1. 图片作为内容和背景的区别
2. 如果图片做为内容，当容器宽度再变化的时候，高度也会进行缩放
3. 如果是做为背景，再加上设置图片居中显示，那么在变化容器宽度的时候，图片可以显示中间部分，但是，当图片的高度与当前容器的高度不相适应的时候，图片可能造成一些区域空白或者图片的某些部分无法显示

Background-size:cover

Background-position:center

1. 组件轮播图使用
2. 在官网【JavaScript插件】中的【Carousel】中复制轮播图的代码
3. Carousel预制组件的说明
   1. data-ride="carousel":指定当前是一个轮播图插件
   2. class="carousel slide"：轮播图样式，如果不添加，js效果可以正常实现，但是样式不一样
   3. data-slide-to="0"控制指定索引的图片
   4. data-target="#carousel-example-generic"说明当前操作的是哪一个轮播图组件
4. 添加想进行展示的图片，同时添加对应数量的li。
5. 调整样式
   1. 在非移动端
      1. 图片会进行等比例缩放，造成原始大于屏幕宽度的图片在非移动端屏幕展示的时候需要进行缩放，造成高度也会等比例缩放，影响显示效果
      2. 解决方案：大图的话只需要改变宽度，高度固定，要让图片居中显示，将图片做成背景图片，通过设置背景图片的居中显示background-position:center，同时设置background-size:cover来达到背景图像完全覆盖背景区域
   2. 在移动端
      1. 添加超链接，在超链接里面添加图片
      2. 设置超链接的宽度为100%，设置图片的宽度也是100%.
   3. 静态的响应式轮播图
      1. 简单思路：在结构代码中添加两种情况下的图片内容
      2. 设置响应式隐藏，如在超小屏幕下:pc端的图片内容不显示 hidden-xs
   4. 静态式的轮播图完成之后的问题
      1. 在移动端也会加载非移动端的图片资源
      2. 在非移动端也会加载移动端的图片资源
      3. 数据固定，不方便动态扩展
   5. 更好的实现方式
      1. 通过后台数据请求，动态的加载资源：前台发送数据请求，后台处理，返回json数据，前台进行解析并显示
      2. 实现过程(使用json文件来模拟后台数据-index.json，再用模板动态生成在页面上，添加ajax请求)
6. artTemplate图片模板
7. 引入jQuery
8. 引入template-native.js
9. Html页面书写一段script标签，可以将type设置为temp/template，并为模板设置一个id名
10. 因为有多张图片，要进行一个循环输出，需要使用模板的【<%%>】形式来书写一段闭合的JS循环代码
11. 从数据取出的数据是一个对象数组，需要将其转换成对象的值来在模板中使用其长度进行循环
12. 写入我们需要添加到页面的html代码，需要修改的地方用【<%=key.value%>】的形式替换
13. 第一个元素要单独处理，在class中用【<%=%>】三元表达式进行判断是否为第一个，是就加上样式
14. 在取得的数据后进行包装时，添加一个标杆，用于判断是否是移动端
15. 注意If-else语句不能分开，否则有问题
16. 获取数据时调用template()方法，第一个参数是模板的id，第二个参数是装饰后的数据
17. 最后用html()方式输出在屏幕上
18. Underscore.js小圆点模板
19. 在获取数据时先读取模板的html内容，生成一个模板对象参数是模板的html字符串文本，

var indicator= \_.template($('#indTemp').html());//indTemp模板id

1. 调用模板对象的方法，传递数据，返回可渲染的html代码

var indHtml=indicator({'items':data});

1. 渲染

$('.carousel-indicators').html(indHtml);

1. 模板

<script type="temp/template" id="indTemp">  
<% \_.each(items,function(item,i){ %>  
 <li data-target="#carousel-example-generic" data-slide-to="<%=i%>" class="<%=i==0?'active':''%>"></li>  
<% }); %>  
</script>