**前端开发规范**

目录

[第1章 WEB开发规范 1](#_Toc452968381)

[1.1规范目的 1](#_Toc452968382)

[1.2目录结构规范 1](#_Toc452968383)

[1.3文件命名规范 3](#_Toc452968384)

[1.3.1 html和模板文件命名规范 4](#_Toc452968385)

[1.3.2 js文件名命名规范 4](#_Toc452968386)

[1.3.3 css文件名命名规范 4](#_Toc452968387)

[1.3.4 图片资源文件命名规范 4](#_Toc452968388)

[1.4代码编写规范 4](#_Toc452968389)

[1.4.1 html和模板代码编写规范 4](#_Toc452968390)

[1.4.2 js代码编写规范 6](#_Toc452968391)

[1.4.3 CSS代码编写规范 7](#_Toc452968392)

[1.4.4 注释编写规范 7](#_Toc452968393)

[第2章common 8](#_Toc452968394)

[2.1 使用说明 8](#_Toc452968395)

[2.1.1 引入 8](#_Toc452968396)

[2.1.2 API使用说明 8](#_Toc452968397)

[2.2 开发说明 10](#_Toc452968398)

[2.1.1 结构说明 10](#_Toc452968399)

[2.1.2 源码模块编写说明 11](#_Toc452968400)

# 第1章 WEB开发规范

## 1.1规范目的

为提高团队协作效率, 便于后台人员添加功能及前端后期优化维护, 输出高质量的文档, 特制订此文档. 本规范文档一经确认, 前端开发人员必须按本文档规范进行前台页面开发. 本文档如有不对或者不合适的地方请及时提出, 经讨论决定后方可更改.

## 1.2目录结构规范

为了方便代码的管理，以及构建工具的使用，前端项目目录结构规范如下图所示：

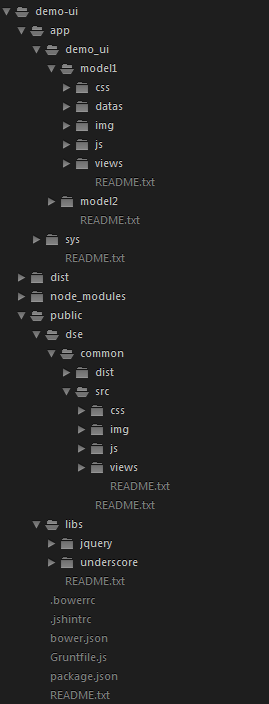


图1-1 前端目录结构图

每一个目录里面都有一个README.txt里面说明这层目录的作用以及结构，方便其他开发人员可以通过该文件了解项目。

app目录为系统源码目录；

sys目录为系统级别模块；

model1目录为指定模块目录

css目录存放的是样式文件

img目录存放的是图片资源文件

datas目录为前端同后端协商好的数据结构数据

js目录存放的是脚本文件

views目录存放的是页面

template目录存放模板文件

projectName目录下面存放的是项目的各个模块；

model1目录为指定模块目录

css目录存放的是样式文件

img目录存放的是图片资源文件

datas目录为前端同后端协商好的数据结构数据

js目录存放的是脚本文件

views目录存放的是页面

template目录存放模板文件

model2

dist目录为系统发布代码目录；

public目录为第三方依赖目录；

dse 目录为公司开发的公共组件库

common 公共组件库

src 源码，由开发common的开发人员去维护

js

css

img

views

template

dist 打包后代码，项目引用的是该目录下的文件

others

libs目录为第三方依赖库

根目录文件说明：

Gruntfile.js grunt配置文件

package.json node依赖配置信息以及项目的信息

.bowerrc bower依赖库存放位置配置文件

bower.json bower第三方类库依赖配置文件

.jshintrc 代码规范检查配置文件

## 1.3文件命名规范

文件名全部必须英文半角，不得使用中文或者全角。

### 1.3.1 html和模板文件命名规范

按照模块进行命名，模板命名则为：模块名称-模板用意-template.html。

例如用户[user]模块：

user.html，新增用户模板名称：user-add-template.html。

### 1.3.2 js文件名命名规范

按照模块名称进行驼峰试命名，每个模块中必须包含两个js文件，一个为\*Sea.js，另一个为\*main.js，有再需要增加的js文件都以模块名称为开头。

例如用户[user]模块：

userSea.js存放seajs的依赖配置，userMain.js为模块的入口文件，如果与另外需要增加js文件，例如增加用户的js文件，则命名为userAdd.js。

