统一支撑软件服务平台

前端开发规范

版本号：STYB-0205-202001

2020年1月

**目 录**

[1项目构建 1](#_Toc33291145)

[1.1源码获取 1](#_Toc33291146)

[1.2配置文件更改 1](#_Toc33291147)

[1.3运行基础框架 2](#_Toc33291148)

[2项目开发 2](#_Toc33291149)

[2.1应用的目录结构 2](#_Toc33291150)

[2.2Html开发 3](#_Toc33291151)

[2.3脚本的引用 3](#_Toc33291152)

[2.4JavaScript开发 3](#_Toc33291153)

[2.5前端CSS引用 4](#_Toc33291154)

[2.6前端CSS自定义开发 4](#_Toc33291155)

[3项目发布与部署 6](#_Toc33291156)

[3.1代码发布 6](#_Toc33291157)

[3.2配置文件修改 7](#_Toc33291158)

[3.3站点部署 7](#_Toc33291159)

[4命名规范 8](#_Toc33291160)

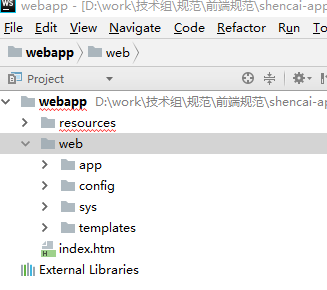
[4.1应用的命名 8](#_Toc33291161)

[4.2Html，JavaScript，CSS文件的命名 8](#_Toc33291162)

## 1项目构建

### 1.1源码获取

从技术组获取最新的公司框架webapp，目录结构如下：



各个文件夹说明：

|--------webapp，项目开发根目录

|--------resources，所有应用公用的前端资源库，包括js, css, images, fonts等等

|--------web，web项目所在的目录

|----------------app，应用所在的目录

|----------------config，公共的配置文件所在的目录

|----------------sys，框架涉及页面

|----------------templates，存放着根据前端规范构建的一些表单，列表，首页，登录，注册等，html示例代码，JavaScript示例，CSS示例代码，可以作为构建你的应用的模板。

### 1.2配置文件更改

* 更改webapp根目录下面的index.htm页面，将里面页面的路径指向你的app所在的页面
* 更改配置文件config->app.config.js里面的serviceSiteRootUrl(后台服务的地址)， uploadServer（上传服务器的地址） ， isSSO（是否启用单点登录）等自定义信息

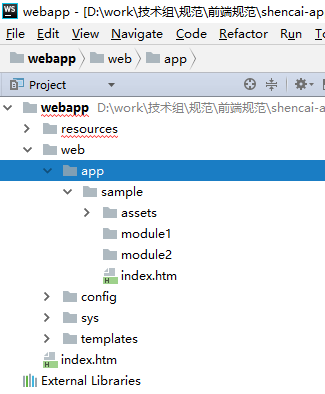
### 1.3运行基础框架

暂时我们推荐webstorm或者Visual studio来进行前端的开发，webstorm打开webapp目录， 打开根目录下面的index.htm页面，Ctrl+shift+F10运行项目

## 2项目开发

### 2.1应用的目录结构

项目开发应该在自己的项目文件夹下进行开发， 展开目录app, 创建一个应用，文件夹名称根据你的项目名称来定, 并由业务负责人审核（比如wry污染源）。可以参考sample应用的目录结构。



|--assets存放sample应用所有应用到的公用资源文件

|--module1，存放sample应用module1的html, js和自定义css文件

|--module2，存放sample应用module2的html, js和自定义css文件

### 2.2Html开发

参考templates目录下面的模板页面，copy到你的应用目录下面相应的文件夹下面，然后修改相应的html，CSS和JavaScript。

### 2.3脚本的引用



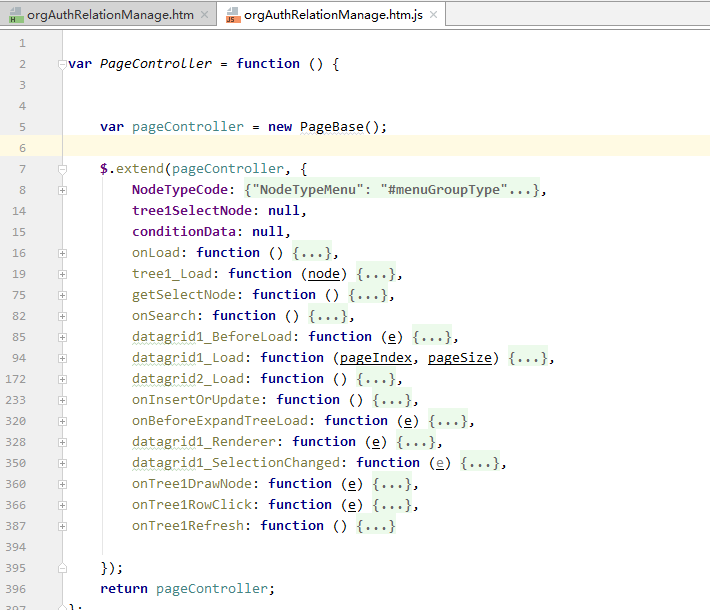
* 页面中应首先引用全局的boot.js（boot.js包含miniui, jquery等的引用）, boot.js内容如下：



