

# CSS教程

chengbenchao



# 目 录

## A.css工程化

### 1.课程介绍

### 2.css基础

#### 2.1非布局样式

#### 2.2使用自定义的字体

#### 2.3行高

#### 2.4背景

#### 2.5边框

#### 2.6overflow

#### 2.7文字折行

#### 2.8文字的装饰属性

#### 2.9cssHack

#### 2.10美化checkbox

#### 2.11css面试

### 3.css进阶

#### 3.1css布局

##### 3.1.1table布局

##### 3.1.2flexbox

##### 3.1.3float

##### 3.1.4inline-block布局

##### 3.1.5响应式布局

##### 3.1.6@media

##### 3.1.7面试题

#### 3.2css效果

##### 3.2.1transform3d

#### 3.3动画

### 4.预处理器

#### 4.1less

#### 4.2sass

##### 第一节 变量和嵌套

##### 第二节 @mixin,%placeholder

##### 第三节 插值#{}

##### 第四节 sass的控制命令

#### 4.3css预处理器框架

#### 4.4面试

### 5.bootstrap

#### 示例

#### bootstrap3模板

#### 第1章 起步

1-1带悬浮高亮的表格

1-2图像

1-3图标

## 第2章 表单

2.1垂直表单

2.2水平表单

2.3内联表单

2.4表单控件input

2.4.1select下拉框

2.4.2textarea

2.4.3checkbox/radio

2.4.4check/radio水平排列

2.4.5按钮

2.4.6表单控件大小

2.4.7表单控件状态(验证)

2.4.8表单提示信息

2.4.9有图标的表单

## 第3章 响应式布局

3-1实现原理

3-2列偏移排序

## 第8章 JS组件

1.modal模态框

## 第4章 菜单,按钮,导航

5-1下来菜单

5-2下拉(分割线)

5-3下拉(菜单标题)

5-4dropup上弹菜单

5-5按钮(按钮组,工具栏)

5-6按钮(嵌套按钮组-下拉)

5-7按钮(垂直分组)

## 第5章 导航

6-1tabs

6-2垂直堆叠导航

6-3自适应导航

6-4下拉导航

6-5breadcrumb导航

## 第6章 导航条

6-1为导航加标题

6-2带表单的导航

6-3固定导航

6-4响应式导航

## 6.三大框架中的css

## 7.css3

### 第一章 3d效果

## B.进阶教程

### 1.flex教程

### 2.css-@import

### 3.grid布局

### 4.bootstrap栅格实现原理

### 5.选择器 倍数写法

## C.sass进阶教程

### 第一节 Sass的函数功能-字符串与数字函数

### 第二节 列表函数

### 第三节 Introspection函数

### 第四节 map

### 第五节 颜色函数

# A.css工程化

## cssTutorial

本教程是关于css工程化的教程

本质上沿用编程思想：代码复用,可维护性

如何在本地查看本教程

```
npm install gitbook-cli -g
gitbook -V
gitbook serve
```

# 1.课程介绍

#A课程概要

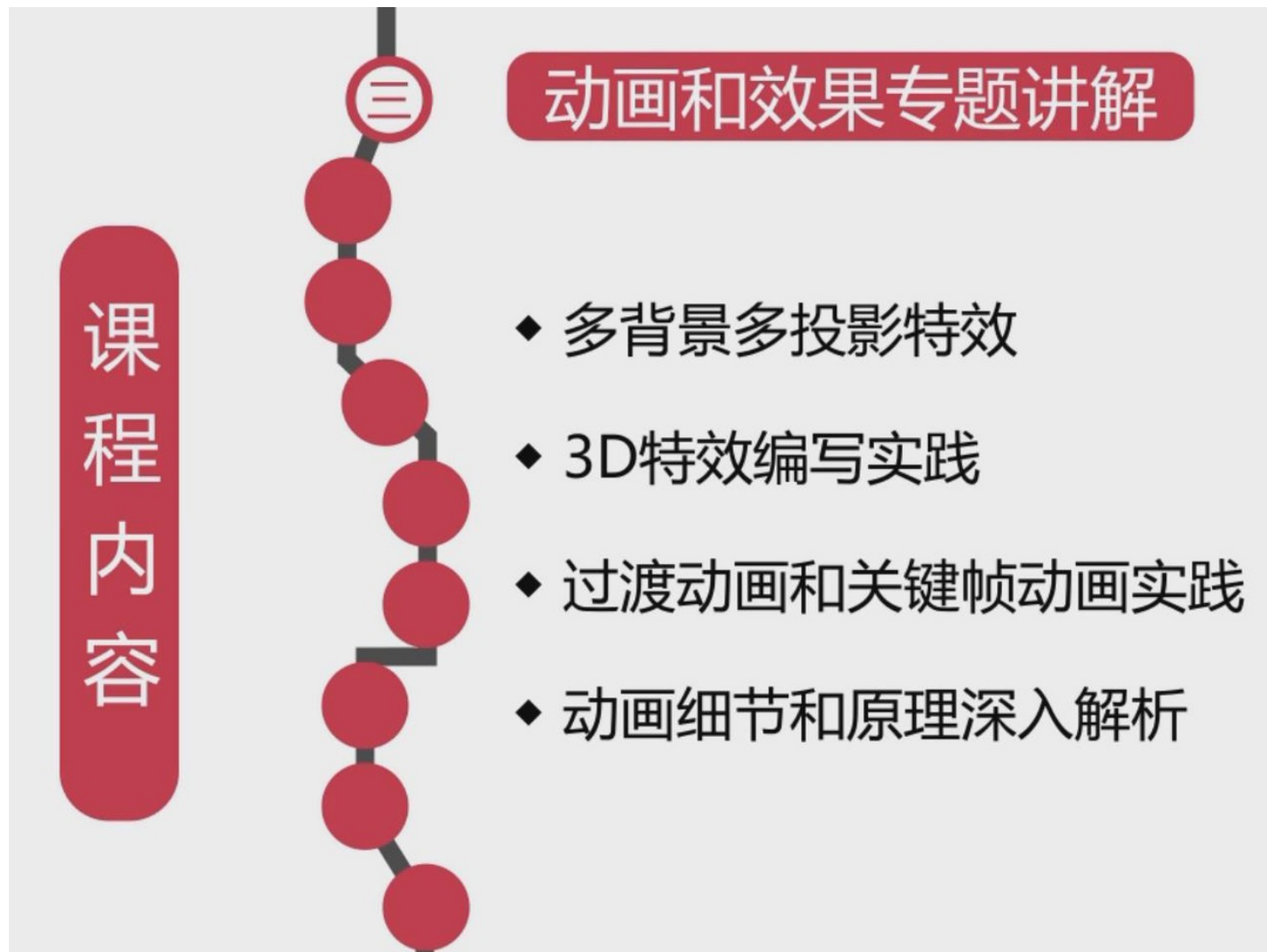
###一、HTML和CSS基础知识



###二、css布局实战



###三、动画

A graphic for course content. On the left, a vertical red rounded rectangle contains the text '课程内容' (Course Content) in white. To its right is a vertical line of red circles connected by a grey line, with a red circle containing a white '三' (Three) at the top. To the right of this line is a red rounded rectangle containing the text '动画和效果专题讲解' (Animation and Effects Special Topic Lecture) in white. Below this, a list of four topics is shown, each preceded by a red diamond symbol.

