前端开发

# 前端流程

首先web.xml中利用sitemesh和freemarker对页面进行装饰。

common.ftl：负责装饰所有请求页面，它确定了整个框架。

主框架结构图：hd头，bd主体，ft脚，content-wrapper左栏。



首先定义了YUI\_VERSION，SITE\_KEY两个变量，然后引入header.core.inc，接着为界面主框架，最后引入了footer.inc。

header.core.inc：主要负责校正浏览器时间和引入jsdeps\_all.inc。

jsdeps\_all.inc：通过定义DOMAIN\_HOST，COMBO\_BASE，YUI\_VERSION，配置JS环境变量，例如：var M = window.M || {};，并引入yui-config-mtoa.js文件

footer.inc：主要负责ga参数初始化和跟踪，控件初始化。Y.mt.widget.init()。

**主要注意2点：**

1. **yui-config-mtoa.js**

重新对yui-min.js进行补充，设计了M代表Meituan JS框架，并重新封装了一些模块，如w-base，mt-base，oa-xxx等。

/\*\*

\* the unique global variable for Meituan JS Framework

\* @class M

\* @static

\*/

var M = global.M;

1. **Y.mt.oa.page(OA\_config)**

page.js文件中，page方法除了负责初始化oa环境变量之外，最重要的是绑定方法。

bindOA()： 将OA与相应方法绑定

//登录后调用loadMenu方法

oa.after(EVT\_LOGIN, function(data) {

var account = oa.account;

account.loadMenu();

});

//所有路径都用pageDispatch分发

router.route('\*', pageDispatch);

pageDispatch：首先从url解析出模块名，如，然后调用syncPage，分别oa.header.sync()，oa.sidebar.sync()

//body的点击由handleClick响应

Y.one('body').delegate('click', handleClick, 'a');

handleClick：从target中解析出url，然后调用oa.fire(EVT\_REDIRECT, evt);

dispatch()：调用EVT\_REDIRECT事件

defRedirect()：加载菜单，

oa.publish(EVT\_REDIRECT, {

emitFacade: true,

defaultFn: defRedirect

});

封装组件

其余都是由ajax发起请求，处理响应

FeListener的作用是在容器初始化时，从static目录下读取所有js文件的依赖关系，写入site\_js\_dep.js中。

# 前端参数

注：[]代表值为数组，{}代表值为对象。

## DataList

DataList表单控件

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 含义 |
| billCode |  | 表单码 |
| flowName |  | 前端未支持 |
| isStat |  |  |
| components | [] | DataList中包含的组件列表 |
| searchBox | {} | 搜索框控件，详情见searchBox表 |
| byTable |  | 表格控件，详情见byTable表 |

## SearchBox

SearchBox搜索框控件

|  |  |  |
| --- | --- | --- |
| 一级参数 | 二级参数 | 三级参数 |
| searchBoxConfig | title：string，标题 |  |
| columnConf：{}，配置列对象 | {  count：列数  } |
| data：[]，显示内容 | [  code：字段码  name：显示名称  type：字段类型，根据不同类型，不同显示方式，如T\_TIMESTAMP，则显示成时间选择控件  tips：提示，控件下面显示一行小字] |
| searchUrl：点击查询按钮后，利用searchUrl发起查询请求 |  |
| url：获取单据模板内容。  默认为Y.mt.oa.URLS. bill\_model\_query\_enter |  |
| dealPos：??? | {  showList：[]  } |

### Q&A

1. **searchUrl和url的区别？**

searchUrl：单击查询按钮后，获取数据的url

url：是获取单据模板的url，即获取oa.pf.bs.model.find.action文件的内容

1. **searchUrl详解？**

控件初始化完毕后，利用searchUrl发起请求，SearchBox会将查询参数封装param。

## byTable

|  |  |  |
| --- | --- | --- |
| byTable表格控件 |  |  |
| queryData：{}，在所有请求的参数中都添加queryData的内容 |  |  |
| byTableConfig：{}，表格配置对象 | headerUrl：string |  |
| submitQueryName：提交查询名称，必须与Controller的@RequestParam(“submitQueryName”)一致，默认名称是param。 |  |
| isSort：boolean，显示是否排序 |  |
| toolbar：{}，表格上方的工具栏，默认实现CRUD，其他按钮要自己实现  ，详情见toolbar表 |  |
| thParams：[]，表格列内容 | code：字段名  showlist：是否在单据的列表状态显示  showcard：是否在单据的卡片状态显示  name：显示名称  type：字段类型  width：字段宽度，可以为百分比 |
| listParams：??? |  |
| checkboxColumn：boolean，每行前面是否有checkbox |  |
|  | linkCell： |  |

dyTable与searchBox联合使用时，若headerUrl与searchBox的url相同，则不必设置该值

## toolbar

|  |  |  |
| --- | --- | --- |
| toolbar表格上方的工具栏 |  |  |
| toolbar：[]，配置工具栏按钮，  默认实现addNew,modifyNew,del,view | name：string，按钮名称 |  |
| title：string，按钮显示名称 |  |
| callback：function，单击按钮后的回调函数，值封装在o.data中 |  |
| importAction：string， |  |
| query：{}，??? |  |
| dynamicFormConf：{}，动态表格配置 | beforeBuildItem：{}，建立item前执行的操作，如校验等。??? | before：function， |
| after：function， |
| afterSetValue：??? |  |
|  |
| beforeSaveSubmit：function，提交保存前调用该方法，数据封装在o.formData中 |  |  |
|  |  |  |

