**高效，极简主义**

1. 屏幕分辨率、缩放dpr和浏览器px
2. meta参数说明
3. rem和定高
4. flex布局
5. Emmet语法
6. 调试工具：Chrome（CSS调试，JS断点调试，手机模拟）
7. 移动端UI框架
8. 常用介绍：

**移动web屏幕适配**

移动端开发最难的不在于代码的编写，而在于屏幕的适配。对于移动端来说，用户的屏幕大小和分辨率各不相同，为了保证统一的用户体验，就要求前端开发人员，对不同尺寸和大小的屏幕进行适配。

举个简单的例子：iPhone6、iPhone6 plus和Nexus 6P，分别代表iOS阵营（普通屏和高分屏（高分辨率屏幕））和Android安卓阵营。

iPhone6：

**主屏尺寸：**4.7英寸

**主屏分辨率：**1334\*750像素

**浏览器分辨率：**667\*375像素

iPhone6 plus：

**主屏尺寸：**5.5英寸

**主屏分辨率：**[1920\*1080像素](http://detail.zol.com.cn/cell_phone_index/subcate57_list_s4328_1.html)

**浏览器分辨率：**[736\*414像素](http://detail.zol.com.cn/cell_phone_index/subcate57_list_s4328_1.html)

Nexus 6P：

**主屏尺寸：**5.7英寸

**主屏分辨率：**[2560\*1440像素](http://detail.zol.com.cn/cell_phone_index/subcate57_list_s5528_1.html)

**浏览器分辨率：**[732\*412像素](http://detail.zol.com.cn/cell_phone_index/subcate57_list_s5528_1.html)

从上边的例子可以看出，手机的屏幕尺寸和分辨率都不相同，iOS自家都不统一，更别说开源、碎片化的Android安卓阵营，各种奇葩的分辨率和尺寸。

但是，我们会提出几个问题：

1. 什么是主屏幕分辨率和浏览器分辨率？
2. 主屏幕分辨率和浏览器分辨率为什么不一致？他们之间有什么关系？
3. 主屏幕分辨率高，浏览器分辨率就一定大吗？

下一节，我们就简单介绍一下这几个基础知识点

1. **移动web基础知识**

回到上一节提出的几个问题，在这里要申明一点：**屏幕分辨率的1px不能等同于CSS中1px**，屏幕分辨率的1px是指的物理像素，而我们CSS中的1px是值得逻辑像素。下面我们看一个iPhone6两个分辨率的关系式：

物理像素：1334 \* 750 px

系数： **2**

逻辑像素： 667 \* 375px

我们看这个结果，是不是满足等式关系。在iPhone6 中，CSS的1px等自身物理像素的2倍。下面引出几个新概念: layout viewport（布局视口） 、visual viewport（视觉视口）。

**1、layout viewport（布局视口）**

一般移动设备的浏览器都默认设置了一个viewport 元标签，定义一个虚拟的layout viewport（布局视口），用于解决早期的页面在手机上显示的问题。iOS, Android基本都将这个视口分辨率设置为 980px，所以pc上的网页基本能在手机上呈现，只不过元素看上去很小，一般默认可以通过手动缩放网页。

