnewui/WebContent/contents/frameControllers/widgets/app/Wlj-frame-function-app.js: 业务模块整体 APP 类;
- 6、/crmnewui/WebContent/contents/frameControllers/widgets/app/Wlj-frame-function-builder.js: 业务模块构建器逻辑, 请求业务模块代码、检查配置、构建 APP;
- 7、/crmnewui/WebContent/contents/frameControllers/widgets/app/Wlj-frame-function-error.js: 调试模块控制代码, 主要包括各级别日志开关;
- 8、/crmnewui/WebContent/contents/frameControllers/widgets/app/Wlj-frame-function-header.js: 业务开发模块头文件, 主要申明各配置项以及各个事件原型;
- 9、/crmnewui/WebContent/contents/frameControllers/widgets/app/Wlj-frame-function-widgets.js: 业务模块各个子组件类原型;

2) 类对象列表

wlj-function.jsp

JSP 入口。

WljFunctionBooter.js

构建必要的 css、js 文件标签

debug.js

构建日志控制台。

Wlj-frame-function-api.js

类名	API（对象）		
说明	API 对象中的每个属性，标识一个 window 域的 API 方法。这些 API 方法会在 APP 构建时发布。在业务逻辑块中可进行调用。		
父类	Object	Xtype	
事件方法属性	<pre>/** * 设置查询条件面板尺寸 * @param obj: { * width: 100, * hieght: 100 * },高宽属性均为可选属性 * @return */ setSearchSize : true, /** * 根据字典类型,获取字典STORE * @param type * @return */ findLookupByType : true, /** * 根据字典类型,把key转换为值 * @param type * @param key</pre>		

```

* @return
*/
translateLookupByKey : true,
/**
 * 根据字典类型,把值转换为key
 * @param type
 * @param value
 * @return
 */
translateLookupByValue : true,
/**
 * 设置查询条件并刷新数据
 * @param params : 参数
 * @param forceLoad : 是否强制刷新,default : true
 * @param add : 是否清除过期条件,default : true
 * @param transType : 是否需要转换字段命名模式,默认为APP的
SEARCHFIELDTRANS属性;1: 转为驼峰命名;2: 转为大写underline模式;3: 不做
转换;
 * @return
 */
setSearchParams : true,
/**
 * 获取查询store
 * @return
 */
getResultStore : true,
/**
 * 刷新当前数据
 * @return
 */
reloadCurrentData : true,
/**
 * 获取结果域中当前选择数据对象。如无选择数据,则返回false;
 * @return
 */
getSelectedData : true,
/**
 * 数据提交
 * @param data : 当前表单捕获数据,json格式;
 * @param comitUrl : 可选参数,提交URL,系统默认为app对象的提交
URL
 * @param needPid : 是否需要返回提交数据主键
 * @return
 */

```

```

comitData : true,
/**
 * 转换数据字段名格式,对于空数据项,进行剪除。返回新副本,不变动传入
参数的结构
 * @param data : 待提交数据
 * @param transtype : 转换类型,1: 从大写下划线转换为驼峰命名;2:
从驼峰命名转换为大写下划线类型;默认为1;3、不做转换;
 * @return
 */
translateDataKey : true,
/**
 * 根据name属性,获取字段配置信息;
 * @param name: 可为String类型或者数组;
 * @return
 */
getFieldsByName : true,
/**
 * 根据name属性,创建字段配置信息副本;
 * @param name: 可为String类型或者数组;
 * @return
 */
copyFieldsByName : true,
/**
 * 创建APP对象整体基础字段配置副本
 * @return
 */
getFieldsCopy : true,
/**
 * 收起查询条件面板
 * @return
 */
collapseSearchPanel : true,
/**
 * 展开查询面板
 * @return
 */
expandSearchPanel : true,
/**
 * 获取边缘面板对象;
 * params : 'left','right','top','bottom'
 */
getEdgePanel : true,
/**
 * 重置查询条件;

```

```
* params : deldy : 是否删除动态字段;
*/

resetCondition : true,
/**
 * 获取当前展示信息VIEW对象,展示列表时,返回undefined;
 */
getCurrentView : true,
/**
 * 隐藏当前展示信息VIEW对象;
 */
hideCurrentView : true,
/**
 * 获取详情VIEW,无详情面板返回false;
 */
getDetailView : true,
/**
 * 获取新增VIEW,无新增面板返回false;
 */
getCreateView : true,
/**
 * 获取修改VIEW,无修改面板返回false;
 */
getEditView : true,
/**
 * 展示详情VIEW,无详情面板返回false;
 */
showDetailView : true,
/**
 * 展示新增VIEW,无新增面板返回false;
 */
showCreateView : true,
/**
 * 展示修改VIEW,无修改面板返回false;
 */
showEditView : true,
/**
 * 根据标题获取自定义信息VIEW
 * @param title : view标题
 * @return
 */
getCustomerViewByTitle : true,
/**
 * 根据顺序获取自定义信息VIEW
 * @param index: 顺序
```


	<pre> * @return */ getCustomerViewByIndex : true, /** * 根据标题展示自定义信息VIEW * @param title: 标题 * @return */ showCustomerViewByTitle : true, /** * 根据顺序展示自定义信息VIEW * @param index: 顺序 * @return */ showCustomerViewByIndex : true </pre>

