链滴

1.Bootstrap

作者：linbainian

原文链接：https://ld246.com/article/1575881338738

来源网站：链滴

1.Bootstrap

1.简介

1 什么是bootstrap?

bootstrap是当下**最流行的前端框架**（界面工具集);

特点是灵活简单、代码优雅、美观大方;

目的在于让web开发更加迅速、敏捷;

由**Twitter**公司的两名前端工程师Mark Otto和Jacob Thornton在2011年发起的，并利用业余时间完了第一个版本的开发。

2 什么是框架?

库 lib library

写的更少做的更多 提供一套较为便捷的操作方式;

将一套功能体系封装到一个单独的文件中的东西;

Bootstrap提供一套前端需要的界面工具集合。

3 为什么使用Bootstarp?

- **生态圈火**，不断地更新迭代;
- 提供一套 **美观大方的界面组件**;
- **响应式界面，移动设备优先;**
- 让我们的 **Web 开发更简单，更快捷**

注意:

1. 使用 Bootstrap 并不代表不用写 CSS 样式，而是不用写绝大多数大家都会用到的样式
2. Bootstrap 不是 BootStrap！这是一个词，不是合成词，其含义为：n. 引导指令,引导程序

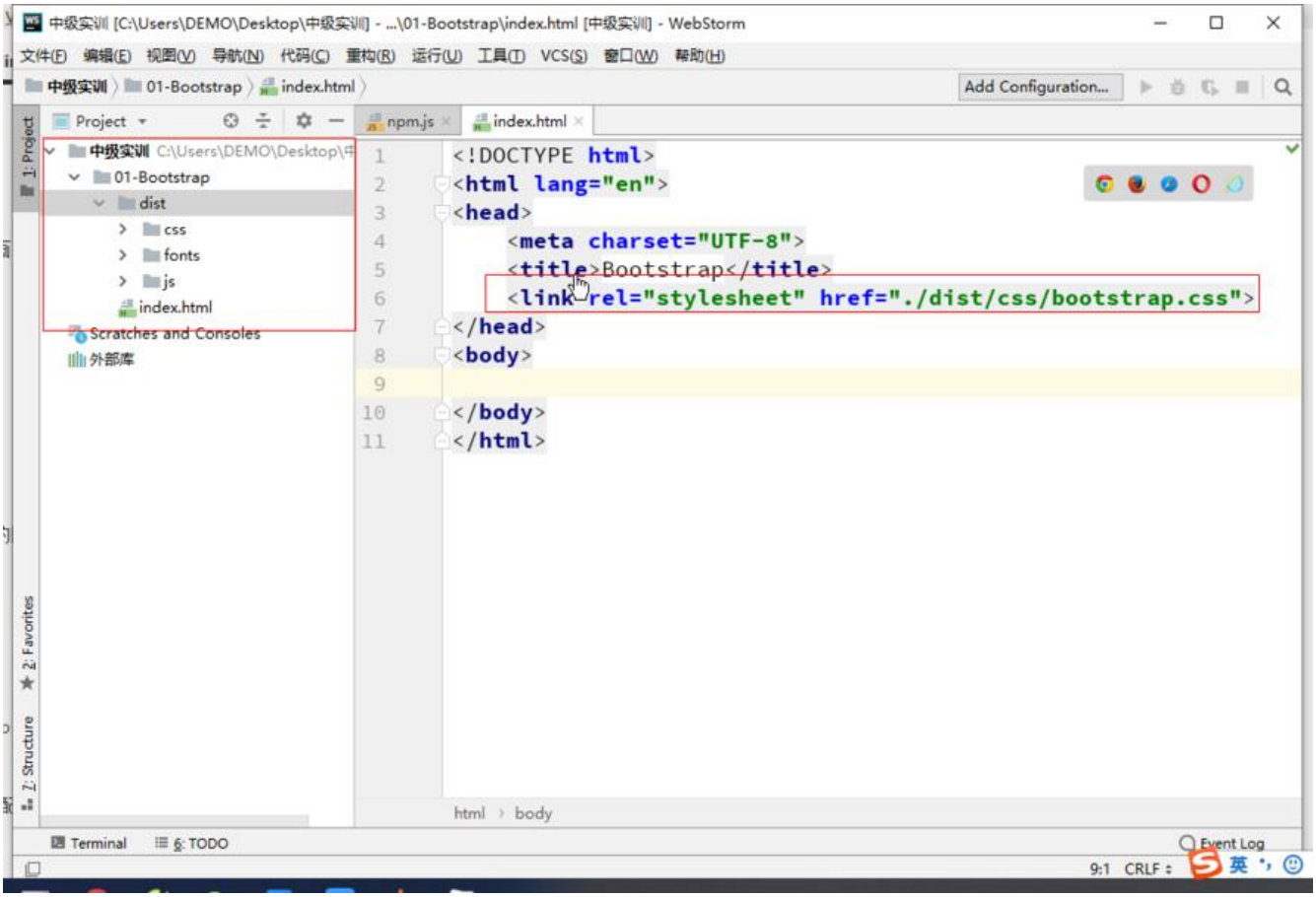
官网： http://www.bootcss.com/

github 地址： https://github.com/twbs/bootstrap

2.Bootstrap初体验

Boostrap 依赖 Jquery

1.项目集成BootStrap



2.Bootstrap初体验

视口viewport：可以说是一个**虚拟的窗口**，默认是**980px**

```
<!DOCTYPE html>
<html lang="zh-CN">
<head>
    <!--设置编码 gbk gb2312 utf-8-->
    <meta charset="utf-8">
    <!--
      告诉IE浏览器以最新的 解析器  去解析当前的页面，
          content="IE=edge"中的edge代表是最新的解析器,也可以说是IE-11。
          content="IE=10" 代表指定使用IE-10的解析器
    -->
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <!--
       兼容移动设备
          视口：虚拟的窗口，默认是980px
    -->
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- 上述3个meta标签*必须*放在最前面，任何其他内容都*必须*跟随其后！ -->
    <title>01-Bootstrap基本模板</title>
```

```
    <!-- 引入Bootstrap -->
    <link href="bootstrap/dist/css/bootstrap.css" rel="stylesheet">

    <!--
      如果IE浏览器版本小于9，将使用下面两个库():
         html5shiv：让版本小于9的IE浏览器，也能够使用h5的标签
         respond： 让版本小于9的IE浏览器，也能够使用C3的样式
    -->
    <!--[if lt IE 9]>
    <script src="js/html5shiv.min.js"></script>
    <script src="js/respond.min.js"></script>
    <![endif]-->

</head>
<body>

<h1>你好，世界！ </h1>

<!-- Bootstrap这个个框架依赖jQuery; jquery从1.9版本以后不再兼任ie9以下浏览器（推荐1.9.1） -
>
<script src="jquery/jquery-3.2.0.js"></script>
<script src="bootstrap/dist/js/bootstrap.js"></script>

</body>
</html>
```

####3.最简单的使用

组件：button ， button -group , input ， progress , 字体图标

```
    <!--按钮-->
    <button class="btn btn-default">普通按钮</button>
    <button class="btn btn-success">成功</button>
    <button class="btn btn-danger">危险</button>


    <!--按钮组-->
