HTML

常用标签

- h1~h6：标题
- hr：水平线
- p：段落
- br：换行
- Ctrl+ / ：注释
- Ctrl+ k ：整理代码格式
- b、strong：加粗
- i、em：倾斜
- u、ins：下划线
- s、del：删除线
- img：图片
- <body background=>：背景图
- a：超链接
- ul+li：无序列表
- ol+li：有序列表
- dl+dt+dd：自定义列表
- table：表格
    - 标签
        - table：表格标签
        - tr：行标签
        - td：单元格标签 列默认是以当前列中最宽的为基准
        - th：表头标签 内容水平居中且加粗的
        - caption：表格标题
    - 属性:
        - border：边框
        - width：宽度
        - height：高度
        - cellspacing：单元格与单元格之间的间距
        - align left：(默认)居左/center 居中/right 居右
        - colspan：夸列合并单元格
        - rowspan：跨行合并单元格
    - 创建一个表格时，如果不指定thead tfoot tbody这些标签，浏览器会自动创建一个tbody标签，然后将所有的tr都放入到tbody中
- form：表单

    - label：标题
        - 属性：for 关联表单中的id值（比如点击姓名，定位到输入框）
    - input
        - 属性：type
            - text：输入框
                - 属性：placeholder：提示文本，（比如请输入）
- password：密码框 - radio：单选按钮 - 属性： - name：是组的概念，将多个单选的name设置相同的名称，实现单选 - checked：默认选中 checked="checked" / checked="" / checked - disabled：不可用 - checkbox 复选框 - 属性： - checked：默认选中 checked="checked" / checked="" / checked - disabled：不可用 - button：普通按钮 - submit：提交按钮 - reset：重置按钮
    - select：下拉菜单
        - option：下拉项
        - optgroup：下拉组
            - 属性：label 下拉组标题
        - 属性
            - selected：默认选中 selected="selected" / selected="" / selected
    - textarea：文本域
    - form：提交表单 将所有被提交的标签包含其中
        - 属性
            - action：提交地址
            - method：提交方式 get/post
    - button：按钮
        - 属性：type
            - button：普通按钮
            - submit：提交按钮（默认值）
            - reset：重置按钮

```
<form action="http://www.baidu.com" method="get">
    <label for="txt1">姓名:</label>
    <input type="text" id="txt1">
    <br><br>
    <label>
        密码:<input type="password">
    </label>
    <br><br>
    性别:
    <input type="radio" name="gender" id="male" checked><label for="male">男</label>
    <input type="radio" name="gender" id="female"><label for="female">女</label>
    <br><br>
    学历: <input type="radio" name="edu">本科
    <input type="radio" name="edu">小学
    <input type="radio" name="edu">幼儿园
    <br><br>
    您的国籍是:
    <select>
```

```
        <option>中国</option>
        <option>日本</option>
        <option selected="selected">韩国</option>
        <option>泰国</option>
    </select>
    <select>
        <optgroup label="亚洲">
            <option>中国</option>
            <option>日本</option>
            <option selected="selected">韩国</option>
            <option>泰国</option>
        </optgroup>
        <optgroup label="欧洲">
            <option>法国</option>
            <option>德国</option>
            <option>意大利</option>
            <option>英格兰</option>
        </optgroup>
    </select>
    <br><br>
    您的兴趣爱好是:
    <input type="checkbox">抽烟
    <input type="checkbox">喝酒
    <input type="checkbox" id="head" checked="checked"><label for="head">烫头</label>
    <br><br>
    公司简介:
    <br><br>
    <textarea></textarea>
    <br><br>
    <input type="button" value="按钮">
    <input type="submit" value="提交">
    <input type="reset" value="重置">
    <button type="submit">按钮</button>
</form>
```

CSS

常用文本属性

- color : blue / rgb(255,255,255) / #FF0000——设置颜色
- font-size : 60px——字号
- font-weight : 400 / normal / 700 / bold——正常 / 加粗
- font-style : italic / normal——倾斜 / 正常
- font-family : "宋体"——字体
- text-align : center / left(默认) / right——文本水平居中 / 左 / 右
    - 如果希望文本、行内元素、行内块元素水平居中，要给它们的父元素设置text-align : center
- text-indent : 2em——首行缩进
- text-decoration : underline / overline / line-through / none——下划线 / 顶划线 / 中划线 / 去掉线

- word-break : break-all——强制换行
- white-space : nowrap——强制不换行
- letter-spacing : 0px——字间距
- line-height
  - 行高，文本在元素中字的大小 + 字顶部 + 字底部（行高可以是倍数，不能是百分比，得是具体的值）
    - 盒子未设置固定高度时，高度会跟着行高的变化而变化，高度就是行高的值
    - 盒子设置固定高度时，高度不会跟着行高的变化而变化
    - 反之：希望元素垂直居中于盒子，行高需等于盒子的高度
  - line-height ——行间距，基线到基线之间的距离（上一个字的底部+下一个字的顶部）
- list-style : none;——去掉列表小圆点

border边框属性

- border-width : 5px; —— 边框粗细 默认值3px
- border-style : dotted;——边框样式 solid实线 / dashed虚线 / dotted点状线
- border-color : red;—— 边框颜色 默认是黑色

background背景属性

- background-color : yellow;——背景色
- background-image : url(liuzhiyuan.jpg);——背景图（不占位置，会在padding中）
- background-repeat : repeat / repeat-x / repeat-y / no-repeat;——平铺方式平铺(默认值) / 水平平铺 / 垂直平铺
- background-position : 100px 0px;——背景图在元素中的位置 水平位置 垂直位置
  - 水平可写单词：left / center / right，可写数值:正值向右，负值向左
  - 垂直可写单词：top / center / bottom，可写数值:正值向下，负值向上
- background-clip : content-box; ——背景裁剪属性

