

移动WEB开发之rem适配布局

掌握 rem 基础知识

课程目标

TARGET


- 能够使用 rem 单位
- 能够使用媒体查询的基本语法
- 能够使用 Less 的基本语法
- 能够使用 Less 中的嵌套
- 能够使用 Less 中的运算
- 能够使用 2 种 rem 适配方案
- 能够独立完成苏宁移动端首页

思考

1. 页面布局文字能否随着屏幕大小变化而变化?
2. 流式布局和 flex 布局主要针对于宽度布局, 那高度如何设置?
3. 怎么样让屏幕发生变化的时候元素高度和宽度等比例缩放?

rem 单位

- rem (root em) 是一个相对单位，类似于 em，em 是父元素字体大小。
- 不同的是 rem 的基准是相对于 <html> 元素的字体大小。
- 比如，根元素 (html) 设置 font-size=12px; 非根元素设置 width:2rem; 转换成 px 表示就是 24px。
- rem 的优势：父元素文字大小可能不一致，但是整个页面只有一个 <html>，可以很好的来控制整个页面的元素大小比例。



```
/* 根html 为 12px */  
html {  
    font-size: 12px;  
}  
/* 此时 div 的字体大小就是 24px */  
div {  
    font-size: 2rem;  
}
```

媒体查询

掌握媒体查询制作方法


媒体查询

媒体查询 (Media Query) 是CSS3新语法。

- 使用 @media 查询，可以针对不同的媒体类型定义不同的样式
- @media 可以针对不同的屏幕尺寸设置不同的样式
- 当你重置浏览器大小的过程中，页面也会根据浏览器的宽度和高度重新渲染页面
- 目前针对很多苹果手机、Android手机，平板等设备都用得到多媒体查询

语法规范

- 用 @media 开头 注意@符号
- mediatype 媒体类型
- 关键字 and not only
- media feature 媒体特性 必须有小括号包含



```
@media mediatype and|not|only (media feature) {  
  CSS-Code;  
}
```


mediatype 媒体类型

- 将不同的终端设备划分成不同的类型，称为媒体类型

值	解释说明
all	用于所有设备
print	用于打印机和打印预览
screen	用于电脑屏幕，平板电脑，智能手机等

关键字

关键字将媒体类型或多个媒体特性连接到一起做为媒体查询的条件。

- and: 可以将多个媒体特性连接到一起, 相当于“且”的意思。
- not: 排除某个媒体类型, 相当于“非”的意思, 可以省略。
- or: 可以测试多个媒体查询的条件, 只要有一个条件成立, 就执行, “或”的意思。
- only: 指定某个特定的媒体类型, 可以省略。

媒体特性

每种媒体类型都具体各自不同的特性，根据不同媒体类型的媒体特性设置不同的展示风格。

我们暂且了解三个。

注意他们要加小括号进行包含。

值	解释说明
width	定义输出设备中页面可见区域的宽度
min-width	定义输出设备中页面最小可见区域宽度
max-width	定义输出设备中页面最大可见区域宽度

