# 移动web和响应式

# 目录

[移动web和响应式 1](#_Toc9263)

[目录 2](#_Toc1716)

[一、响应式网站的概念 3](#_Toc21180)

[1.1 欣赏一下响应式网站 3](#_Toc14282)

[1.2 原理 5](#_Toc348)

[二、BootStrap 7](#_Toc32231)

[2.1 概述 7](#_Toc29233)

[2.2 部署 7](#_Toc13955)

[2.3 敏捷开发 7](#_Toc10543)

[三、栅格系统 8](#_Toc26804)

[3.1 容器 8](#_Toc8255)

[3.2 12列栅格系统 8](#_Toc8512)

[四、Canvas 9](#_Toc18819)

[4.1 基本用法 9](#_Toc10324)

[4.2 矩形 10](#_Toc14534)

[4.3 路径 11](#_Toc9649)

[4.4 圆弧 12](#_Toc21850)

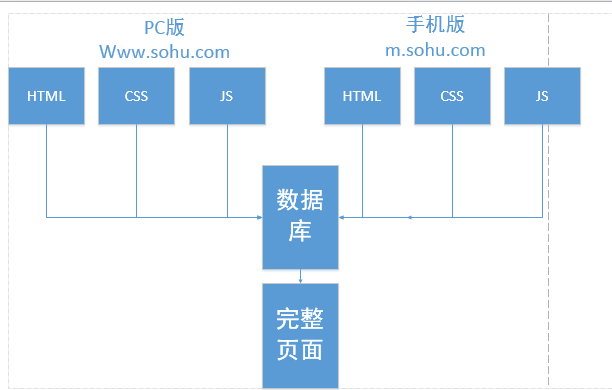
[4.5 文本 12](#_Toc12181)

[4.6 绘制图片 13](#_Toc16366)

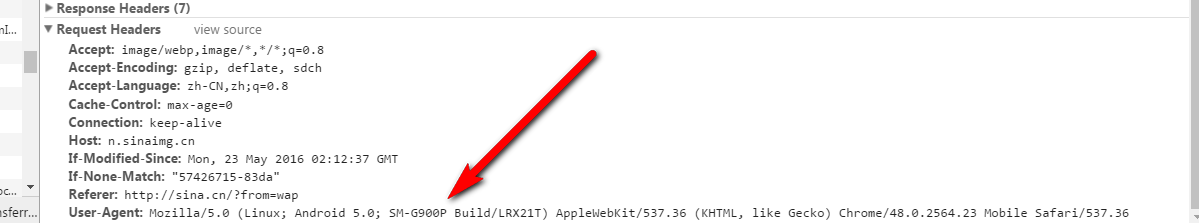
# 一、响应式网站的概念

## 1.1 欣赏一下响应式网站

移动web是一个独立的站点，HTML、CSS、JS都是与PC网站独立的。唯一和PC网站有关系的就是数据。



当用户用手机输入www.sohu.com的时候，由于的你上行HTTP请求中，有User-Agent标记，所以后台工程师可以写代码检测设备是什么，进行相应的跳转。



我们今天要学习的响应式网站不是这样子的，而是同一份HTML、CSS、JS，交付不同的屏幕呈递。



所谓的响应式网站，responsive web design（RWD），指的是根据用户浏览器的视口宽度不同，来由不同的版心。这样的话，大屏幕用户感觉自己的屏幕被撑满了，网站大气，能够看见更多的内容。小屏幕用户，不会出现横向滚动条。