复合属性

- 当单属性和复合属性同时存在时，要先写复合属性再写单属性，否则会被复合属性中的值层叠。
- font：是否倾斜 是否加粗 字号(必填项) / 行高 字体(必填项)
  - font : italic 700 40px / 500px "宋体";
- border：粗细 样式 颜色
  - border : 5px solid red;
  - border-left / border-top / border-right / border-bottom——左 / 上 / 右 / 下
  - border-bottom : none;——设置边框没有
- background：背景色 背景图 平铺方式 水平位置 垂直位置
  - background: yellow url(liuzhiyuan.jpg) no-repeat 0px 0;

盒子三属性

```
div{
    width : 200px;
    height : 200px;
    background : red;
  }
```

选择器

- 标签选择器：标签名{...}
- 类选择器：.class类名{...}
- id选择器：#id名{...}
- 权重：
  - **继承0<标签选择器1<类选择器/状态伪类10<id选择器100<行内样式1000<!important无穷**
- 多类名调用

```
<span class="blue font150 abc">G</span>
```

- 后代选择器

```
.h5 div h3{
                color:yellowgreen;
            }
```

- 一级后代选择器

```
/* 一级后代选择器 > */
.box>ul>li{
    border:1px solid red;
}
```

- 并集选择器：多个选择器逗号隔开，实现共同属性

```
.box span,.box p,.box .one{
                border: 1px solid red;
            }
```

- 交集选择器：同时具备多个选择器条件

```
.box p.current{
                border:1px solid red;
            }
```

块/行内/行内块—标准流

- 块元素：html、body、div、h1-h6、p、ul、ol、li、dl、dt、dd、hr、for

    - 面积组成：margin + border + padding + content
    - 独占一行（多个会换行），设置宽高起作用
    - 没有设置固定宽度时，宽度和父元素的content宽度一样
    - 没有设置固定高度时，高度由内容撑开
- 行内元素：span、i、em、u、ins、s、del、a、b、strong
    - 一行有多个（多个不换行），设置宽高不起作用，尺寸由内容决定，没有内容时，宽高是0
    - 代码换行会有默认间距
    - 沿基线对齐(底对齐)
    - 设置自身内外（垂直）边距，受基线对齐的限制，垂直方向的内外边距不起作用
- 行内块元素：img、表单
    - 一行有多个（多个不换行），设置宽高起作用
    - 代码换行会有默认间距
    - 沿基线对齐(底对齐)
    - 设置自身内外（垂直）边距，会影响周围的元素（垂直方向）
- 转换
    - display : inline-block;——其它转换成行内块
    - display : block;——其它转换成块级

浮动—浮动流

- float : left / right;——按照标签的书写顺序，依次排列在包含块（父元素）的左 / 右上。
    - 父元素有padding时，在父元素的content区域浮动
    - 用作横向布局，代码换行，没有间距
    - 设置自身内外（垂直）边距，不会影响周围的元素（垂直方向）
    - 设置margin: 0 auto; 不会水平居中
    - 元素设置浮动后，不再具备之前的显示模式，设置宽高起作用，不设置宽高时，由内容决定的
    - 浮动元素的父元素宽度不够时，会自动换行
    - 元素设置浮动后，后面的文本对浮动元素进行文本环绕
    - 当子元素设置浮动后，高度超出父元素时，会影响到下面父元素中的浮动元素（需要给子元素加高度）
- 影响
    - 子元素浮动，父元素没有设置固定高度，由于子元素是漂浮状态，父元素会认为没有内容撑开自身的高度，此时造成高度塌陷，父元素的高度是0。
    - 解决方案：
        - 给父元素设置overflow属性，overflow : scroll;
            - hidden 溢出隐藏
            - scroll 溢出滚动
            - auto 当内容超出元素时，自动设置滚动条

```
/* overflow-x 水平隐藏
   overflow-y 垂直隐藏*/
overflow-x: hiddn;
```

        - 额外标签法

```
<div class="four"></div>
.four{
    /* height: 100px; */
    background:yellow;
    /* 父元素的最后 */
    /* 清除(关闭,结束)之前的三个浮动造成影响,让浮动元素占位置 */
    clear: both;
}
```

- 父元素调用clearFix类名

```
.clearFix::after{
    content:"";
    display: block;
    clear:both;
}
```

- 给父元素设置浮动（此方案要正巧赶上父元素需要设置浮动）

状态伪类

- a : link{}——未访问
- a : visited{}——访问后
- a : hover{}——鼠标移入

```
/* 鼠标移入a时,让后代的span........ */
.box .banner a:hover span {
    color: #ff8500;
}
```

  - cursor : pointer——鼠标样式
- a : active{}——鼠标按下

盒子模型padding/margin

- 盒子在网页中的尺寸 = content区域（设置的宽高） + padding区域 + border区域
- 内边距padding：盒子与内容之间的间距
  - 单属性
    - padding-left : 10px;——左内边距
    - padding-top : 10px;——上内边距
    - padding-right : 10px;——右内边距
    - padding-bottom : 10px;——下内边距
  - 复合属性
    - padding : 10px;——一个值 上右下左
    - padding : 10px 20px;——两个值 上下 左右
    - padding : 10px 20px 30px;——三个值 上 左右 下

        - padding : 10px 20px 30px 40px;——四个值 上 右 下 左
- padding减宽度的场景
    - 块元素未设置固定宽度，宽度和父元素一样，设置水平方向的padding，不会撑宽盒子，会从content自动减去padding值，宽度的尺寸不变
    - 块元素设置固定宽度，设置水平方向的padding，盒子的尺寸会变大
- 外边距margin：设置盒子与盒子之间的间距
    - 单属性
        - margin-left : 100px;——左外边距
        - margin-top : 100px;——上外边距
        - margin-right : 100px;——右外边距
        - margin-bottom : 100px;——下外边距
    - 复合属性
        - margin: 10px;——一个值 上右下左
        - margin: 10px 20px;——两个值 上下 左右
            - **margin: 0 auto;——块元素水平居中于父元素，水平左右外边距自适应**
        - margin: 10px 20px 30px;——三个值 上 左右 下
        - margin: 10px 20px 30px 40px;——四个值 上 右 下 左
    - 外边距合并：垂直排列的两个块元素，分别给上面的盒子设置向下的外边距和给下面的盒子设置向上的外边距，形成外边距合并。
        - 当两个值相同时，就是该值
        - 当两个值不同时，是较大的那个值
    - 外边距塌陷：嵌套的两个块元素，给子元素（第一个）设置向上的外边距，此时父元素会跟着掉下来，形成外边距塌陷。
        - 解决方案：
            - 给父元素设置上边框
            - 给父元素设置overflow属性，overflow : scroll;
                - hidden 溢出隐藏
                - scroll 溢出滚动
                - auto 当内容超出元素时,自动设置滚动条
            - 给父元素设置浮动
            - 给子元素设置浮动