Wlj-frame-function-header.js

类名			
说明	该文件中定义所有开发人员可配置项原型。Listeners 对象申明所有可自定义事件逻辑片段原型。		
父类	Object	Xtype	
事件方法属性	<pre> /** * url : * 类型: string; * 说明: 数据查询URL地址; * 必选: 是; */ var url = false; /** * comitUrl : * 类型: string; * 说明: 新增、修改提交地址,若未配置该项,则使用查询URL做为提交地址; * 必选: 否; */ var comitUrl = false; </pre>		

```
/**
 * fields :
 *      类型: array[object];
 *      说明: 页面主工作区域所涉及到的字段申明; 该申明, 将被运用于查询面板、结果列表、增删查改的展示、逻辑运算等;
 *      字段对象: 申明一个字段的业务逻辑、展示等的各类配置信息; 主要可配属性如下:
 *          name :
 *              类型: string;
 *              说明: 字段所对应的业务逻辑名称, 该属性应与查询结果中的字段列相同; 该属性将做为数据接收、前台控制、后台提交等的关键属性;
 *              必选: 是;
 *          hidden :
 *              类型: boolean;
 *              说明: 字段展示情况, 该属性用于字段在列表、查询表单、新增、修改表单的强制隐藏;
 *              必选: 否;
 *          text :
 *              类型: string;
 *              说明: 字段的中文名称, 将被用于表单中的字段标签、以及列表中的表头; 当字段没有这个属性时, 该字段将被隐藏;
 *              必选: 否;
 *          searchField :
 *              类型: boolean;
 *              说明: 该字段是否做为查询字段, 在查询条件域中展示;
 *              必选: 否;
 *          viewFn :
 *              类型: function;
 *              说明: 字段在列表中, 展示时的特殊展示类型; 该function将在数据行渲染时, 接收到字段的值, 返回值将做为该字段的展示效果;
 *              必选: 否;
 *          dataType :
 *              类型: string;
 *              说明: 数据类型;
 *              必选: 否;
 *          translateType :
 *              类型: string;
 *              说明: 字段涉及到的映射字典项; 该字典项将做为查询结果中的字段映射依据, 以及表单面板中的下拉框的选择值;
 *              必选: 否;
 *          resutlWidth :
 *              类型: int;
 *              说明: 字段在查询结果列中展示的宽度, 默认150;
 *              必选: 否;
```

```

*      必选: 是;
*/
var fields = false;

/**
 *  createView|editView|detailView
 *  createView :
 *      类型: boolean;
 *      说明: 是否需要新增面板;默认为: true;
 *      必选: 否;
 *  editView :
 *      类型: boolean;
 *      说明: 是否需要修改面板;默认为: true;
 *      必选: 否;
 *  detailView :
 *      类型: boolean;
 *      说明: 是否需要详情面板;默认为: true;
 *      必选: 否;
 */
var createView = false;
var editView = false;
var detailView = false;

/**
 *
 *
formViewers[createFormViewer|editFormViewer|detailFormViewer] :
 *      类型: array[object{fields[string],fn,columnCount}];
 *      说明: 新增、修改、详情业务数据表单分组;form表单, 基础配置,
createFormViewer|editFormViewer|detailFormViewer可单独对新增、修改、
详情面板进行配置;
 *          字段业务分组:
 *              fields: 数组对象, 某个分组中所需要的字段对象; 必选;
 *              fn: 分组字段初始化逻辑。该function可依次接收到[fields]数
组中声明的字段;
 *                      返回经过业务逻辑处理后的字段数组, 该数组中, 字段的顺序将
成为分组面板中字段的展示顺序;
 *                      必选;
 *              columnCount: 字段面板中, 字段列数。可选, 默认值为, 面板宽度
大于1024时为4, 小于1024时, 为3
 *      必选: 是;
 */
var formViewers = false;

```

```

var createFormViewer = false;
var editFormViewer = false;
var detailFormViewer = false;

/**
 * validates[createValidates|editValidates] :
 *     类型: array[object{dataFields[string],fn]];
 *     说明: 新增、修改提交时, 字段校验逻辑, 逐条配置, 编写; 校验对象基础配置,
createValidates|editValidates可单独对新增、修改校验进行配置;
 *     校验规则对象(validate object) : 包含两个属性;
 *         dataFields: 数组对象, 申明该校验规则中, 涉及到的字段;
 *         fn: 校验逻辑function, 可依次接收到[dataFields]中申明的
字段的值;
 *             当且仅当返回为: false时, 校验失败;
 *     必选: 否;
 */
var validates = false;
var createValidates = false;
var editValidates = false;

/**
 * linkages[createLinkages|editLinkages] :
 *     类型: object{fieldName:{fields[string],fn}};
 *     说明: 新增、修改面板数据域联动逻辑, 逐条编写, 联动逻辑将在字段数据发生
变化, 失去焦点时触发; 联动对象基础配置, createLinkages|editLinkages可单独
对新增、修改进行配置;
 *     联动对象{linkages object} :
 *         fieldName: 联动逻辑触发字段;
 *         fields: 数组对象, 申明该联动对象影响到的字段对象;
 *         fn: 逻辑function, 可依次接收到[fieldName]声明的字段
对象以及[fields]中申明的所有的字段对象;
 *     必选: 否;
 */
var linkages = false;
var createLinkages = false;
var editLinkages = false;

/**
 * lookupTypes|localLookup
 *     lookupTypes :
 *         类型: array[string];