# SiteMesh&FreeMarker

SiteMesh:<http://lichen.blog.51cto.com/697816/159468>

FreeMarker:<http://www.cnblogs.com/sdauraul/archive/2012/12/19/2824643.html>

## web.xml配置

WEB-INF/web.xml

**//为使sitemesh支持freemarker，必须添加FreemarkerDecoratorServlet**

   <servlet>

        <servlet-name>sitemesh-freemarker</servlet-name>

        <servlet-class>com.opensymphony.module.sitemesh.freemarker.FreemarkerDecoratorServlet</servlet-class>

        <init-param>

            <param-name>TemplatePath</param-name>

            <param-value>/</param-value>

        </init-param>

**//UTF-8编码**

        <init-param>

            <param-name>default\_encoding</param-name>

            <param-value>UTF-8</param-value>

        </init-param>

**//第一个启动**

        <load-on-startup>1</load-on-startup>

    </servlet>

    <servlet-mapping>

        <servlet-name>sitemesh-freemarker</servlet-name>

        <url-pattern>\*.ftl</url-pattern>

    </servlet-mapping>

**//必须要配置SiteMeshFilter，否则装饰器不起作用**

    <filter>

        <filter-name>sitemesh</filter-name>

        <filter-class>com.opensymphony.sitemesh.webapp.SiteMeshFilter</filter-class>

    </filter>

    <filter-mapping>

        <filter-name>sitemesh</filter-name>

        <url-pattern>/\*</url-pattern>

        <dispatcher>REQUEST</dispatcher>

        <dispatcher>FORWARD</dispatcher>

    </filter-mapping>

## sitemesh配置文件

WEB-INF/sitemesh.xml

**必须放在WEB-INF下**

<?xml version="1.0" encoding="UTF-8"?>

<sitemesh>

**//设置decorators.xml位置**

    <property name="decorators-file" value="/WEB-INF/conf/decorators.xml" />

    <excludes file="${decorators-file}" />

**//page-parsers：负责读取stream的数据到一个page对象，然后被SiteMesh解析**

    <page-parsers>

        <parser content-type="text/html"

            class="com.opensymphony.module.sitemesh.parser.HTMLPageParser" />

    </page-parsers>

**//decorator-mappers：配置不同装饰器**

    <decorator-mappers>

        <mapper class="com.opensymphony.module.sitemesh.mapper.PageDecoratorMapper">

            <param name="property.1" value="meta.decorator" />

            <param name="property.2" value="decorator" />

        </mapper>

        <mapper

            class="com.opensymphony.module.sitemesh.mapper.FrameSetDecoratorMapper">

        </mapper>

        <mapper class="com.opensymphony.module.sitemesh.mapper.AgentDecoratorMapper">

            <param name="match.MSIE" value="ie" />

            <param name="match.Mozilla [" value="ns" />

            <param name="match.Opera" value="opera" />

            <param name="match.Lynx" value="lynx" />

        </mapper>

**//打印装饰器，当用http://localhost/aaa/a.html?printable=true访问时给出原始页面以供打印（去掉header，footer）**

        <mapper

     class="com.opensymphony.module.sitemesh.mapper.PrintableDecoratorMapper">

            <param name="decorator" value="printable" />

            <param name="[parameter.name](http://parameter.name/)" value="printable" />

            <param name="parameter.value" value="true" />

        </mapper>

        <mapper class="com.opensymphony.module.sitemesh.mapper.RobotDecoratorMapper">

            <param name="decorator" value="robot" />

        </mapper>

        <mapper

            class="com.opensymphony.module.sitemesh.mapper.ParameterDecoratorMapper">

            <param name="decorator.parameter" value="decorator" />

            <param name="[parameter.name](http://parameter.name/)" value="confirm" />

            <param name="parameter.value" value="true" />

        </mapper>

        <mapper class="com.opensymphony.module.sitemesh.mapper.FileDecoratorMapper">

        </mapper>

**//指定decorators.xml的存放路径和名称，默认为/WEB-INF/decorators.xml**

        <mapper

            class="com.opensymphony.module.sitemesh.mapper.ConfigDecoratorMapper">

            <param name="config" value="${decorators-file}" />

        </mapper>

    </decorator-mappers>

</sitemesh>

## WEB-INF/conf/decorators.xml

<?xml version="1.0" encoding="UTF-8"?>

**//defaultdir：包含装饰器页面的目录**

<decorators defaultdir="/WEB-INF/decorators">

**//excludes：包含以下字符串的url，不需要使用装饰器修饰**

      <excludes>

        <pattern>\*bpm.html\*</pattern>

        <pattern>\*oa.html\*</pattern>

        <pattern>\*TestCaseApp.html\*</pattern>

        <pattern>/oa\*</pattern>

        <pattern>/api\*</pattern>

        <pattern>/service\*</pattern>

        <pattern>/localhost\*</pattern>

    </excludes>

**//name：别名，page：装饰器页面，pattern：路径**

**//对所有/\*路径都使用common.ftl装饰**

    <decorator name="common" page="common.ftl">

        <pattern>/\*</pattern>

    </decorator>

</decorators>