案例：根据页面宽度改变背景变色

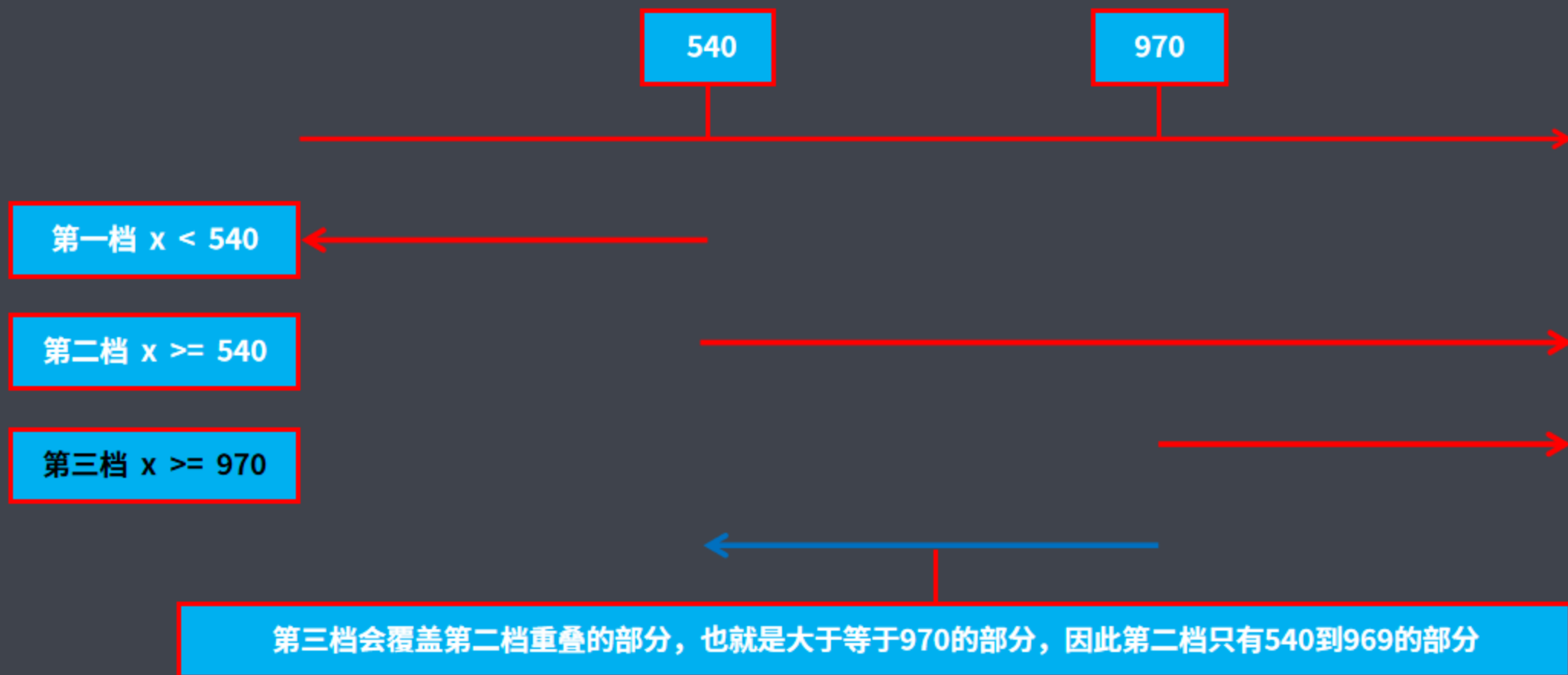


案例：实现思路

- ① 按照从大到小的或者从小到大的思路
- ② 注意我们有最大值 max-width 和最小值 min-width 都是包含等于的
- ③ 当屏幕小于 540 像素，背景颜色变为蓝色 ($x \leq 539$)
- ④ 当屏幕大于等于 540 像素 并且小于等于 969 像素的时候背景颜色为绿色 ($540 \leq x \leq 969$)
- ⑤ 当屏幕大于等于 970 像素的时候，背景颜色为红色 ($x \geq 970$)