### 1.3.3 css文件名命名规范

按照模块名称进行驼峰试命名。

### 1.3.4 图片资源文件命名规范

所有页面元素类图片均放入img文件夹;图片格式仅限于gif || png || jpg;命名全部用小写英文字母 || 数字 || \_ 的组合，其中不得包含汉字 || 空格 || 特殊字符；尽量用易懂的词汇, 便于团队其他成员理解; 命名分头尾两部分, 用下划线隔开, 比如ad\_left01.gif || btn\_submit.gif;

尽量避免使用半透明的png图片(若使用, 请参考css规范相关说明);

## 1.4代码编写规范

### 1.4.1 html和模板代码编写规范

1）文档声明采用html5，编码统一用utf-8；代码要缩进，属性值采用双引号。

2）语义化html, 如 标题根据重要性用h(同一页面只能有一个h1),

3）段落标记用p, 列表用ul, 内联元素中不可嵌套块级元素;尽可能减少div嵌套。

4）书写链接地址时, 必须避免重定向，例如：href=”http://www.example.com/”, 即须在URL地址后面加上“/”，空链接用代码“javascript:void(0)”代替原来的“#”标记；

5）在页面中尽量避免使用style属性,即style=”…”；

6）必须为含有描述性表单元素(input, textarea)添加label；

7）能以背景形式呈现的图片, 尽量写入css样式中;重要图片必须加上alt属性; 给重要的元素和截断的元素加上title;

8）给区块代码及重要功能(比如循环)加上注释, 方便后台添加功能;

9）特殊符号使用: 尽可能使用代码替代: 比如 <(<) & >(>) & 空格( ) & »(») 等等；

10）每个模块采用三层去规范代码,第一层为总体包裹div类名为dse-continer;里面有头[<div calss="dse-header">]、内容[<div class="dse-content">]、底部[<div class="dse-footer">]三层,模块超出50行的必须加上注释，注释采用模块说明+空格+start，模块结束时结尾也需要加注释，同样为模块说明+空格+end。方便代码块的检索与区分,自定义属性采用data-开头，一般情况下使用data-实体名称[entity]-属性名称[propname]。如下例子[具体可以参考common模块里的blank.html]：

<html>

<head>

<meta charset="UTF-8">

<title>Document</title>

<!-- 第三方插件 样式 start -->

<link rel="stylesheet" type="text/css" href="" />

<!-- 第三方插件 样式 end -->

<!-- common 样式 start -->

<link rel="stylesheet" type="text/css" href="">

<!-- common 样式 end -->

<!-- ${module} 为具体的模块名称，build为打包后的路径 -->

<!-- 业务 样式 start -->

<!-- build:css /app/info/${module}/css/main.min.css -->

<link rel="stylesheet" href="/app/info/${module}/css/main.css">

<!-- endbuild -->

<!-- 业务 样式 end -->

<!-- 业务 脚本 start -->

<!-- build:js /app/info/${module}/js/main.min.js -->

<script src="/app/info/${module}/js/${module}Sea.js"></script>

<!-- endbuild -->

<!-- 业务 脚本 end -->

</head>

<body>

<!-- module 具体功能 start -->

<!-- 容器,容器放的为相对独立的一个构件，容器可以嵌套容器，header、content、footer根据实际情况可以选择用 -->

<div class="dse-continer">

<div data-moduel-propName="" class="dse-header">

<!-- 头部内容 -->

</div>

<div class="dse-content">

<!-- 具体内容 -->

</div>

<div class="dse-footer">

<!-- 底部内容 -->

</div>

</div>

<!-- module 具体功能 end -->

</body>

</html>

### 1.4.2 js代码编写规范

1）js代码依赖采用bower去安装管理依赖，具体安装命令为：bower install 插件名称#版本号 --save；卸载命令为bower uninstall 插件名称；

