#### **常见的屏幕适配方案及其优劣**

##### **1.固定元素适配**

就是在页面布局及元素设置的时候固定宽度或高度，如：320；一般web网页端采用px作为固定宽度单位，在布局时采用固定像素值，设置页面布局的最大宽度，然后在超出宽度的部分以白底显示；这种做法在早期的互联网页面比较常见，比如早期的淘宝webpage，页面设置成320的宽度，超出部分留白，但是随着不同尺寸的设备的出现，这种方式会导致页面在不同设备上显示差异较大；相对于大屏幕设备来说，空白边太大，相对于高分辨率的设备来说图片、字体等元素显示较小，这种方法已经相对落后淘汰了；

##### 2.流式布局**适配**

就是固定高度，宽度用百分比来设定元素和布局；这种做法的好处就是可以使元素撑满设备屏幕，不至于留白或被隐藏；当然，缺点就是容易导致元素的内容不协调（如图片拉伸变形、文字溢出等），影响视觉效果。这种方法只对少部分设备屏幕适配较好，在科技产品爆炸的当下，单纯的这种做法已经不太合适用于屏幕适配了；

##### 3.利用第三方插件布局**适配**

利用第三方的一些现成的前端框架（如:Bootstrap、jQueryUi等），以其开源、配套成熟、应用广泛等特点，将其应用到移动端的页面中，使用媒体查询，只需在CSS中添加@media screen属性,根据浏览器宽度判断并输出不同的长宽值，能够有比较好的适配效果；但是，毕竟受第三方插件的影响，难免会有难以修复的未知问题，应用到移动端页面具有一定的风险性，而且，各插件的体量和性能各不相同，没有明确的官方标准，可维护性较差，对开发者的要求也较高，开发成本高；媒体查询对于屏幕元素的适配也是马马虎虎，虽然大家通常用到，但是，在协调性上还有一定的差距；目前，一些大型的主流网站喜欢用这种方式；

##### 4.响应式设计适配

目前，在移动设备上进行网页的重构或开发中常用的做法就是设置viewport meta 标签（viewport 是用户网页的可视区域），将页面的宽度固定，然后获取设备的宽度，可以得到我们之前设定的宽度与设备宽度的比例，再使用HTML5新增的viewport来对页面进行缩放，并固定不允许用户再重新缩放，让网页适配或响应各种不同分辨率的移动设备。使用方法如下：

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

在苹果的规范中，meta viewport 有6个属性(暂且把content中的那些东西称为一个个属性和值)，如下：

width：控制 viewport 的大小，可以指定的一个值，如 600，或者特殊的值，如 device-width 为设备的宽度（单位为缩放为 100% 时的 CSS 的像素）。

height：和 width 相对应，指定高度。

initial-scale：初始缩放比例，也即是当页面第一次 load 的时候缩放比例，，为一个数字，可以带小数。

maximum-scale：允许用户缩放到的最大比例，为一个数字，可以带小数。

minimum-scale：允许用户缩放到的最小比例，为一个数字，可以带小数。

user-scalable：用户是否可以手动缩放，是否允许用户进行缩放，值为"no"或"yes", no 代表不允许，yes代表允许。

这些属性可以同时使用，也可以单独使用或混合使用，多个属性同时使用时用逗号隔开就行了。

当然，这种做法就是简单高效，基本满足屏幕的适配需求，对于一些简单而要求不高的h5页面是够用了；但是，对于一些复杂有精度要求的h5页面来说，还是有优化的空间的；设置viewport 只是解决常见的屏幕适配布局的问题，因为安卓版本或浏览器的不同，其对这一属性的支持性也有差异； 而且在用户体验上，随着手机屏幕的分辨率已经越来越高，高像素但是屏幕尺寸却没有发生太大变化，对屏幕位置元素精度（元素的精度与显示的清晰度有关）的要求也越来越高；所以，在一些设备上缩放页面，难免会有元素显示模糊的现象；

##### 5.采用rem的响应式布局适配

结合以上的适配方法，我们可进一步提炼，采用css3新引入的单位—**rem**（相对长度单位。相对于html根元素的font-size计算值的倍数），作为设置页面元素宽高单位，简而言之就是通过动态设置html根元素的font-size，等比缩放元素大小来自适应移动设备。这种做法，就可以完美解决屏幕适配元素显示清晰化的问题，这也就是本案例要讲的阿里团队目前最普遍使用的高清方案，也是现在移动端普遍使用的高清方案。