```

```

*      说明：远程数据字典类型；APP对象会在页面渲染之前自动请求所有数据字典
项，并纳入数据字典管理器；
*      必选：否；
*      localLookup :
*      类型： object{sring:array};
*      说明：本地静态数据字典项；APP对象会在页面渲染之前将这些数据字典项同
远程字典项一并纳入数据字典管理器；
*      必选：否；
*/
var lookupTypes = false;
var localLookup = false;

/**
*      edgeVies :
*      类型： object;
*      说明：四个属性，分别配置上下左右四个边缘配置信息；
*      top:
*      类型： object;
*      说明：工作区上部边缘配置；配置信息同： Ext.form.FormPanel;
默认配置参考：
Wlj.frame.functions.app.App.prototype.edgeViewBaseCfg.top;
*      必选：否；
*      left:
*      类型： object;
*      说明：工作区上部边缘配置；配置信息同： Ext.Panel，默认布局采
用accordion；默认配置参考：
Wlj.frame.functions.app.App.prototype.edgeViewBaseCfg.left;
*      必选：否；
*      right:
*      类型： object;
*      说明：工作区上部边缘配置；配置信息同： Ext.Panel，默认布局采
用accordion；默认配置参考：
Wlj.frame.functions.app.App.prototype.edgeViewBaseCfg.right;
*      必选：否；
*      bottom:
*      类型： object;
*      说明：工作区上部边缘配置；配置信息同： Ext.TabPanel；默认配
置参考：
Wlj.frame.functions.app.App.prototype.edgeViewBaseCfg.bottom;
*      必选：否；
*
*      必选：否；
*/

```

```

var edgeVies = false;

/**
 * customerView :
 *     类型: array[object];
 *     说明: 查询结果域扩展功能面板;触发按钮将被渲染在`新增`,`修改`,`详情`按钮之后;配置同Ext各类组件对象;
 *     必选: 否;
 */
var customerView = false;

/**
 * treeLoaders|treeCfgs
 *     treeLoaders :
 *         类型: array[object]
 *         说明: 页面中可能会使用的树形结构的loader对象配置, 配置参考:
Com.yucheng.bcrm.ArrayTreeLoader; loader对象会在APP初始化的时候进行
创建, 且加载好数据结构。
 *         必选: 否;
 *     treeCfgs :
 *         类型: array[object]
 *         说明: 页面中可能用到的树形面板对象预配置, 配置参考:
Com.yucheng.bcrm.TreePanel; tree对象构建调用TreeManager对象的
createTree方法;
 *         其中, root数据不做配置, 由rootCfg代替, 为简单json对象;
 *         必选: 否;
 */
var treeLoaders = false;
var treeCfgs = false;

/**
 * tbar :
 *     类型: array[object];
 *     说明: 用户扩展工具栏按钮, 按钮配置同Ext.Button;
 *     必选: 否;
 */
var tbar = false;

var listeners = {

```

```
/**APP初始化**/  
/**  
 * APP初始化之前触发;  
 * params : app: 当前APP对象;  
 * return : false: 阻止页面初始化; 默认为true;  
 */  
beforeinit : true,  
/**  
 * APP初始化之后触发;  
 * params : app: 当前APP对象;  
 */  
afterinit : true,  
  
/**查询条件域事件**/  
/**  
 * 查询条件域对象初始化之前触发, 此时对象尚未渲染;  
 * params : con: 查询条件面板对象;  
 * app: 当前APP对象;  
 */  
beforeconditioninit : true,  
/**  
 * 查询条件域对象初始化之后触发, 此时对象尚未渲染;  
 * params : con: 查询条件面板对象;  
 * app: 当前APP对象;  
 */  
afterconditioninit : true,  
/**  
 * 查询条件域对象渲染之前触发, 此时对象尚未渲染;  
 * params : con: 查询条件面板对象;  
 * app: 当前APP对象;  
 */  
beforeconditionrender : true,  
/**  
 * 查询条件域对象渲染之后触发;  
 * params : con: 查询条件面板对象;  
 * app: 当前APP对象;  
 */  
afterconditionrender : true,  
/**  
 * 当数据字段被动态拖动到查询条件框时触发;  
 * params : fCfg: 添加之前默认生成的数据项配置;  
 * columnIndexT: 将要被添加的列数;  
 * searchPanel: 查询条件form面板;
```

```
* return : true阻止条件添加事件; 默认为true;
*/
beforeconditionadd : true,
/**
 * 当数据字段被添加为查询条件后触发
 * params : field: 被添加后的字段对象;
 *          searchPanel: 查询面板表单;
 */
conditionadd : true,
/**
 * 当一个动态数据条件被移除前触发
 * params : field: 将要被移除的查询条件字段对象;
 *          searchPanel: 查询条件面板对象;
 * return : false, 阻止移除事件; 默认为true;
 */
beforeconditionremove : true,
/**
 * 当一个动态数据条件被移除后触发
 * params : searchPanel被移除字段后的查询条件表单;
 */
conditionremove : true,
/**
 * 当动态数据条件被全部移除前触发;
 * params : searchDomain: 查询域对象;
 *          searchpanel: 查询条件面板;
 *          dyfield: 移除前动态字段数组;
 */
beforeddyfieldclear : true,
/**
 * 当动态数据条件被全部移除后触发;
 * params : searchDomain: 查询域对象;
 *          searchpanel: 查询条件面板;
 *          dyfield: 移除后动态字段数组;
 */
afterddyfieldclear : true,
/**
 * 查询条件域收起前触发;
 * params: panel: 查询条件域面板;
 * return: false: 阻止查询条件域收起事件, 默认为true,
 */
beforeconditioncollapse : true,
/**
 * 查询条件域收起后触发;
 * params: panel: 查询条件域面板;
```



```

*/
afterconditioncollapse : true,
/**
 * 查询条件域收展开触发;
 * params: panel: 查询条件域面板;
 * return: false: 阻止查询条件域展开事件, 默认为true,
 */
beforeconditionexpand : true,
/**
 * 查询条件域展开后触发;
 * params: panel: 查询条件域面板;
 */
afterconditionexpand : true,

/**查询结果域操作**/
/**
 * 查询结果翻页后触发;
 * params : store: 结果域数据源;
 *          pageIndex: 当前页数;
 */
turnpage : true,
/**
 * 数据行被选择时触发;
 * params : record:被选择的数据对象;
 *          store:数据所在数据源对象;
 *          tile:结果面板中数据行的瓷贴对象;
 */
recordselect : true,
/**
 * 数据行双击事件;
 * params : tile:被双击数据行瓷贴对象;
 *          record: 被双击数据对象;
 */
rowdblclick : true,
load : true,
/**
 * 设置当前查询条件前触发;
 * params : params:追加查询条件项;
 *          forceLoad: 是否强制刷新当前数据;
 *          add: 是否清理之前查询条件;
 *          transType: 查询条件key值转换类型
 * return : true: 阻止查询条件设置动作; 默认为true;
 */