**2、visual viewport（视觉视口）和物理像素**

visual viewport（视觉视口）备物理屏幕的可视区域，屏幕显示器的物理像素，同样尺寸的屏幕，像素密度大的设备，硬件像素会更多。

**3、ideal viewport（理想视口）和 dip （设备逻辑像素）**

ideal viewport（理想视口）通常是我们说的屏幕分辨率。

dip （设备逻辑像素）跟设备的硬件像素无关的。一个 dip 在任意像素密度的设备屏幕上都占据相同的空间。

满足如下公式：

逻辑像素宽度 \* 倍率 = 物理像素宽度

1. **meta标签**

**1、meta基础知识**

H5页面窗口自动调整到设备宽度，并禁止用户缩放页面  
<**meta** name=**"viewport"**

content=**"width=device-width,initial-scale=1.0,minimum-scale=1.0,maximum-scale=1.0,user-scalable=no"** />  
忽略将页面中的数字识别为电话号码  
<**meta** name=**"format-detection"** content=**"telephone=no"** />  
忽略Android平台中对邮箱地址的识别  
<**meta** name=**"format-detection"** content=**"email=no"** />  
当网站添加到主屏幕快速启动方式，可隐藏地址栏，仅针对ios的safari  
<**meta** name=**"apple-mobile-web-app-capable"** content=**"yes"** />  
*<!-- ios7.0版本以后，safari上已看不到效果 -->*将网站添加到主屏幕快速启动方式，仅针对ios的safari顶端状态条的样式  
<**meta** name=**"apple-mobile-web-app-status-bar-style"** content=**"black"** />  
*<!-- 可选default、black、black-translucent -->*

**2、viewport模板——通用**

<!DOCTYPE html>  
<**html** lang=**"en"**>  
<**head**>  
 <**meta** charset=**"UTF-8"**>  
 <**meta** name=**"viewport"**

content=**"width=device-width,initial-scale=1.0,maximum-scale=1.0,user-scalable=no"**/>  
 <**meta** name=**"apple-mobile-web-app-capable"** content=**"yes"**/>  
 <**meta** name=**"apple-mobile-web-app-status-bar-style"** content=**"black"**/>  
 <**meta** name=**"format-detection"** content=**"telephone=no, email=no"**/>  
 <**link** rel=**"stylesheet"** type=**"text/css"** href=**"mystyle.css"**>  
 <**title**>标题</**title**>  
</**head**>  
<**body**>  
这里是开始内容  
</**body**>  
</**html**>

这里是每个属性的详细介绍：

| **属性名** | **取值** | **描述** |
| --- | --- | --- |
| width | 正整数或 device-width | 定义视口的宽度，单位为像素 |
| height | 正整数 或 device-height | 定义视口的高度，单位为像素，一般不用 |
| initial-scale | [0.0-10.0] | 定义初始缩放值 |
| minimum-scale | [0.0-10.0] | 定义缩小最小比例，它必须小于或等于maximum-scale设置 |
| maximum-scale | [0.0-10.0] | 定义放大最大比例，它必须大于或等于minimum-scale设置 |
| user-scalable | yes/no | 定义是否允许用户手动缩放页面，默认值yes |

#### **width**

width属性被用来控制layout viewport（布局视口）的宽度，layout viewport（布局视口）宽度默认值是设备厂家指定的。iOS, Android基本都将这个视口分辨率设置为 980px。我们可以 width=320 这样设为确切的像素数，也可以设为device-width这一特殊值，一般为了自适应布局，普遍的做法是将width设置为device-width。例如：

<**meta** name=**"viewport"** content=**"width=device-width, initial-scale=1, user-scalable=no, minimal-ui"**/>

width=device-width 也就是将layout viewport（布局视口）的宽度设置 ideal viewport（理想视口）的宽度。网页缩放比例为100%时，一个CSS像素就对应一个 dip（设备逻辑像素），而layout viewport（布局视口）的宽度，ideal viewport（理想视口）的宽度（通常说的分辨率），dip 的宽度值是相等的。

**height**

与width类似，但实际上却不常用。

**initial-scale**

initial-scale用于指定页面的初始缩放比例：

<**meta** name=**"viewport"** content=**"initial-scale=1.5"**/>

initial-scale=1 表示将layout viewport（布局视口）的宽度设置为 ideal viewport（理想视口）的宽度，initial-scale=1.5 表示将layout viewport（布局视口）的宽度设置为 ideal viewport（理想视口）的宽度的1.5倍。