    <div class="btn-group" role="group" aria-label="...">
       <button type="button" class="btn btn-default">Left</button>
       <button type="button" class="btn btn-success">Middle</button>
       <button type="button" class="btn btn-default">Right</button>
    </div>

    <div class="btn-toolbar" role="toolbar" aria-label="...">
       <div class="btn-group" role="group" aria-label="...">
          <button type="button" class="btn btn-default">1</button>
          <button type="button" class="btn btn-default">2</button>
          <button type="button" class="btn btn-default">3</button>
       </div>
       <div class="btn-group" role="group" aria-label="...">
          <button type="button" class="btn btn-default">4</button>
          <button type="button" class="btn btn-default">5</button>
          <button type="button" class="btn btn-default">5</button>
```

```
        </div>
        <div class="btn-group" role="group" aria-label="...">
            <button type="button" class="btn btn-default">6</button>
            <button type="button" class="btn btn-default">7</button>
            <button type="button" class="btn btn-default">8</button>
        </div>
    </div>

    <!--文本框-->
    <div class="input-group" style="width: 300px">
        <span class="input-group-addon" id="basic-addon1" style="width: 100px">@ @ </spa
>
        <input type="text" class="form-control" placeholder="Username" aria-describedby="ba
ic-addon1">
    </div>
    <div class="input-group" style="width: 300px">
        <span class="input-group-addon" id="basic-addon3" style="width: 100px">@</span>
        <input type="text" class="form-control" placeholder="Username" aria-describedby="ba
ic-addon1">
    </div>

    <div class="input-group">
        <input type="text" class="form-control" placeholder="Recipient's username" aria-descri
edby="basic-addon2">
        <span class="input-group-addon" id="basic-addon2">@example.com</span>
    </div>


    <!--进度条-->
    <div class="progress">
        <div class="progress-bar" role="progressbar" aria-valuenow="60" aria-valuemin="0" ari
-valuemax="100" style="width: 40%;">
            40%
        </div>
    </div>
```