```

```

beforesetsearchparams : true,
/**
 * 设置当前查询条件之后，数据刷新之前触发；
 * params : params:追加查询条件项；
 *         forceLoad: 是否强制刷新当前数据；
 *         add: 是否清理之前查询条件；
 *         transType: 查询条件key值转换类型
 */
setsearchparams : true,
/**
 * 查询结果域对象初始化之前触发，此时对象尚未渲染；
 * params : con: 查询条件面板对象；
 *         app: 当前APP对象；
 */
beforeresultinit : true,
/**
 * 查询结果域对象初始化之后触发，此时对象尚未渲染；
 * params : con: 查询条件面板对象；
 *         app: 当前APP对象；
 */
afterresultinit : true,
/**
 * 查询结果域对象渲染之前触发，此时对象尚未渲染；
 * params : con: 查询条件面板对象；
 *         app: 当前APP对象；
 */
beforeresultrender : true,
/**
 * 查询结果域对象渲染之后触发；
 * params : con: 查询条件面板对象；
 *         app: 当前APP对象；
 */
afterresultrender : true,

/**查询结果域附加面板事件**/
/**
 * 新增面板渲染之前触发
 * params : view:新增面板对象
 */
beforecreateviewrender : true,
/**
 * 新增面板渲染之后触发
 * params : view:新增面板对象

```

```

*/
aftercreateviewrender : true,
/**
 * 修改面板渲染之前触发
 * params : view:修改面板对象
 */
beforeeditviewrender : true,
/**
 * 修改面板渲染之后触发
 * params : view:修改面板对象
 */
aftereditviewrender : true,
/**
 * 详情面板渲染之前触发
 * params : view:详情面板对象
 */
beforedetailviewrender : true,
/**
 * 详情面板渲染之后触发
 * params : view:详情面板对象
 */
afterdetailviewrender : true,
/**
 * 结果域面板滑入前触发:
 * params: theview : 当前滑入面板;
 * return: true, 阻止面板滑入操作; 默认为true;
 */
beforereviewshow : true,
/**
 * 结果域面板滑入后触发:
 * params: theview : 当前滑入面板;
 */
viewshow : true,
/**
 * 结果域面板滑出前触发:
 * params: theview : 当前滑出面板;
 * return: false, 阻止面板滑出操作; 默认为ture;
 */
beforeviewhide : true,
/**
 * 结果域面板滑出后触发:
 * params: theview : 当前滑出面板;
 */
viewhide : true,

```

```

/**
 * 新增、修改面板提交之前数据校验前置事件
 * params : view:面板对象;
 *          panel:面板对象内部form表单面板对象;
 * return : true, 阻止校验以及提交;
 */
beforevalidate : true,
/**
 * 新增、修改面板提交之前数据校验后置事件
 * params : view:面板对象;
 *          panel:面板对象内部form表单面板对象;
 *          error: 校验结果, 布尔型
 */
validate : true,
/**
 * 数据提交之前触发
 * params : data:提交的数据对象;
 *          cUrl: 提交地址;
 * return : true, 阻止提交动作; 默认为true
 */
beforecommit : true,
/**
 * 数据提交之后触发
 * params : data:提交的数据对象;
 *          cUrl: 提交地址;
 *          result: 提交成功失败结果, 布尔型;
 */
afertcommit : true,
/**
 * 修改表单滑入, 加载当前选择数据之前触发;
 * params : view: 修改表单;
 *          record : 当前选择的数据;
 * return : false: 阻止数据加载事件, 默认为true;
 */
beforeeditload : true,
/**
 * 修改表单滑入, 加载当前选择数据之后触发;
 * params : view: 修改表单;
 *          record : 当前选择的数据;
 */
aftereditload : true,
/**
 * 详情表单滑入, 加载当前选择数据之前触发;
 * params : view: 详情表单;