#### 阿里高清屏幕适配方案分析及运用

##### 方案介绍

阿里的高清方案就是利用rem的特性（我们知道默认情况下html的1rem = 16px），根据设备屏幕的DPR（设备像素比，又称DPPX，比如DPR=2时，表示1个CSS像素由4个物理像素点组成）根据设备DPR调整页面的压缩比率（即：1/DPR），同时动态设置 html 的font-size为（50 \* DPR)，进而达到高清效果。

###### 核心源码

!**function** (e) {  
 **function** *t*(a) {  
 **if** (i[a]) **return** i[a].**exports**;  
 **var** n = i[a] = {**exports**: {}, **id**: a, **loaded**: !1};  
 **return** e[a].call(n.**exports**, n, n.**exports**, *t*), n.**loaded** = !0, n.**exports** }  
  
 **var** i = {};  
 **return** *t*.**m** = e, *t*.**c** = i, *t*.**p** = **""**, *t*(0)  
}([**function** (e, t) {  
 **"use strict"**;  
 Object.defineProperty(t, **"\_\_esModule"**, {**value**: !0});  
 **var** i = **window**;  
 t[**"default"**] = i.flex = **function** (normal, e, t) {  
 **var** a = e || 100, n = t || 1, r = i.**document**, o = **navigator**.**userAgent**,  
 d = o.match(/Android[\S\s]+AppleWebkit\/(\d{3})/i), l = o.match(/U3\/((\d+|\.){5,})/i),  
 c = l && parseInt(l[1].split(**"."**).join(**""**), 10) >= 80,  
 p = **navigator**.**appVersion**.match(/(iphone|ipad|ipod)/gi), s = i.devicePixelRatio || 1;  
 p || d && d[1] > 534 || c || (s = 1);  
 **var** u = normal ? 1 : 1 / s, m = r.querySelector(**'meta[name="viewport"]'**);  
 m || (m = r.createElement(**"meta"**), m.setAttribute(**"name"**, **"viewport"**), r.**head**.appendChild(m)), m.setAttribute(**"content"**, **"width=device-width,user-scalable=no,initial-scale="** + u + **",maximum-scale="** + u + **",minimum-scale="** + u), r.**documentElement**.**style**.**fontSize** = normal ? **"50px"** : a / 2 \* s \* n + **"px"** }, e.**exports** = t[**"default"**]  
}]);  
flex(**false**, 100, 1);

###### 方案优势

* + 引用简单，布局简便（只要把js代码贴到head标签里面或引用这部分核心代码）。
  + 根据设备屏幕的DPR，自动设置最合适的高清缩放。
  + 保证了不同设备下视觉体验的一致性。（常用方案是屏幕越大元素越大；此方案是屏幕越大，看的越多）。
  + 有效解决移动端真实1px问题（这里的1px 是设备屏幕上的物理像素）。

##### 使用案例—h5编写移动端“周边服务”模块

###### Rem换算

这里“周边服务”模块是集图片广告轮播和服务菜单为一体的上下结构的布局；由于引用的是阿里高清方案，这个要求对具体元素固定尺寸的单位用rem来表示；通常手机端网页的宽度为750或640，而这个方案默认是1rem = 100px，也就是说如果UI设计效果图是640，高清方案要完整还原效果图，只需要将设置元素宽度设置为6.4rem，如果元素的宽度超过效果图宽度的一半（效果图宽为640或750），果断使用百分比宽度，或者flex布局。示例如下：

这里设备屏幕的DPR=1，html的font-size为50px，此时 1rem = 50px，则8px单位换算为rem就是8/50=0.16rem；为定义一组图片轮播的原点按钮，在css中对应rem设置如下：

.**unslider-nav ol li** {  
 **display**: **inline-block**;  
 **width**: 0.16**rem**;  
 **height**: 0.16**rem**;  
 **margin**: 0 0.01**rem**;  
 **background**: **#ffffff**;  
 **border-radius**: 0.08**rem**;  
 **overflow**: **hidden**;  
 **text-indent**: -999**em**;  
 **border**: 0.01**rem solid #000000**;  
 **cursor**: **pointer**;  
}