4.视口ViewPort

什么是视口？　窗口

每个网页默认都会有一个视口，视口其实是一个**虚拟的窗口**，默认的尺寸是980像素;

为了兼容移动设备，一般让**网页视口的宽度和设备的宽度的比例为 1:1，并且不允许用户缩放网页**;

```
<!--
viewport: 视口
width=device-width：  视口的宽 等于 设备的宽
user-scalable：  不允许网页进行缩放
initial-scale：初始化缩放为1
maximum-scale=1.0：  最大缩放为1
minimum-scale=1.0：  最小缩放为1
-->
<meta name="viewport"
      content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0,
```

```
minimum-scale=1.0">
```

5.Bootstrap常用的组件

1.面板

```
<div class="panel panel-danger">
        <div class="panel-heading">
           今天天气很好!
        </div>
        <div class="panel-body">
           风和日丽!
        </div>
        <div class="panel-footer">
           2019-10-21
        </div>
    </div>
```

2.徽章

```
<span class="badge" style="background-color: red">10</span>
 <div class="badge" style="background-color: red">20</div>
<button type="button" class="btn btn-danger">
    <span class="badge" style="background-color: red">30</span>
</button>
 <a href="">input
     <span class="badge" style="background-color: red">23</span>
     </a>
```

3.模态框

```
    <!--弹出模态框
    data-target="#myModel" 指定弹出的目标 (值是: 选择器)
    data-toggle="model"    以模态框的方式弹出
    data-dismiss="modal"   以模态框的方式消失
    -->
    <button class="btn btn-success" data-toggle="modal" data-target="#myModal">立即登
</button>
    <!--模态框-->
    <div class="modal fade" id="myModal" tabindex="-1" role="dialog">
        <div class="modal-dialog" role="document">
            <div class="modal-content">
                <div class="modal-header">
                    <button type="button" class="close" data-dismiss="modal" aria-label="Close">
span aria-hidden="true">&times;</span></button>
                    <h4 class="modal-title">欢迎登陆考试中心</h4>
                </div>
                <div class="modal-body">
                    <div>
                        用户名:<input type="text" placeholder="请输入用户名">
                    </div>
                    <div>
                        密   码:<input type="password" placeholder="请输入密码">
                    </div>
```

```
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-default" data-dismiss="modal">取消</button>
                <button type="button" class="btn btn-primary">登录</button>
            </div>
        </div><
    </div>
</div>
```

1.显示或者隐藏模态框方式一(添加属性):

```
  //调出模态框:
  //data-target="#myModal"   指定弹出的目标（值是：选择器）
  //data-toggle="modal"      指定以模态框的方式弹出
  //data-dismiss="modal"     指定以模态框的方式消失
<button type="button" data-toggle="modal" data-target="#myModal">Launch modal</button>
```

2.显示或者隐藏模态框方式二 (调用 js 代码):

```
$('#myModal').modal('show'); // 通过js代码显示模态框
```

3.监听模态框的显示和隐藏

```
$('#myModal').on('show.bs.modal', function (e) {
  console.log('模态框显示时回调');
})

$('#myModal').on('hidden.bs.modal', function (e) {
   console.log('模态框隐藏时回调');
})
```

2.Bootstrap小案例

1新建项目

使用bootstrap 与 Jquery 框架技术 ;

