**绽放工场前端复用文档**

# 功能组件

解释说明在对应的文件里面

## 微信公众号授权

文件：wxAuth.js

使用：export default auth，函数参数请查看文件代码说明，该方法目的是处理微信公众号页面授权功能，基于前后端分离。

## 微信公众号签名

文件：wxSignature.js

使用：export default wx，函数参数请查看文件代码说明，该方法目的是处理微信公众号页面签名功能，基于前后端分离。

## 小程序授权

文件：wxSignature.js

使用：export default authorize，函数参数请查看文件代码说明，该方法目的是处理微信小程序授权功能，基于前后端分离。

# 业务组件

## 左图标右文字

**示例**：menu.saveimg.savepath20190119175958

<div class="ww-graphic">

<figure class="ww-icon">

<img src="" width="100%" height="100%">

</figure>

<p class="ww-text"></p>

</div>

.ww-graphic

display flex

flex-flow row nowrap

align-items center

.ww-icon

font-size 0

width 20px

height 20px

.ww-text

margin-left 10px

font-size 32px

color #333333

## 上图标下文字

**示例**：menu.saveimg.savepath20190119180037

<div class="ww-graphic">

<figure class="ww-icon">

<img src="" width="100%" height="100%">

</figure>

<p class="ww-text"></p>

</div>

.ww-graphic

display flex

flex-flow column nowrap

align-items center

.ww-icon

font-size 0

width 20px

height 20px

.ww-text

margin-top 10px

font-size 32px

color #333333

## 左图文右文字

示例：

<template>

<div class="content\_box" @click="port" @touchstart="portTouchStart" @touchmove="portTouchMove" @touchend="portTouchEnd" :class="{textActive: textActive}">

<div class="left\_img" :class="size">

<img :src="list.imageUrl?list.imageUrl:require('../../../static/imgs/ees.png')" alt="" width="100%">

</div>

<div class="right\_text">

<div class="right\_text\_title" :class="size + 't'">

{{list.title}}

</div>

<div class="right\_text\_cont" :class="size + 'c'">

<p>{{list.context}}</p>

</div>

</div>

</div>

</template>

<script>

export default {

name: 'imageText',

props:{

list: Object,

size: String

},

data () {

return {

textActive: false,

};

},

methods: {

port () {

if (this.list.funtionUrl === '/') {

this.$emit('showClose')

} else {

this.$router.push(this.list.funtionUrl)

}

},

portTouchStart () {

this.textActive = true

},

portTouchMove () {

if (this.textActive) {

this.textActive = false

}

},

portTouchEnd () {

this.textActive = false

},

}

}

</script>

<style lang='stylus' scoped>

.content\_box

padding 34px 0px

border-bottom 1px solid #f3f4f6

touch-action none

.left\_img

display inline-block

position relative

// width 230px

// height 174px

.big

width 230px

height 174px

.right\_text

vertical-align top

display inline-block

width 460px

margin-left 20px

.right\_text\_title

// font-size 32px

// height 52px

// line-height 52px

.bigt

font-size 32px

height 52px

line-height 52px

.right\_text\_cont

margin-top 60px

// font-size 24px

// line-height 36px

color #999999

overflow hidden

text-overflow ellipsis

display -webkit-box

-webkit-line-clamp 2

-webkit-box-orient vertical

.bigc

font-size 24px

line-height 36px

.textActive

background-color #f1f1f1

</style>

## 星级评价

示例：

<template>

<div class="evaStar">

<span

v-for="(starsList,index) in starsList"

:key="index"

:class="[starsList,sizes]"

class="star-item"

@click="stars(index)"

></span>

</div>

</template>

<script>

export default {

props: {

size: String

},

data() {

return {

score: 0,

};

},

computed: {

sizes(){

return this.size;

},

starsList() {

let result = [];

let score = Math.floor(this.score \* 2) / 2;

let integer = Math.floor(score);

for (let i = 0; i < integer; i++) {

result.push("on");

}

while (result.length < 5) {

result.push("off");

}

return result;

}

},

methods: {

stars: function(index) {

this.score = index + 1;