img底部留白处理

- 由于img是行内块元素，底部和文本的基线对齐，所有会有一部分留白。

```
.box{
                                border: 1px solid red;
                                /* 第一种 将box的字体设置成0*/
                            font-size: 0;
                        }
```

```
.box img{
                                    /* 第二种 将img转换成块元素,display:block*/
                                    display: block;
                            }
```

样式初始化

```
/* 基础设置 */
body,h1,h2,h3,h4,h5,h6,p,ul,ol,dl,li,dt,dd{
        margin: 0;
        padding: 0;
}
ul,ol{
/* 去掉小圆点 */
        list-style: none;
}
h1,h2,h3,h4,h5,h6{
        font-weight: 400;
        /* 父元素字号的百分比 */
        font-size: 100%;
}
b,strong{
        font-weight: 400;
}
i,em{
        /* 不倾斜 */
        font-style: normal;
}
u,ins,s,del{
        text-decoration: none;
}

table{
        /* width: 500px;
        height: 300px;
        border: 1px solid red; */
        /* 相当于cellspacing */
        border-spacing: 0;
        /* 1px边框 */
        border-collapse: collapse;
}
th,td{
        border: 1px solid red;
        text-align: center;
}
img{
        /* 底部留白 */
        display: block;
}
input,button{
/* 去除轮廓线 */
```

```
        outline: none;
        border:none;
}
input{
        /* 去掉表单的默认内边距 */
        padding: 0;
}

.clearFix::after{
        content:"";
        display: block;
        clear:both;
}

/* 风格设置 */
body{
        font: 12px/1.3 "Microsoft YaHei",Tahoma, Helvetica, Arial, "\5b8b\4f53", sans-serif;
        color:#333;
}
a{
        color:#333;
        text-decoration: none;
}
a:hover{
        color:yellowgreen;
}
```

圆角属性

/* 左上 右上 右下 左下 */

border-radius: 20px 0 0 20px;

border-radius : 50%;——圆

超出的文本用省略号代替

```
/* 强制不换行 */
white-space: nowrap;
overflow: hidden;
/* 用省略号代替溢出部分 */
text-overflow: ellipsis;
```

定位

 image-20220708173813973

- position : relative; ——相对定位（占位定位）por
- 通过偏移量，根据当前在标准流自身的位置为参考点移动，在移动时，是在z轴的空间移动,此时不会影响周围的元素，但是在标准流的位置还是存在的

    - 水平偏移量
        - left 正值向右 负值向左 优先
        - right 正值向左 负值向右
    - 垂直偏移量
        - top 正值向下 负值向上 优先
        - bottom 正直向上 负值向下
    - left / right / top / bottom ： auto;——让之前设置的值不起作用
- position : absolute;——绝对定位：脱离标准流，默认起始参考点是浏览器的第一屏 / 初始包含块 poa
    - 当元素设置绝对定位后，设置宽高起作用
    - 参考点：绝对定位的起始参考点是设置定位属性的父元素(离自身最近的)，如果没有设置相对定位的父元素，默认参考点是body初始包含块
- position : fixed;——固定定位：将元素设置到浏览器窗口的一个具体坐标位置，固定定位的参考点永远是浏览器窗口
- z-index——层级
    - 层级相同时，后写的标签压在先写标签的上面，当层级不同时，谁的层级高谁在上面
    - 取值范围是整数，默认层级是0，层级是负数时，是低于标准流的元素
    - 父元素设置层级：
        - 层级相同时，后写的标签连同子元素，压在先写的标签连同子元素的上面
        - 层级不同时，较大层级的标签连同子元素，压在较低层级的标签连同子元素的上面，父元素的层级将决定子元素的层级的高低
    - 父元素不设置层级：
        - 后写的标签连同子元素，压在先写的标签连同子元素的上面
        - 子元素层级的高低，将决定自身是否能压在后面标签的上面

居中方式

- 第一种
    - 水平居中：设置left:50%，父元素宽度的一半，margin-left：负的自身宽度的一半，向回走
    - 垂直居中：设置top:50%，父元素高度的一半，margin-top：负的自身高度的一半，向回走

```
position: absolute;
/* 父元素宽度的百分比 */
left:50%;
/* 从当前left值作为起始点 */
margin-left: -50px;
/* 父元素高的百分比 */
top:50%;
/*  从当前top值作为起始点 */
margin-top: -50px;
```

- 第二种

```
position: absolute;
/* 水平且垂直居中 给元素设置宽高*/
left:0;
right:0;
top:0;
bottom:0;
margin: auto;
/*水平居中*/
left:0;
right:0;
margin: 0 auto;
```

- 第三种

```
position: absolute;
/* 水平垂直居中 */
left:50%;
top:50%;
/*              自身宽度的百分比,自身高度的百分比 */
transform: translate(-50%,-50%);
```

- 定位和浮动的选择：批量的选择浮动，个人的、会压住其他元素、可能出现可能消失的、满屏飞选择定位
    - 把要出现的框（子）写在要点击的框（父）里面，设置父元素相对定位，子元素绝对定位，子元素写好后先display:none，在父元素的动作事件中写子元素display:block

开启bfc（不存在塌陷）

- 开启bfc：block formatting context(块级格式化上下文)，是一个独立的渲染区域，脱离标准流
    - overflow: hidden/scroll
    - float:left/right
    - position: absolute,fixed

隐藏

```
/* 占位隐藏 */
visibility: hidden;
/* 不占位隐藏 */
display: none;
/* 其他 */
opacity: 0;
width: 0;
heigth: 0;
transform: translate(-2000);
transform: scale(0);
```