其中boostrap 依赖 Jquery

2 BootStrap 页面的简单配置

1.配置： 网页的宽等于设备的宽

2.初始化网页的缩放比例为 1

3 实现导航条组件

注意: 导航条 组件的nav默认的高度是50px 并且 margin-bottom:20px

```
<!--导航条-->
<!--
   data-toggle="collapse"  指定以折叠的方式切换目标
   data-target="#nav_content"  指定要折叠的目标

   data-toggle="modal"     指定以模态框的方式打开目标
   data-toggle="dropdown"  指定以下拉的方式打开目标

   role: 是用来增强语义性，没有什么其它功能，可以删除。
   sr-only 和 aria-xx : 是屏幕阅读辅助，兼容残障人使用的设备-->
<nav class="navbar navbar-default">
   <div class="container-fluid">
      <!-- Brand and toggle get grouped for better mobile display -->
      <!--品牌 和 一个移动端展示的按钮-->
      <div class="navbar-header">
         <button type="button" class="navbar-toggle collapsed" data-toggle="collapse"

               data-target="#nav_content" aria-expanded="false">

            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
         </button>
         <a class="navbar-brand" href="#">
            <img src="./img/header_logo.png" alt="" style="width: 100px">
         </a>
      </div>
      <!--导航条的内容-->
      <!-- Collect the nav links, forms, and other content for toggling -->
      <div class="collapse navbar-collapse" id="nav_content">
         <ul class="nav navbar-nav">
            <li class="active"><a href="#">首页 <span class="sr-only">(current)</span></a>
/li>
            <li><a href="#">蛋糕</a></li>
            <li><a href="#">下午茶</a></li>
            <li><a href="#">精选食材</a></li>
            <li><a href="#">最新活动</a></li>
            <li><a href="#">企业专区</a></li>
            <li><a href="#">会员中心</a></li>

         </ul>
         <ul class="nav navbar-nav navbar-right">
```

```
            <li><a href="#">登录</a></li>
            <li><a href="">注册</a></li>
          </ul>
        </div><!-- /.navbar-collapse -->
    </div><!-- /.container-fluid -->
</nav>
```

4设计导航条样式

- 添加 .navbar-fixed-top 类可以让**导航条固定在浏览器的顶部**

注意:固定的导航条会遮住页面上的其他的内容，可给<body> 设计padding

- 导航条包含一个 .container 或者.container-fluid容器，从而让**导航条居中显示**或者**自适应显示**
- 添加 .navbar-inverse 可以改变**导航栏显示的背景颜色**

5实现 轮播图 （js插件）和 样式


```
<!--
 以下容器就是整个轮播图组件的整体，
 注意该盒子必须加上 class="carousel slide" data-ride="carousel" 表示当前是一个轮播图
 bootstrap.js会自动为当前元素添加图片轮播的特效
-->
<!--广告轮播图-->
<div id="sz-banner" class="carousel slide" data-ride="carousel">
    <!-- 点 -->
    <!-- ol标签是图片轮播的控制点 -->
    <ol class="carousel-indicators">
        <!--
         每一个li就是一个单独的控制点
           data-target属性就是指定当前控制点控制的是哪一个轮播图，其目的是如果界面上有多个轮
图，便于区分到底控制哪一个
           data-slide-to属性是指当前的li元素绑定的是第几个轮播项
         注意，默认必须给其中某个li加上active，展示的时候就是焦点项目
        -->
        <li data-target="#sz-banner" data-slide-to="0" class="active"></li>
        <li data-target="#sz-banner" data-slide-to="1"></li>
        <li data-target="#sz-banner" data-slide-to="2"></li>
        <li data-target="#sz-banner" data-slide-to="3"></li>
    </ol>
```

```
    <!-- 内容 -->
    <!--
    .carousel-inner是所有轮播项的容器盒子，
    注意role="listbox"代表当前div是一个列表盒子，作用就是给当前div添加一个语义
  -->
    <div class="carousel-inner" role="listbox">
        <!-- 每一个.item就是单个轮播项目，注意默认要给第一个轮播项目加上active，表示为焦点 -->
        <div class="item active">
            <!-- 轮播项目中展示的图片 -->
            <img src="./img/banner1.jpg" alt="...">
            <!-- 标题或说明性文字，如果不需要，直接删除当前div.carousel-caption -->
            <div class="carousel-caption">
            </div>
        </div>
        <div class="item">
            <img src="./img/banner2.jpg" alt="...">
            <div class="carousel-caption">
            </div>
        </div>
        <div class="item">
            <img src="./img/banner3.jpg" alt="...">
            <div class="carousel-caption">
            </div>
        </div>
        <div class="item">
            <img src="./img/banner4.jpg" alt="...">
            <div class="carousel-caption">
            </div>
        </div>
    </div>

    <!-- 控制器 上一张 -->
    <!-- 图片轮播上左右两个控制按钮，分别点击可以滚动到上一张和下一张 -->
    <!-- 此处需要注意的是 该a链接的href属性必须指向需要控制的轮播图ID -->
    <!-- 另外a链接中的data-slide="prev"代表点击该链接会滚到上一张，如果设置为next的话则相反
->
    <a class="left carousel-control" href="#sz-banner" role="button" data-slide="prev">
        <span class="glyphicon glyphicon-chevron-left" aria-hidden="true"></span>
        <span class="sr-only">Previous</span>
    </a>
    <!--下一张-->
    <a class="right carousel-control" href="#sz-banner" role="button" data-slide="next">
        <span class="glyphicon glyphicon-chevron-right" aria-hidden="true"></span>
        <span class="sr-only">Next</span>
    </a>
</div>
```

