CSS部分

5. 如何创建块级格式化上下文BFC，有什么用?

- BFC(Block Formatting Context)是一个独立的渲染区域，让处于BFC内部的元素与外部的元素相互隔离，使内外元素的定位不会相互影响

> 解决Margin塌陷和Margin合并：让父级变成BFC

触发条件(以下任意一条)

1. position:absolute
2. display:inline-block
3. **float:left/right:** 浮动元素产生了浮动流，块级元素看不到，产生了BFC的元素，文本类属性(inline)和文本可以看到
4. overflow:hidden

- BFC布局与普通文档布局有什么区别?
  普通文档流
  - 浮动的元素是不会被父级计算高度
  - 非浮动元素会覆盖浮动元素的位置
  - margin会传递给父级元素
  - 两个相邻元素上下的margin会重叠
    BFC布局
  - 浮动的元素会被父级计算高度(父级元素触发了BFC)
  - 非浮动元素不会覆盖浮动元素的位置(非浮动元素触发了BFC)
  - margin不会传递给父级(父级触发BFC)
  - 属于同一个BFC的两个相邻元素上下margin会重叠
    开发中的应用
  - 阻止margin重叠
  - 可以包含浮动元素 —— 清除内部浮动(清除浮动的原理是两个 div都位于同一个 BFC 区域之中)
  - 自适应两栏布局
  - 可以阻止元素被浮动元素覆盖

55.外边距折叠(collapsing margins)

- 毗邻的两个或多个margin会合并成一个margin，叫做外边距折叠
  - 普通流的块元素垂直方向的margin会折叠

- 触发了BFC的元素，不会和它的子元素折叠
- 浮动流、inline-block和绝对定位的元素margin在垂直方向上不会折叠
- 元素自身的margin-top和bottom相邻时，也会折叠

86.设置元素浮动后，该元素的display值会如何改变

- 设置元素浮动后，display会变成block

11.CSS盒子模型

- 标准W3C盒模型：元素宽度=width+padding+border+margin(box-sizing:content-box)
- 怪异IE盒模型：元素宽度= width+margin(box-sizing:border-box)

通过设置CSS3的box-sizing：inherit继承父元素的属性

12. CSS优先级算法如何计算?

- !important(Infinity)>行间样式(1000)>id(100)>class,属性,伪类(10)>tag,伪元素(1)>通配符*

89.display:inline-block什么时候会显示间距

- 相邻的inline-block元素之间有换行或者空格分隔的情况下会产生间距
- 非inline-block水平元素设置为inline-block也会有水平间距
- 可以借助vertical-align:top消除垂直间距
- **在父级中加font-size:0，在子元素里设置需要的字体大小，消除垂直间距**
- li标签写在同一行可以消除垂直间距，但代码可读性差

16.display:inline-block什么时候不会显示间隙?

- 移除空格
- margin为负值
- font-size为0
- letter-spacing和word-spacing

18.行内元素float:left后是否变为块级元素?

- 行内元素设置浮动后，会更像inline-block元素，这个时候默认宽度不是100%，且可以改变宽高和padding等

25. CSS在性能优化方面的实践

- CSS文件放在head里，不要用@import
- CSS压缩与合并、Gzip压缩

44.知道CSS中有个content属性吗？有什么作用？有什么应用？

> css中的content专门应用在before/after的伪类上，用来插入生成内容。常见的应用是清除浮动

```
.clearfix:after{
    content:' ';
    display:block;
    visibility:hidden;
    height:0;
    clear:both;
}
.clearfix{
    *zoom:1;
}
```

46.水平垂直居中

- 单行文本

```
{
    height:100px;
    line-height:100px;
    text-align:center;
}
```

- 已知高度的块级子元素

```
.container{
    position:relative;
}
.vertical{
    height:300px;
    position:absolute;
    top:50%;
    left:50%;
    margin-top:-150px;
    margin-left:-150px;
}
```

- 绝对定位配合CSS3位移

```
{
    position:absolute;
    top:50%;
    left:50%;
    transform:translate(-50%,-50%)
}
```

- CSS3弹性盒子

```
{
    display:flex;
    justify-content:center;
    align-items:center;
}
```

47.如何使用CSS实现硬件加速?

> 硬件加速是指通过创建独立的复合图层，让GPU来渲染这个图层，从而提高性能

- 一般触发硬件加速的CSS属性有transform（用translateZ(0)来“欺骗”，使硬件加速）、opacity、filter、3D、video和canvas

48.重绘和重排是什么，如何避免?

- 重绘：当渲染树中的元素外观（颜色）发生改变，不影响布局时，产生重绘
- 重排：当渲染书中的元素的布局（尺寸，位置，隐藏状态）发生改变时，产生重排
- JS获取Layout属性值也会引起回流，因为浏览器需要通过回流计算最新值

如何最小化重绘和重排

1. 对元素有复杂操作时，可以先display:none，操作完成后再显示
2. 创建多个DOM节点时，使用DocumentFragment一次性加入document
3. 对Layout进行缓存赋值
4. 避免使用table布局，因为table中一旦触发重排就会导致对其他所有table元素重排
5. 避免使用CSS表达式，因为每次调用都会重新计算值
6. 尽量使用CSS属性简写
7. 批量修改元素样式

58.居中一个div；居中一个浮动元素；居中一个绝对定位的div

居中一个div

```
.div{
    width:200px;
    margin:0 auto;
}
```

居中一个浮动元素

```
.div{
    width:500px;
    height:300px;
    margin:-150px 0 0 -250px;
    position:relative;
    left:50%;
    top:50%;
}
```

居中一个绝对定位div

```
.div{
    position:absolute;
    width:1200px;
    margin: 0 auto;
    top:0;
    bottom:0;
    left:0;
    right:0;
}
```

59.画一个三角形

```
#demo{
    width:0;
    height:0;
    border-width:20px;
    border-style:solid;
    border-color:transparent transparent red transparent;

}
```

71.浏览器是怎样解析CSS选择器的

- 从右到左

75.元素竖向的百分比设定是相对于容器的高度吗

- 元素竖向的百分比是相对容器的宽度，不是高度

90.一个高度自适应的div，里面包含一个100px的div和一个填满剩下高度的div

```
.sub{
    height:calc(100% - 100px)
}


.container{
    position:relative;
}
.sub{
    position:absolute;
    top:100px;
    bottom:0;
    }


.contatiner{
    display:flex;
    flex-direction:column;
}
.sub{
    flex:1;
}
```