**maximum-scale**

maximum-scale用于指定用户能够放大的最大比例，例如：

<**meta** name=**"viewport"** content=**"initial-scale=1,maximum-scale=3"** />

假设页面的默认缩放值initial-scale是1，那么用户最终能够将页面放大到这个初始页面大小的3倍。

**minimum-scale**

类似maximum-scale的描述，不过minimum-scale是用来指定页面缩小比例的。通常情况下，不会定义该属性的值，页面太小将难以阅读。

**user-scalable**

user-scalable来控制用户是否可以通过手势对页面进行缩放。该属性的默认值为yes，可被缩放，你也可以将该值设置为no，表示不允许用户缩放网页。例如：

<**meta** name=**"viewport"** content=**"user-scalable=no"**/>

1. **定高适配和rem适配**

目前常用的两种适配手机屏幕的方案就是：定高法和rem法。

定高法：顾名思义就是高度固定，通过增加元素间的空白进行适配。优点简单易学，屏幕越大显示的内容越多。

参考：大众点评网https://m.dianping.com/

rem法：以及html元素根节点的font-size为基准，通过监测浏览器的宽度，动态改变font-size的大小，达到动态适配不同宽度屏幕的效果。优点最大程度的保证不同屏幕效果的统一。

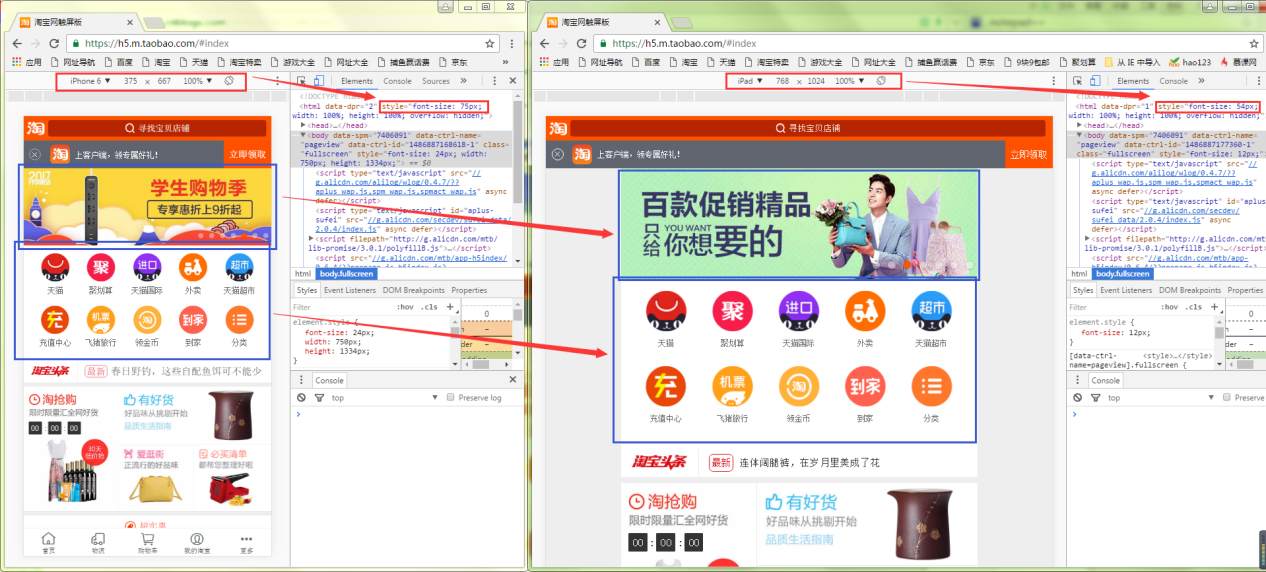
参考：手机淘宝触屏版https://h5.m.taobao.com/#index

1. **定高法**



结合给出的对比图，能够看到在iPhone6、iPad和PC上同时打开大众点评手机版时，所框出的区域高度都是一样的，这就是定高法。而图片img在移动端的width一般都设为100%，所以随着屏幕的变大高度会增加。