注意：为了防止混乱，媒体查询我们要按照从小到大或者从大到小的顺序来写,但是**我们最喜欢的还是从小到大来写，这样代码更简洁**

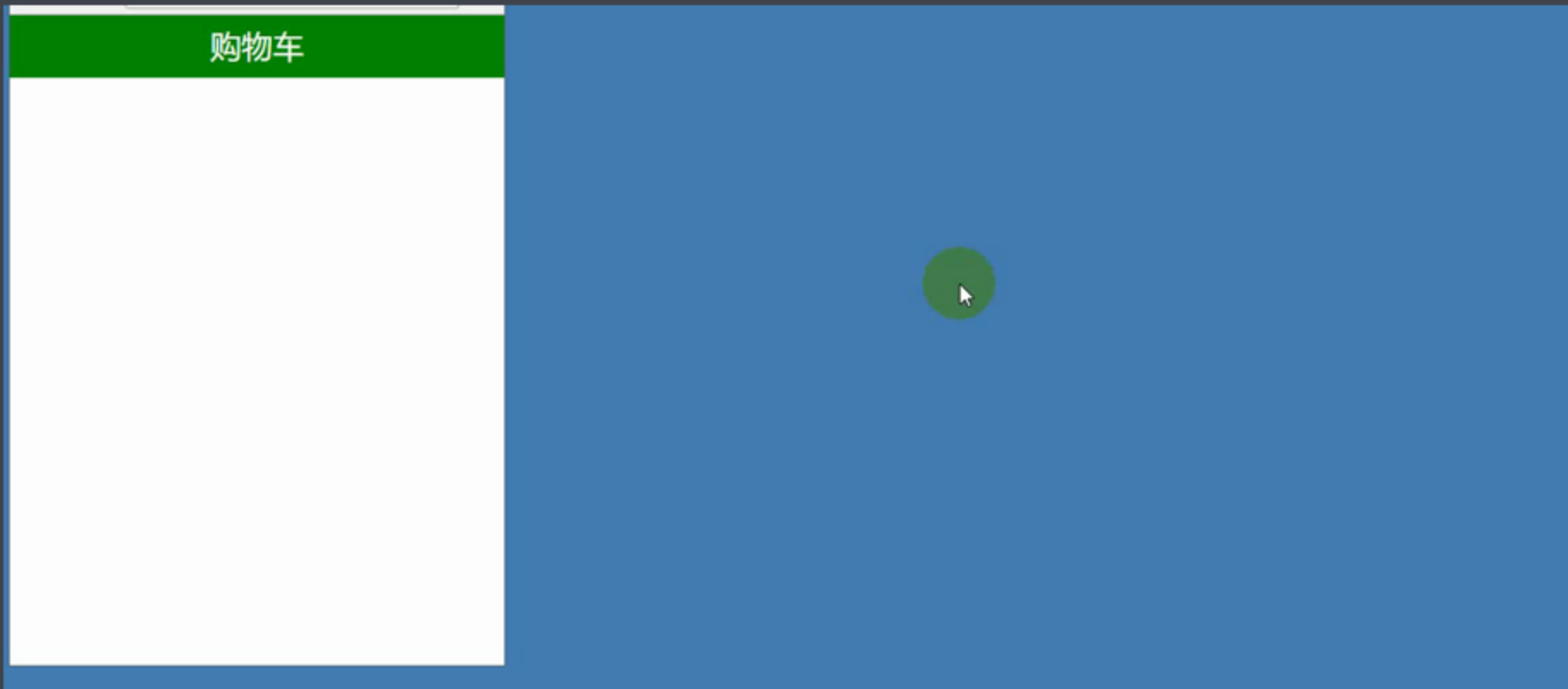
案例：媒体查询从小到大优势代码分析



媒体查询+rem 实现元素动态大小变化

- rem 单位是跟着 <html> 来走的，有了 rem 页面元素可以设置不同大小尺寸
- 媒体查询可以根据不同设备宽度来修改样式
- 媒体查询 + rem 就可以实现不同设备宽度，实现页面元素大小的动态变化

案例：媒体查询+rem实现元素变化



引入资源（理解）

- 当样式比较繁多的时候，我们可以针对不同的媒体使用不同 stylesheets（样式表）。
- 原理，就是直接在 <link> 中判断设备的尺寸，然后引用不同的 css 文件。

1. 语法规则

```
<link rel="stylesheet" media="mediatype and|not|only (media feature)" href="mystylesheet.css">
```

2. 示例

```
<link rel="stylesheet" href="styleA.css" media="screen and (min-width: 400px)">
```

Less 基础

掌握 Less 的使用方法

维护 css 的弊端

CSS 是一门非程序式语言，没有变量、函数、SCOPE（作用域）等概念

- CSS 需要书写大量看似没有逻辑的代码，CSS 冗余度是比较高的。
- 不方便维护及扩展，不利于复用。
- CSS 没有很好的计算能力
- 非前端开发工程师来讲，往往会因为缺少 CSS 编写经验而很难写出组织良好且易于维护的 CSS 代码项目。