优酷：

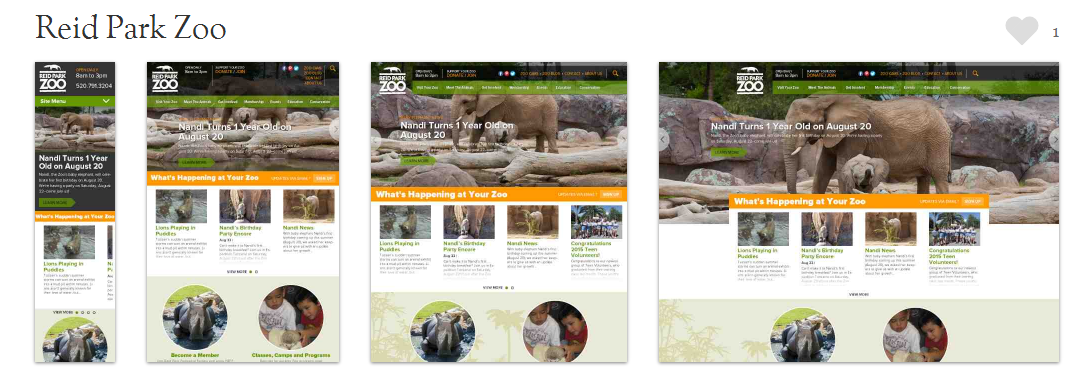
|  |  |
| --- | --- |
| 1340px版心宽度： | 1120px版心宽度 |

淘宝：

|  |  |
| --- | --- |
| 大屏幕用户： | 中屏幕用户： |

看看国外的网站。有一个网站，专门收集了全球各地的比较漂亮的响应式网站：<http://mediaqueri.es/>

中国人（不包括台湾地区）挺丢人，没有收录一个。



## 1.2 原理

如何知道用户的屏幕宽度，太简单了。我们可以用

|  |
| --- |
| 1. document.documentElement.clientWidth |

检测。或者干脆用CSS3媒体查询：

|  |
| --- |
| 1. @media screen and(min-width:1200px){} |

媒体查询：

|  |
| --- |
| 1. <link rel="stylesheet" type="text/css" href="css/reset.css" /> 2. <link rel="stylesheet" type="text/css" href="css/index1200.css" /> 3. <link rel="stylesheet" type="text/css" href="css/index980.css" media="screen and (min-width:980px) and (max-width:1200px)"/> 4. <link rel="stylesheet" type="text/css" href="css/index640.css" media="screen and (min-width:640px) and (max-width:980px)"/> 5. <link rel="stylesheet" type="text/css" href="css/index320.css" media="screen and (max-width:640px)"/> |

必须有一个link标签没有media属性的，因为IE6、7、8会认为有media属性的link标签是错误标签。所以一定要有一个link标签“留活口”。其他的css层叠这个css。

JS也可以实现响应式：给一个link标签加id：

<link rel="stylesheet" href="" id="rwdlink" />

然后用js检测屏幕的宽度，更改link的href属性：

|  |
| --- |
| 1. function setCSS(){ 2. var windowWidth = document.documentElement.clientWidth; 3. if(windowWidth >= 1200){ 4. rwdlink.href = ""; 5. }else if(windowWidth >= 980){ 6. rwdlink.href = "css/index980.css"; 7. }else if(windowWidth >= 640){ 8. rwdlink.href = "css/index640.css"; 9. }else{ 10. rwdlink.href = "css/index320.css"; 11. } 12. } |

实际上实现响应式不难，恶心的就是不优雅。需要大量的层叠，需要大量的书写同样的CSS。

所以，现在大家都用BootStrap来实现响应式页面，这个也是我们今天课程的重点。

# 二、BootStrap

## 2.1 概述

你面试的时候，你可以说你不会angular，你可以说你不会canvas。但是你要是说你不会boot，你就基本没戏了。

bootstrap就是计算机术语中的“引导程序”，来源于Twitter。让响应式网站变得简单，它的哲学是：**你不用关心响应式的事情，响应式已经完成了。**

中文官网：<http://www.bootcss.com/>

BootStrap可以称为“框架”，和之前学习的iQuery不同，它不仅仅是js，而且还是css。网站中的所有东西，比如标签（标签的类名不能随便起了）、CSS、js都是建筑在这个bootstrap上的。所以称为“框架”。

## 2.2 部署

去www.bootcss.com去下载boot的包，然后在页面中进行引用。

抽象出来：

|  |
| --- |
| 1. <!DOCTYPE html> 2. <html lang="zh-CN"> 3. <head> 4. <meta charset="utf-8"> 5. **<link href="css/bootstrap.min.css" rel="stylesheet">** 6. </head> 7. <body> 8. **<script src="js/jquery-1.12.3.min.js"></script>** 9. **<script src="js/bootstrap.min.js"></script>** 10. </body> 11. </html> |