```

```

*         record : 当前选择的数据;
* return : false: 阻止数据加载事件, 默认为true;
*/
beforedetailload : true,
/**
* 详情表单滑入, 加载当前选择数据之后触发;
* params : view: 详情表单;
*         record : 当前选择的数据;
*/
afterdetailload : true,


/**边缘面板事件**/
beforetophide : true,
tophide : true,
beforetopshow : true,
topshow : true,
beforelefthide : true,
lefthide : true,
beforeleftshow : true,
leftshow : true,
beforebuttonhide : true,
buttonhide : true,
beforebuttonshow : true,
buttonshow : true,
beforerighthide : true,
righthide : true,
beforerightshow : true,
rightshow : true,


/**数据字典事件**/
/**
* 一个远程数据字典项被加载完毕之后触发;
* params : key:字典项类型键值;
*         store:数据字典store;
*/
lookupinit : true,
/**
* 一个本地数据字典项被加载完毕之后触发;
* params : key:字典项类型键值;
*         store:数据字典store;
*/

```

	<pre>locallookupinit : true, /** * 数据字典项全部加载完毕之后触发; * <u>params</u> : lookupManager: 数据字典管理器 */ alllookupinit : true, /**属性面板事件**/ beforetreecreate : true, treecreate : true };</pre>

Wlj-frame-function-app.js

类名	Wlj.frame.functions.app.App		
说明	开发框架主体 APP 原型。由构建器创建。主要参数参看头文件中的配置项。		
父类	Object	Xtype	
事件方法属性	<pre>/** * 页面整体APP对象 * @param cfg : APP对象整体参数 * <u>url</u> : * 类型: string; * 说明: 数据查询URL地址; * 必选: 是; * * formViewers[createFormViewer editFormViewer detailFormViewer] : * 类型: array[object{fields[string], <u>fn</u>, columnCount}]; * 说明: 新增、修改、详情业务数据表单分组;form表单, 基础配置, * createFormViewer editFormViewer detailFormViewer可单独对新增、修改、 * 详情面板进行配置; * 字段业务分组: * fields: 数组对象, 某个分组中所需要的字段对象; 必选; * <u>fn</u>: 分组字段初始化逻辑。该function可依次接收到[fields] 数组中声明的字段; * 返回经过业务逻辑处理后的字段数组, 该数组中, 字段的顺序将</pre>		

成为分组面板中字段的展示顺序;

* 必选;

* columnCount: 字段面板中, 字段列数。可选, 默认值为, 面板宽度大于1024时为4, 小于1024时, 为3

* 必选: 是;

* fields :

* 类型: array[object];

* 说明: 页面主工作区域所涉及到的字段申明; 该申明, 将被运用于查询面板、结果列表、增删查改的展示、逻辑运算等;

* 字段对象: 申明一个字段的业务逻辑、展示等的各类配置信息; 主要可配属性如下:

* name :

* 类型: string;

* 说明: 字段所对应的业务逻辑名称, 该属性应与查询结果中的字段列相同; 该属性将做为数据接收、前台控制、后台提交等的关键属性;

* 必选: 是;

* hidden :

* 类型: boolean;

* 说明: 字段展示情况, 该属性用于字段在列表、查询表单、新增、修改表单的强制隐藏;

* 必选: 否;

* text :

* 类型: string;

* 说明: 字段的中文名称, 将被用于表单中的字段标签、以及列表中的表头; 当字段没有这个属性时, 该字段将被隐藏;

* 必选: 否;

* searchField :

* 类型: boolean;

* 说明: 该字段是否做为查询字段, 在查询条件域中展示;

* 必选: 否;

* viewFn :

* 类型: function;

* 说明: 字段在列表中, 展示时的特殊展示类型; 该function将在数据行渲染时, 接收到字段的值, 返回值将做为该字段的展示效果;

* 必选: 否;

* dataType :

* 类型: string;

* 说明: 数据类型;

* 必选: 否;

* translateType :

* 类型: string;

* 说明: 字段涉及到的映射字典项; 该字典项将做为查询结果中的字段映射依据, 以及表单面板中的下拉框的选择值;

* 必选: 否;

```

*          resutlWidth :
*              类型: int;
*              说明: 字段在查询结果列中展示的宽度,默认150;
*              必选: 否;
*      必选: 是;
*      comitUrl :
*          类型: string;
*          说明: 新增、修改提交地址,若未配置该项,则使用查询URL做为提交地址;
*          必选: 否;
*      pageSize :
*          类型: int;
*          说明: 查询结果每页数据条数,默认为10;
*          必选: 否;
*      tbar :
*          类型: array[object];
*          说明: 用户扩展工具栏按钮,按钮配置同Ext.Button;
*          必选: 否;
*      edgeVies :
*          类型: object;
*          说明: 四个属性,分别配置上下左右四个边缘配置信息;
*          top:
*              类型: object;
*              说明: 工作区上部边缘配置;配置信息同: Ext.form.FormPanel;
默认配置参看:
Wlj.frame.functions.app.App.prototype.edgeViewBaseCfg.top;
*          必选: 否;
*          left:
*              类型: object;
*              说明: 工作区上部边缘配置;配置信息同: Ext.Panel,默认布局采用accordion;默认配置参看:
Wlj.frame.functions.app.App.prototype.edgeViewBaseCfg.left;
*          必选: 否;
*          right:
*              类型: object;
*              说明: 工作区上部边缘配置;配置信息同: Ext.Panel,默认布局采用accordion;默认配置参看:
Wlj.frame.functions.app.App.prototype.edgeViewBaseCfg.right;
*          必选: 否;
*          button:
*              类型: object;
*              说明: 工作区上部边缘配置;配置信息同: Ext.TabPanel;默认配置参看:
Wlj.frame.functions.app.App.prototype.edgeViewBaseCfg.button;
*          必选: 否;