2）尽量确保页面没有js代码，全部都使用引入js文件；

3）采用seajs进行模块化编程，约定模块配置为"模块名称Sea.js";具体模块业务入口文件为"模块名称Main.js";

4）js代码规范需基本通过jshint的代码检查；

5）如果使用ES6规范进行开发，局部变量必须用let，常量必须用const；

6）类的命名首字母要大写，方法首字母应小写，ES5规范也要这样。

7）对于jquery命名采用$开头以作为标示；私有成员以及方法采用\_开头标示，私有的都不为外部调用，作为一种约定使用。

### 1.4.3 CSS代码编写规范

1）语义化class(类)名和ID；不要使用标签选择器；一般情况下ID不应该被应用于样式。

2）应该始终考虑使用class，而不是id，除非只使用一次。

3）应该尽可能使用缩写属性（如 font 字体）。

4）始终使用小写的十六进制数字。（如：#fffccc）。

5）为了保证一致性和可扩展性，每个声明应该用分号结束，每个声明换行。

6）属性名的冒号后使用一个空格。出于一致性的原因，属性和值（但属性和冒号之间没有空格）的之间始终使用一个空格。

7）属性选择器或属性值用双引号（" "）;URI值（url()）不要使用引号。

### 1.4.4 注释编写规范

1） html注释: 注释格式 <!--这儿是注释-->, '--'只能在注释的始末位置,不可置入注释文字区域;

2）css注释: 注释格式 /\*这儿是注释\*/;

3）JavaScript注释, 单行注释使用'//这儿是单行注释' ,多行注释使用 /\* 这儿有多行注释 \*/;方法的注释必须说明方法用途已经参数说明，包含类型。

# 第2章common

## 2.1 使用说明

### 2.1.1 引入

只需要在"模块Sea.js"中配置依赖即可，如下代码所示：

'dseCommon':'dse/common/dist/js/main.dse.min.js'

使用编写如下代码即可获取所有common提供的服务：

let DseCommon = reqiure('dseCommon').DseCommon;

### 2.1.2 API使用说明

#### 2.1.2.1 DseRequest

请求封装类，包含客户端请求，下载请求，以及对后台返回统一接口对象进行处理以及对模板进行处理；后台返回的结构为：

{

data: Obj,

status: Number,

message : {msg: String,...}

}

依赖jquery和underscore

##### 2.1.2.1.1 request

1）说明：请求后台统一接口，主要对success和error进行处理。

2）调用：DseRequest.request(params);

3）参数：

类型:Object

值：{

success: function(data,msgObj){},

error: function(msgObj){}

}

4）返回结果：this;

5）示例：

##### 2.1.2.1.2 getTemplate

1）说明：获取模板文件，传入要渲染的数据，返回渲染后的结果。

2）调用：DseRequest.getTemplate(params);

3）参数：

类型:Object

值：{

data:Object,

success: function(resolverResultData){}

}

4）返回结果：this;

5）示例：

##### 2.1.2.1.3 requestDownLoad

1）说明：请求后台下载文件。

2）调用：DseRequest.requestDownLoad(params);

3）参数：

类型:Object

值：{

url:String

}

4）返回结果：this;

5）示例：

#### 2.1.2.2 DseUtils

##### 2.1.2.2.1 ObjectUtil

用于处理对象的工具类，如创建对象属性，复制对象等。

###### 2.1.2.2.1.1 createReadOnlyProp

1）说明：创建只读对象属性。

2）调用：DseUtils.ObjectUtil.createReadOnlyProp (obj,propName,value);

3）参数：

类型:Object 被创建对象

类型:String 属性名称

类型:Object 属性的值

4）返回结果：DseUtils.ObjectUtil;

5）示例：

###### 2.1.2.2.1.2 depthCopy

1）说明：深度复制一个对象为全新的对象,不存在引用。

2）调用：DseUtils.ObjectUtil.depthCopy(obj);

3）参数：

类型:Object 被复制对象

4）返回结果：Object 新对象,如果传入的为非对象,则返回一个空对象{};