## 2.3 敏捷开发

就是利用人家写好的css，和必要的HTML结构，我们往自己的页面里面ctrl+v。

敏捷开发是现在的一个流行趋势，让做页面变得简单。缺点也非常明显：不方便自定义。

所以boot搭建的网站，都是非常简单的，因为boot的哲学就是简单。页面中的所有元素都简单，美其名曰扁平化、商业化、normal。

设计师也需要懂BootStrap，然后设计出来的东西，就是一个高保真的一个模型图。

# 三、栅格系统

## 3.1 容器

boot中container是一个非常好用的容器，自己有4段的响应式变化：

|  |
| --- |
| 1. <div class="container"></div> |

在下表中能够看见container的宽度：



## 3.2 12列栅格系统

栅格系统：把页面看做12列，用类名：

|  |
| --- |
| 1. col-lg-\*\*\* |

来规定这个盒子占了多少列。加起来等于12的列，将并排在一个行显示。

下面的div是并排显示的：

|  |
| --- |
| 1. <div class=”col-lg-3”></div> 2. <div class=”col-lg-3”></div> 3. <div class=”col-lg-3”></div> 4. <div class=”col-lg-3”></div> |

如果想在手机端进行变化，可以写不同的占有列的数值：

|  |
| --- |
| 1. <div class=”col-lg-3 col-sm-6”></div> 2. <div class=”col-lg-3 col-sm-6”></div> 3. <div class=”col-lg-3 col-sm-6”></div> 4. <div class=”col-lg-3 col-sm-6”></div> |

国外比较好的boot模板：

<http://www.bootstrapzero.com/>http://startbootstrap.com/

# 四、Canvas

好消息，我们又回到了JS语境了。CSS3和手机端，说白了，又回到了margin、padding的时代，一定要多做东西。多去总结东西。我们后面的canvas游戏，也是兼容手机的。换句话说，就是为手机做的，扫描一个二维码就可以直接玩儿。

手册：https://developer.mozilla.org/zh-CN/docs/Web/API/Canvas\_API/Tutorial

今天把canvas的API介绍完毕，周四开始就直接用canvas做项目了。

canvas是html5新增的标签，表示“油画布”。是HTML5中最最浓墨重彩的一笔，能够颠覆Flash。

## 4.1 基本用法

|  |
| --- |
| 1. <canvas width="600" height="300">对不起，您不支持画布，请升级浏览器！</canvas> |

<canvas> 元素创造了一个固定大小的画布，它公开了一个或多个“渲染上下文”，“渲染上下文”可以用来绘制和处理要展示的内容。有两种“渲染上下文”，2D的，3D的。我们将会将注意力放在2D渲染上下文中，因为3D兼容极差。也会稍微介绍WebGL ，就是3D上下文。

JS中必须要书写的两个语句是：

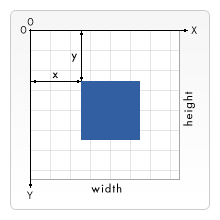
|  |
| --- |
| 1. //得到画布 2. **var canvas = document.getElementById("canvas");** 3. //得到上下文 4. **var ctx = canvas.getContext("2d");** |

之后的所有绘制，都是ctx的方法，而不是canvas的方法。

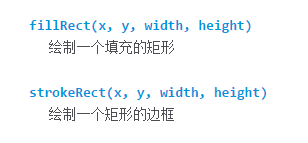
|  |
| --- |
| 1. ctx.fillRect(); 2. ctx.fillArc(); |

## 4.2 矩形

坐标系：



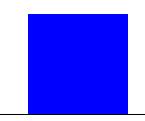
canvas中，有两种东西，一种叫做stroke（描边、笔触）和fill（填充、内容）。



|  |
| --- |
| 1. ctx.strokeStyle = "red"; //更改笔触的颜色 2. ctx.lineWidth = "4"; //更改笔触的宽度 3. ctx.strokeRect(100,100,200,50); //画一个矩形笔触，没有填充 |