课程内容

动画和效果专题讲解

- ◆ 多背景多投影特效
- ◆ 3D特效编写实践
- ◆ 过渡动画和关键帧动画实践
- ◆ 动画细节和原理深入解析

###四、css工程化



## 课程内容

四

### 框架集成和CSS工程化

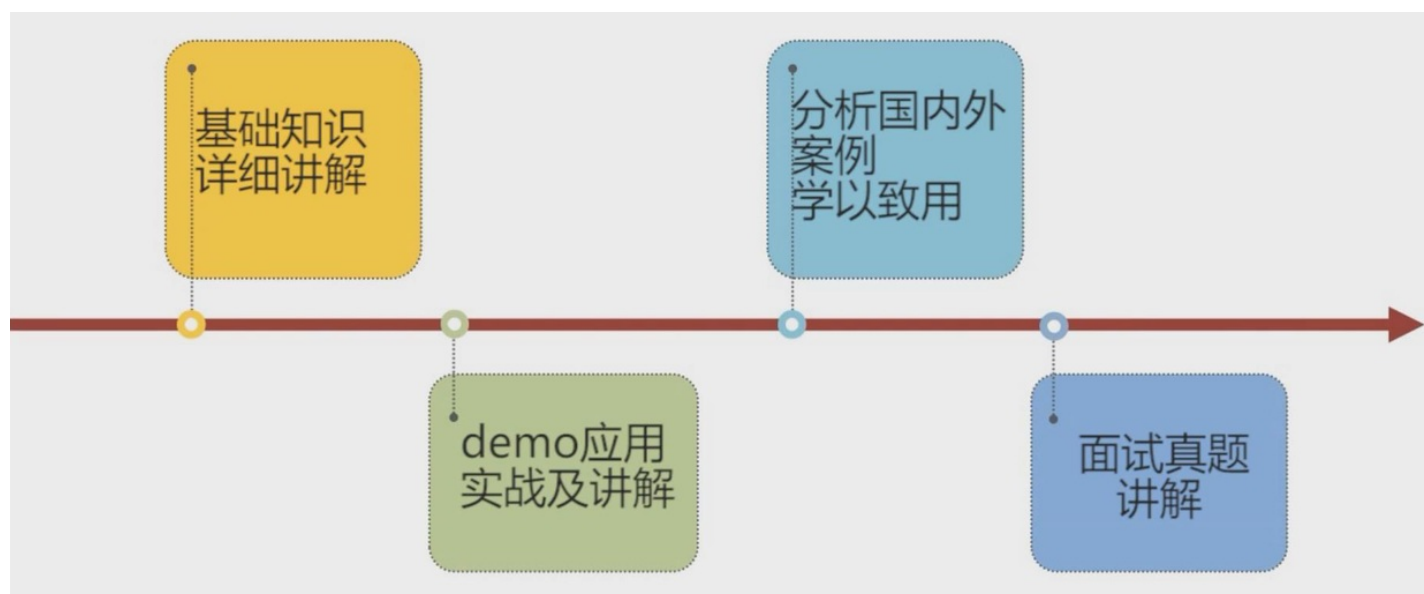
- ◆ 预处理器作用和原理
- ◆ Less/Sass代码实践
- ◆ Bootstrap原理和用法
- ◆ CSS工程化实践方式
- ◆ JS框架中的CSS集成实践

#B.课程目标

## 课程目标

- ◆ 学会布局、特效、动画 轻松应对日常前端工作
- ◆ 全面深入的学习CSS体系 掌握原理 举一反三
- ◆ 会用现代化CSS工具 学习CSS代码维护和架构
- ◆ 全面覆盖CSS面试相关问题

### #C.课程流程



### #D.面向人群



- ◆ web前端初级、中级工程师
- ◆ 重构工程师（UI开发工程师）
- ◆ web前端初学者、爱好者

#E.预备知识



## 2.css基础

---

css非布局的一些样式

## 2.1非布局样式

---

### 非布局样式

- 字体，字重，行高，颜色，大小
- 背景，边框
- 滚动，换行
- 粗体，斜体，下划线

### 1.字体的预备机制

---

```
//简书方案
body{
    font-family:-apple-system,SF UI Text,Arial,PingFang SC,Hiragino Sans GB,Mic
rosoft YaHei,
    WenQuanYi Micro Hei,sans-serif;
}
```



实际使用的，看调试工具使用的是微软雅黑



## 2.2使用自定义的字体

```
<p class="custom-font">hello world</p>
```

```
//css
.custom-font{
  font-family:"IF"
}
@font-face{
  font-family:"IF";
  src:url("./path")
}
```

倘若字体文件过大怎么解决

### 1.安装font-spider

```
npm install font-spider -g
```

### 2.使用font-spider

```
font-spider *.html
```

## 2.3行高

```
<div>
  <span class="c1">hello world</span>
</div>
```

```
//css
.c1{
  line-height: 60px;
}
span{
  background: red;
}
div{
  border: 1px solid #333;
}
```

### 1.内联元素在块元素中垂直居中,使用\_line-height\_

显示结果



### 2.vertical-align设置文字的垂直对齐属性

```
vertical-align:baseline
//默认基线对齐
```

eg:

```
//css
span{
  background: red;
}
.c1{
  font-size: 20px;
}
.c2{
  font-size: 30px;
}
.c3{
  font-size: 40px;
}
```

## 2.3行高

```
//html
<div>
  <span class="c1">第1</span>
  <span class="c2">第2</span>
  <span class="c3">第3</span>
</div>
```



如何改为居中对齐,以下代码

```
span{vertical-align:middle}
```



## 3.图片与文字会有空隙 原理是什么

```
<div>
  <span>美女</span>
  
</div>
//css
div{background:pink}
```



原因：将图片识别为文字，默认是基线对齐的，所以会有3px左右的偏差。

如何解决：将图片设为基线对齐

```
img{vertical-align:bottom}
```





## 2.4背景

- 背景颜色
- 渐变色背景

<https://webgradients.com/>

- 多背景叠加
- 背景图片和属性(雪碧图)
- base64和性能优化
- 多分辨率适配

### 1.纯的背景色

- 1.1 hsla

```
//色相, 饱和度, 明度, 透明度  
background:hsla(230,50%,50%,.6);
```

- 1.2rgba

```
background:rgba(20,30,243,.5)
```

- 1.3背景的简写

```
background:color img repeat position;
```

### 2.雪碧图

```
background-position:x y  
//去调整图片的位置
```

### 3.base64 一串文本

好处:减少http连接数

坏处:

- 1.文件的体积会增大 会增大1/3
- 2.解码难度增大

适用场合:只适合小图片

## 2.4背景

在项目中通过构建的方式,转换为base64

## 2.5边框

### 1.边框的简写

```
border:width style color
```

### 2.边框的连接(三角形)

```
//html
<div></div>
//css
div{
  width:0;
  height:0;
  border:15px solid transparent;
  border-top-color:#333;
}
```

结果如下图所示



## 2.overflow

---

1.overflow:hidden

溢出隐藏

2.overflow:auto;

//溢出部分滚动



3.overflow:scroll

//不管如何滚动条都会出现



## 2.7文字折行

- overflow-wrap(word-wrap)对不可分割的长单词换行

值	描述
normal	只在允许的断字点换行（浏览器保持默认处理）。
break-word	在长单词或 URL 地址内部进行换行。

- word-break属性规定自动换行的处理方法

```
word-break: normal | break-all | keep-all;
```

值	描述
normal	使用浏览器默认的换行规则
break-all	允许在单词里换行
keep-all	只能在词与词的空格之间换行

- white-space 内容是否换行

```
white-space: nowrap;
```

### 文字以省略号结尾

```
//超出内容隐藏
overflow: hidden;
//设置内容不换行
white-space: nowrap;
//溢出部分以省略号结尾
text-overflow: ellipsis;
```

# 2.8文字的装饰属性

- 字重font-weight

值	描述
normal	默认。定义标准的字符。
bold	定义粗体字符。
bolder	定义更粗的字符。
lighter	定义更细的字符。
100-200-300-400-500-600-700-800-900	定义由粗到细的字符。400 等同于 normal，而 700 等同于 bold。

- 斜体font-style:italic
- 下划线text-decoration:
- 指针cursor

# 2.9cssHack

---

兼容IE7,8

## CSS Hack

---

- Hack即不合法但生效的写法
- 主要用于区分不同浏览器
- 缺点：难理解 难维护 易失效
- 替代方案：特性检测
- 替代方案：针对性甲class

<https://www.jianshu.com/p/682a7776be84>

## 2.10美化checkbox

```
//html
<div class="checkbox">
  <input type="checkbox" id="c">
  <label for="c">我很牛</label>
</div>
```

```
//css
input{
  display: none;
}
.checkbox label{
  background: url("off.png") no-repeat;
  padding-left: 20px;
}
.checkbox input:checked+label{
  background: url("on.png") no-repeat;
}
```

### 在线实例

点击checkbox在这两张图片之间切换





## 2.11css面试

### 1.css选择器的优先级别排序

### 2.雪碧图的作用

- 原理：运用background-position
- 优势：减少HTTP请求数,提高加载性能

### 3.自定义字体的使用场景

- 宣传/品牌/banner等固定文案
- 字体图标

```
npm install font-spider -g
```

### 4.base64使用

- 原理：将图片变为文本,内嵌到css中使用
- 优势：

1.减少http请求,增加加载性能

- 应用场景:小图片
- 劣势：体积会增大1/3左右

### 5.伪类和伪元素的区别

- 伪类表示状态

eg:

```
p:hover{}  
//表示鼠标悬停
```

- 伪元素是真的元素

### 6.如何美化checkbox

- label[for]和id
- 隐藏原生input
- :checked+label

# 3.css进阶

---

[3.1css布局](#)

[3.2css效果](#)

[3.3动画](#)