Less 介绍

- Less (Leaner Style Sheets 的缩写) 是一门 CSS 扩展语言, 也成为CSS预处理器。
- 做为 CSS 的一种形式的扩展, 它并没有减少 CSS 的功能, 而是在现有的 CSS 语法上, 为CSS加入程序式语言的
- 特性。
- 它在 CSS 的语法基础之上, 引入了变量, Mixin (混入), 运算以及函数等功能, 大大简化了 CSS 的编写, 并且
- 降低了 CSS 的维护成本, 就像它的名称所说的那样, Less 可以让我们用更少的代码做更多的事情。

Less中文网址: <http://lesscss.cn/>

常见的CSS预处理器: Sass、Less、Stylus

一句话: Less 是一门 CSS 预处理语言, 它扩展了CSS的动态特性。

Less 安装

- ① 安装 nodejs, 可选择最新的版本, 网址: <https://nodejs.org/en/download/>
- ② 检查是否安装成功, 使用 cmd 命令 (win10 是 window + r 打开 运行输入 cmd) --- 输入 " node -v " 查看版本即可
- ③ 基于 node.js 在线安装 Less, 使用 cmd 命令 " npm install -g less " 即可
- ④ 检查是否安装成功, 使用 cmd命令 " lessc -v " 查看版本即可



Less 使用

我们首先新建一个后缀名为 .less 的文件, 在这个 .less 文件里面书写 Less 语句。

- Less 变量
- Less 编译
- Less 嵌套
- Less 运算

Less 变量

- 变量是指没有固定的值，可以动态改变的。因为我们 CSS 中的一些颜色和数值等经常使用。
- @变量名:值;

1. 变量命名规范

- 必须有@为前缀
- 不能包含特殊字符
- 不能以数字开头
- 大小写敏感

@color: pink;

2. 变量使用规范

//直接使用

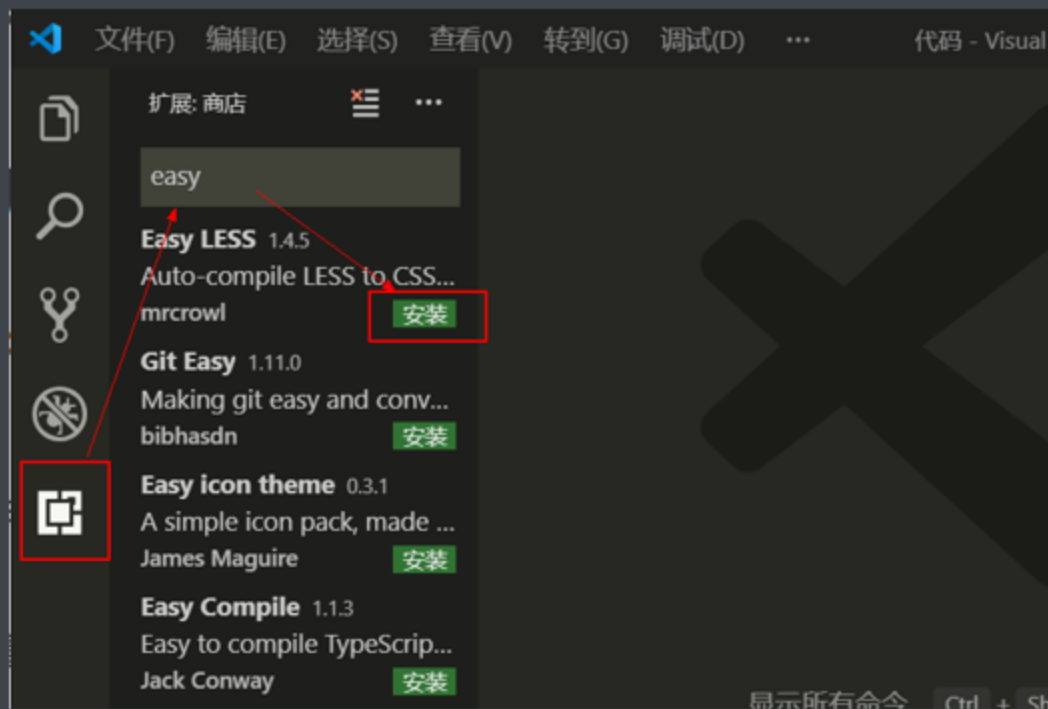
```
body {  
    color: @color;  
}  
  
a:hover {  
    color: @color;  
}
```