6栅格布局(一)

1.*container 响应式容器类

- 定义一个 **固定宽度**且有居中的版心

- container是具备 **响应式**的能力
- 宽度有四个档位 1170 970 750 100%

主要作用

1. 在随时可能的宽度变化(响应式)中提供宽度限制。当页面宽度变化，container 的宽度也随之变。并且其中的 column 的宽度是基于百分比，所以他们的值不需要变化。
2. 提供一个水平方向的 padding，使其内部的内容不会接触到浏览器的边界，大小为15px

1.原生JS实现container响应式布局原理-resize

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        *{
            margin: 0;
            padding: 0;
        }
        .container{
            height: 30px;
            background-color: red;
            margin:  0 auto;
        }
    </style>
</head>
<body>
    <div class="container">
    </div>
    <!--
        四个档位 1170 970 750 100%
    -->
<script>

    window.onload=function () {
        //1.拿到container标签
        var container = document.querySelector('.container');

        windowChange();

        //2.监听窗口发生改变...
        window.addEventListener('resize',windowChange)

        function windowChange() {
            //3.拿到口的宽度
            var windowWidth=window.innerWidth; //width + padding

            if(windowWidth>=1170){
                container.style.width=1170+'px';
            }else if(windowWidth>=970){
```

```
            container.style.width=970+'px';
        }else if(windowWidth>=750){
            container.style.width=750+'px';
        }else{
            //...
            container.style.width='100%';
        }
    }
  }
</script>
</body>
</html>
```

2.媒体查询实现container响应式布局原理

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        *{
            margin: 0;
            padding: 0;
        }

        .container{
            height: 30px;
            background-color: red;
            margin:  0 auto;
        }

        /*... 750*/
        @media  (max-width: 750px){
            .container{
                width: 100%;
            }

        }

        @media  (min-width: 750px) and (max-width: 970px){
            .container{
                width: 750px;
            }
        }

        @media  (min-width: 970px) and (max-width: 1170px){
            .container{
                width: 970px;
            }
        }

        @media  (min-width: 1170px) {
            .container{
```

```
            width: 1170px;
        }
    }

  </style>
</head>
<body>

  <div class="container">

  </div>

  <!--
    四个档位 1170 970 750 100%
  -->
</body>
</html>
```

2.row类

row类代表一行，一般都是与container类一起结合使用。

并且在栅格系统中，所有“列（column）必须放在 ” .row 内;

总结:

container 左右padding 15px ; row 左右margin左右-15px , 为什么要这么折腾呢?

1.row中嵌套的是column, column 左右padding 15px ;

2.在进行多层嵌套时，column可以充当container容器，省了一层嵌套。

附加注意： column 除了左右有 padding 15px 默认还有一个相对定位。

3.栅shan格系统

所有“列（column）必须放在 ” .row 内; 并且.row一般放在.container内

Bootstrap中的栅格系统就是将一行划分为12列
我们通过col-*-*的类名控制某个元素在某种屏幕的情况下展示几列
col-md-6 在中等屏幕下 占1/2
col-xs-12 在超小屏幕下 占全部

4.col-** -* *类

在某种屏幕尺寸下控制列的占比

col-xs-[列数]：在超小屏幕下展示几份
col-sm-[列数]：在小屏幕下展示几份
col-md-[列数]：在中等屏幕下展示几份
col-lg-[列数]：在大屏幕下展示几份
__xs__ : 超小屏幕 手机 (<768px)

__sm__ : 小屏幕 平板 (≥768px)
__md__ : 中等屏幕 桌面显示器 (≥992px)
__lg__ : 大屏幕 大桌面显示器 (≥1200px)

该布局代码：

```
<!--中间的栅栏布局-->
<div id="main">
    <div class="container-fluid">
        <div class="row">
            <div class="col-md-4">.col-md-4</div>
            <div class="col-md-4">.col-md-4</div>
            <div class="col-md-4">.col-md-4</div>
        </div>
    </div>
</div>
```

