# CSS3，主讲：汤小洋

## 一、CSS3简介

### 1. 什么是CSS3

CSS3 是最新的 CSS 标准，增加了新的特性

### 2. 兼容性

目前主流浏览器已经支持绝大部分 CSS3 属性

旧版本的浏览器对CSS3支持性会较差，浏览器厂商一般通过添加CSS3前缀来解决，也称为浏览器厂商前缀

| 浏览器 | 厂商前缀 |
| --- | --- |
| Chrome、Safari | -webkit- |
| Firefox | -moz- |
| IE | -ms- |
| Opera | -o- |

### 3. 新增单位rem

em：根据父元素字体大小进行设置，是父元素的倍数

rem：root em 根据HTML根元素字体的大小进行设置，是根元素的倍数

注意：

- HTML根元素默认字体大小为16px，也称为基础字体大小
- Chrome浏览器不支持12px以下的字体大小，其他浏览器没问题
- rem不仅可以设置字体大小，也可以设置元素的宽高，对于适配移动端，可以使用rem

## 二、新增选择器

### 1. 结构性伪类

根据页面中元素的结构来选择元素

#### 1.1 child相关

```
E:first-child/last-child     父元素中的首/末E元素
E:nth-child(n)/nth-last-child(n)     父元素中的正数/倒数第n个E元素，n还支持xn+y、odd、even写法
E:only-child     父元素中唯一的子元素E
```

### 1.2 type相关

```
E:first-of-type/last-of-type     同类型、同级别中的首/末E元素
E:nth-of-type(n)/nth-last-of-type(n)     同类型、同级别中的正数/倒数第n个E元素
E:only-of-type   同类型、同级别中的唯一元素E
```

## 2. 元素状态伪类

根据元素所处的状态来选择元素，类似于 a:hover

```
E:enabled    可用状态的E元素
E:disabled   禁用状态的E元素
E:read-only 只读状态的E元素
E:focus 得到焦点的E元素
E:checked    选中状态的E元素
E::selection     当前选中(通过鼠标画选或Ctrl+A)的E元素，支持的元素、样式有限
```

## 3. 属性选择器

```
E[attr] 有属性attr的E元素
E[attr="val"]    属性attr的值为val的E元素
E[attr^="val"]   属性attr的值以val开头的E元素
E[attr$="val"]   属性attr的值以val结尾的E元素
E[attr*="val"]   属性attr的值包含val的E元素
```

## 4. 其他选择器

```
E:empty 没有任何内部内容(无论是子元素还是文本)的元素
E:not(s)     否定/取反，选择所有不匹配选择器s的E元素
E ~ F    选择同级别中E元素之后的F元素
:root    文档根元素，即<html>元素
:target 选择当前活动的目标锚点
```

# 三、新增样式

## 1. 文本效果

### 1.1 文本阴影

语法：

```
text-shadow: h-shadow v-shadow blur color;
    h-shadow     必需，水平阴影的偏移位置，右偏是正方向
    v-shadow     必需，垂直阴影的偏移位置，下偏是正方向
    blur     可选，模糊的距离
    color    可选，阴影颜色
注：可以指定多组数据，多组数据之间以逗号隔开
```

### 1.2 文本裁剪

语法：

```
text-overflow: clip|ellipsis|string;
    clip     裁剪多余文字
    ellipsis     以省略号表示裁剪文字
    string   以指定字符串表示裁剪文字（非所有浏览器都支持，Firefox支持）
```

### 1.3 单词换行

都是针对西文而言，用来设置是否对长单词进行断句，对中文没意义

语法：

```
word-wrap: normal|break-word;
    normal   使用浏览器默认的换行规则，有可能产生字符串溢出
    break-word   允许在长单词或URL地址中间换行
```

```
word-break: normal|break-all;
    normal 使用浏览器默认的换行规则，有可能产生字符串溢出
    break-all    允许在长单词或URL地址中间换行
```

共同点：都是用来对长单词进行强行断句

不同点：

- word-wrap:break-word 会将长单词放在一个新行里，如果新行还是放不下，则会对长单词进行强制断句
- word-break:break-all 不会把长单词放在一个新行里，如果当前行放不下，则会直接进行强制断句

## 2. 边框

### 2.1 圆角边框

语法：

```
border-radius: 左上 右上 右下 左下;
border-radius是一个简写属性，用于设置四个 border-*-radius 属性，如 border-top-left-radius
```

### 2.2 图片边框

使用图片来创建边框，语法：

```
border-image: border-image-source|border-image-slice|border-image-width|border-image-outset | border-image-repeat
border-image是一个简写属性，用于设置以下属性：
    border-image-source 用在边框的图片的路径
    border-image-slice  图片边框向内偏移
    border-image-width  图片边框的宽度
    border-image-outset 边框图像区域超出边框的量
    border-image-repeat 图像边框是否应平铺(repeat)、铺满(round)或拉伸(stretch)
```