}

}

};

</script>

<style>

.evaStar {

padding-top: 40px;

}

.star-item {

display: inline-block;

background-repeat: no-repeat;

margin-left: 15px;

background-size: 100%;

}

.small{

width: 24px;

height: 24px;

}

.big{

width: 48px;

height: 48px;

}

.star-item.on {

background-image: url("../../../static/imgs/col\_active.png");

}

.star-item.off {

background-image: url("../../../static/imgs/col.png");

}

</style>

使用组件：<stars-score size="big"></stars-score>

# CSS组件

该CSS组件依赖于stylus

## 一行省略

one-line()

overflow hidden

text-overflow hidden

white-space nowrap

## 多行省略

many-line(line)

overflow hidden

display -webkit-box

-webkit-line-clamp line

-webkit-box-orient vertical

## 占满父元素的绝对定位

absolute-full()

position absolute

top 0

right 0

bottom 0

left 0

## 不定宽高的绝对定位居中

absolute-center()

position absolute

top 50%

left 50%

transform translate(-50%, -50%)

text-align center

## PC端到移动端的媒体查询

@media only screen and (max-width: 1440px) {

}

@media only screen and (max-width: 1068px) {

}

@media only screen and (max-width: 736px) {

}

## 三角形

.triangle

width 0

height 0

border-top 20px solid #333

border-left 20px solid transparent

border-right 20px solid transparent

## 内容左右垂直居中

centered()

display flex

justify-content center

align-items center

# 前端移动端集成框架（Vue）使用说明

**集成框架：**几乎每个项目都会重复使用的功能代码都会集成起来，配合基础Vue框架，可以快速进入开发阶段，从而避免重复搭框架消耗时间，而且一个完善的集成框架能更利于项目的开发。

## 功能插件选型

1. 表单验证：vee-validate
2. 动画：gsap
3. 触摸动效：vue-touch-ripple
4. 组件过渡动效：vue2-animate
5. 图表制作：v-charts
6. JS工具库：lodash
7. 页面滚动：better-scroll
8. HTTP请求：axios
9. 本地存储：localforage

## 移动端适配

**代码使用单位：**px

**转换宽度标准：**375px

**说明：**

为解决移动端适配问题，主要解决的是移动端样式（css)单位不适配，目前主流解决方案有3种，分别是rem、百分比、视窗单位，在此选择的是最主流移动端适配方案，视窗单位，不单单目前表现出色，而且未来数年也不会改变它的地位。

由于不同厂家的浏览器会产生差异，chrome、safari、QQ浏览器等等，有时候我们需要编写不同的css前缀来进行兼容，例如你需要禁止选中文本，你需要编写以下css:

-webkit-user-select:none;

-moz-user-select:none;

-khtml-user-select:none;

user-select:none

为了兼容，本来只需要一条代码，却需要写四条，这令人烦躁不是吗？优秀的是，在此已经帮你解决了这个问题，你只需要写下不带前缀的css样式

user-select:none

，其他的，框架会帮你处理好的。

对于其他开发者来说，以上使用起来是无感的，即是你无需为此做出什么改变或者重新学习某些知识，它降低了开发难度，简化过程，你不必知道代码内部发生了什么，你只需要在项目里，按照UI设计图进行编写你熟悉的像素（px）单位，最终打包编译后的单位是vw，即是视窗百分比，它会根据不同移动端尺寸给页面元素转换成对应的百分比宽高，就是如此轻松。

**注意：**

为了减少移动端兼容性问题，在页面布局上，选择flex、grid都是不错的选择，请尽量避免给元素直接固定宽度，固定宽度会使它失去适应性，除非你就需要它这么宽。

ios上不要直接对<img>标签定义宽高，vw会失效，你可以给img标签套上一层figure或者div，然后控制它。

以上所做的事情并不足以解决所有兼容问题，它能保证的是基础的兼容，保证在不同尺寸的设备上元素表现一致。事实上，兼容问题总是千奇百怪，没人能保证有一套方案能完全避免兼容问题，如果你发现了一个新的兼容问题，请描述在目录5里面，以帮助后来的开发人员能快速解决同样的问题。