7栅格布局（二）

1.每一个item布局的代码：

```
<!--栅格布局-->
<div id="main">
  <div class="container-fluid">
    <div class="row">
      <div class="col-md-4 col-sm-6">
        <img src="./img/cake1.jpg" alt="">
        <h3>90蛋糕</h3>
        <button class="btn btn-success">立即购买</button>
      </div>
      <div class="col-md-4 col-sm-6">
        <img src="./img/cake2.jpg" alt="">
        <h3>樱桃芝士蛋糕</h3>
        <button class="btn btn-success">立即购买</button>
      </div>
      <div class="col-md-4 col-sm-6">
        <img src="./img/cake3.jpg" alt="">
        <h3>牛油果低脂蛋糕</h3>
        <button class="btn btn-success">立即购买</button>
      </div>
    </div>
```

```
  </div>
</div>
```

8栅格布局样式

```
body {
    padding-top: 50px;
}

#main {
    padding: 20px;
}

/*水平居中*/
#main .container-fluid .row .col-md-4 {
    text-align: center;
}

/*图片大小*/
#main .container-fluid .row .col-md-4 img {
    width: 180px;
}

#main .container-fluid .row .col-md-4 h3 {
    font-size: 15px;
    font-weight: bolder;
}
```

9标签添栅格布局

该布局的代码：

```
<!--底部的tabBar-->
    <div id="tabBar">
        <!-- Nav tabs -->
        <!-- Tab panes -->
        <div class="tab-content">
            <div role="tabpanel" class="tab-pane active" id="nine">
                <div class="row">
                    <div class="col-md-8">.col-md-8</div>
                    <div class="col-md-4">.col-md-4</div>
```

```
            </div>
        </div>
        <div role="tabpanel" class="tab-pane" id="Cherry">
            <div class="row">
                <div class="col-md-4">.col-md-4</div>
                <div class="col-md-8">.col-md-8</div>
            </div>
        </div>
        <div role="tabpanel" class="tab-pane" id="Avocado">
            <div class="row">
                <div class="col-md-8">.col-md-8</div>
                <div class="col-md-4">.col-md-4</div>
            </div>
        </div>
    </div>
</div>
```

10标签栅格布局的完善

该布局对应的代码：

```
<div id="tabBar">
<!-- 标签选项卡 -->
<!-- data-toggle="tab" ： 面板切换的方式。 （modal , tab , dropdown ,collapse..）-->
<ul class="nav nav-tabs" role="tablist">
    <li role="presentation" class="active"><a href="#nine" aria-controls="home" role="tab" data-toggle="tab">70蛋糕</a></li>
    <li role="presentation"><a href="#Cherry" aria-controls="profile" role="tab" data-toggle="tab">樱桃芝士蛋糕</a></li>
    <li role="presentation"><a href="#Avocado" aria-controls="messages" role="tab" data-toggle="tab">牛油果低脂蛋糕</a></li>
</ul>
<!-- 标签面板 -->
<div class="tab-content">
    <div role="tabpanel" class="tab-pane fade in active" id="nine">
        <div class="row">
            <div class="col-md-8">
                <h2>回归|70蛋糕</h2>
                <p class="lead" >收起散落在路上的珍贵点滴，每个年代都有属于自己的童年记忆，
                    70的淡定、80的率性、90的不羁??
                    思绪总是在不曾提起的故事里静静流淌，掠过往事，找寻简单的快乐。</p>
            </div>
            <div class="col-md-4">
                <img class="img-responsive" src="./img/cake1.jpg" alt="">
            </div>
        </div>
    </div>
    <div role="tabpanel" class="tab-pane fade" id="Cherry">
        <div class="row">
            <div class="col-md-4">
                <img class="img-responsive" src="./img/cake2.jpg" alt="">
            </div>
            <div class="col-md-8">
                <h2>桃芝|樱桃芝士蛋糕</h2>
```

```
                <p class="lead">有种美好本在人生际遇之处，
                    那些初次的余香，俨如口中混合着的丝缕酸甜，
                    就像樱桃和百利甜一样，成了没人遗忘的天生一对。</p>
            </div>

        </div>
    </div>
    <div role="tabpanel" class="tab-pane fade" id="Avocado">
        <div class="row">
            <div class="col-md-8">
                <h2>初心|牛油果低脂蛋糕</h2>
                <p class="lead">牛油果的清香弥漫在口腔，和着乳清奶酪咀嚼，
                    仿佛热带的阳光，洒满心房，也惊叹牛油果的神奇，
                    包容所有的味道，升华所有的味觉，伟大的感动，总在不言中。</p>
            </div>
            <div class="col-md-4">
                <img class="img-responsive" src="./img/cake3.jpg" alt="">
            </div>
        </div>
    </div>
</div>
</div>
```