* sys.js, main.htm.js为本页面服务的JavaScript，应放在和该html页面相同的文件夹
* 如果有自定义CSS，可以自行定义，不做规定

### 2.4JavaScript开发

参考sys->orgAuthRelationManage.htm.js 的写法，对于其他写法暂不推荐，格式如下：



注意：所有JavaScript代码应该包含在一个执行块里面，避免全局变量的污染，应该尽量避免注册新的全局变量和全局函数。

### 2.5前端CSS引用

全局CSS应该在boot.js里引入，自己页面的CSS按照常规引入方式。

### 2.6前端CSS自定义开发

* **如何选择id 和class的问题**

避免使用#id来定义样式（除最外层），id定位网页位置倾向于结构性，功能性选择器也尽量使用id。 功能性class 与样式性class区分，功能性 .f\_开头，例: <input type=”button” value=”提 交” class=”btn” />

如果jQuery绑定提交事件：

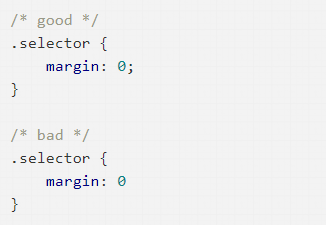
1. 优先使用id如#f\_submit；

2. 如果页面多个类似按纽同时绑定.f\_submit。

* **多selector 时，每个选择器声明必须独占一行**



* **属性定义后必须以分号结尾**



* **尽量不使用 !important 声明**
* **避免使用标签选择器和层级选择器**

原因: 层级选择器违反样式结构分离原则, 不利于html后期修改

如: 不建议

.mycontainer>div

div.each

* **块级功能的独有样式**

建议按html结构写全，让.class一定有id环境可读性

#mycontainer

#mycontainer .leftwrap

#mycontainer .rightwrap

#mycontainer .leftwrap .title

#mycontainer .leftwrap .body

* **0 与 0px**

省略“0”值后面的单位，不要在0值后面使用单位，除非有值。

正：padding-bottom: 0;

反：padding-bottom: 0px;

* **引号问题**

样式内引号全采用双引号，URI值（url()）不要使用引号。

* **CSS属性的声明顺序**

下面是一个选择器内书写CSS属性的顺序，这是为了保证更好的可读性，作为最佳实践，建议按照下表的顺序。

结构性属性：

display

position, left, top, right etc.

overflow, float, clear etc.

margin, padding

表现性属性：

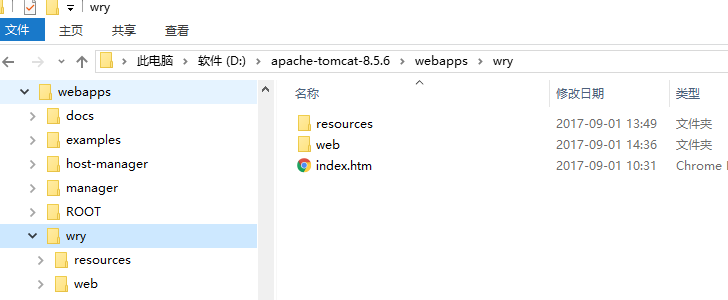
background, border etc.

font, text

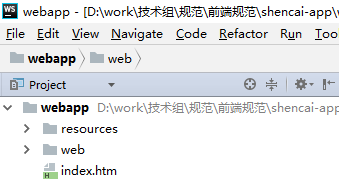
## 3项目发布与部署

### 3.1代码发布

* 在服务器tomcat 目录下的webapps目录下创建应用文件夹，如wry

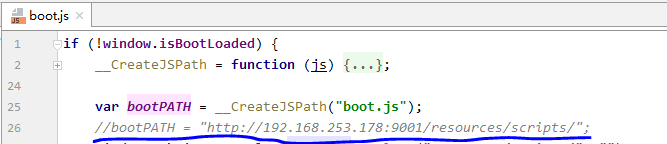


* 将开发目录webapp中的所有文件（如下图）copy到服务器上你的应用目录



### 3.2配置文件修改

* 根据业务实际服务器地址，修改app.config.js文件
* 如引用公司外部发布的fw框架，请修改boot.js里面下面一行



### 3.3站点部署

站点部署应该根据项目文档及客户需求，比如部署相应的消息队列，RPC服务器，负载均衡，数据库服务等，此处不涉及。

## 4命名规范

### 4.1应用的命名

要根据你的实际项目名称来定，最好是英文的，并且在项目组之间有较好的辨识度，具体参考公司的代码规范。

### 4.2Html，JavaScript，CSS文件的命名

文件命名应该根据页面的功能来定义，如转移危废功能页面, transferWaste.html，transferWaste.htm.js，transferWaste.css，避免中英文混杂，缩写等，具体参考公司的代码规范。