# 3.1css布局

---

## css布局

- css知识体系的核心

## css布局的发展

- 早期以table为主(简单)  
劣势：表格解析不是像水流一样的流式加载,它是等整个table表格加载完了,再进行渲染。
- 后期技巧性布局div+css(难)
- 现在有flexbox/grid(简单)
- 响应式布局

## 常用的布局方法

- table表格布局 07年之前使用广泛,可以说是唯一的布局方式
- float浮动+margin(经典)
- inline-block布局——(有小问题)
- flexbox布局(正统的布局方式)

## 3.1.1table布局

table-cell	元素会作为表格单元格显示(类似td,th)
table-row	元素会作为一个表格行显示(类似tr)

```
<table>
  <tr>
    <td class="left">left</td>
    <td class="right">right</td>
  </tr>
</table>
```

```
//css
table{width:600px;height:300px;border-collapse:collapse}
.left{background:red}
.right{background:yellow}
```

显示如下



如何用div达到以上效果

```
//html
<div class="table">
  <div class="table-row">
    <div class="left table-cell">
      left
    </div>
    <div class="right table-cell">
      right
    </div>
  </div>
</div>
```

```
//CSS
.table{
  display: table;
  width: 800px;
  height: 200px;
}
.table-row{
  display: table-row;
}
.table-cell{
```

### 3.1.1table布局


```
    display: table-cell;
    vertical-align: middle;
}
.left{
    background: red;
}
.right{
    background: yellow;
}
```

## 3.1.2flexbox



```
<div class="container">
  <div class="flex">flex</div>
  <div class="flex">flex</div>
  <div class="flex">flex</div>
  <div class="flex">flex</div>
</div>
```

```
//css
.container{
    width:800px;
    height:200px;
    border:1px solid #333;
    display: flex;
}
.flex{
    flex:1;
    background:red;
    margin:10px;
}
```

//container里的四个div被分成四等分,如下图所示 

### 2.左边给固定宽度,右边自适应

```
<div class="container">
  <div class="left"></div>
  <div class="right"></div>
</div>
```

```
.container{
    display:flex;
    width:800px;
    height:200px;
    border:1px solid #333;
}
.left{
    display:flex;
    width:200px;
    background-color: red;
}
```

### 3.1.2flexbox

```
.right{  
  background:yellow;  
  flex:1;  
}
```



## 3.1.3float

---

float诞生的目的就是为了作图文混排的

<https://gitee.com/frontendLol/cssSkill/tree/master/float>图文混排



## 3.1.4inline-block布局

```
<div class="container">
  <div class="left"></div>
  <div class="right"></div>
</div>
```

Tip:将left,right设置为inline-block时,他们之间会有间隙,浏览器把它们当文字在处理,因为字体之间的空隙导致的空白;只要将container的font-size设置为0,就可以解决。

```
.container{
  font-size: 0;
  width:800px;
  border:1px solid #333;
}
.left{
  display: inline-block;
  width:300px;
  background:red;
  height:200px;
}
.right{
  display:inline-block;
  width:500px;
  background:yellow;
  height:200px;
}
```

## 3.1.5响应式布局

---

### 响应式设计和布局

- 在不同设备上正常使用
- 一般主要处理屏幕大小问题
- 主要方法:
  - 1.隐藏+折行+自适应空间
  - 2.具体方法:rem/viewport/media query

```
@media (max-width:640px){  
    .left{  
        display:none;  
    }  
}
```

<https://gitee.com/frontendLol/cssSkill/tree/master/@media>让对应的页面隐藏

## 3.1.6@media

@media可以针对[不同的媒体类型](#),定义不同的样式



```
@media only screen and (max-width: 500px) {  
    /*css code*/  
}  
  
@media screen and (min-width:300px) and (max-width:500px) {  
    /* CSS 代码 */  
}
```

max-width	定义输出设备页面最大的可见区域宽度
max-height	定义输出设备页面最大的可见区域高度

## 3.1.7面试题

### 1.常用的布局方式

- 1.表格布局
- 2.float+margin布局
- 3.inline-block布局(将父元素的font-size设置为0)
- 4.flexbox布局

### 2.position:absolute/fixed有什么区别?

- 前者相对最近的absolute/relative的父元素
- 后者相对屏幕(viewport)

### 3.display:inline-block的间隙

- 原因:字符间距
- 解决方案:

1. 消灭字符标签与标签之间紧挨着 eg<div>hello</div><div>hello</div>
2. 消灭间隙      父元素的font-size设置为0

### 4.如何清除浮动

- overflow:hidden;
- 伪元素

```
.row:after{
  content:"";
  display:table;
  clear:both;
}
```

### 5.如何适配移动端的页面

- viewport
- rem/viewport/media
- 设计上:隐藏 折行 自适应

## 3.2css效果

- box-shadow
- text-shadow
- border-radius
- background
- clip-path

### 1.box-shadow

```
box-shadow:offsetX offsetY blur size color inset
```

### 2.text-shadow

```
text-shadow:offsetX offsetY size color
```

```
p{  
    text-indent: 30px;  
    text-shadow: 1px 1px 2px #333;  
    font-family: STKaiti  
}
```

### 3.border-radius

```
border-radius:topLeft topRight bottomRight bottomLeft
```

```
<div></div>  
//css  
div{  
    width:200px;  
    height:30px;  
    border-bottom:1px solid #333;  
    border-radius: 10px 10px 30px 20px;  
}
```



### 4.clip-path

- clip-path:inset(x,y)裁剪矩形

```
//x,y 表示横坐标,和纵坐标,裁剪的位置  
clip-path: inset(x,y)
```

- clip-path:circle(radius at x y) 裁剪圆形

```
//radius表示半径 x,y表示裁剪的坐标
```

## 3.2.1transform3d

### 本节参考张鑫旭

#### 1.perspective透视

- CSS3 3D transform的透视点是在浏览器的前方

实现3d效果有两种写法

```
<div class="parent">
  <div class="child">
    </div>
  </div>
```

```
//1.给父元素透视
.parent{
//保留3d效果
transform-style:preserve-3d
perspective:300px;
}
```

```
//2.用在当前动画元素上
.child{
  transform: perspective(600px) rotateY(45deg);
}
```

#### 2.3D transform中有下面这三个方法：

- `rotateX( angle )`
- `rotateY( angle )`
- `rotateZ( angle )`
- 这三个样式可以帮助理解三位坐标

单杠运动是 `rotateX`



钢管舞 `rotateY`



转盘`rotateZ`



3.translateZ帮你寻找透视位置

近大远小,translateZ()的值设置的越小,视觉上里我们就越远

在线demo

<https://gitee.com/frontendLol/cssJiaoCheng/blob/master/04translateZ.html>



## 3.3动画

动画的原理：

- 1.视觉暂留作用
- 2.画面逐渐变化

动画的作用-愉悦感

css中的动画类型

- 1.transition补间动画——从一种状态变成另一种状态
- 2.keyframe关键帧动画
- 3.逐帧动画

### 1.transition

```
//transition指定多个属性
transition: width 2s,height 3s;
```

transition-timing-function 属性规定过渡效果的速度曲线。

```
transition-timing-function: linear|ease|ease-in|ease-out|ease-in-out
```

### 2.animation

animation-fill-mode:属性规定动画在播放之前或之后，其动画效果是否可见

animation-timing-function:指定时间和动画进度之间的关系

```
animation-timing-function: steps(1)
//steps(n)指定关键帧之间有几个动画
```

github上好用的动画库:

<https://daneden.github.io/animate.css/>

animation demo:

<https://gitee.com/frontendLol/cssJiaoCheng/blob/master/05-animal.html>

### 3.面试

- css动画的实现方式有几种

```
transition
```

`keyframes(animation)`

- 如何实现逐帧动画

1. 使用关键帧动画
2. 去掉补间(steps)

- css动画性能

1. 性能不坏
2. 部分情况下优于js
3. 但js可以做的更好
4. 部分高危属性`box-shadow`等

## 4.预处理器

---

- less——基于node

优势：  
js写的，编译快

- sass——基于ruby

### 1.预处理器的优势

---

- 嵌套——反映层级和约束
- 变量和计算 减少重复代码
- Extend和Mixin代码片段
- 循环 适用于复杂又规律的样式
- import CSS文件模块化--可以将头尾样式分拆

# 4.1less

less的设计是尽量做到和原生的css相同,无论变量的设计还是mixin的设计

## 1.嵌套

```
div{
  background:red;
  .content{
    width:100px;
    height:100px;
    //&表示.content 同时的意义
    &:hover{
      height:200px;
    }
  }
}
```

```
//编译为
div {
  background: "red";
}
div .content {
  width: 100px;
  height: 100px;
}
div .content:hover {
  height: 200px;
}
```