Less 编译

- 本质上, Less 包含一套自定义的语法及一个解析器, 用户根据这些语法定义自己的样式规则, 这些规则最终会通过解析器, 编译生成对应的 CSS 文件。
- 所以, 我们需要把我们的 .less 文件, 编译生成为 .css 文件, 这样我们的 html 页面才能使用。
- 推荐方法 (nodejs): 在当前文件夹, 使用 cmd 命令 “lessc style.less > style.css”

Vscode Less 插件 ★

- Easy LESS 插件用来把less文件编译为css文件
- 安装完毕插件，重新加载下 vscode。
- 只要保存一下Less文件，会自动生成CSS文件。



Less 嵌套

我们经常用的选择器

```
#header .logo {  
    width: 300px;  
}
```

Less 嵌套写法

```
#header {  
    .logo {  
        width: 300px;  
    }  
}
```

Less 嵌套

如果遇见（交集|伪类|伪元素选择器）

- 内层选择器的前面没有 & 符号，则它被解析为父选择器的后代；
- 如果有 & 符号，它就被解析为父元素自身或父元素的伪类。

我们经常用的选择器



```
a:hover{  
    color:red;  
}
```

Less 嵌套写法



```
a{  
    &:hover{  
        color:red;  
    }  
}
```

Less 运算 ★

- 任何数字、颜色或者变量都可以参与运算。Less 提供了加 (+)、减 (-)、乘 (*)、除 (/) 算术运算。

```
/*Less 里面写*/
@width: 10px + 5;
div {
    border: @width solid red;
}
/*生成的css*/
div {
    border: 15px solid red;
}
/*Less 甚至还可以这样 */
width: (@width + 5) * 2;
```

Less 运算 ★

注意:

- 乘号 (*) 和除号 (/) 的写法
- 对于两个不同的单位的值之间的运算, 运算结果的值取第一个值的单位
- 如果两个值之间只有一个值有单位, 则运算结果就取该单位
- 运算符中间左右有个空格隔开 `1px + 5rem`

rem 适配方案

了解 rem 适配方案

rem 适配方案 思考

- 我们适配的目标是什么？
- 怎么去达到这个目标的？
- 在实际的开发当中使用？

答案

- 让一些不能等比自适应的元素，达到当设备尺寸发生改变的时候，等比例适配当前设备。
- 使用媒体查询根据不同设备按比例设置html的字体大小，然后页面元素使用rem做尺寸单位，当html字体大小变化元素尺寸也会发生变化，从而达到等比缩放的适配。

rem 实际开发适配方案

- ① 按照设计稿与设备宽度的比例，动态计算并设置 html 根标签的 font-size 大小；（媒体查询）
- ② CSS 中，设计稿元素的宽、高、相对位置等取值，按照同等比例换算为 rem 为单位的值；



rem 适配方案技术使用（市场主流）

技术方案1

- less
- 媒体查询
- rem

技术方案2（推荐）

- flexible.js
- rem

总结：

1. 两种方案现在都存在。
2. 方案2 更简单，现阶段大家无需了解里面的js代码。

rem 实际开发适配方案1

rem + 媒体查询 + less 技术

1. 设计稿常见尺寸宽度

设备	常见宽度
iphone 4.5	640px
iphone 678	750px
Android	常见320px、480px、540px、600px、720px、768px、800px、1080px 目前市场主流设备尺寸按照 1080px 设计

一般情况下，我们以一套或两套效果图适应大部分的屏幕，放弃极端屏或对其优雅降级，牺牲一些效果
现在基本以750为准。

rem 实际开发适配方案1

动态设置 html 标签 font-size 大小

- ① 假设设计稿是750px
- ② 假设我们把整个屏幕划分为15等份（划分标准不一可以是20份也可以是10等份）
- ③ 每一份作为html字体大小，这里就是50px
- ④ 那么在320px设备的时候，字体大小为 $320/15$ 就是 21.33px
- ⑤ 用我们页面元素的大小 除以不同的 html 字体大小会发现他们比例还是相同的