5）示例：

###### 2.1.2.2.1.3 getPropValue

1）说明：返回对象的属性的值，用于不清楚传入对象是否为空的情况，如果对象为空则返回undefined。

2）调用：DseUtils.ObjectUtil.getPropValue(obj, propName);

3）参数：

类型:Object 目标对象

类型:String 对象的属性名称

4）返回结果：Object 对象的值;

5）示例：

###### 2.1.2.2.1.4 getPropValueInt

1）说明：返回对象的属性的值，用于不清楚传入对象是否为空的情况，如果对象为空则返回0。

2）调用：DseUtils.ObjectUtil.getPropValueInt(obj, propName);

3）参数：

类型:Object 目标对象

类型:String 对象的属性名称

4）返回结果：Object 对象的值，如果对象为空则返回0;

5）示例：

###### 2.1.2.2.1.5 arrayToTree

1）说明：将数组转换成树。

2）调用：DseUtils.ObjectUtil.arrayToTree(target,config);

3）参数：

类型:Array 目标数组

类型: Object 配置项

{

idField:String,id的属性名称，如果没有默认拿id

pidField:String,pid的属性名称,如果没有默认拿pid

childrenField:String,children的属性名称，如果没有默认拿children

}

4）返回结果：Object 返回树结构对象，如果传入为非数组，则返回空对象;

5）示例：

###### 2.1.2.2.1.6 treeToArray

1）说明：将数组转换成树。

2）调用：DseUtils.ObjectUtil.treeToArray(tree,config);

3）参数：

类型: Object||Array 目标对象

类型: Object 配置项

{

childrenField:String 子节点的名称,如果没有默认为children }

4）返回结果：Array 转换后数组;

5）示例：

###### 2.1.2.2.1.7 treeToMap

1）说明：将数组转换成树。

2）调用：DseUtils.ObjectUtil.treeToMap(array,config);

3）参数：

类型: Array 树数组

类型: Object 配置项

{

keyField: 对象的key属性，如果没有默认为id的值作为key

valueField: key对应的值属性，如果没有传，默认采用的是对象 }

4）返回结果：Array 返回一个key对应的对象，如果传入非数组，则返回空对象{};

5）示例：

###### 2.1.2.2.1.8 isString

1）说明：检查是否为字符串。

2）调用：DseUtils.ObjectUtil.isString(target);

3）参数：

类型: Object 被检查对象

4）返回结果：Boolean true为字符串，false为非字符串;

5）示例：

###### 2.1.2.2.1.9 isObject

1）说明：检查是否为对象。

2）调用：DseUtils.ObjectUtil.isObject(target);

3）参数：

类型: Object 被检查对象

4）返回结果：Boolean true为对象，false为非对象;

5）示例：

###### 2.1.2.2.1.10 isEmpty

1）说明：判断目标对象是否为空，[],{}等都为空。

2）调用：DseUtils.ObjectUtil.isEmpty(target);

3）参数：

类型: Object||Array 被检查对象

4）返回结果：Boolean true表示为空，false表示不为空;

5）示例：

###### 2.1.2.2.1.11 isFalse

1）说明：判断对象是否为false值。

2）调用：DseUtils.ObjectUtil.isFalse(target);

3）参数：

类型: Object||Array 被检查对象

4）返回结果：Boolean

如果为'',null,'null',false,'false',undefine,'undefine',0;

5）示例

##### 2.1.2.2.2 StringUtil

字符串处理工具类，如获取真实长度、替换指定字符串、转换字符串大小、是否包含某子串、移除最后一个字符串等。

###### 2.1.2.2.2.1 realLength

1）说明：获取字符串真实的长度，即汉子为两个字节的长度。

2）调用：DseUtils.StringUtil.realLength(str);

3）参数：

类型:String 目标字符串

4）返回结果：Number 字符串长度:-1表示传入的是非字符串;

5）示例：

###### 2.1.2.2.2.2 htmlEscape

1）说明：转义HTML字符串，替换&, <, >, ", ', 和 /字符。

2）调用：DseUtils.StringUtil.htmlEscape(str);