## 2.变量

优势:可以避免写相同的值,可以参与计算

```
@fontSize:12px;
@bgColor:red;
div{
  background:lighten(@bgColor,40%);
  fontsize:@fontSize+2px;
}
```

实际开始可以将一些常用的样式先定义好 eg:

```
@headFontSize:16px;
```

```
@contentFs:14px;
@textColor:#333;
@linkColor:yellow;
```

### 3.mixin ——大段代码复用

```
@fontSize:12px;
//定义一段复用的代码
.font(@fontSize){
    border:1px solid red;
    font-size: @fontSize;
}
.box{width:100px;}
.nav{
    //调用
    .box();
    .font(14px);
}
```

### 4.extend

```
.block{
    border:1px solid red;
    width:100px;
    height:100px;
}
//第一种方式
.box:extend(.block){};
//第二种方式
.content{
    &:extend(.block);
}
```

### 5.loop

```
.gen-col(@n) when(@n>0){

    .col-@{n}{
        width:100%/12*@n;
    }

    .gen-col(@n - 1);
}
.gen-col(12);
```

4.1less

## 6.import

可以将css拆分成不同的模块,用import去加载对应的css

# 4.2sass

## 1.嵌套

```
div{
  h1{
    width:100px;
    height:100px;
    background:yellow;
    //&表示h1
    &:hover{
      background:red;
    }
  }
}
```

```
//编译后
div h1 {
  width: 100px;
  height: 100px;
  background: yellow;
}

div h1:hover {
  background: red;
}
```

## 2.变量

```
$bg:red;
$fontSize:12px;
```

## 3.mixin

- 1.使用@mixin定义代码块
- 2.使用@include引用代码块

```
$bg:red;
@mixin bg($bg){
  background:$bg;
  line-height:40px;
  text-align: center;
}
.nav{
```

```
@include bg(yellow);
}
```

## 4.extend

```
.block{
  width:100px;
  height:100px;
}
.nav{
  @extend .block;
}
```

## 5.loop

```
@mixin gen-col($n){
  @if $n > 0 {
    @include gen-col($n - 1);
    .col-#{ $n }{
      width:100%/12*$n;
    }
  }
}
@include gen-col(12)
;
```

推荐使用

//sass之处for循环

```
@for $i from 1 through 12 {
  .col-#{ $i }{
    width:100%/12*$i
  }
}
```

## 6.import

可以将css拆分成不同的模块,用import去加载对应的css



# 第一节 变量和嵌套

## 1.变量

```
$bg:red;
div{
  $color:blue;
  border:1px solid $color;
}
```

## 2.嵌套

### 2.1选择器嵌套

```
nav {
  a {
    color: red;
    head & {
      color:yellow;
    }
  }
}
```

```
//编译后
nav a {
  color: red;
}

header nav a {
  color: green;
}
```

### 2.2属性嵌套

```
.box {
  font: {
    size:12px;
    family:"微软雅黑"
  }
}
```

```
.box {  
  font-size: 12px;  
  font-family: "微软雅黑";  
}
```

## 2.3 伪类嵌套

---

```
.row{  
  &::before,&::after{  
    content:"";  
    display: table;  
  }  
  &::after{  
    clear: both;  
  }  
}
```

```
.row::before, .row::after {  
  content: "";  
  display: table;  
}  
  
.row::after {  
  clear: both;  
}
```

## 第二节 @mixin,%placeholder

### 1.@mixin混合宏的参数

#### 1.1 传一个 不带值 的参数

```
@mixin bs($bs){
  border-radius: $bs;
}
div{
  @include bs(3px);
}
```

#### 1.2带一个传值的参数

```
@mixin bs($bs:5px){
  border-radius: $bs;
}
.one{
  @include bs;
}
div{
  @include bs(3px);
}
```

#### 1.3有一个特别的参数 “...” 。当混合宏传的参数过多之时，可以使用参数来替代

```
@mixin box-shadow($shadows...){
  @if length($shadows) >= 1 {
    box-shadow: $shadows;
  } @else {
    $s: 1 0 2px rgba(#000, .25);
    box-shadow: $s;
  }
}
```

## 2.占位符 %placeholder

%placeholder 声明的代码，如果不被 @extend 调用的话，不会产生任何代码

```
%mt5{  
    margin-top: 5px;  
}  
div{  
    @extend %mt5;  
}
```

## 第三节 插值#{}

- 1.使用变量
- 2.使用变量构建选择器
- 3.配合@extend使用

```
$props:(margin,padding);
@mixin set-value($value){
  @each $prop in $props {
    #{$prop}:$value
  }
}
div{
  @include set-value(12px);
}
```

### 1.使用变量

```
@mixin m-v($margin,$v){
  #{$margin}:$v;
}
div{
  @include m-v(top,10px)
}
```

### 2.使用变量构建选择器

```
@mixin g-z($class,$w){
  #{$class}{width:$w};
}
@include g-z(".one",100px )
```

### 3.配合@extend使用

```
%w-h{
  width:100px;
  height:100px;
}
$h:"h";
div{
  @extend %w-#{ $h}
}
```

## 第四节 sass的控制命令

Sass中控制命令指的是@if、@each、@for和@while

### 1.@if--@else

```
@mixin v-h ($b:true){
  @if $b {
    display: block
  }@else{
    display: none;
  }
}
.visible{
  @include v-h;
}
.hide{
  @include v-h(false)
}
```

### 2.@for

```
@for $i from <start> through <end>
@for $i from <start> to <end>
```

- \$i 表示变量
- start 表示起始值
- end 表示结束值

//区别

这两个的区别是关键字 through 表示包括 end 这个数，而 to 则不包括 end 这个数。

### @while

```
$i:12;
@while $i > 0 {
  .col-#{ $i } { width: 100%/12 * $i; }
  $i: $i - 1;
}
```

## @each

---

```
$list:(width,height,margin-top,border);
@mixin whmb{
  @each $key in $list {
    #{ $key}:20px;
  }
}
div{
  @include whmb();
}
```

## 4.3css预处理器框架

- SASS-Compass
- Less - Lesshat/EST
- 提供现成的mixin
- 类似JS类库 封装常用功能

### 一、Compass是什么？

Sass本身只是一个编译器，Compass在它的基础上，封装了一系列有用的模块和模板，补充Sass的功能。它们之间的关系，有点像Javascript和jQuery

### 二、安装

Compass是用Ruby语言开发的，所以安装它之前，必须安装Ruby

```
gem install compass
```

### 三、项目初始化

接下来，要创建一个你的Compass项目，假定它的名字叫做myproject，那么在命令行键入

```
compass create myproject
```

```
cd myproject
```

### 四、编译

在命令行中输入

```
compass compile
```

会将sass子目录中的scss文件，编译成css文件，保存在stylesheets子目录中

```
compass compile --output-style compressed  
//编译出来的css带大量注释,使用折行命令,可以删除注释
```

### 五、Compass的模块

```
* reset  
* css3
```



- \* layout
- \* typography
- \* utilities

## 4.4面试

---

### 1.常见的css预处理器

- less(node.js)
- sass(Ruby)

### 2.预处理的作用

- 更好的帮开发者组织css代码(核心功能:通过变量,mixin复用代码,可维护性增加);

### 3.预处理器的能力

- 嵌套——反映层级和约束
- 变量和计算——减少重复代码
- Extend和Mixin代码片段
- 循环 适用于复杂有规律的样式
- import css文件模块化

# 5.bootstrap

---

## bootstrap

- 一个css框架
- twitter出品
- 提供通用的基础样式

## bootstrap4

- 1.兼容ie10+
- 2.适用flexbox布局
- 3.抛弃Nomalize.css
- 4.提供布局和reboot版本

bootstrap分为三个部分

基础样式——常用组件——js插件

本教程以bootstrap4为模板讲解

# 示例

## ☐ 1.水平表单

```
<h1 class="text-center popover-header">注册</h1>
<form action="" id="myForm" class="col-6 offset-3">
  <div class="row form-group">
    <label for="" class="col-sm-2 col-form-label">用户名</label>
    <div class="col-sm-10">
      <input type="text" class="form-control">
    </div>
  </div>
  <div class="row form-group">
    <label for="" class="col-sm-2 col-form-label">密码</label>
    <div class="col-sm-10">
      <input type="password" class="form-control">
    </div>
  </div>
  <div>
    <button type="submit" class="btn btn-primary ">提交</button>
  </div>
</form>
```

```
form
  div class="row form-group"
    label class="col-sm-2 col-form-label"
    div class="col-sm-10"
      input class="form-control"
```