比如我们以 750为标准设计稿：

- ① 一个100*100像素的页面元素 在 750屏幕下， 就是 $100 / 50$ 转换为rem 是 $2\text{rem} * 2\text{rem}$ 比例是 1比1
- ② 320屏幕下， html 字体大小为 21.33 则 $2\text{rem} = 42.66\text{px}$ 此时宽和高都是 42.66 但是 宽和高的比例还是 1比1
- ③ 但是已经能实现不同屏幕下 页面元素盒子等比例缩放的效果

rem 实际开发适配方案1

元素大小取值方法

- ① 最后的公式： 页面元素的rem值 = 页面元素值 (px) / (屏幕宽度 / 划分的份数)
- ② 屏幕宽度/划分的份数 就是 html font-size 的大小
- ③ 或者： 页面元素的rem值 = 页面元素值 (px) / html font-size 字体大小

苏宁首页案例制作

掌握 rem 布局技巧

苏宁网移动端首页

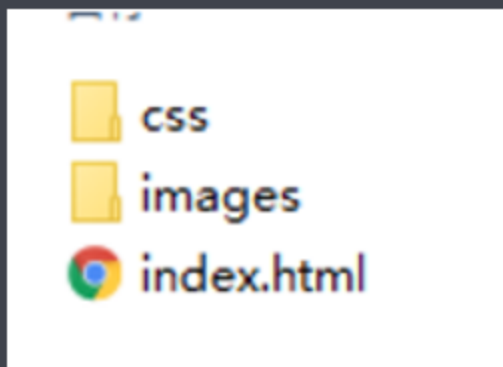
- 访问地址: m.suning.com



技术选型

- 方案：我们采取单独制作移动页面方案
- 技术：布局采取rem适配布局 (less + rem + 媒体查询)
- 设计图：本设计图采用 750px 设计尺寸

搭建相关文件夹结构



设置视口标签以及引入初始化样式

```
<meta name="viewport" content="width=device-width, user-scalable=no,  
initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
```

```
<link rel="stylesheet" href="css/normalize.css">
```

设置公共common.less文件

1. 新建 common.less 设置好最常见的屏幕尺寸，利用媒体查询设置不同的html字体大小，因为除了首页其他页面也需要。
2. 苏宁网站首页的开发尺寸有 320px、360px、375px、384px、400px、414px、424px、480px、540px、720px、750px等。
3. 划分的份数我们定为 15 等份。
4. 因为我们 pc 端也可以打开我们苏宁移动端首页，我们默认 html 字体大小为 50px，注意这句话写到最上面。

新建index.less文件

1. 新建 index.less 这里面写首页的样式
2. 将刚才设置好的 common.less 引入到 index.less 里面
语法如下：

```
// 在 index.less 中导入 common.less 文件  
@import "common"
```

3. 生成index.css 引入到 index.html 里面

body样式

```
body {  
  min-width: 320px;  
  width:15rem;  
  margin: 0 auto;  
  line-height: 1.5;  
  font-family: Arial,Helvetica,STHeiTi,sans-serif;  
  background: #F2F2F2;  
}
```

简洁高效的rem适配方案flexible.js

技术方案1

- less
- 媒体查询
- rem

技术方案2（推荐）

- flexible.js
- rem

简洁高效的rem适配方案flexible.js

手机淘宝团队出的简洁高效 移动端适配库

我们再也不需要在写不同屏幕的媒体查询，因为里面js做了处理

它的原理是把当前设备划分为10等份，但是不同设备下，比例还是一致的。

我们要做的，就是确定好我们当前设备的html 文字大小就可以了

比如当前设计稿是 750px，那么我们只需要把 html 文字大小设置为 75px($750\text{px} / 10$) 就可以

里面页面元素rem值： 页面元素的px 值 / 75

剩余的，让flexible.js来去算

github地址：<https://github.com/amfe/lib-flexible>

使用适配方案2制作苏宁移动端首页

1. 技术选型

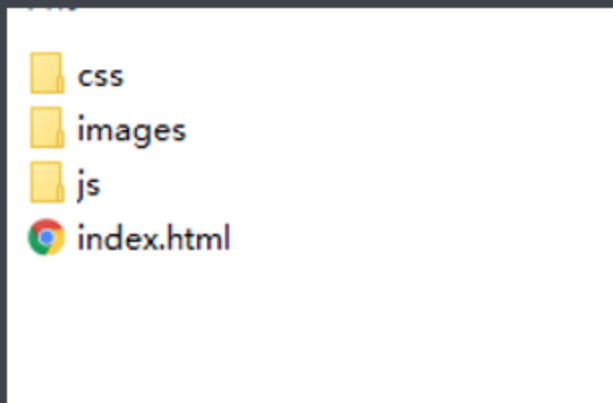
方案：我们采取单独制作移动页面方案

技术：布局采取rem适配布局2 (flexible.js + rem)