```



```

*
*      必选: 否;
*      createView :
*      类型: boolean;
*      说明: 是否需要新增面板;默认为: true;
*      必选: 否;
*      editView :
*      类型: boolean;
*      说明: 是否需要修改面板;默认为: true;
*      必选: 否;
*      detailView :
*      类型: boolean;
*      说明: 是否需要详情面板;默认为: true;
*      必选: 否;
*      lookupTypes :
*      类型: array[string];
*      说明: 远程数据字典类型;APP对象会在页面渲染之前自动请求所有数据字典
项,并纳入数据字典管理器;
*      必选: 否;
*      localLookup :
*      类型: object{sring:array};
*      说明: 本地静态数据字典项;APP对象会在页面渲染之前将这些数据字典项同远
程字典项一并纳入数据字典管理器;
*      必选: 否;
*      SEARCHFIELDTRANS :
*      类型: int;
*      说明: 查询条件提交时,数据字段name属性转换方式,可选参数见:
Wlj.frame.functions.app.App.prototype.TRANS_TYPE;默认为:
Wlj.frame.functions.app.App.prototype.TRANS_TYPE.NO_TRANS;
*      必选: 否;
*      VIEWCOMMITTRANS :
*      类型: int;
*      说明: 新增修改提交时,数据字段name属性转换方式,可选参数见:
Wlj.frame.functions.app.App.prototype.TRANS_TYPE;默认为:
Wlj.frame.functions.app.App.prototype.TRANS_TYPE.TO_JAVA;
*      必选: 否;
*      validates[createValidates|editValidates] :
*      类型: array[object{dataFields[string],fn}];
*      说明: 新增、修改提交时,字段校验逻辑,逐条配置,编写;校验对象基础配置,
createValidates|editValidates可单独对新增、修改校验进行配置;
*      校验规则对象(validate object) : 包含两个属性;
*      dataFields: 数组对象,申明该校验规则中,涉及到的字段;
*      fn: 校验逻辑function,可依次接收到[dataFields]中申明的字
段的值;

```


父类		Xtype	
事件方法属性	onReady: 根据 resId 在父页面中获取该菜单对应的业务逻辑 js 代码; __build: 逐项检查配置; buildApp: 根据检查之后的配置, 创建 APP 对象;		

Wlj-frame-function-error.js

类名	Wlj.error.debuggerConfig (单态对象, 简化句柄: JDEBUG)		
说明	JDEBUG 对象控制各个级别日志 API 的开闭		
父类		Xtype	
事件方法属性	属性: ERROR: ERROR 级别日志开关; WARN: 警告级别日志开关; INFO: 普通日志开关; 方法: initLogTool: 根据配置关闭相应 API; 由 debug.js 内部方法调用;		

Wlj-frame-function-widgets.js

定义各类子组件原型。

类名	Wlj.frame.functions.app.widgets.SearchContainer		
说明	查询条件表单原型。可动态添加查询条件, 并根据查询项的多少修复表单面板尺寸。		
父类	Ext.Panel	Xtype	searchcontainer
事件方法属性	属性: columnCount: 查询条件列数, 默认为四, 根据面板尺寸, 在渲染时重定义; WCLP: 内部属性, 由 API 操纵, 标识查询条件面板是否被收起; Dyfield: 内部属性, 被动态添加的条件字段数组; 方法: fixSearchHeight: 重构面板高度, 调用系统 API, 将触发查询结果容器的高度变化; removeAllDyField: 移除所有动态添加的条件字段;		

类名	Wlj.frame.functions.app.widgets.ResultContainer		
说明	查询结果容器。该容器将展示查询结果列表, 系统的增删查改面板、开发人员扩展面板, 及分页栏信息等。		