# bootstrap3模板

```
<!DOCTYPE html>
<html lang="zh-CN">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <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>Bootstrap 101 Template</title>

    <!-- Bootstrap -->
    <link href="https://cdn.bootcss.com/bootstrap/3.3.7/css/bootstrap.min.css"
rel="stylesheet">

    <!-- HTML5 shim 和 Respond.js 是为了让 IE8 支持 HTML5 元素和媒体查询 (media quer
ies) 功能 -->
    <!-- 警告: 通过 file:// 协议 (就是直接将 html 页面拖拽到浏览器中) 访问页面时 Respond.
js 不起作用 -->
    <!--[if lt IE 9]>
      <script src="https://cdn.bootcss.com/html5shiv/3.7.3/html5shiv.min.js"></s
cript>
      <script src="https://cdn.bootcss.com/respond.js/1.4.2/respond.min.js"></s
cript>
    <![endif]-->
  </head>
  <body>
    <h1>你好, 世界! </h1>

    <!-- jQuery (Bootstrap 的所有 JavaScript 插件都依赖 jQuery, 所以必须放在前边) -->

    <script src="https://cdn.bootcss.com/jquery/1.12.4/jquery.min.js"></script>
    <!-- 加载 Bootstrap 的所有 JavaScript 插件。你也可以根据需要只加载单个插件。 -->
    <script src="https://cdn.bootcss.com/bootstrap/3.3.7/js/bootstrap.min.js"></
script>
  </body>
</html>
```

浏览器兼容ie8+

# 第1章 起步

---

## 1.列偏移

```
//向右偏移2个单位  
.col-md-push-2
```

```
//向左偏移  
.col-md-pull-2
```

# 1-1带悬浮高亮的表格

```
<table class="table table-bordered table-hover table-responsive">
  <caption class="text-center">手机品牌</caption>
  <tr class="active">
    <th>手机</th>
    <th>手机</th>
  </tr>
  <tr class="warning">
    <td>小米</td>
    <td>小米</td>
  </tr>
  <tr class="success">
    <td>小米</td>
    <td>小米</td>
  </tr>
  <tr class="danger">
    <td>小米</td>
    <td>小米</td>
  </tr>
</table>
```

```
table class="table table-bordered table-hover"
```



bootstrap为表格提供了以下几种样式

1.table:基础表格

2.table-striped:斑马线表格

3.table-bordered:带边框的表格

4.table-hover:鼠标悬停高亮的表格

5.table-condensed:紧凑型表格

6.table-responsive:响应式表格

# 1-2图像

1、img-responsive :

响应式图片，主要针对于响应式设计

2、img-rounded :

圆角图片

3、img-circle :

圆形图片

4、img-thumbnail :

缩略图片

```
//bootstrap
.img-responsive{
    display:block;
    max-width:100%;
    height:auto;
}
```

```
<div class="row">
    <div class="col-sm-4">
        
    </div>
    <div class="col-sm-4">
        
    </div>
    <div class="col-sm-4">
        
    </div>
</div>
```





# 1-3图标

---

<https://v3.bootcss.com/components/#glyphicons>

bootstrap提供了200多个icon

## 第2章 表单

---

```
form
  div.form-group
    .form-control
```

Tip:给表单元素input,select,textarea增加样式class="form-control"

## 2.1垂直表单



```
<form role="form">
  <div class="form-group">
    <label for="">用户名</label>
    <input type="text" class="form-control">
  </div>
  <div class="form-group">
    <label for="">密码</label>
    <input type="text" class="form-control">
  </div>
  <div class="checkbox">
    <label for="">
      <input type="checkbox">记住密码
    </label>
  </div>
  <button type="submit" class="btn btn-default">提交</button>
</form>
```

## 2.2水平表单

```
<form action="" role="form" class="form-horizontal">
  <div class="form-group">
    <label for="" class="col-xs-2">用户名</label>
    <div class="col-xs-10">
      <input type="text" class="form-control ">
    </div>
  </div>
  <div class="form-group">
    <label for="" class="col-xs-2">密码</label>
    <div class="col-xs-10">
      <input type="text" class="form-control">
    </div>
  </div>
  <div class="form-group">
    <div class="checkbox col-xs-10 col-xs-offset-2">
      <label for="">
        <input type="checkbox">记住密码
      </label>
    </div>
  </div>
  <div class="form-group">
    <div class="col-xs-10 col-xs-offset-2">
      <button type="submit" class="btn btn-default">提交</button>
    </div>
  </div>
</form>
```

```
form class="form-horizontal"
  div class="form-group"
```



## 2.3内联表单



```
<form class="form-inline" role="form">
  <div class="form-group">
    <label class="sr-only" for="exampleInputEmail2">邮箱</label>
    <input type="email" class="form-control" id="exampleInputEmail2"
placeholder="请输入你的邮箱地址">
  </div>
  <div class="form-group">
    <label class="sr-only" for="exampleInputPassword2">密码</label>
    <input type="password" class="form-control" id="exampleInputPas
sword2" placeholder="请输入你的邮箱密码">
  </div>
  <div class="checkbox">
    <label>
      <input type="checkbox"> 记住密码
    </label>
  </div>
  <button type="submit" class="btn btn-default">进入邮箱</button>
</form>
```

```
from class="form-inline"
```

## 2.4表单控件input

---

为了让控件在各种表单风格中样式不出错，需要添加类名 “form-control ”

```
<form role="form">
  <div class="form-group">
    <input type="email" class="form-control" placeholder="Enter email">
  </div>
</form>
```



## 2.4.1 select 下拉框

```
<form >
  <div class="form-group">
    <select class="form-control">
      <option value="武汉">武汉</option>
      <option value="长沙">长沙</option>
      <option value="贵阳">贵阳</option>
    </select>
  </div>
</form>
```

```
form
  div class="form-group"
    select class="form-control"
```



```
<form action="" >
  <div class="form-group">
    <select class="form-control" multiple>
      <option value="武汉">武汉</option>
      <option value="长沙">长沙</option>
      <option value="贵阳">贵阳</option>
    </select>
  </div>
</form>
```

显示如下



## 2.4.2 textarea

```
<form action="">
  <div class="form-group">
    <textarea class="form-control" rows="5"></textarea>
  </div>
</form>
```

```
form
  div.form-group
    textarea.form-control
```





## 2.4.3checkbox/radio

```
<form action="">

  <div class="checkbox">
    <label>
      <input type="checkbox">记住密码
    </label>
  </div>

  <div class="radio">
    <label >
      <input type="radio" name="love">喜欢
    </label>
  </div>
  <div class="radio">
    <label >
      <input type="radio" name="love">不喜欢
    </label>
  </div>

</form>
```



## 2.4.4 check/radio 水平排列

```
<form action="">
  <div class="form-group">
    <label class="checkbox-inline">
      <input type="checkbox">篮球
    </label>
    <label class="checkbox-inline">
      <input type="checkbox">足球
    </label>
    <label class="checkbox-inline">
      <input type="checkbox">羽毛球
    </label>
  </div>
</form>
```

```
form
  div.form-group
    label.checkbox-inline
```



## 2.4.5 按钮

```
<button class="btn btn-default">default</button>
<button class="btn btn-info">info</button>
<button class="btn btn-primary">primary</button>
<button class="btn btn-warning">warn</button>
<button class="btn btn-success">success</button>
<button class="btn btn-danger">danger</button>
<button class="btn btn-inverse">inverse</button>
```



```
<button class="btn btn-danger" disabled>提交</button>
```



## 2.4.6 表单控件大小

- 1、input-sm: 让控件比正常大小更小
- 2、input-lg: 让控件比正常大小更大

```
<input type="text" class="form-control input-lg">  
<input type="text" class="form-control input-sm">
```



## 2.4.7 表单控件状态(验证)

```
<form role="form">
  <div class="form-group has-success">
    <label class="control-label" for="inputSuccess1">成功状态</label>

    <input type="text" class="form-control" id="inputSuccess1" placeholder="成功状态">
  </div>
  <div class="form-group has-warning">
    <label class="control-label" for="inputWarning1">警告状态</label>

    <input type="text" class="form-control" id="inputWarning1" placeholder="警告状态">
  </div>
  <div class="form-group has-error">
    <label class="control-label" for="inputError1">错误状态</label>
    <input type="text" class="form-control" id="inputError1" placeholder="错误状态">
  </div>
</form>
```

```
form
  div class="form-group has-success"
    label class="control-label"
    input class="form-control"
```