11标签栅格布局的样式

1.设计字体样式

```
/*图片距离顶部*/
#tabBar .tab-content .tab-pane .row .col-md-4 {
    margin-top: 50px;
}

/*字体距离顶部*/
#tabBar .tab-content .tab-pane .row .col-md-8 {
    margin-top: 100px;
}
/*h2 字体颜色*/
#tabBar .tab-content .tab-pane .row .col-md-8 h2{
    color:  red;
}
```

2.自适应布局

```
/*自适应布局--> 其中screen and 可以省略*/
@media screen and (max-width: 650px) {
    /**字体距离顶部*/
    #tabBar .tab-content .tab-pane .row .col-md-8 {
        margin-top: 40px;
    }
}
```

12响应式小工具

```
<div class="topbar hidden-xs hidden-sm"></div>
// 或者
<div class="topbar visible-md visible-lg"></div>
```

hidden-xx : 在某种屏幕下隐藏(display : ‘none’)

visible-xx-xxx : 在某种屏幕尺寸下显示

visible-md-xx：指的是中等屏幕可见，不是中等屏幕以上

visible-md-block visible-lg-block == hidden-sm hidden-xs

1. *探究-响应式工具hidden

根据hidden-xxx的特点，我们可以通过**媒体查询**的方式探究探究：

```
<!--
.hidden-xs  < 768
.hidden-sm  [768, 992)
.hidden-md  [992, 1200)
.hidden-lg  >= 1200
-->

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        .container{
            width: 300px;
            height: 300px;
            background-color: red;
            margin: 20px auto;
        }
        //screen and 是可以省略的
        @media screen and (max-width: 768px){
            .hidden-xs{
                display: none;
            }
        }

        @media screen and (min-width: 768px) and (max-width: 992px){
            .hidden-sm{
                display: none;
            }
        }

        @media screen and (min-width: 992px) and (max-width: 1200px){
            .hidden-md{
                display: none;
            }
        }
```

```
        @media screen and (min-width: 1200px){
            .hidden-lg{
                display: none;
            }
        }
    </style>
</head>

<!--
.hidden-xs   < 768
.hidden-sm   [768, 992)
.hidden-md   [992, 1200)
.hidden-lg   >= 1200
-->

<body>
    <div class="container hidden-sm ">
    </div>
</body>
</html>
```

2 *探究-响应式工具visible

2.根据visible-xxx的特点，我们可以通过**媒体查询**的方式探究探究：

```
<!--
    .visible-xs-block
    .visible-xs-inline
    .visible-xs-inline-block
-->

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        .container{
            width: 300px;
            height: 300px;
            background-color: red;
            margin: 20px auto;
            display: none;
        }

        @media screen and (max-width: 768px){
            .visible-xs-block{
                display: block;
            }
            .visible-xs-inline{
                display: inline;
            }
            .visible-xs-inline-block{
                display: inline-block;
```

```
            }
        }

        @media screen and (min-width: 768px) and (max-width: 992px){
            .visible-sm-block{
                display: block;
            }
            .visible-sm-inline{
                display: inline;
            }
            .visible-sm-inline-block{
                display: inline-block;
            }
        }

        @media screen and (min-width: 992px) and (max-width: 1200px){
            .visible-md-block{
                display: block;
            }
            .visible-md-inline{
                display: inline;
            }
            .visible-md-inline-block{
                display: inline-block;
            }
        }

        @media screen and (min-width: 1200px){
            .visible-lg-block{
                display: block;
            }
            .visible-lg-inline{
                display: inline;
            }
            .visible-lg-inline-block{
                display: inline-block;
            }
        }
    </style>
</head>
<!--
    .visible-xs-block
    .visible-xs-inline
    .visible-xs-inline-block
-->

<body>
    <div class="container visible-xs-block">

    </div>
    <p>我是p</p>
</body>
</html>
```