父类	Ext.Panel	Xtype	resultcontainer
事件方法属性	<p>属性:</p> <p>CREATE_VIEW: 静态字符: createView;</p> <p>EDIT_VIEW: 静态字符: editView;</p> <p>DETAIL_VIEW: 静态字符: detailView;</p> <p>GRID_VIEW: 静态字符: gridView;</p> <p>createView : 用户配置项, 是否构建新增面板, 由createView初始化</p> <p>editView : 用户配置项, 是否构建修改面板, 由editView初始化;</p> <p>detailView : 用户配置项, 是否构建详情面板, 由 detailView 初始化;</p> <p>createFieldsCopy: 新增所需字段的配置副本, 由 APP 根据字段元数据配置拷贝;</p> <p>editFieldsCopy: 修改所需字段的配置副本, 由 APP 根据字段元数据配置拷贝;</p> <p>detailFieldsCopy: 详情所需字段的配置副本, 由 APP 根据字段元数据配置拷贝;</p> <p>createFormViewer: 新增面板表单设计配置信息, 由 <u>createFormViewer</u> 或者 <u>formViewers</u> 初始化;</p> <p>editFormViewer: 修改面板表单设计配置信息, 由 editFormViewer 或者 formViewers 初始化;</p> <p>detailFormViewer: 详情面板表单设计配置信息, 由 detailFormViewer 或者 formViewers 初始化;</p> <p>createValidates: 新增校验配置信息, 由 createValidates 或者 validates 初始化;</p> <p>editValidates: 修改校验配置信息, 由 editValidates 或者 validates 初始化;</p> <p>createLinkages: 新增面板字段联动逻辑配置信息, 由 createLinkages 或者 linkages 初始化;</p> <p>editLinkages: 修改面板字段联动逻辑配置信息, 由 editLinkages 或者 linkages 初始化;</p> <p>createViewText: 静态字符: 新增;</p> <p>editViewText: 静态字符: 修改;</p> <p>detailViewText: 静态字符: 详情;</p> <p>gridViewText: 静态字符: 列表;</p> <p>NEXT_PAGE: 静态字符: nextPage;</p> <p>PRE_PAGE: 静态字符: prePage;</p> <p>FIRST_PAGE: 静态字符: firstPage;</p> <p>LAST_PAGE: 静态字符: lastPage;</p> <p>nextPageText: 静态字符: 下一页;</p> <p>prePageText: 静态字符: 上一页;</p> <p>firstPageText: 静态字符: 首页;</p> <p>lastPageText: 静态字符: 尾页;</p> <p>turnToPageText: 静态字符: 转到;</p> <p>nextPage: 是否显示下一页按钮;</p> <p>prePage: 是否显示上一页按钮;</p> <p>firstPage: 是否显示首页按钮;</p>		

	<p>lastPage: 是否显示尾页按钮;</p> <p>pageSize: 单页数据条数, 默认为 10, 由 pageSize 初始化;</p> <p>currentPage: 内部变量, 当前页数;</p> <p>totalPage: 内部变量, 总页数;</p> <p>totalLength: 内部变量, 总数据条数;</p> <p>url: 数据查询 URL, 由 url 初始化;</p> <p>dataFields: 数据列配置属性, 由 APP 对象从 fields 属性拷贝副本初始化;</p> <p>jsonRoot: 请求数据根节点, 默认为 json.data;</p> <p>jsonCount: 请求数据总数路径;</p> <p>store: 内部变量, 列表数据源;</p> <p>currentParams: 内部变量, 当前查询条件;</p> <p>gridContainerTpl: 数据布局容器模版对象;</p> <p>gridScrollTpl: 数据布局滚动调模版对象;</p> <p>viewPanel.createView: 新增面板句柄;</p> <p>viewPanel.editView: 修改面板句柄;</p> <p>viewPanel.detailView: 详情面板句柄;</p> <p>customerViewPanels: 扩展面板对象数组;</p> <p>方法:</p> <p>createPageButtons: 创建翻页按钮;</p> <p>initViews: 初始化信息面板;</p> <p>createCustomerViews: 初始化用户自定义信息面板;</p> <p>showView: 滑入某个信息面板;</p> <p>hideCurrentView: 滑出当前信息面板;</p> <p>initDataEvent: 绑定数据源数据源事件;</p> <p>initStore: 初始化数据源对象;</p> <p>resetContainer: 计算数据列表高度;</p> <p>resetTiles: 刷新数据列表展示;</p> <p>booterDataTiles: 渲染所有数据行;</p> <p>turnToCurrentPage: 跳到当前页;</p> <p>gridMoveOut: 隐藏数据列表页;</p> <p>gridMoveIn: 展示数据列表页;</p> <p>nextPageHandler: public, 转到下一页;</p> <p>prePageHandler: public, 转到上一页;</p> <p>firstPageHandler: public, 转到第一页;</p> <p>lastPageHandler: public, 转到最后一页;</p> <p>gridViewHandler: public, 转到数据列表页;</p>

类名	Wlj.frame.functions.app.widgets.RecordView		
说明	数据行展示对象, 用于生成数据行瓷贴对象或者数据行展示所需的 HTML 代码;		
父类	Ext.util.Observable	Xtype	
事件方法属性	属性: alwaysField: 字段数据为空时, 是否依然创建占位符;		

	<p>showLabel: 是否展示字段名称;</p> <p>cellTemplate: 内部变量, 根据字段 key 生成的数据字段展示模版对象;</p> <p>float: 瓷贴模式下, 数据瓷贴左悬浮;</p> <p>方法:</p> <p>formatFieldData: 数据显示格式化, 将调用用户对该字段的展示逻辑;</p> <p>translateFieldData: 数据字典映射逻辑;</p> <p>createRecordHTML: 以 HTML 格式展示时, HTML 生成逻辑;</p> <p>createFieldTiles: 以瓷贴格式展示时, 瓷贴生成逻辑;</p> <p>createRecordTile: 单个字段瓷贴生成逻辑;</p>

类名	Wlj.frame.functions.app.widgets.View		
说明	各类容器面板对象的基类;		
父类	Ext.Panel	Xtype	basicview
事件方法属性			