<https://v3.bootcss.com/components/#glyphicons-glyphs>

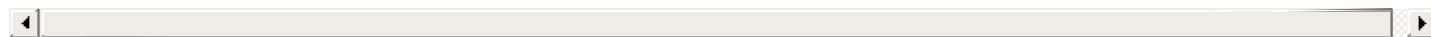
带图片的状态框,要将fonts文件夹放到项目中



```
<form role="form">
  <div class="form-group has-success has-feedback">
    <label class="control-label" for="inputSuccess1">成功状态</label>

    <input type="text" class="form-control" id="inputSuccess1" placeholder="成功状态">
    <span class="glyphicon glyphicon-ok form-control-feedback"></span>
  </div>

</form>
```



form

```
div class="form-group has-sucess has-feedback"
  label class="control-label"
  input class="form-control"
  span class="form-control-feedback"
```

## 2.4.8 表单提示信息

```
<form role="form">
  <div class="form-group has-success has-feedback">
    <label for="" class="control-label">用户名</label>
    <input type="text" class="form-control">
    <span class="help-block">你输入的信息是正确的</span>
    <span class="glyphicon glyphicon-ok form-control-feedback"></span>
  </div>
</form>
```



## 2.4.9 有图标的表单

```
<form >
  <div class="form-group has-success has-feedback">
    <div class="input-group">
      <span class="input-group-addon">@</span>
      <input type="text" class="form-control" id="user">
    </div>
    <span class="glyphicon glyphicon-ok form-control-feedback"></sp
an>
  </div>
</form>
```

```
.input-group
  .input-group-add
  .form-control
```





## 第3章 响应式布局

---



响应式布局是bootstrap中最精华的部分

# 3-1实现原理

## 1.实现原理

Bootstrap框架中的网格系统就是将容器平分成12份。

## 2.工作原理

- 1.row必须包含在container中

```
<div class="container">
  <div class="row"></div>
</div>
```

- 2.在行(.row)中可以添加列(.column)，但列数之和不能超过平分的总列数，比如12

```
<div class="row">
  <div class="col-sm-3">
    col-3
  </div>
  <div class="col-sm-9">
    col-9
  </div>
</div>
```

# 3-2列偏移排序

## 1.偏移

```
.col-sm-offset-3  
//向右偏移
```

## 2.排序

```
//向左偏移  
.col-sm-pull-3  
//向右偏移  
.col-sm-push-3
```

## 3.嵌套

```
<div class="row">  
  <div class="col-sm-3"></div>  
  <div class="col-sm-9">  
    //此处嵌套一个网格  
    <div class="row">  
      <div class="col-sm-6"></div>  
      <div class="col-sm-6"></div>  
    </div>  
  </div>  
</div>
```

## 第8章 JS组件

---

- 使用方式
  - 1.基于data-\*属性
  - 2.基于JS API

# 1.modal模态框

```

    <button type="button" class="btn btn-primary" data-toggle="modal" data-target="#exampleModal">
      data-target弹窗
    </button>
    <button type="button" class="btn btn-danger" data-toggle="modal" id="showModal">js调出弹窗</button>

    <!-- Modal -->
    <div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">
      <div class="modal-dialog" role="document">
        <div class="modal-content">
          <div class="modal-header">
            <h5 class="modal-title" id="exampleModalLabel">Modal title</h5>
            <!-- data-dismiss表示隐藏模态框 -->
            <button type="button" class="close" data-dismiss="modal" >
              <span aria-hidden="true">&times;</span>
            </button>
          </div>
          <div class="modal-body">
            ...
          </div>
          <div class="modal-footer">
            <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
            <button type="button" class="btn btn-primary">Save changes</button>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>

```

```

//js
$("#showModal").click(function(){
    $("#exampleModal").modal();
})

```

```

<div class="modal fade">
  <div class="modal-dialog">
    <div class="modal-content">

      <div class="modal-header">

```

## 1.modal模态框

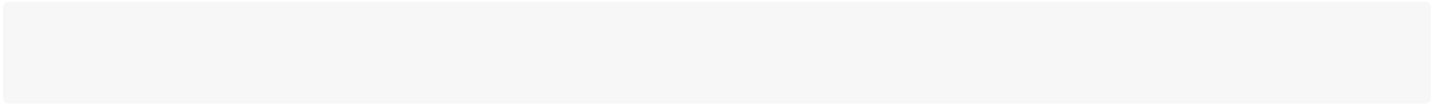
```
        <h5 class="modal-title">Title</h5>
        <button data-dismiss="modal"></button>
    </div>

    <div class="modal-body">
    </div>

    <div class="modal-footer">
        <button data-dismiss="modal">取消</button>
        <button >更新</button>
    </div>
</div>
</div>
</div>
```

# 第4章 菜单,按钮,导航

---



## 5-1下来菜单

```
<div class="dropdown">
  <button class="btn btn-default dropdown-toggle" type="button" data-
toggle="dropdown">
    手机品牌
    <span class="caret"></span>
  </button>
  <ul class="dropdown-menu" role="menu">
    <li><a href="#">苹果</a></li>
    <li><a href="#">小米</a></li>
    <li><a href="#">魅族</a></li>
  </ul>
</div>
```

1. 给以class为dropdown的容器 `<div class="dropdown"></div>`
2. 用btn按钮作用父菜单, 定义类名dropdown-toggle, 自定义属性data-toggle="dropdown"
3. 下拉菜单的类名定义为dropdown-menu





## 5-2下拉(分割线)

```
<div class="dropdown">
  <button class="btn btn-default dropdown-toggle" data-toggle="dropdo
wn">
    手机品牌
    <span class="caret"></span></button>

    <ul class="dropdown-menu">
      <li><a href="#">小米</a></li>
      <li><a href="#">苹果</a></li>
      //分割线
      <li class="divider"></li>
      <li><a href="#">华为</a></li>
    </ul>
  </div>
```



## 5-3下拉(菜单标题)

```
<div class="dropdown">
  <button class="btn btn-default dropdown-toggle" data-toggle="dropdo
wn">手机品牌
  <span class="caret"></span></button>

  <ul class="dropdown-menu">
    //菜单标题
    <li class="dropdown-header">国产手机</li>
    <li><a href="#">小米</a></li>
    <li><a href="#">华为</a></li>
    <li><a href="#">vivio</a></li>
    <li class="divider"></li>
    <li class="dropdown-header">外国品牌</li>
    <li><a href="#">苹果</a></li>
  </ul>
</div>
```



## 5-4dropup上弹菜单

```
<div class="dropup">
  <button class="btn btn-default dropdown-toggle" data-toggle="dropdo
wn">手机品牌
  <span class="caret"></span></button>

  <ul class="dropdown-menu">
    <li class="dropdown-header">国产手机</li>
    <li><a href="#">小米</a></li>
    <li><a href="#">华为</a></li>
    <li><a href="#">vivio</a></li>
    <li class="divider"></li>
    <li class="dropdown-header">外国品牌</li>
    <li><a href="#">苹果</a></li>
  </ul>
</div>
```



# 5-5按钮(按钮组,工具栏)

## 1.按钮组



```
<div class="btn-group">
  <button class="btn btn-default">
    <span class="glyphicon glyphicon-arrow-left"></span>
  </button>
  <button class="btn btn-default">
    <span class="glyphicon glyphicon-arrow-right"></span>
  </button>
</div>
```

## 2.按钮工具栏 btn-toolbar



```
<div class="btn-toolbar">
  <div class="btn-group">
    <button class="btn btn-default">
      <span class="glyphicon glyphicon-search"></span>
    </button>
    <button class="btn btn-default">
      <span class="glyphicon glyphicon-music"></span>
    </button>
  </div>

  <div class="btn-group">
    <button class="btn btn-default">
      <span class="glyphicon glyphicon-align-left"></span>
    </button>
    <button class="btn btn-default">
      <span class="glyphicon glyphicon-align-center"></span>
    </button>
    <button class="btn btn-default">
      <span class="glyphicon glyphicon-align-right"></span>
    </button>
  </div>
</div>
```

## 5-6按钮(嵌套按钮组-下拉)



```
<div class="btn-group">
  <button class="btn btn-default">关于我们</button>
  <button class="btn btn-default">公司介绍</button>
  <div class="btn-group">
    <button class="btn btn-info dropdown-toggle" data-toggle="dropdown">产品<span class="caret"></span>
    </button>
    <ul class="dropdown-menu">
      <li><a href="#">电脑</a></li>
      <li><a href="#">手机</a></li>
      <li><a href="#">平板</a></li>
    </ul>
  </div>
</div>
```