透明

```
/* 第一种:透明属性,取值范围是0-1之间的小数,默认值是1(不透明),内容会一起透明*/
opacity: .5;
/* 第二种: css3中的属性 */
background:rgba(0,0,0,.5);
```

色饱和度

```
/* 色饱和度 */
border-color: hsla(0,0%,100%,.3);
background: rgba(0,0,0,.4);
```

精灵图

css精灵：css sprite 雪碧图，是背景图的技术。

```
background: url(abcd.jpg) no-repeat;
background-position:-372px -277px;
```

图标字体

- https://www.iconfont.cn/ 阿里巴巴矢量字体库下载
  - 添加至项目——选择Unicode/font class——下载至本地——放至项目——demo_index.html查看用法
- http://www.fontawesome.com.cn/fontawesome图标字体
  - 下载旧版本——开始使用查看用法——放至项目——link<font-awesome.css>——用*调用*

生成导航图标

http://www.bitbug.net/

HTML5

新增标签（块）

```
<header>头部标签</header>
<nav>导航标签</nav>
<section>
    <aside>侧边栏</aside>
    <article>文章标签</article>
</section>
<footer>底部标签</footer>
```

新增属性

```
<!-- 禁用属性  disable 三种写法和checked一样-->
<input type="text" disabled="disabled">
<input type="button" value="按钮" disabled="disabled">
<!-- 非空验证 -->
<form action="http://www.baidu.com">
    <input type="text" required="required">
    <!-- 自动获取焦点 -->
    <input type="text" autofocus="autofocus">
    <!-- 提示文本 -->
    <input type="text" placeholder="充气气球">
    <!-- 自动补全
                off 关闭(默认值)
                                on  开启
                                配合name属性使用,根据name的属性值,将成功提交的文本存储到下拉列表
                        -->
    <input type="text" autocomplete="on" name="zzz">
    <input type="submit" value="提交">
</form>
```

新增表单类型

```
<form action="http://www.baidu.com">
    <!-- 邮箱类型 -->
    <input type="email">
    <!-- 网址类型 -->
    <input type="url">
    <!-- 本地时间类型 -->
    <input type="datetime-local">
    <!-- 月类型 -->
    <input type="month">
    <!-- 日类型 -->
    <input type="date">
    <!-- 周类型 -->
    <input type="week">
    <!-- 颜色 -->
    <input type="color">
    <!-- 滑块 -->
    <input type="range">
    <input type="submit" value="提交">
</form>
```

音频标签

```
<!--
音频标签: audio
source多个文件引入标签
属性:
src 音频文件路径
controls 控制面板
loop 循环播放
```

```
autoplay 自动播放,目前ie是可以使用
-->
<!-- 引入单个文件 -->
<audio src="music/yinyue.ogg" controls="controls" loop="loop" autoplay="autoplay"></audio>
<!-- 引入多个文件 -->
<audio controls="controls" loop="loop" autoplay="autoplay">
    <source src="music/yinyue.ogg"></source>
    <source src="music/yinyue.MP3"></source>
</audio>
```

视频标签

```
<!--
视频标签: video
source多个文件引入标签
属性:
src 音频文件路径
controls 控制面板
loop 循环播放
autoplay 自动播放,目前ie是可以使用
-->
<!-- 单个文件引入 -->
<video src="video/movie.mp4" controls="controls" loop="loop" autoplay>
    <!-- 多个文件引入 -->
    <video controls="controls" loop="loop" autoplay>
        <source src="video/movie.webm"></source>
        <source src="video/movie.ogg"></source>
        <source src="video/movie.mp4"></source>
</video>
```

CSS3

关系选择器

```
/* 下一个紧挨着的兄弟元素选择器 */
h1+b{
    border: 1px solid red;
}
/* 后面所有的兄弟元素选择器 */
h1~p{
    border: 1px solid red;
}
.one~.two {
    border: 1px solid red;
}
```

属性选择器

```
/* 表示有class属性名的元素 */
div[title]{
    color:blue;
}
/* 表示class属性值等于box的 */
div[class="box"]{
    color:red;
}
/* 表示class属性值包含box的 */
div[class *= "box"]{
    color:red;
}
/* 表示class属性值以box开始的... */
div[class ^= "box"]{
    color:red;
}
/* 表示class属性值以box结束的... */
div[class $= "box"]{
    color:red;
}
/* 表示class属性值包含独立单词box的... */
div[class ~= "box"]{
    color:red;
}
```

伪类选择器

- 受其他元素影响的

```
/* 第一个儿子元素li */
.list li:first-child{
        color:red;
}
/* 最后一个儿子元素li  */
.list li:last-child{
        color:red;
}
/*
  根据编号选择,编号可以是数字,从1开始,
  还可以是单词,odd是基数,even是偶数
  还可以是表达式,表达式必须是n,n从0开始
 */
.list li:nth-child(2n+3){
        color:red;
}
/* 倒数根据编号选择li */
.list li:nth-last-child(2n){
        color:red;
}
/* 唯一一个儿子元素li */
.box span:only-child{
```

```
        color:red;
}
```

- 不受其他标签影响

```
/* 第一个儿子元素li */
.list li:first-of-type{
        color:red;
}
/* 最后一个儿子元素li  */
.list li:last-of-type{
        color:red;
}
/*
  根据编号选择,编号可以是数字,从1开始,
  还可以是单词,odd是基数,even是偶数
  还可以是表达式,表达式必须是n,n从0开始
 */
/* 根据编号选择li */
.list li:nth-of-type(odd){
        color:red;
}
/* 倒数根据编号选择li */
.list li:nth-last-of-type(1){
        color:red;
}
/* 唯一一个儿子元素li */
.box span:only-of-type{
        color:red;
}
```

- 内容是空(内容包括:文本,标签,空格)

```
.box2:empty{
    width: 100px;
    height: 100px;
    background: red;
}
```