## 页面过渡的左右滑动效果

**使用：**在组件对应的路由里面设置meta: { level: 1 }，level代表路由的层级，层级大会右滑进入，反之左滑进入。

**说明：**不必强行使用它，它只是个可选项，你可以使用它作为页面的过渡效果，使app更像原生app的感觉，但它不是完美的，在低端移动设备上，它表现得并不出色（通常是千元机内），另一个选择是使用骨架屏过渡，它得适应性更优秀，体验也不错（我还没把它写出来）。

## 第三方UI框架（Vant）

**使用：**

**默认按需引入**：

<script>

import { Button } from 'vant'

export default {

name: 'xxx',

components: {

[Button.name]: Button

}

}

</script>

**全局引入：**这不是一个优秀的选择，它会降低程序的性能，尽管它很方便，不推荐，如果 你非得使用，请参照官方文档引入项目。

**修改主题：**在vue.config.js文件内修改配置，你可以参照官方文档提供的信息操作。

**说明：**我尝试了数款移动端UI框架，最终选择了它（Vant），因为它能更搭配目前的架构，而且功能更为丰富，善于利用它绝对能大幅度提升你的开发效率，当然你也可以对着官方文档进行复制粘贴，这是最简单的使用方式，但无论如何，你都需要知道它到底提供了什么功能。

## HTTP请求工具

**说明：**默认集成axios，内部设置了请求拦截器和响应拦截器，如果你希望每次请求之后/之前都做些什么，/src/plugins/axios.js里面修改，后台需要返回固定的格式。

## 全局URL设置

**说明：**在/src/tool/baseUrl.js文件内修改全局请求默认url前缀，凡是url类型的均在此管理，方便修改。

## 页面缓存keep-alive设置

**说明：**默认全组件均为可缓存，最大缓存数为3，如果某组件不希望进行缓存，请在单组件内的name后添加 -nka，例子： name: ‘home-nka’，缓存页面能提升用户体验，不必每进入一个页面就重新渲染和请求数据，但缓存太多会导致低端手机卡顿，所以我设最大值为3，如果需要修改，它在app.vue里面，你能很轻易得找到它。

# 移动端兼容问题记录

## 安卓手机的圆角失效

background-clip: padding-box;

## 横屏时字体加粗不一致

text-size-adjust: 100%;

## 长时间按住页面出现闪退

-webkit-touch-callout: none;

## 安卓和IOS在微信公众号背景色不一致

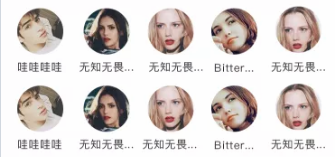
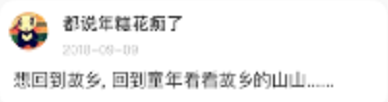
统一设定共同背景色background-color

# 开发计划

## UI组件开发

1. 信息卡片，开发时间：02/18 - 02/23，负责人：胡安生



1. 多标签选择器，开发时间：02/18 - 02/23，负责人：吴上海 
2. 评价星星， 开发时间：02/25 - 03/01，负责人：胡安生  
    
3. 分类栏(右侧更多)，开发时间：02/25 - 03/01，负责人：吴上海  
    
4. 多图文排列，开发时间：02/25 - 03/08，负责人：胡安生  
    
5. 导航条，开发时间：02/25 - 03/08，负责人：吴上海  
    
6. 小分类，开发时间：03/11 - 03/16，负责人：胡安生  
    a)  
    b)
7. 切换卡，开发时间：03/11 - 03/16，负责人：吴上海  
    
8. 带用户信息卡片，开发时间：03/18 - 03/22，负责人：胡安生  
    
9. 左右露出部分的轮播图，开发时间：03/18 - 03/22，负责人：吴上海   
    

11. 进程零件与标签，开发时间：03/25 - 03/30，负责人：吴上海  
 

## 框架迭代计划

1. 预加载骨架屏组件  
    开发时间：02/18 - 04/01  
    负责人：吴上海