总之，固定尺寸的元素如果要实现高清方案就是将其像素单位换算成对应比例的rem；

###### 应用方法

首先，我们将核心源码贴到head标签里面（案例中是引用这部分核心代码），因为核心源码是根据设备屏幕的DPR动态设置 html 的font-size，引用如下：

注意，不要手动设置viewport，该方案自动帮你设置。所以将viewport注释了。

<**title**>周边服务</**title**>  
 <**script type="text/javascript" src="<%=**path**%>/static/common/mobile.ali.adaptation.design.js"**></**script**>  
*<%-- <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no"/>--%>*

其次，将h5页面内固定尺寸的元素换算成rem单位。

例1．使用媒体查询的情况下，设置菜单p元素的font-size如下：

*/\*480分辨率以下（小于479px）\*/***@media only screen and** (**max-width**: 479**px**) {  
 **p** {  
 **font-size**: 0.24**rem**;  
 }  
}

例2.对屏幕整体布局的div可设置如下：

<**div class="lay-all"**>  
 <**div class="lay-top"**>

……

.**lay-all** {  
 **width**: 100%;  
 **min-width**: 6.40**rem**;  
 **height**: 100%;  
 **min-height**: 9.60**rem**;  
 **margin**: 0 **auto**;  
 **background**: **#f3f5f9**;  
 **position**: **absolute**;  
}

最后，我们可以根据需要可以对图片、页面字体等元素进行等比伸缩，伸缩方法不外乎设置百分比或使用媒体查询（例1和例2已经展示，当然使用该方案后已经不需要这么做了）等，以更好的达到页面及元素对设备屏幕的完整匹配。

###### 注意事项

1. 并不是所有用px的地方都要用rem，rem布局只针对固定宽度，对于尺寸比较大的元素，应该考虑用百分比。rem做单位的元素在任何设备下都是固定大小。
2. 使用高清方案代码进行rem单位换算时，因为其是根据设备的dpr动态设置html 的 font-size，如果dpr=1(如电脑端），则html的font-size为50px，此时 1rem = 50px，则对应元数p的font-size如果为14px，则换算为14/50=0.28rem，dpr的其他情况也是以此类推进行换算；
3. 默认情况下，高清方案的页面是开启压缩的（在核心源码最后一行flex(false, 100, 1)），如果引用了其他的UI库，其他类库也许是以px为单位的，为了兼容高清方案，最简单的做法就是禁止压缩的页面执行window.flex(true)或者将flex(false, 100, 1)中的false改为true，禁止所有引用的h5页面压缩，而rem的用法保持不变。
4. 在X5新内核Blink中，在排版页面的时候，会主动对字体进行放大，会检测页面中的主字体，当某一块字体在我们的判定规则中，认为字号较小，并且是页面中的主要字体，就会采取主动放大的操作。然而这不是我们想要的，可以采取给最大高度解决，如下：

**\*** {  
 **box-sizing**: **border-box**;  
 **max-height**: 100000**px**;  
}  
  
**\***:**before**, **\***:**after** {  
 **box-sizing**: **border-box**;  
 **max-height**: 100000**px**;  
}

1. 这个方案可以结合以上介绍的常用适配方法，根据具体需求混用设置对应元素。在项目实战中，不要仅仅满足于浏览器的模拟器，还是要用真机去看自己的适配换算效果；

##### 案例应用效果展示

###### 1.对于480以下分辨率的设备，以iphone4的320\*480为例，效果如下：



###### 2.对于720分辨率的设备，以Nexus7 600\*960为例，效果如下：



###### 对于720以上的分辨率的设备，以ipad Pro 1024\*1366为例，效果图如下：



#### 总结

总的来说，学习到阿里通用的高清方案还是比较强大和实用的，不管是对移动端的h5页面开发，还是对移动端的应用软件开发，都是一个有价值的实用方案；在强大的互联网背景下，阿里的一些通用方案都是比较成熟稳定的，这些方案值得我们开发者学习和积累，在今后的项目实战中加以运用并不断的丰富和完善；