状态选择器

- focus伪类

```
/*focus伪类: 获取焦点伪类*/
input[type="text"]{
        background: yellow;
        color:red;
}
input[type="text"]:focus{
        background: yellowgreen;
        color:blue;
}
```

- checked伪类

```
/* 清除表单的默认样式  快捷键  app tab键*/
/* 带前缀的属性属于测试阶段的属性,每个内核的实现会有区别,或者根本没有实现 */
-webkit-appearance: none;
-moz-appearance: none;
appearance: none;
/*
    浏览器    内核                前缀
        chrome    webkit(blink)    -webkit-
        safari    webkit           -webkit-
        opear     webkit           -o-
        firefox   gecko            -moz-
        ie        trident          -ms-
 */
}
/* checked伪类  选中状态 */
input[type="checkbox"]:checked{
        background: blue;
}
```

- 自定义复选框

```
input[type="checkbox"]{
        width: 22px;
        height: 22px;
        -webkit-appearance: none;
        -moz-appearance: none;
        appearance: none;
        background: url(icon_check.png) no-repeat  0 0;
        outline: none;
}
input[type="checkbox"]:checked{
        width: 24px;
        height: 24px;
        background-image: url(icon_checked.png);
}
```

- target伪类

```
/*target 目标伪类，  锚点链接的目标*/
p:target{
        width: 200px;
        height: 200px;
        background: yellow;
}
```

```
<p id="one">第一层</p>
<a href="#one">去第一层</a>
```

边框模拟三角形

```
/*边框模拟三角形:元素的相邻两个边框同时存在时,交点会形成平分,此时将content的区域的宽和高,设置成0,此时会形成一个三角形,将其他的边框的颜色设置成透明*/
.box{
        width: 0px;
        height: 0px;
        border-left:50px solid red;
        border-top:50px solid red;
        border-right:50px solid transparent; /* 透明 */
        border-bottom:50px solid transparent;
}
```

圆角属性

- 单属性：可以是像素值，可以是百分比（宽和高的百分比）

```
/* 左上                  水平和垂直*/
border-top-left-radius: 100px;
/* 左上                  水平    垂直 */
border-top-left-radius: 100px 50px;
/* 右上 */
border-top-right-radius: 100px;
/* 右下 */
border-bottom-right-radius:  20px;
/* 左下 */
border-bottom-left-radius: 30px;
```

- 复合属性

```
/* 一个值  左上右上右下左下*/
border-radius:100px ;
/* 两个值  左上右下 右上左下*/
border-radius:10px 20px ;
/* 三个值  左上  右上左下 右下*/
border-radius:10px 20px 30px ;
/* 四个值  左上 右上 右下 左下*/
border-radius:10px 20px 30px 40px ;
/*              水平/垂直 */
border-radius:100px/50px;
/*         左上 右上 右下 左下水平/左上 右上 右下 左下垂直*/
border-radius: 100px 100px 0 0/50px 50px 0 0;
```

伪元素

元素内部一前一后的两个盒子，伪元素默认是行内显示模式，就是元素中嵌套的元素，伪元素中的文本不会被搜索引擎抓取到

```
.box::before{
        /* 必写项 */
        content: "我是before";
        background:yellow;
        width: 100px;
        height: 100px;
        /* display: block;
        margin: 50px auto; */
        /* float:left; */
        /* position: absolute; */
        /* left:0; */
        /* top:0; */
}
.box::after{
        content:"" ;
        width: 100px;
        height: 100px;
        background: blue;
        display: block;
}
```

```
/* 鼠标移入li让a的伪元素..... */
.box ul li:hover a::after{
        display: block;
}
/* 鼠标移入a让a的伪元素..... */
.box ul li a:hover::after{
        display: block;
}
```

怪异盒子模型

- 标准盒子模型 box-sizing: content-box;
  - 盒子在网页中的尺寸=content+padding+border，当盒子设置padding和border后盒子的尺寸会变大
- 怪异盒子模型 box-sizing: border-box;
  - 盒子在网页中的尺寸=content+padding+border，当盒子设置padding和border后盒子的尺寸不会变大，会从content的区域自动减去，保证和之前width、height的值一样
  - 使用怪异盒子模型一定要设置width和height

文字效果

- 文字阴影属性

```
/* 文字阴影属性  可以设置多组,用逗号隔开*/
/*       水平偏移  垂直偏移  模糊程度(羽化) 颜色 */
text-shadow:  10px 6px 28px red,-10px -10px -10px yellowgreen;
```

- 凹凸效果

```
text-shadow: -1px -1px 0px #000,1px 1px 0px #fff;/*凹*/text-shadow: -1px -1px 0px
#fff,1px 1px 0px #000;/*凸*/
```

- 文字边框

```
/*测试属性 文字边框        边框的粗细    颜色*/
-webkit-text-stroke: 2px yellowgreen;
```

- 文字裁剪

```
background: url(mm.jpg) no-repeat 0 0;
/* 文本裁剪属性  测试属性 */
-webkit-background-clip:text ;
color:transparent
```

盒子效果

- 盒子阴影属性

```
/* 盒子阴影属性 ,可以设置多组,用逗号隔开*/
/*      水平偏移  垂直偏移  模糊程度  扩展 颜色 向内阴影(默认向外的)*/
box-shadow: 0px 0px 0px 0px blue,0px 0px 0px 0px yellowgreen inset;
/* 立体球*/
background: #fff;
border-radius: 50%;
box-shadow: -30px -29px 55px 8px #000 inset;
```

背景效果

- 背景裁剪：背景色/平铺背景图会铺满content+padding+border

不平铺背景图在content+padding

```
/*
   border-box(默认值)  从边框区域可见
   padding-box         从padding区域可见（border不可见）
   content-box         从content区域可见（padding+border不可见）
*/
background-clip: content-box;
```

- 背景起始位置

```
/*
   padding-box(默认值) 起始位置在 padding区域
   border-box 起始位置在border区域
   content-box 起始位置在content区域
*/
background-origin: content-box;
```