## 3. 背景

### 3.1 背景图片

设置背景图片的大小，语法：

```
background-size: length|percentage|cover|contain;
    length  设置背景图像的宽度和高度，第一个值表示宽度，第二个值表示高度。如果只设置宽度，则高度会根据图片宽高比自动进行缩放
    percentage  以父元素的百分比来设置背景图像的宽度和高度
    cover   保持图片宽高比不变，铺满整个容器的宽高，而图片多出的部分则会被截掉
    contain 保持图片宽高比不变，缩放至图片自身能完全显示出来，所以容器会有留白区域
```

设置多个背景图片，语法：

```
background: url(…) position repeat, url(…) position repeat;
注：可以指定多组数据，多组数据之间以逗号隔开
      多个背景图片重叠时，遵循先占先得原则
```

### 3.2 渐变背景

线性渐变，语法：

```
background-image: linear-gradient(direction,fromColor,toColor...);
    direction    渐变的方向或角度，取值：to top、to right、度数(以deg为单位，按顺时针方向，指定过渡在哪个方向结束)
    fromColor    渐变开始的颜色，从...颜色到...
    toColor 渐变结束的颜色
注：还可以在颜色后面再加一个值，指定各颜色的位置
```

放射性渐变，语法：

```
background-image: radial-gradient(type-position,fromColor,toColor...);
    type-position
    (1)渐变的形状
            circle 圆
      ellipse 椭圆
    (2)设置半径的长度
        closest-side 指定径向渐变的半径从圆心到离圆心最近的边
        closest-corner 指定径向渐变的半径从圆心到离圆心最近的角
        farthest-side 指定径向渐变的半径从圆心到离圆心最远的边
        farthest-corner 指定径向渐变的半径从圆心到离圆心最远的角
    (3)渐变圆心的位置
        关键字：at top、at center...
            具体值：at 100px 100px
    fromColor    渐变开始的颜色，从...颜色到...
    toColor 渐变结束的颜色
```

重复性渐变，语法：

```
background-image: repeating-linear-gradient(direction,fromColor,toColor number);   重复的线性渐变
background-image: repeating-radial-gradient(type-position,fromColor,toColor number);
重复的放射性渐变
```

# 4. 盒子属性

## 4.1 盒子大小

默认情况下，按照盒子模型的规则，一个盒子所占空间为：width/height + padding + border + margin

设置元素的width和height是否包含 内边距padding 和 边框border，语法：

```
box-sizing: content-box|border-box
    content-box 不包含内边距和边框(默认值)
    border-box  包含边框和内边距
注：并不包含margin，也就是说margin是要单独计算大小的
```

### 4.2 尺寸调节

对元素的尺寸进行调整，语法：

```
resize: none|both|horizontal|vertical;
    none      无法调整元素的尺寸
    both      可调整元素的高度和宽度
    horizontal  可调整元素的宽度
    vertical    可调整元素的高度
注：需要配合overflow属性使用，一般设置为auto
```

### 4.3 盒子阴影

为元素添加阴影，语法：

```
box-shadow: h-shadow v-shadow blur spread color inset
    h-shadow     必需，水平阴影的偏移位置，右偏是正方向
    v-shadow     必需，垂直阴影的偏移位置，下偏是正方向
    blur     可选，模糊的距离
    spread   可选，阴影的尺寸
    color    可选，阴影的颜色
    inset    可选，将默认的外部阴影 (outset) 改为内部阴影（inset）
```

## 四、新增效果

## 1. 变形transform

transform 单词本义是改变、转换，能够对元素进行移动、缩放、转动、拉长或拉伸，称为变形或转换

包含以下方法：

- 平移：translate(x位置,y位置)

  语法：

  ```
  /* 分别设置横向和纵向移动多少 */
  transform:translateX(x);
  transform:translateY(y);
  /* 横向移动x，纵向移动y */
  transform:translate(x,y);
  ```

- 缩放：scale(x倍数,y倍数)

  语法：

```
/*分别设置横向和纵向缩放多少*/
transform:scaleX(x);
transform:scaleY(y);
/*横向缩放x，纵向缩放y*/
transform:scale(x,y);
```

- 旋转：rotate(角度deg)

  语法：

```
transform:rotate(度数);
```

- 倾斜：skew(角度deg)

  语法：

```
/* 分别设置横向和纵向倾斜多少 */
transform:skewX(x);
transform:skewY(y);
/* 横向倾斜x，纵向倾斜y */
transform:skew(x,y);
```

- 可以叠加多个变形操作，多个之间以空格隔开

```
transform:scale(2,2) translate(100px,100px) rotate(30deg) skew(10deg,20deg);
```