## 5-7按钮(垂直分组)



```
//btn-group-vertical
<div class="btn-group-vertical">
  <button class="btn btn-default">关于我们</button>
  <button class="btn btn-default">公司介绍</button>
  <div class="btn-group">
    <button class="btn btn-info dropdown-toggle" data-toggle="dropd
own">产品<span class="caret"></span>
    </button>
    <ul class="dropdown-menu">
      <li><a href="#">电脑</a></li>
      <li><a href="#">手机</a></li>
      <li><a href="#">平板</a></li>
    </ul>
  </div>
</div>
```

# 第5章 导航

.nav .nav-tabs

## 1.nav nav-tabs



```
<ul class="nav nav-tabs">
  <li><a href="#">苹果</a></li>
  <li><a href="#">小米</a></li>
  <li><a href="#">华为</a></li>
</ul>
```

## 2.nav nav-pills 胶囊式导航

```
<ul class="nav nav-pills">
  <li><a href="#">苹果</a></li>
  <li><a href="#">小米</a></li>
  <li><a href="#">华为</a></li>
</ul>
```



# 6-1tabs



```
<ul class="nav nav-tabs">
  <li class="active"><a href="#">苹果</a></li>
  <li><a href="#">小米</a></li>
  <li class="disabled"><a href="#">华为</a></li>
</ul>
```



## 6-2垂直堆叠导航



```
//nav stacked
<ul class="nav nav-pills nav-stacked">
  <li class="active"><a href="#">苹果</a></li>
  <li><a href="#">小米</a></li>
  <li><a href="#">华为</a></li>
</ul>
```

## 6-3自适应导航

---

### 1.nav nav-tabs nav-justified

//加上nav-justified这行样式,导航自适应

```
<ul class="nav nav-tabs nav-justified">
  <li class="active"><a href="#">苹果</a></li>
  <li><a href="#">小米</a></li>
  <li><a href="#">华为</a></li>
  <li><a href="#">中兴</a></li>
</ul>
```

## 6-4下拉导航



```
<ul class="nav nav-tabs">
  <li class="active"><a href="#">苹果</a></li>
  <li class="dropdown">
    <a href="#" class="dropdown-toggle" data-toggle="dropdown">小米
    <span class="caret"></span></a>
    <ul class="dropdown-menu" >
      <li><a href="#">小米手机</a></li>
      <li><a href="#">小米电脑</a></li>
      <li><a href="#">小米平板</a></li>
    </ul>
  </li>
  <li><a href="#">华为</a></li>
</ul>
```

## 6-5breadcrumb导航



breadcrumb

```
<ul class="breadcrumb">
  <li class="active"><a href="#">苹果</a></li>
  <li><a href="#">小米</a></li>
  <li><a href="#">华为</a></li>
</ul>
```

## 第6章 导航条

```
.navbar .navbar-default/.navbar-inverse  
.nav .navbar-nav
```



```
<div class="navbar navbar-default">  
  <ul class="nav navbar-nav">  
    <li class="active"><a href="#">小米</a></li>  
    <li><a href="#">苹果</a></li>  
    <li><a href="#">华为</a></li>  
  </ul>  
</div>
```

navbar-inverse反色导航——背景色为黑色

## 6-1为导航加标题

```
.navbar-header  
  .navbar-brand
```



```
<div class="navbar navbar-default">  
  <div class="navbar-header">  
    <div class="navbar-brand">极客营</div>  
  </div>  
  <ul class="nav navbar-nav">  
    <li><a href="#">Java</a></li>  
    <li><a href="#">PHP</a></li>  
    <li><a href="#">HTML5</a></li>  
  </ul>  
</div>
```

## 6-2带表单的导航



```
<div class="navbar navbar-default">
  <!-- navbar-header -->
  <div class="navbar-header">
    <a href="#" class="navbar-brand">极客营</a>
  </div>
  <!-- navbar-nav -->
  <ul class="nav navbar-nav">
    <li><a href="#">JAVA</a></li>
    <li><a href="#">PHP</a></li>
    <li><a href="#">HTML5</a></li>
  </ul>
  <!-- navbar-form -->
  <form action="" class="navbar-form navbar-right">
    <div class="form-group">
      <input type="text" class="form-control">
    </div>
    <button class="btn btn-default">搜索</button>
  </form>
</div>
```

## 6-3固定导航

---

- navbar-fixed-top
- navbar-fixed-bottom

存在bug及解决方法:

```
body{  
  padding-top:70px;  
  padding-bottom:70px;  
}
```



## 6-4响应式导航



Tip:需要引入js文件

```
<script src="https://cdn.bootcss.com/jquery/1.12.4/jquery.min.js"></script>
<script src="https://cdn.bootcss.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
```

- 1.将需要折叠的内容,包裹在一个div内,并为div加入collapse,navbar-collapse两个类名
- 2.保证在窄屏的时候要显示的图片(固定写法)

```
<button class="navbar-toggle" type="button" data-toggle="collapse">
  <span class="sr-only">Toggle Navigation</span>
  <span class="icon-bar"></span>
  <span class="icon-bar"></span>
  <span class="icon-bar"></span>
</button>
```

- 3.button添加data-target=".类名/#id名", 究竟是类名还是id名呢? 由需要折叠的div来决定。

```
<div class="navbar navbar-default">
  <div class="navbar-header">
    <!-- navbar-brand -->
    <a href="#" class="navbar-brand">极客营</a>
    <!-- navbar-toggle -->
    <button class="navbar-toggle" data-target="#collapse" data-toggle="collapse">
      <span class="sr-only">Toggle Navbar</span>
      <span class="icon-bar"></span>
      <span class="icon-bar"></span>
      <span class="icon-bar"></span>
    </button>
  </div>

  <!-- collapse -->
  <div class="collapse navbar-collapse" id="collapse">
    <ul class="nav navbar-nav">
      <li><a href="#">JAVA</a></li>
      <li><a href="#">PHP</a></li>
      <li><a href="#">HTML5</a></li>
    </ul>
  </div>
```

```
</div>
```

```
.navbar .navbar-default  
  .navbar-header  
    .navbar-brand  
    .navbar-toggle data-toggle="collapse" data-target=""  
  .collapse .navbar-collapse  
    .nav .navbar-nav
```

## 6.三大框架中的css

---

# 7.css3

## css3基础教程

---

### 1.3d转换

# 第一章 3d效果

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

```
//css  
div{  
    perspective: 800px;  
    border-radius: 10px;  
    width:300px;  
    height:400px;  
    border:1px solid #eee;  
    padding:15px;  
}  
img{  
    width:300px;  
    height:400px;  
    animation: rotate 2s infinite;  
}  
@keyframes rotate{  
    from{  
        transform: rotateY(0deg)  
    }  
    to{  
        transform: rotateY(180deg)  
    }  
}
```



## B.进阶教程

---

1.flex教程

2.css-@import

3.grid布局

4.bootstrap栅格实现原理

5.选择器 倍数写法

# 1.flex教程

阮一峰语法篇

阮一峰实战篇

```
justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly;
```

- space-between 两端对齐，子元素之间的间隔相等
- space-around 子元素之间的间隔,比子元素到父元素边框的间隔大一倍
- space-evenly 父元素的边界到子元素之间的间隔,和子元素与子元素之间的间隔相等(所有的间隔相等)

```
.box{  
  justify-content: space-evenly;  
}
```

```
//设置垂直方向的对齐  
align-items: flex-start | flex-end | center | stretch  
//stretch在没有高度的情况下自动拉升
```

## 教程总结

## flex布局总结

```
<div class="box">  
</div>  
//css  
.box{  
  display: flex  
}
```

```
//能够给父元素设置的属性  
justify-content -- 设置子元素水平方向  
align-items -- 设置子元素垂直方向  
flex-direction -- 设置子元素的排列方向  
flex-wrap -- 子元素是否换行
```

```
//能够给子元素设置的属性  
order
```



```
flex  
align-self
```

## 2.css-@import

---

```
//head.css
div{
border:1px solid #333;
}
```

```
//index.css
@import 'head.css';
```

## 3.grid布局

### grid布局

```
//grid单元格中元素水平对齐方法
justify-items: center;
//grid单元格中元素垂直对齐方法
align-items:center;
```



实现一个这样的布局

```
.button-list{
  padding: 20px;
  display: grid;
  grid-template-columns: 1fr 1fr 1fr;
  justify-items: center;
  grid-row-gap: 10px;
}
```