3）参数：

类型:String 目标字符串

4）返回结果：String 转换结果;

5）示例：

###### 2.1.2.2.2.3 htmlUnEscape

1）说明：和htmlEscape相反。转义HTML字符串，替换&, &lt;, &gt;, &quot;, &#96;, 和 &#x2F;字符。

2）调用：DseUtils.StringUtil.htmlUnEscape(str);

3）参数：

类型:String 目标字符串

4）返回结果：String 转换结果;

5）示例：

###### 2.1.2.2.2.4 relpaceAll

1）说明：替换字符串中所有指定的字符为新字符。

2）调用：DseUtils.StringUtil.relpaceAll(str,oldStr,newStr);

3）参数：

类型:String 目标字符串

类型:String 要被替换的字符串

类型:String 替换为新的字符串

4）返回结果：String 返回替换结果字符串

5）示例：

###### 2.1.2.2.2.5 capitalize

1）说明：将字符串首字符大写，之后的小写,如果isFirst为true则只对第一个进行大写转换。

2）调用：DseUtils.StringUtil. capitalize(str,isFirst);

3）参数：

类型:String 目标字符串

类型: Boolean 是否只转换第一个字符

4）返回结果：String 转换结果

5）示例：

###### 2.1.2.2.2.6 isContain

1）说明：是否包含某个字符串。

2）调用：DseUtils.StringUtil.isContain(str, destStr);

3）参数：

类型:String 目标字符串

类型: String 包含字符串

4）返回结果：Boolean true为包含，false不包含

5）示例：

###### 2.1.2.2.2.7 removeLast

1）说明：移除字符串最后一个字符。

2）调用：DseUtils.StringUtil.removeLast(str);

3）参数：

类型:String 目标字符串

4）返回结果：String 返回移除后的字符串，如果传入为非字符串，则返回空字符''。

5）示例：

###### 2.1.2.2.2.8 toLowerCase

1）说明：将字符串转换成小写，如果为空字符串则返回空字符。

2）调用：DseUtils.StringUtil.toLowerCase(str);

3）参数：

类型:String 目标字符串

4）返回结果：String 转换结果，如果传入为非字符串，则返回空字符''。

5）示例：

###### 2.1.2.2.2.9 propToLowerCase

1）说明：将对象的属性名称转换成小写,如果传入为非对象，则返回空对象。

2）调用：DseUtils.StringUtil.propToLowerCase(target);

3）参数：

类型:String 目标字符串

4）返回结果：String 转换结果，如果传入为非字符串，则返回空字符''。

5）示例：

###### 2.1.2.2.2.10 nullToSP

1）说明：将空对象替换成指定的字符。

2）调用：DseUtils.StringUtil.nullToSP(str,sp);

3）参数：

类型:String 目标字符串

类型: String 转后的字符(默认'--')

4）返回结果：String 替换后的字符串。

5）示例：

##### 2.1.2.2.3 DateUtil

时间日期处理工具类。

###### 2.1.2.2.3.1 oneDayMs

1）说明：一天时间戳。

2）调用：DseUtils.DateUtil.oneDayMs();

3）参数：空

4）返回结果：Number 一天的时间戳值。

5）示例：

###### 2.1.2.2.3.2 now

1）说明：获取当前客户端时间戳。

2）调用：DseUtils.DateUtil.now();

3）参数：空

4）返回结果：Number 当前客户端时间戳。

5）示例：

###### 2.1.2.2.3.3 getLastDayOfYearMonth

1）说明：获取某年某月的最后一天时间戳。

2）调用：DseUtils.DateUtil.getLastDayOfYearMonth(year,month);

3）参数：

类型:Number 年

类型:Number 月

4）返回结果：Number 那一天的时间戳,如果传入为非时间则返回-1。

5）示例：

###### 2.1.2.2.3.4 getNearDayTimeStamp

1）说明：获取离当前时间近n天的时间戳。

2）调用：DseUtils.DateUtil.getNearDayTimeStamp(n);