- 背景缩放属性

```
/*               宽度       高度 */
background-size: 600px 300px;
/*             宽度和高度 */
background-size:590px ;
/*              元素宽度的百分比   元素高度的百分比 */
background-size: 50% 50%;
/*           元素宽度的百分比,高度等比例缩放 */
background-size:50% ;
/* 图片按照盒子的最长边铺满,为了保证图片不变形,图片会有一部分被截掉 */
background-size: cover;
/* 图片按照盒子的最短边铺满,为了保证图片不变形,盒子会有一部分留白 */
background-size: contain;
```

- 滚动背景和固定背景

```
/*
scroll:（默认值)滚动背景,背景图默认在盒子内部的左上,当页面滚动时,盒子会跟着页面移动,
背景图会跟着盒子移动
fixed: 固定背景,背景图参考点永远是浏览器窗口,当页面滚动时,背景图是不动的,给哪个元素
设置了该背景图,当元素经过浏览器窗口位置的时候,能看到该图片
 */
background-attachment: fixed;
```

- 多重背景

```
/* 背景图可以引入多张,用逗号隔开 */
background: url(img/1.png) no-repeat,url(img/atguigu.png) no-repeat 0 0,url(img/bg.png)
no-repeat 0 0;
```

- 过滤器模糊属性

```
/* 过滤器 */
filter: blur(10px);
/* 全屏遮罩 */
position: absolute;
width: 100%;
height: 100%;
background: rgba(0,0,0,.5);
left:0;
top:0;
/* 过滤器:黑白网页 */
filter: grayscale(100%);
```

- 线性渐变

```
/*线性渐变:沿着一条轴线,从一个颜色值逐渐过渡到另一个颜色值,最少是两个颜色,否则报错
 方向:
 单词:
 to bottom 由上至下
 to top    由下至上
 to right  由左至右
 to left  由右至左
 to left top 到左上
 to right top 到右上
 to left bottom 到左下
 to right bottom 到右下
 角度: 单位deg
 正直是顺时针
 负值时逆时针*/
/*从0px-50px是red,从50px-100px是red到yellow的渐变,从100px-150px是yellow到blue的渐变,从150px-300px是blue*/
background-image: linear-gradient( to right,red 50px,yellow 100px,blue 150px);
/* 省略开头和结束 */
background-image: linear-gradient( to right,red 0px ,red 50px,yellow 100px,blue 150px,blue 300px);
/*从0%-50%是red,从50%-70%是red到yellow的渐变,从70%-90%是yellow到blue的渐变,从90%-100%是blue*/
background-image: linear-gradient( to right,red 50%,yellow 70%,blue 90%);
background-image: linear-gradient( to right,red 0%,red 50%,yellow 70%,blue 90%,blue 100%);
/* 用渐变实现左边是纯红,右边是纯蓝 */
background-image: linear-gradient( to right,red 50%,blue 50%);
background-image: linear-gradient( to right,red 0%,red 50%,blue 50%,blue 100%);
```

- 重复线性渐变

```
/*重复线性渐变:将线性渐变进行平铺效果*/
background-image:repeating-linear-gradient(to right,red 50px,yellow 100px,blue 150px);
```

- 径向渐变

```
/*径向渐变: 从原点到半径之间的一个渐变
 图形:
```

```
 正方向盒子: circle圆 默认值/ellipse椭圆
 长方向盒子: circle圆/ellipse椭圆 默认值
 closest-side:  指定径向渐变的半径长度为从圆心到离圆心最近的边
 closest-corner:  指定径向渐变的半径长度为从圆心到离圆心最近的角
 farthest-side:  指定径向渐变的半径长度为从圆心到离圆心最远的边
 farthest-corner:  指定径向渐变的半径长度为从圆心到离圆心最远的角 */
div:nth-of-type(1){
width: 300px;
height: 300px;
background-image: radial-gradient(circle farthest-side,red 50px,yellow 100px,blue
150px);                  }
div:nth-of-type(2){
width: 300px;
height: 200px;
/*图形  到哪的半径  圆心水平坐标  圆心的垂直坐标, 颜色 值,.. */
background-image: radial-gradient(circle closest-corner at 50px 50px,red 30%,yellow
50%,blue 80%);
margin-top: 5px;}
```

- 重复径向渐变

```
/* 重复径向渐变: 将径向渐变进行重复平铺 */
background-image: repeating-radial-gradient(circle farthest-side,red 20px,yellow
30px,blue 50px);
```

- 圆锥渐变

```
background: conic-gradient(deeppink 0, deeppink 30%, yellowgreen 30%, yellowgreen 70%,
teal 70%, teal 100%);
/*0-30% 的区间使用 deeppink
0-70% 的区间使用 yellowgreen
0-100% 的区间使用 teal
而且，先定义的颜色的层叠在后定义的颜色之上。*/
background: conic-gradient(deeppink 0 30%, yellowgreen 0 70%, teal 0 100%);
```

过渡transition

- 过渡：元素中的属性值，在单位时间内，从一个值逐渐到达某个值，如果希望过渡有来有去，需要将过渡属性写在默认状态下，将过渡最后呈现的状态写在动作事件中
  - 单属性

```
transition-property: width,height,background-color,opacity,display;
transition-property: all;
/* 过渡的时间  单位: 秒s   毫秒ms   1秒=1000毫秒*/
transition-duration: 1s;
/* 过渡的方式     https://cubic-bezier.com/#.05,-0.4,1,1.49 贝塞尔曲线的网址
linear:  线性过渡。等同于贝塞尔曲线(0.0, 0.0, 1.0, 1.0)
ease:  平滑过渡。等同于贝塞尔曲线(0.25, 0.1, 0.25, 1.0)
ease-in:  由慢到快。等同于贝塞尔曲线(0.42, 0, 1.0, 1.0)
ease-out:  由快到慢。等同于贝塞尔曲线(0, 0, 0.58, 1.0)
ease-in-out:  由慢到快再到慢。等同于贝塞尔曲线(0.42, 0, 0.58, 1.0)*/
transition-timing-function: cubic-bezier(.05,-0.4,1,1.49);
/* 过渡的延时 */
transition-delay: .5s;
```