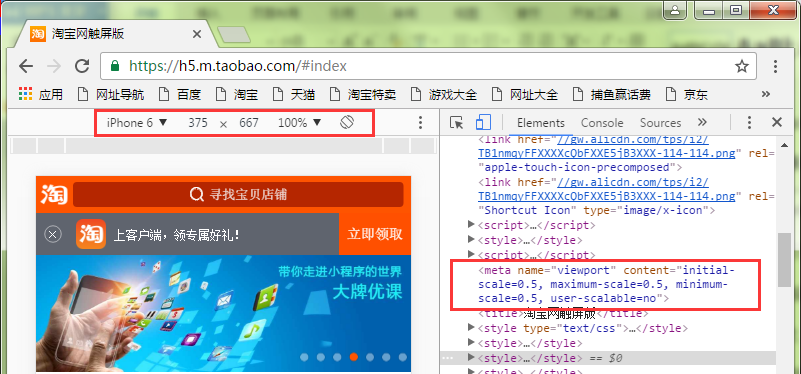
**2、rem法**



从图片的对比中可以看出。在iPhone6和iPad上，同时打开淘宝触屏版时，iPhone6显示的元素较小，iPad上显示较大，并且这个是整体放大。不像定高法那么粗暴，显得更加柔和一点，体验更好。并且观察代码也可以看到，在 iPhone6打开时，html标签的font-size：75px；

iPad打开时，html标签的font-size：54px；

发现确实通过改变font-size来实现屏幕适配的，但是有人会说不对呀，应该屏幕越大，font-size越大，元素才会变大，为什么在iPad上的font-size小反而显示的更大呢，继续看代码：



它在iPhone6的时候设置了页面缩放initial-scale=0.5，也就是缩小2倍，上面的font-size等同于37.5px，所以在iPhone6上整体UI风格会小一点。同时还注意到页面顶部的搜索输入栏是采用的定高法来实现的，所以，方法都是不是固定的，要根据实际情况来具体对待。

**3、设计稿宽度转化为rem**

***rem的计算公式是：px=系数 \* font-size；***

UI设计师提供的设计稿宽度一般以iPhone6为基准，设计稿宽度为750px。假设其中有一元素宽度为300px，高度为200px，下面请看转化过程：

1）在750px的设计稿当中是[300px 200px]，以实际浏览器宽度为375px的iPhone6为开发基准时要除以2，所以CSS因该为[150px 100px]；

2）定义1rem=10px，所以转化rem就是[15rem 10rem]，此时我们的页面是与设计稿1:1还原的；

3）为了转化方便省去计算rem的复杂性，我们定义

<html style="font-size:50px">，此时上边的[300px 200px] 直接除以100即可，变成[3rem, 2rem]。

***注意：在谷歌浏览器中最小font-size是12px，所以小于12px只能在火狐浏览器上调试。***

4）使用规则：整行区域划分，比如导航栏分成三部分[150px 450px 150px ] ，一般使用百分比形式[20% 60% 20%]。

1. Sublime有插件直接帮你自动完成px-->rem的计算【cssrem】，简直是一大利器。安装教程：http://www.wozhuye.com/index.php?m=content&c=index&a=show&catid=22&id=298

6）JS动态更改根节点字体大小

**var *fontSize*** = 50;  
**var *\_w*** = **document**.**body**.**clientWidth**;  
**document**.getElementsByTagName(**"html"**)[0].**style**.**fontSize** = ***\_w***/7.5+**"px"**;  
**window**.onresize = **function**(){  
 **var** \_w = **document**.**body**.**clientWidth**;  
 **document**.getElementsByTagName(**"html"**)[0].**style**.**fontSize** = \_w/7.5+**"px"**;  
}

7）Jquery动态更改根节点字体大小

**var *\_w*** = $(**window**).**width**();  
$(**'html'**).css(**'font-size'**,***\_w***/7.5+**'px'**);  
$(**window**).**resize**(**function**(){  
 **var** \_w = $(**window**).**width**();  
 $(**'html'**).css(**'font-size'**,\_w/7.5+**'px'**);  
});

1. rem计算优化

根据浏览器窗口动态改变font-size的时候，会出现小数位，由于js的四舍五入产生进位，会导致rem计算超出浏览器宽度，从而产生错位现象。此时我们可以优化计算方法为，JS四舍五入不进位且保留一位小数。

***font-size = parseInt(\_w/7.5\*10)/10；***