3）参数：

类型:Number 整数，代表n天,如果为负数，表示当前天之后几天

4）返回结果：Number n天的时间戳,如果传入的不是数值，则返回-1。

5）示例：

###### 2.1.2.2.3.5 timeStampFormatDateObj

1）说明：将时间戳转换为时间对象。

2）调用：DseUtils.DateUtil.timeStampFormatDateObj(timeStamp);

3）参数：

类型:Number 时间戳

4）返回结果：Object 时间对象

5）示例：

###### 2.1.2.2.3.6 timeStampFormatDate

1）说明：将时间戳转换成yyyy-MM-dd。

2）调用：DseUtils.DateUtil.timeStampFormatDate(timeStamp);

3）参数：

类型:Number 时间戳

4）返回结果：String 返回转换后格式,如果传入为非数值类型,则返回''字符串

5）示例：

###### 2.1.2.2.3.7 timeStampFormatDateTime

1）说明：将时间戳转换成yyyy-MM-dd hh:mm:ss。

2）调用：DseUtils.DateUtil.timeStampFormatDateTime(timeStamp);

3）参数：

类型:Number 时间戳

4）返回结果：String 返回转换后格式,如果传入为非数值类型,则返回''字符串

5）示例：

##### 2.1.2.2.4 ArrayUtil

数组处理工具，如并集、交集等

###### 2.1.2.2.4.1 unique

1）说明：去除重复数组。

2）调用：DseUtils.DateUtil.unique(array);

3）参数：

类型: Array 目标数组

4）返回结果：Array 返回去除重复数据的数组,如果传入的为非数组则返回一个空数组

5）示例：

###### 2.1.2.2.4.2 union

1）说明：返回多个数组的并集，并去除重复的。

2）调用：DseUtils.DateUtil.unique(array,array2,...);

3）参数：

类型: Array 变形数组

4）返回结果：Array 并集数组

5）示例：

###### 2.1.2.2.4.3 intersection

1）说明：返回多个数组的交集。

2）调用：DseUtils.DateUtil.intersection(array,array2,...);

3）参数：

类型: Array 变形数组

4）返回结果：Array 交集数组，如果存入非数组返回空数组[]

5）示例：

###### 2.1.2.2.4.4 removeFalse

1）说明：移除数组中为false的值，即移除 false, null, 0, "", undefined 和 NaN值。

2）调用：DseUtils.DateUtil.removeFalse(array);

3）参数：

类型: Array 目标数组

4）返回结果：Array 结果数组,如果传入的为非数组则返回一个空数组[]

5）示例：

###### 2.1.2.2.4.5 without

1）说明：从数组中，移除指定值。

2）调用：DseUtils.DateUtil.without(array);

3）参数：

类型: Array 目标数组

4）返回结果：Array 结果数组，如果传入为非数组则返回一个空数组[]

5）示例：

###### 2.1.2.2.4.6 insertOfIndex

1）说明：将元素插入到指定位置，如果没传index默认插入到首位。

2）调用：DseUtils.DateUtil.without(array,obj,index);

3）参数：

类型: Array 目标数组  
 类型: Object 元素

类型: int 插入位置，0为首位

4）返回结果：this ArrayUtil

5）示例：

##### 2.1.2.2.5 ScrollUtil

滚动条生成工具

###### 2.1.2.2.5.1 defaultScroll

1）说明：生成默认的滚动条样式。

2）调用：DseUtils.ScrollUtil.defaultScroll();

3）参数：空

4）返回结果：空

5）示例：

##### 2.1.2.2.6 FunUtil

函数处理工具

###### 2.1.2.2.6.1 recursion

1）说明：递归数据型结构数据。

2）调用：DseUtils.FunUtil.recursion(root,fun,config);

3）参数：

类型: Array||Object 被递归对象

类型:Function 业务处理方法

类型: Object 配置信息

{

childrenField:String,子节点的属性名称如果没有默认为children

}

4）返回结果：空

5）示例：

##### 2.1.2.2.7 PathUtil

处理有关路径信息的工具类

###### 2.1.2.2.6.1 resolvePath

1）说明：解析地址上面的参数。

2）调用：DseUtils.FunUtil.recursion(root,fun,config);