设计图：本设计图采用 750px 设计尺寸

使用适配方案2制作苏宁移动端首页

2. 搭建相关文件夹结构



使用适配方案2制作苏宁移动端首页

设置视口标签以及引入初始化样式还有js文件

```
<meta name="viewport" content="width=device-width, user-scalable=no,  
initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
```

```
<link rel="stylesheet" href="css/normalize.css">
```

```
<link rel="stylesheet" href="css/index.css">
```

我们页面需要引入 这个js文件

在 index.html 中 引入 flexible.js 这个文件

```
<script src= “js/flexible.js” > </script>
```

使用适配方案2制作苏宁移动端首页

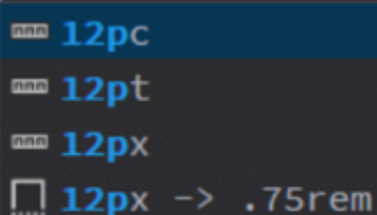
body样式

```
body {  
  min-width: 320px;  
  width: 15rem;  
  margin: 0 auto;  
  line-height: 1.5;  
  font-family: Arial, Helvetica, STHeiTi, sans-serif;  
  background: #F2F2F2;  
}
```

VSCode px 转换rem 插件 cssrem

这是一个神奇的插件

```
1 // "cssrem.rootFontSize": 16
2 body {
3     font-size: .75rem;
4     margin: .625rem;
5     padding: 1.875rem;
6     left: 12p
7 }
```



The screenshot shows the VS Code editor with a CSS file. The cursor is at the end of the line 'left: 12p'. A dropdown menu is open, showing four suggestions: '12pc', '12pt', '12px', and '12px -> .75rem'. The '12px -> .75rem' option is highlighted, indicating the plugin's functionality to convert pixels to REM units based on the root font size.