类名	Wlj.frame.functions.app.widgets.RView		
说明	展示于查询结果容器内置信息面板对象的基类。提供滑入滑出事件的初始化, 以及在容器面板中添加滑入按钮;		
父类	Wlj.frame.functions.app.widgets.View	Xtype	resultview
事件方法属性			

类名	Wlj.frame.functions.app.widgets.CView		
说明	<p>用户新增、修改、删除的面板类原型;</p> <p>1、根据配置构建表单面板;</p> <p>2、处理字段的校验、联动、提交逻辑;</p> <p>3、提供数据行接入展示逻辑;</p>		
父类	Wlj.frame.functions.app.widgets.RView	Xtype	cview
事件方法属性	<p>属性:</p> <p>openValidate: 是否开启数据校验逻辑;</p> <p>svButton: 是否需要保存按钮;</p> <p>fields: 面板字段信息;</p> <p>validates: 面板校验信息;</p> <p>linkages: 面板联动信息;</p> <p>record: 面板当前展示数据, false 为无数据;</p> <p>方法:</p> <p>setRecord: 设置面板展示数据;</p>		

	<p>reset: 重置面板;</p> <p>buildContent: 构建内部表单面板;</p> <p>createColumnsCfg: 构建表单列数组;</p> <p>buildFormField: 构建表单面板内部所有分组字段;</p> <p>getFieldsByName: 根据 name 属性获取字段对象;</p> <p>validateData: 校验表单数据;</p> <p>linkaging: 字段联动逻辑触发函数;</p> <p>getLinkFields: 根据字段获取附属于该字段的联动逻辑所控制的其他字段;</p>

类名	Wlj.frame.functions.app.widgets.BView		
说明	结果容器扩展面板类，用户自定义扩展面板均采用此类面板做为容器;		
父类	Wlj.frame.functions.app.widgets.RView	Xtype	buisnessview
事件方法属性			

类名	Wlj.frame.functions.app.widgets.ComboTree		
说明	下拉树组件		
父类	Ext.form.ComboBox	Xtype	wcombotree
事件方法属性	<p>属性:</p> <p>innerTree: 下拉树, string 或者 Object; 当为 string 类型时, 值指向 treeManager 配置的一个 tree 面板的 key。</p>		

类名	Wlj.frame.functions.app.widgets.TreeManager (单态句柄: TreeManager)		
说明	树形结构管理器		
父类	Ext.util.Observable	Xtype	
事件方法属性	<p>方法:</p> <p>addLoader: 向树形管理器添加一个数据源配置, 配置信息见: Com.yucheng.bcrm.ArrayTreeLoader;</p> <p>initTree: 添加一个受控树形面板配置;</p> <p>createTree: 参数为 string 或者 Object。根据 key 或者配置获取一个树形面板实例;</p> <p>loadTreesCfgs: 根据业务逻辑代码初始化树形结构配置;</p> <p>checkTreesCfgs: 检查树形结构配置;</p>		

4. 开发细则

功能模块业务逻辑开发，以变量申明和方法申明的方式进行的编写，并可在必要的逻辑环节添加自定义日志代码，用以控制台日志输出。

1) 第三方类库的引用：

对于使用到第三方类库的代码，比如系统的一些公共组件，公共选择器等，需要在业务逻辑代码开始的地方申明，具体申明例子如下：

```
imports([
  '/contents/pages/com.yucheng.bcrm/com.yucheng.bcrm.js',
  '/contents/pages/common/Com.yucheng.bcrm.common.Annacommit.js'
]);
```

2) 变量的定义：

业务逻辑代码中的变量定义将做为 APP 构建的参数使用；有效的变量定义，已经在 Wlj-frame-function-header.js 文件中申明了原型。

在头文件定义的变量句柄之外定义的变量，将被 APP 忽略，但可做为全局变量，在业务逻辑片段中使用。

头文件中所有变量原型的含义、格式、是否必输均有说明。如为配置必输项或者必输项不符合格式，系统将报错，代码中断执行。

3) 方法申明：

- 1、Wlj-frame-function-header.js 文件中 `listeners` 对象申明了所有可用事件原型；
- 2、业务逻辑代码中所有与 `listeners` 对象属性同名函数将被做为事件监听器装载在 APP 对象上，在一定条件下触发，并可接收到事件参数；
- 3、Listeners 对象中定义的原型以外的函数将不可被系统触发，但可做为全局方法，被其他函数调用。

小技巧：在 INFO 级别日志功能开启时，所有的事件触发均会有日志记录，可根据此日志记录确定业务逻辑方法触发的时机；

自定义全局方法可以用于编写公共的逻辑，并在事件触发函数内部调用，增加代码复用率。

4) 系统 API 的使用：

- 1、Wlj-frame-function-api.js 文件中 API 对象定义了所有可用系统 API 方法；

- 2、系统 API 方法作用域均为 window 对象，可直接通过方法名调用；
- 3、API 方法均由 APP 构建，所以在顺序代码中，API 方法尚未构建，不可调用；在事件方法、内部调用方法中可调用。

5) 日志 API 的使用：

- 1、日志 API 在系统代码加载完毕之后即构建完成，在业务逻辑代码文件中随时可以调用；
- 2、Wlj-frame-function-error.js 文件中配置了各个级别日志代码的开关，打开开关，即可在控制台输出相关日志信息；
- 3、系统的代码检查以及的事件触发等均进行了日志记录，打开各个级别控制台后，即可看到的页面实时运行情况。