## 4.bootstrap栅格实现原理

```

@mixin fh {
    float: left;
    height: 100%;
}
// xs手机screen<767
@media (max-width: 767px) {
    .hidden-xs {
        display: none !important;
    }
    @for $i from 1 through 12 {
        .col-xs-#{$i} {
            width: 100% / 12 * $i;

            @include fh;
        }
    }
}

// sm平板 768<screen<991
@media (min-width: 768px) and (max-width:991px) {
    .hidden-sm {
        display: none !important;
    }
    .container {
        width: 750px;
    }

    @for $i from 1 through 12 {
        .col-sm-#{$i} {
            width: 100% / 12 * $i;

            @include fh;
        }
        .col-sm-offset-#{$i}{
            margin-left:100%/12*$i;
        }
    }
}

// md电脑 992<screen<1119
@media (min-width: 992px) and (max-width:1119px) {
    .hidden-md {
        display: none !important;
    }
    @for $i from 1 through 12 {
        .col-md-#{$i} {
            width: 100% / 12 * $i;

```

```

        @include fh;
    }
}
.container {
    width: 970px;
}
}
// lg大屏电脑 screen>1200
@media (min-width: 1200px) {
    .hidden-lg {
        display: none !important;
    }
    .container {
        width: 1170px;
    }
    @for $i from 1 through 12 {
        .col-lg-#{ $i } {
            width: 100% / 12 * $i;

            @include fh;
        }
    }
}

.container {
    margin-left: auto;
    margin-right: auto;
}
*{margin:0;padding:0}
.row::before{
    content:"";
    display: table;
}
.row::after{
    content:"";
    display: block;
    clear: both;
}
div{
    box-sizing: border-box;
}

```

## 5.选择器 倍数写法

```
//3的倍数
ul li:nth-child(3n+3) {
    color: #ccc;
}
```

# C.sass进阶教程

---

## sass进阶教程

# 第一节 Sass的函数功能-字符串与数字函数

- 字符串函数
- 数字函数
- 列表函数
- 颜色函数
- Introspection 函数
- 三元函数等
  - [1.字符串函数](#)
  - [2.数字函数](#)

## 1.字符串函数

`unquote($string)` : 删除字符串中的引号 ;  
`quote($string)` : 给字符串添加引号。  
`To-upper-case()` : 转为大写  
`To-lower-case()` : 转为小写

```
div{
  content:unquote($string: "hello")
}
div{
  content:quote($string: good)
}
```

```
div{
  content:to-upper-case($string: good)
}
```

```
div{
  content:to-lower-case($string: GOOD)
}
```

## 2.数字函数

`percentage($value)` : 将一个不带单位的数转换成百分比值 ;  
`round($value)` : 将数值四舍五入, 转换成一个最接近的整数 ;  
`ceil($value)` : 将大于自己的小数转换成下一位整数 ;



`floor($value)`：将一个数去除他的小数部分；  
`abs($value)`：返回一个数的绝对值；  
`min($numbers...)`：找出几个数值之间的最小值；  
`max($numbers...)`：找出几个数值之间的最大值；  
`random()`：获取随机数

```
div{
  width:percentage($number: .1)
}
```

```
//四舍五入取整
div{
  width:round($number: 14.5)
}
```

```
//上取整
div{
  width:ceil($number: 12.3)
}
```

```
//下取整
div{
  width:floor($number: 12.3)
}
```

```
//绝对值
div{
  width:abs($number: -3px)
}
```

```
//取最大值
div{
  width:max(1,2,3)
}
//取最小值
div{
  width:min(1,2,3)
}
```

```
div{
  width:random()
}
```



## 第二节 列表函数

- `length($list)`  
返回一个列表的长度值；
- `nth($list, $n)`  
返回一个列表中指定的某个标签值
- `join($list1, $list2, [$separator])`  
将两个列给连接在一起，变成一个列表；
- `append($list1, $val, [$separator])`  
将某个值放在列表的最后；
- `zip($lists...)`  
将几个列表结合成一个多维的列表；
- `index($list, $value)`  
返回一个值在列表中

```
$list:1px solid #333;
@if length($list)>1{
  div{
    border:$list;
  }
}
```

### nth()函数

#### 语法

```
nth($list,$n)
```

```
$list:1px solid #333;
@if length($list)>1{
  div{
    border:nth($list,1 );
  }
}
```

### join()

```
$list:1px solid;
@if length($list)>1{
```

```
div{  
  border:join($list,#333 )  
}  
}
```

## append()

---

```
$list:1px solid;  
@if length($list)>1{  
  div{  
    border:append($list,#333)  
  }  
}
```

## 第三节 Introspection函数

type-of() 函数主要用来判断一个值是属于什么类型：

返回值：

number 为数值型。

string 为字符串型。

bool 为布尔型。

color 为颜色型。

```
>> type-of(100)
"number"
>> type-of(100px)
"number"
```

```
$color:red;
@if type-of($value: $color) == color{
  div{
    color:$color;
  }
}
```

### unit() 函数

函数主要是用来获取一个值所使用的单位

```
$n:12px;
@if unit($number: $n) == px{
  div{
    width:$n;
  }
}
```

### comparable()函数

comparable() 函数主要是用来判断两个数是否可以进行“加，减”以及“合并”。如果可以返回的值为 true，如果不可以返回的值是 false

```
@mixin name {
  @if comparable(12, 11 ){
    width:100px;
  }
}
```

```
    }  
  }  
  div{  
    @include name  
  }
```

## Miscellaneous函数

---

在这里把 Miscellaneous 函数称为三元条件函数，主要因为他和 JavaScript 中的三元判断非常的相似。他有两个值，当条件成立返回一种值，当条件不成立时返回另一种值：

语法

```
if($condition,$if-true,$if-false)
```

```
@mixin name {  
  width:if(false,100px,200px)  
}  
div{  
  @include name  
}
```

## 第四节 map

Sass 的 map 常常被称为数据地图，也有人称其为数组，因为他总是以 key:value 成对的出现，但其更像是一个 JSON 数据

```
$social-colors: (
  dribble: #ea4c89,
  facebook: #3b5998,
  github: #171515,
  google: #db4437,
  twitter: #55acee
);
```

```
//使用@each去遍历
$social-colors: (
  dribble: #ea4c89,
  facebook: #3b5998,
  github: #171515,
  google: #db4437,
  twitter: #55acee
);
@each $k,$v in $social-colors {
  .btn-#{ $k }{
    color:$v;
  }
}
```

map-get  
//获取map类型的值

```
$colors:(
  bg:#eee,
  text-color:#333,
  primary:#ff2d51
);
div{
  color:map-get($map: $colors, $key: text-color)
}
```

```
//自定义函数
@function color($key) {
  @if map-has-key($colors, $key) {
```

```

    @return map-get($colors, $key);
  }

  @warn "Unknown `${$key}` in $colors.";

  @return null;
}

```

```

div:{
  background:color(bg)
}

```

- map-get(\$map,\$key) : 根据给定的 key 值 , 返回 map 中相关的值。
- map-merge(\$map1,\$map2) : 将两个 map 合并成一个新的 map。
- map-remove(\$map,\$key) : 从 map 中删除一个 key , 返回一个新 map。
- map-keys(\$map) : 返回 map 中所有的 key。
- map-values(\$map) : 返回 map 中所有的 value。
- map-has-key(\$map,\$key) : 根据给定的 key 值判断 map 是否有对应的 value 值 , 如果有返回 true , 否则返回 false。
- keywords(\$args) : 返回一个函数的参数 , 这个参数可以动态的设置 key 和 value。

## 1.map-has-key

map-has-key(\$map,\$key) 函数将返回一个布尔值。当 \$map 中有这个 \$key , 则函数返回 true , 否则返回 false。

```
$colors: (bg: #ea4c89, text-color: #333, primary: #ff2d51);
```

```

@if map-has-key($map: $colors, $key: bg){
  div{
    color:map-get($map: $colors, $key:text-color )
  }
}

```

## 2.map-keys(\$map)

函数将会返回 \$map 中的所有 key。这些值赋予给一个变量 , 那他就是一个列表

```

$social-colors: (
  dribble: #ea4c89,

```



```
    facebook: #3b5998,
    github: #171515,
    google: #db4437,
    twitter: #55acee
  );
$list:map-keys($social-colors);
@each $var in $list {
  .#{$var}{
    width:100px;
  }
}
```

### 3.map-merge

---

```
$bg:(
  bg:red
);
$color:(
  color:blue
);

$bc:map-merge($bg,$color);
div{
  background:map-get($map: $bc, $key:bg );
  color:map-get($map: $bc, $key:color )
}
```

## 第五节 颜色函数

---

```
//打开命令行  
sass i  
rgba(#333, .5)
```

### 1. rgba()函数

---

```
$bg:red;  
div{  
  background: rgba($bg, .3)  
}
```

### 2.RGB颜色函数-red()、green()、blue()函数

---

Red()获取颜色重red的色值