3）参数：

类型: Object 配置信息

{

childrenField:String,子节点的属性名称如果没有默认为children

}

4）返回结果：Object 返回对象，如果传入的为非字符串则返回空对象

5）示例：

#### 2.1.2.3 DseUnderWin

点击出现在元素下方出现弹出控件,包含创建DseDistWin等对象

##### 2.1.2.3.1配置项

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名称 | 属性值类型:默认值 | 是否必填 | 描述 |
| target | Object | Y | 目标Dom，为Jquery对象，默认将弹出框显示到该元素下方 |
| \_data | Array | Y | 渲染数据 |
| displayField | String:text | N | 节点显示字段的属性名称，如果没有默认用text字段进行显示 |
| idField | String:id | N | id属性名称，如果没有则拿id |
| pidField | String:pid | N | pid属性名称，如果没有则拿pid |
| appendTo | Object | N | Jquery对象，将生成的弹出框dom放于该元素下，如果没有值则默认生成到body下，如果填写则会自动给target添加点击显示弹窗事件 |
| groupInfo | Object | N | 分组信息{  groups:Array,按照数组中的顺序进行分组，数组的值则为分组显示的值  } |
| allInfo | Object | N | 层级全部显示信息  {  text:String,全部显示的信息,如：text:'全部'  } |
| position | Object | N | 弹框的位置配置项{  offsetLeft:int，左偏移量  offsetTop:int，上偏移量  width:int，宽度  height:int，高度  } |
| isClickReset | Boolean | N | 是否隐藏弹框后重置面板初始状态，默认false |
| isMultiSelect | Boolean | N | 是否节点可以多选，默认false |

##### 2.1.2.3.2 事件配置项

|  |  |  |
| --- | --- | --- |
| 事件名称 | 参数 | 描述 |
| onClick | node,nodes,allNodes | 单击节点事件，node为被点击节点，nodes为所有的节点，allNodes是当有allInfo配置项时，点击allInfo配置项节点会返回该层的所有节点 |
| onDbClick | node,nodes,allNodes | 双击节点事件，node为被点击节点，nodes为所有的节点，allNodes是当有allInfo配置项时，点击allInfo配置项节点会返回该层的所有节点 |

##### 2.1.2.3.3 属性（包含配置项都是其属性）

|  |  |  |
| --- | --- | --- |
| 属性名称 | 属性值类型:默认值 | 描述 |
| targetInfo | Object | 目标target的样式信息对象  {  left:int左边距，  top:int上边距，  width:int宽度,  height:int高度  zIndex:z-index的值，可以通过修改改值来改变面板的值  } |
| zIndex | int:1000 | 面板的z-index值 |
| elementAttr | Object | 生成dom的属性信息，用于标示哪个dom  {  panel: 'data-flag-panel'，面板的属性标示  } |
| classInfo | Object | 样式类信息 |

##### 2.1.2.3.4 方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名称 | 参数 | 返回值 | 描述 |
| createPanel |  | panelId | String，面板的id |
| resetPanel | panelId | this | 重置面板到初始状态 |
| getPanelOfId | Id | panel | 根据id返回panel的dom对象 |
| getDistWinOfId | Id | DseDistWin | 根据id返回DseDistWin对象 |
| findElement | panel,element | element | 查找panel下的元素 |
| createDistWin |  | this | 创建上下层级下拉框,此时可以通过this.distWin获取DseDistWin对象 |

#### 2.1.2.4 DseDistWin

上下层级弹出框，上下级基于树形结构的数据，配置信息基于DseUnderWin对象

##### 2.1.2.4.1 事件

|  |  |  |
| --- | --- | --- |
| 事件名称 | 参数 | 描述 |
| onClick | node,nodes,allNodes | 单击节点事件，node为被点击节点，nodes为所有的节点，allNodes是当有allInfo配置项时，点击allInfo配置项节点会返回该层的所有节点 |
| onDbClick | node,nodes,allNodes | 双击节点事件，node为被点击节点，nodes为所有的节点，allNodes是当有allInfo配置项时，点击allInfo配置项节点会返回该层的所有节点 |