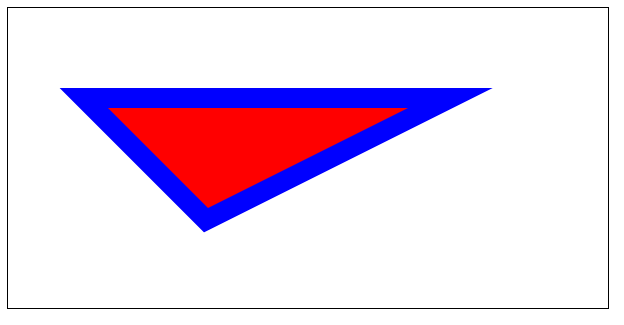
|  |
| --- |
| 1. ctx.fillStyle = "blue"; //更改填充的颜色 2. ctx.fillRect(200, 200, 100, 100); //画一个矩形填充，没有笔触 |



## 4.3 路径

路径就是线，路径没有fill，只有stroke。路径没有填充色，只有描边。

|  |
| --- |
| 1. //得到画布 2. var canvas = document.getElementById("canvas"); 3. //得到上下文 4. var ctx = canvas.getContext("2d"); 6. ctx.beginPath(); //表示我现在要开始绘制路径 7. ctx.moveTo(100,100); //把绘制点移动到100，100的位置去 8. ctx.lineTo(400,100); //绘制一条线。这条线就是(100,100)-(400,100) 9. ctx.lineTo(200,200); //绘制一条线。这条线就是(400,100)-(200,200) 10. ctx.closePath(); //闭合路径，系统会帮我们多绘制一条线(200,200)-(100,100) 12. ctx.strokeStyle = "blue"; 13. ctx.lineWidth = "40"; 14. ctx.stroke(); //把刚刚绘制的所有线添加笔触，显示出来 15. ctx.fillStyle = "red" 16. ctx.fill(); //把刚刚绘制的所有线，添加填充色。 |

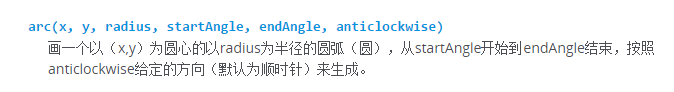


画线是虚的，只有stroke()、fill()了，才能看见。

stroke()方法表示把刚刚画好的线描绘出来； fill()表示把刚刚画好的线，填充颜色。

比如绘制矩形，有几种方法：

## 4.4 圆弧



比如：

|  |
| --- |
| 1. ctx.arc(200,200,60,0,1, false); |

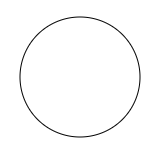
圆心是200,200。 半径是60 ， 从0弧度，画到1弧度。 顺时针。



画一个完整的圆：

|  |
| --- |
| 1. ctx.arc(200,200,60,0,6.3, false); |

6.3就是 Math.PI \* 2

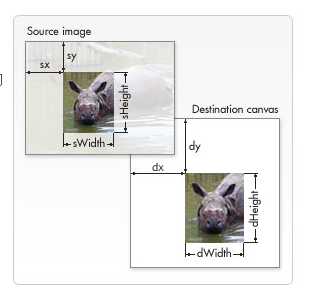
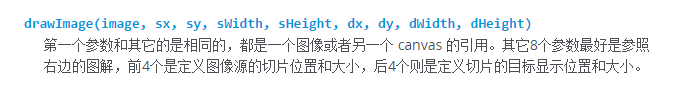


## 4.5 文本

|  |
| --- |
| 1. ctx.font = "40px 微软雅黑"; 2. ctx.fillText("你好",100,100); //100,100是位置 |

## 4.6 绘制图片

9个参数：



drawImage就是绘制图片，第一个参数一定要记住，是一个图片对象，而不是路径。

|  |
| --- |
| 1. //得到画布 2. var canvas = document.getElementById("canvas"); 3. //得到上下文 4. var ctx = canvas.getContext("2d"); 5. var image = new Image(); //创建一个图片对象 6. image.src = "images/0.jpg"; //图片的JS变量，索引 8. image.onload = function(){ 9. ctx.drawImage(image, 100,100 , 130, 130); 10. } |

|  |
| --- |
| 1. ctx.drawImage(image, 100,100 , 130, 130); |



|  |
| --- |
| 1. ctx.drawImage(image, 202,74,48,44,100,100,400,400); |