- 可以指定变形的基准点，默认为元素中心点

```
transform-origin: x轴 y轴;
    x轴  left center right,
    y轴  top center bottom
注：除了可以使用关键字，也可使用像素、百分比来设置
```

## 2. 过渡transition

transition 单词本义就是过渡，过渡就是当元素从一种样式变换为另一种样式时为元素添加效果

包含以下属性：

- transition-property 过渡的CSS属性名
- transition-duration 过渡的时间，默认是 0
- transition-timing-function 过滤的速度曲线，默认是 ease，另有linear、ease-in、ease-out、ease-in-out等
- transition-delay 过渡延时，默认是 0
- transition 简写属性 `transition: property duration timing-function delay;`

## 3. 动画animation

animation 单词本义就是动画，动画是使元素从一种样式逐渐变化为另一种样式的效果

步骤：

1. 定义动画

   使用 `@keyframes` 定义动画过程中的关键帧，其实就是定义不同时间点的CSS样式

   语法：

```
@keyframes 动画的名称{
  百分比{
    样式名:样式值;
  }
  ......
}
注：使用百分比来定义时间点，或通过关键词"from"和"to"，等价于0%和100%，建议使用百分比形式
    0%是动画的开始时间，100%动画的结束时间。
```

2. 调用动画

   在某个元素上应用动画，包含以下属性：

   - animation-name 使用@keyframes定义的动画名称
   - animation-duration 持续时间，默认是 0
   - animation-timing-function 速度曲线，默认是 ease
   - animation-delay 延时时间，默认是 0
   - animation-iteration-count 播放次数，默认是 1，可以是数字，也可以是infinite(无限次)
   - animation-direction 播放方向，默认是 normal表示正常播放，alternate表示正反向轮流播放
   - animation 简写属性 `animation: name duration timing-function delay iteration-count direction;`

## 五、弹性布局

### 1. 简介

Flex弹性布局是CSS3的一种新的页面布局方案，可以简便、快速的实现各种页面布局

- 采用flex布局的元素，称为弹性容器(Flex Container)，简称为容器，也称为弹性盒子
- 容器中的子元素，称为弹性项目(Flex Item)，简称为项目
- 容器中默认存在两条轴，默认水平显示的 `主轴` 和始终要垂直于主轴的 `侧轴` (也叫交叉轴)
- 容器中所有的子元素都是沿着主轴方向显示的

### 2. 常用属性

先设置容器为弹性布局 `display:flex` ，然后设置弹性盒子的相关属性

| 属性名 | 属性值 | 含义 |
| --- | --- | --- |
| flex-direction | row(默认)、row-reverse、column、column-reverse | 主轴的方向，即项目的排列方向 |

| 属性名 | 属性值 | 含义 |
| --- | --- | --- |
| flex-wrap | nowrap(默认)、wrap、wrap-reverse | 如果一行放不下，是否换行 |
| flex-flow | 简写属性 | flex- direction和flex- wrap的简写属性 |
| justify-content | flex-start(默认)、flex-end、center、 space-between、space-around、space-evenly | 项目在主轴上的对齐方式 |
| align-items | stretch(默认)、flex-start、flex-end、center | 项目在交叉轴上的对齐方式 |
| align-content | flex-start(默认)、flex-end、center、 space-between、space-around、space-evenly | 元素换行后设置多行对齐方式，容器必须设置flex-wrap换行 |
| flex | 默认为0，表示不伸缩 | 定义项目的伸缩比例 |
| order | 默认为0，数值越小，排列越靠前 | 定义项目的排列顺序 |

注：前6个为容器的属性，后2个为项目的属性

# 六、网格布局

## 1. 简介

Grid网格布局是一个二维的基于网格的布局系统，它将网页划分成一个个网格，可以任意组合不同的网格，做出各种各样的布局

- 采用grid布局的元素，称为Grid容器(Grid Container)，简称为容器
- 网格容器中的子元素，称为网格项(Grid Item)
- 组成网格线的分界线，称为列网格线和行网格线
- 两个相邻的列网格线和两个相邻的行网格线组成的单元格，称为网格单元格

## 2. 常用属性

先设置容器为网格布局`display:grid`，然后设置网格布局的相关属性

| 样式名 | 属性值 | 含义 |
| --- | --- | --- |
| grid-template-columns | px、%、auto、fr、repeat()、auto-fill | 定义每一列的列宽 |
| grid-template-rows | px、%、auto、fr、repeat()、auto-fill | 定义每一列的行高 |
| grid-column-gap | px | 列与列的间隔 |
| grid-row-gap | px | 行与行的间隔 |
| justify-items | start、end、center、stretch(默认) | 单元格内容的水平对齐 |
| align-items | start、end、center、stretch(默认) | 单元格内容的垂直对齐 |
| justify-content | start(默认)、end、center、space-between、space-around、 space-evenly | 整个网格在容器中的水平对齐 |
| align-content | start(默认)、end、center、space-between、space-around、space-evenly | 整个网格在容器中的垂直对齐 |
| grid-auto-flow | row(默认)、column | 自动放置元素的顺序 |

| 样式名 | 属性值 | 含义 |
| --- | --- | --- |
| grid-column-start/grid-column-end | 简写属性：grid-column | 列的开始和结束 |
| grid-row-start/grid-row-end | 简写属性：grid-row | 行的开始和结束 |