- 复合属性

```
/*        过渡的属性    过渡时间    过渡方式   过渡延时 */
transition: width 1s ease-in .5s;
/*  当制定某个属性过渡时,要用多组的形式,用逗号隔开 */
transition: width 1s ease-in .5s,height 1s ease-in .5s;
transition: all 1s ease-in .5s;
transition-property: width,height;
```

2d转换transform

- 位移

方法：1.定位中的偏移量 2.设置外边距（以前）

```
/* 和相对定位类似 */
/*                      水平,垂直   */
transform: translate(-100px,-100px);
/*              自身宽度的百分比,自身高度的百分比 */
transform: translate(50%,50%);
```

- 旋转：写在动作事件中

```
/* 旋转    单位:deg角度   沿着z轴旋转, 正值是顺时针旋转,负值时逆时针旋转*/
transform: rotate(360000deg);
```

- 旋转轴：写在动作事件中

```
/*  rotate(360deg) 沿着z轴旋转, 正值是顺时针旋转,负值时逆时针旋转*/
/*  rotateX(-180deg) 沿着x轴旋转,从右往左看, 正值是顺时针旋转,负值时逆时针旋转*/
/*  rotateY(-180deg) 沿着y轴旋转,从下往上看, 正值是顺时针旋转,负值时逆时针旋转*/
transform: rotate(360deg);
transform: rotateX(-180deg);
transform: rotateY(-180deg);
```

- 旋转中心点：写在默认状态下

```
/*单词:
 水平: left/center/right
 垂直: top/center/bottom
 数值: 水平 垂直
 沿着z轴旋转有无数个中心点
 沿着x轴旋转,旋转中心点在y轴上
 沿着Y轴旋转,旋转中心点在x轴上*/
transform-origin:50px top ;
```

- 当旋转和位移同时存在时的问题

```
/* 当旋转和位移同时存在时,先写位移,再写旋转 */
transform: translate(500px,300px) rotate(90deg) ;
```

- 缩放

```
/* 缩放     缩放系数*/
/*       一个值:宽度和高度 */
transform: scale(0.2); */
/*           宽度,高度 */
transform: scale(1.2,1.5);
```

动画animation

- 单属性

  ```
  /* 动画名称 */
  animation-name: donghua;
  /* 动画时间 */
  animation-duration: 1s;
  /* 动画方式和过渡一样*/
  animation-timing-function:linear ;
  /* 动画的方向
     normal(默认值) 正常
     reverse 反向  从to到from
     alternate 有来有去,需要两次
     alternate-reverse 先反后正,需要两次
  */
  animation-direction:alternate-reverse;
  /* 动画的次数
     默认1次
     infinite 持久动画
  */
  animation-iteration-count:  infinite;
  /* 动画延时 */
  animation-delay: 1s;
  ```

- 复合属性

```
/* 动画名称 动画时间 动画方式 动画延时 动画方向 动画次数 */animation: donghua 2s ease 1s alternate infinite;
```

- 定义一个动画：定义一个关键帧区间

```
@keyframes donghua{
                        /* 开始 */
                        from{
                                transform: rotate(0deg);
                        }
                        /* 结束 */
                        to{
                                transform: rotate(360deg);
                        }
                        /* 0%{
                                transform: rotate(0deg);
                        }
                        25%{
                                width: 500px;
                                height: 500px;
                        }
                        50%{
                                background: red;
                                transform: rotate(-720deg);
                        }
                        75%{
                                opacity: .5;
                        }
                        100%{
                                transform: rotate(3600deg);
                        } */
                }
```

- 动画库 https://www.dowebok.com/demo/2014/98/

```
<link rel="stylesheet" type="text/css" href="animate.css"/>
```

```
<div class="animated bounceInRight"></div>
```

自适应布局

- 左侧固定右侧自适应

```
.box{
        height: 600px;
        background: #ccc;
}
.left{
        width: 200px;
        height: 600px;
```

```
        background: yellow;
        float:left;
        margin-left: -100%;
}
.right{
        width:100%;
        height: 600px;
        background: yellowgreen;
        float:left;
    /* box-sizing: border-box; */
}
.content{
        padding-left:200px ;
}
```

```
<div class="box">
        <div class="right">
                <div class="content">右</div>
        </div>
        <div class="left">左</div>
</div>
```

- 双飞翼布局

```
.box{
        /* 最小宽度 */
        min-width:800px ;
        /* 最大宽度 */
        /* max-width: 1200px; */
        height: 600px;
        background: #ccc;
}
.left{
        width: 200px;
        height: 600px;
        background: yellow;
        float:left;
        margin-left: -100%;
}
.center{
        width:100%;
        height: 600px;
        background: yellowgreen;
        float:left;
}
.right{
        width: 200px;
        height: 600px;
        background: red;
        float:left;
        margin-left: -200px;
}
```

```
.content{
        padding: 0 200px;
}
```

```
<div class="box">
        <div class="center">
                <div class="content">
                        中间
                </div>
        </div>
        <div class="left">左</div>
        <div class="right">右</div>
</div>
```

- 圣杯布局

```
.box{
        /* 最小宽度 */
        min-width:800px ;
        /* 最大宽度 */
        max-width: 1200px;
        height: 600px;
        background: #ccc;
        margin: 0 200px;
}
.left{
        width: 200px;
        height: 600px;
        background: yellow;
        float:left;
        margin-left: -100%;
        position: relative;
        left:-200px;
}
.center{
        width:100%;
        height: 600px;
        background: yellowgreen;
        float:left;
}
.right{
        width: 200px;
        height: 600px;
        background: red;
        float:left;
        margin-left: -200px;
        position: relative;
        right:-200px;
}
```

```
<div class="box">
    <div class="center">
        中间
    </div>
    <div class="left">左</div>
    <div class="right">右</div>
</div>
```