VSCode px 转换rem 插件 cssrem

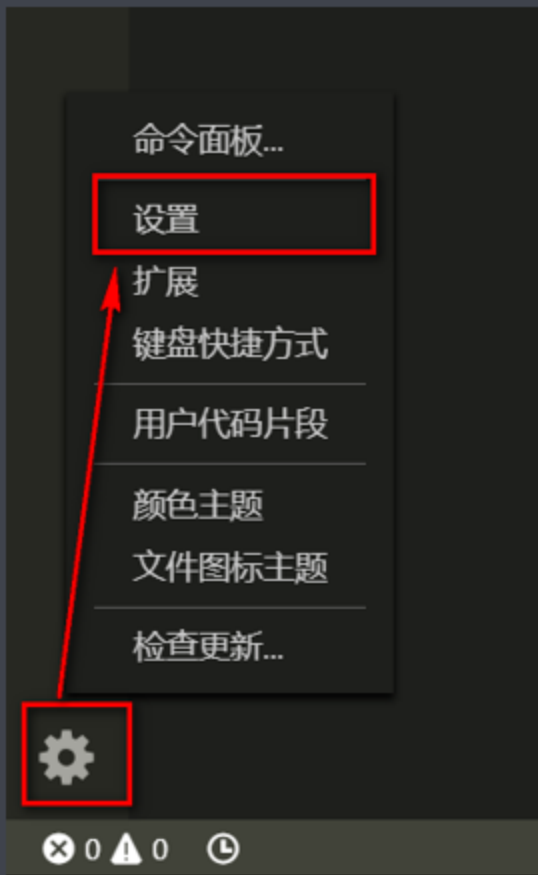
这是一个神奇的插件



VSCode px 转换rem 插件 cssrem

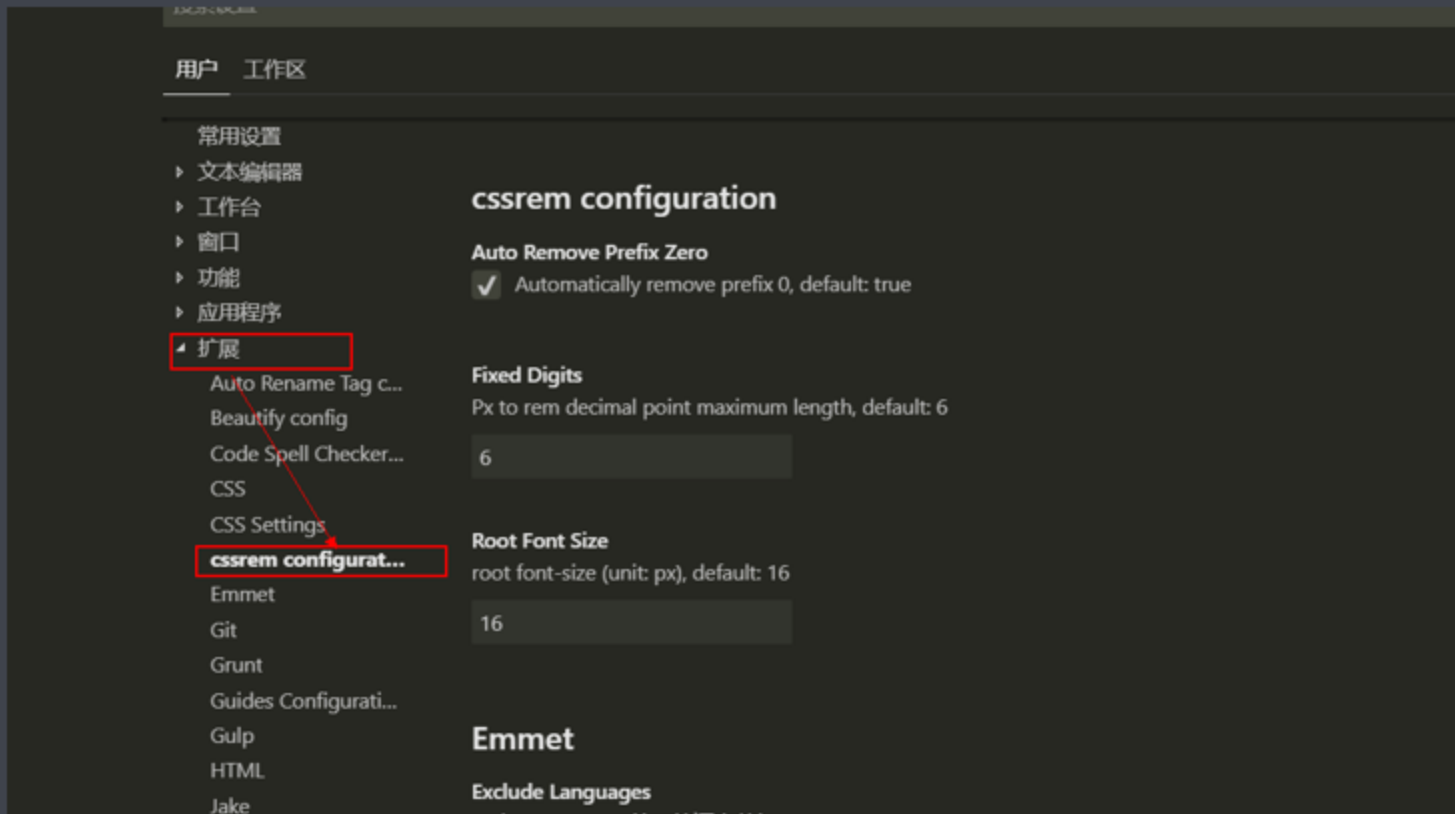
设置html字体大小基准值

1. 打开 设置 快捷键是 ctrl + 逗号



VSCode px 转换rem 插件 cssrem

设置html字体大小基准值



拉勾教育

— 互联网人实战大学 —



下载「拉勾教育App」
获取更多内容