Grid布局和Flex布局的区别：

- Grid 布局与 Flex 布局有一定的相似性，都可以指定容器内部多个项目的位置。但是，它们也存在区别
- Flex 布局是轴线布局，只能指定"项目"针对轴线的位置，可以看作是一维布局
- Grid 布局则是将容器划分成"行"和"列"，产生单元格，然后指定"项目所在"的单元格，可以看作是二维布局
- Grid 布局远比 Flex 布局强大，但兼容性不如Flex布局

## 七、多列布局

### 1. 简介

多列布局就是创建多个列来对文本进行布局，其实就是将文本分成多列进行展示，就像报纸排版那样。

### 2. 常用属性

| 属性名 | 属性值 | 含义 |
| --- | --- | --- |
| column-width | px | 列的宽度 |
| column-count | | 列的数量 |
| columns | 简写属性 | column-width 和 column-count的简写属性 |
| column-fill | balance(默认)、auto | 列的填充方式 |
| column-gap | px | 列之间的间隔 |
| column-rule | 类似于border属性的设置 | 列之间分隔线的宽度、样式、颜色 |
| column-span | 1(默认)、all | 元素是否横跨多列 |

## 八、字体

### 1. 简介

在CSS3之前，网页上只能使用已在用户计算机上安装好的字体。

在CSS3中提供了 `@font-face` 规则，通过 `@font-face` 可以在网页上引用外部的独立字体文件

语法：

```
/* 通过@font-face定义加载外部字体文件 */
@font-face {
    font-family: 自定义字体名称;
    src: 字体文件的路径
}
```

常见的字体文件类型：`.ttf`、`.otf`、`.woff`、`.eot`、`.svg`等

## 2. 阿里矢量图标库

https://www.iconfont.cn

使用步骤：

1. 引入项目下面生成的 fontclass 代码：

```
<link rel="stylesheet" href="./iconfont.css">
```

2. 挑选相应图标并获取类名，应用于页面：

```
<span class="iconfont icon-xxx"></span>
```

# 九、其他

## 1. 媒体查询

`@media`媒体查询可以针对不同的屏幕尺寸设置不同的样式，能够判断设备尺寸进行不同样式的切换，是响应式布局的基础

语法：

```
@media 媒体类型 and （媒体属性） {
  选择器{
    样式规则
  }
}
```

媒体类型：

- all　　所有媒体（默认值）
- screen　　电脑屏幕、平板电脑、手机等彩色屏幕
- print　　打印机
- tv　　电视
- ……

媒体属性：

- min-width　最小宽度
- max-width　最大宽度
- min-height　最小高度
- max-height　最大高度
- ……

## 2. 移动设备响应式

viewport视口是用户网页的可视区域，浏览器实际显示内容的区域，也称为视窗 `Viewport`

为页面增加以下meta标签以适应移动设备

```
<meta name="viewport" content="width=device-width,initial-scale=1.0,minimum-scale=1.0,maximum- scale=1.0,user-scalable=no">
```

常用设置：

- width=device-width 页面大小与屏幕等宽
- initial-scale=1.0 初始缩放比例为1.0（不缩放）
- minimum-scale=1.0 最小缩放比例
- maximum-scale=1.0 最大缩放比例
- user-scalable=no 是否允许用户缩放

## 3. 视口单位

视口单位是根据视口大小来定义的一种相对单位，按照百分比来计算，也称为视窗单位

视口单位主要包括以下4个：

- vw 视口宽度的百分比， `1vw` 表示是视口宽度的 `1%`
- vh 视口高度的百分比 ， `1vh` 表示是视口高度的 `1%`
- vmin 选取当前vw和vh中较小的那个
- vmax 选取当前vw和vh中较大的那个

注：视口单位是相对于视口的高度和宽度设置的，而不是父元素的（CSS百分比是相对于包含它的最近的父元素的高度和宽度）

## 4. CSS3原生变量

CSS中原生的定义变量的语法是： `--变量名:变量值` ，调用变量的语法是： `var(--变量名)`

- 变量的定义和使用，只能在声明块 `{ }` 里面
- 变量命名时不能包含 `$` 、 `[` 、 `^` 、 `(` 、 `%` 等字符

```
:root{
    /* 定义变量 */
  --a:#ccc;
}
div{
  /* 调用变量 */
  background:var(--a);
}
```