##### 2.1.2.4.2 属性（包含配置项都是其属性）

|  |  |  |
| --- | --- | --- |
| 属性名称 | 属性值类型:默认值 | 描述 |
| \_underWin | Object | DseUnderWin对象 |
| sourceData | Object | 元数据 |
| panel | Object | panel对象 |
| elementAttr | Object | 生成dom的属性信息，用于标示哪个dom |
| nodeMap | Object | nodeId对应对象 |
| groupNodes | Array | 分组节点对象 |
| continer | Element | 容器dom对象 |
| header | Element | 头部dom对象 |
| nodeContent | Element | 节点容器dom对象 |

##### 2.1.2.4.3 方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名称 | 参数 | 返回值 | 描述 |
| bulid |  | this | 创建弹框 |
| buildNodeContent | nodes | this | 需要创建的节点数组，将节点创建到容器上显示 |
| buildEvent |  | this | 给所有节点添加事件 |
| buildSelectedNode |  | this | 根据节点的selected属性进行初始化节点显示 |
| buildSelectText |  | this | 构建节点选中展示到头部 |
| clearAllStat |  | this | 清楚所有节点的状态信息 |
| clearChildHtml | parentNode | this | 清除下级节点的dom |
| clearAllChildStat | parentNode | this | 清除所有下级节点的状态，包含兄弟节点的下级 |
| setBrothersStat | targetNode | this | 设置兄弟节点的状态，根据是否配置了可以同层多选 |
| setNodeStat | node,isSelected | this | 设置节点的状态 |

##### 2.1.2.4.4 例子

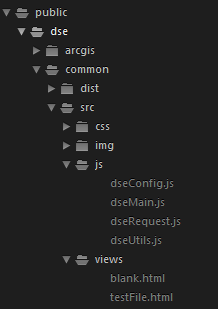
具体例子可以参照baseweb下的webapp/app/baseweb/DseUnderWin/的示例，效果图如下：



## 2.2 开发说明

采用seajs去模块化开发，再通过seajs去对外提供服务。

### 2.2.1 结构说明



src目录存放源码，dist是打包后代码提供给外部使用的也是打包后的代码，view放的是UI示例代码。

模块存放在js目录下，新的模块开发完成后，如果需要对外提供服务，则配置在dseMain.js文件里。

### 2.2.2 源码模块编写说明

文件名称以dse开头，以标示为common里的，代码编写命名空间也以Dse开头。

#### 2.2.2.1 js目录

##### 2.2.2.1.1 dseMain.js

需要提供给外部使用的模块配置在这个文件里；

示例代码：公开DseRequest模块对外提供服务：

module.exports.DseCommon = {

DseRequest : require('dseRequest').DseRequest,/\*请求服务\*/

}

##### 2.2.2.1.1 dseConfig.js

常量等配置信息，如与后台交互返回的状态编码信息；采用"模块名称:{具体常量:具体常量值}"编写；

const DseConstant = {

/\*请求配置项\*/

DseRequest: {

SUCCESS\_CODE : 1,/\*成功状态码\*/

ERROR\_CODE : 0,/\*失败状态码\*/

}

}

##### 2.2.2.1.1 dseUtils.js

工具类实现模块，具体命名为："处理对象Util:{具体处理方法:function(){}}"；

示例代码如下所示:

const DseUtils = {

/\*对象工具类\*/

ObjectUtil : {

/\*创建对象只读属性\*/

createReadOnlyProp : function(obj,propName,value){

Object.defineProperty(obj,propName,{

value : value,

writable : false,

enumerable: false,

configurable: false

});

}

}

}

#### 2.1.2.2 css目录

1）dseMain.css

此文件目前分三块：

1. 重置样式。
2. 通用按钮样式。页面中包含的查询（dse-btn-search）、重置（dse-btn-reset）、删除（dse-btn-delete）等按钮，开发中尽量用此通用样式。其中dse-btn-common可以和其他背景色搭配组合，所以在实际开发中“导入”的通用按钮样式是橘黄色，如果与整体页面颜色不符，整体页面需要搭配蓝色按钮，可以通过（dse-btn-common dse-bg-blue）组合实现。（如下图）





（3）增加的其他公用样式，如table样式等。