媒体查询语句

css语句，检测屏幕的宽高来布局，实际上就是一个条件语句，会遵循css的特性

```
/*@media 定义媒体查询的关键字
min-width 最小宽度 大于等于500*/
@media (min-width:500px){
        .box{
                width: 800px;
                height: 800px;
                background: yellowgreen;
        }
}
/* max-width 最大宽度 小于等于800*/
@media (max-width:800px){
        .box{
                width: 80px;
                height: 80px;
                background: yellowgreen;
        }
}
/*
and 条件连接关键字,and前后最少一个空格,否则不起作用
 */
@media (min-width:500px) and (max-width:800px) {
        .box{
                width: 80px;
                height: 80px;
                background: yellow;
        }
}
/* 横竖屏
   orientation
   landscape 横屏
   portrait 竖屏

   only 只有
   screen 彩屏设备
*/
@media (orientation:portrait) and only screen{
        .box{
                width: 80px;
                height: 80px;
                background: yellow;
```

```
        }
}
```

外链式

```
<link rel="stylesheet" type="text/css" href="pad.css" media="(max-width:1000px)"/>
<link rel="stylesheet" type="text/css" href="phone.css" media="(min-width:500px) and (max-width:700px)"/>
```

弹性布局

- 父元素和子元素之间的关系，也叫伸缩布局、弹性盒子、伸缩盒子、flex布局
  - 父元素叫做弹性空间
  - 子元素叫做弹性元素、弹性项
- 父元素开启弹性空间

```
display: flex;
```

- 容器的属性
  - 弹性元素设置主轴/侧轴的方向
    - row（默认）：主轴是水平方向，左边开始，右边结束 侧轴是垂直方向，上边开始，下边结束，相当于是左浮动
    - row-reverse：主轴是水平方向，右边开始，左边结束 侧轴是垂直方向,上边开始，下边结束，相当于是右浮动
    - column：主轴是垂直方向，上边开始，下边结束 侧轴是水平方向，左边开始，右边结束
    - column-reverse：主轴是垂直方向，下边开始，上边结束 侧轴是水平方向，左边开始，右边结束

```
flex-direction: column-reverse;
```

  - 弹性元素是否换行
    - nowrap（默认）：不换行，当弹性元素超出父元素时，会自身进行压缩
    - wrap：换行，当弹性元素超出父元素时，会换行
    - wrap-reverse：反向换行

```
flex-wrap: wrap-reverse;
```

  - flex-direction 属性和flex-wrap 属性的简写形式
    - row nowrap（默认）

```
flex-flow: flex-direction flex-wrap;
```

  - 弹性元素在主轴上的位置分布
    - flex-start（默认）：居开始
    - center：中
    - flex-end：居结束

        - space-around：空间包含元素（项目之间的间隔比项目与边框的间隔大一倍）
        - space-between：元素包含空间
        - space-evenly：空间包含元素（项目之间的间隔比项目与边框的间隔一样大）

```
justify-content:space-evenly ;
```

    - 弹性元素在侧轴的位置分布（在当前行的位置分布）
        - stretch（默认）：在换行时，当弹性元素不设置宽度和高度时，如果主轴是水平方向，宽度是内容撑开的，高度和当前行的高度一样，如果主轴是垂直方向时，高度是内容撑开的，宽度和当前行的宽度一样
        - flex-start：居开始
        - center：居中
        - flex-end：居结束

```
align-items: center;
```

    - 弹性元素在侧轴的位置分布（整体的）
        - stretch（默认）：在换行时，当弹性元素不设置宽度和高度时，如果主轴是水平方向，宽度是内容撑开的，高度和当前行的高度一样，如果主轴是垂直方向时，高度是内容撑开的，宽度和当前行的宽度一样
        - flex-start：居开始
        - center：居中
        - flex-end：居结束
        - space-around：空间包含元素（项目之间的间隔比项目与边框的间隔大一倍）
        - space-between：元素包含空间
        - space-evenly：空间包含元素（项目之间的间隔比项目与边框的间隔一样大）
        - 注意：当align-items和align-content同时存在时，align-items失效

```
align-content:flex-start;
```

- 项目的属性
    - 弹性元素在侧轴的位置分布（当前行，设置个人）

```
align-self: flex-end;
```

    - 弹性元素在主轴的份数，默认为0

```
flex-grow: 1;
```

    - 弹性元素的排序，在整体的最后按编号顺序排列，数值越小，排列越靠前，默认为0

```
order: 1;
```

    - 弹性元素的压缩率（收缩因子），默认值是1
        - 每份压缩的数值px = 超出的数值(200px)/总分数(6) = 33.33px
        - 元素剩余的宽度 = 自身宽度-每份压缩的数值*压缩的份数 = 166.67

```
flex-shrink: 0;
```

移动端适配

- em单位：1em等于当前一个字号的大小
- rem单位：1rem等于html元素一个字号的大小
- 最小字号：浏览器识别的最小字号是12px，如今浏览器识别的最小字号是1px
- 视口：可视窗口，在移动端，由于浏览器默认是屏幕的大小，所以在移动端视口指的就是平铺的大小，移动设备默认出厂时视口的宽度是980px。苹果公司提出，将手机视口的宽度设置和屏幕的宽度一样

```
<!-- 重新定义视口的宽度和设备的宽度一样  meta:vp tab键 -->
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

- vw单位

  1vw = 当前屏幕的1% 100vw = 屏幕的宽度

```
/* 在750px屏幕下（设计稿750px）*/
1vw = 7.5px
100vw = 750px
1px = 0.13333vw
60px = 60*0.13333vw = 8vw
```

- vw单位结合less结合rem单位

```
html {
     /*font-size:0.13333vw 扩大40倍--5.33332vw*/
  font-size: 5.33332vw;
}
.box {
     /*  1vw = 当前屏幕的1%
                  100vw = 屏幕的宽度
                  1.在750px屏幕下
                    1vw = 7.5px
                    100vw = 750px
          2.计算750屏下的60px的盒子占多少vw(设计图)，即所有尺寸下这个盒子占多少vw
                    1px = 0.13333vw
                    60px = 60*0.13333vw = 8vw
  /*设置font-size:0.13333vw时，  width: 60rem;(即8vw)*/
  /*font-size扩大40倍，  width: 60/40rem;(即8vw)*/
  width: 60/40rem;
  height: 1rem;
  background: red
}
```