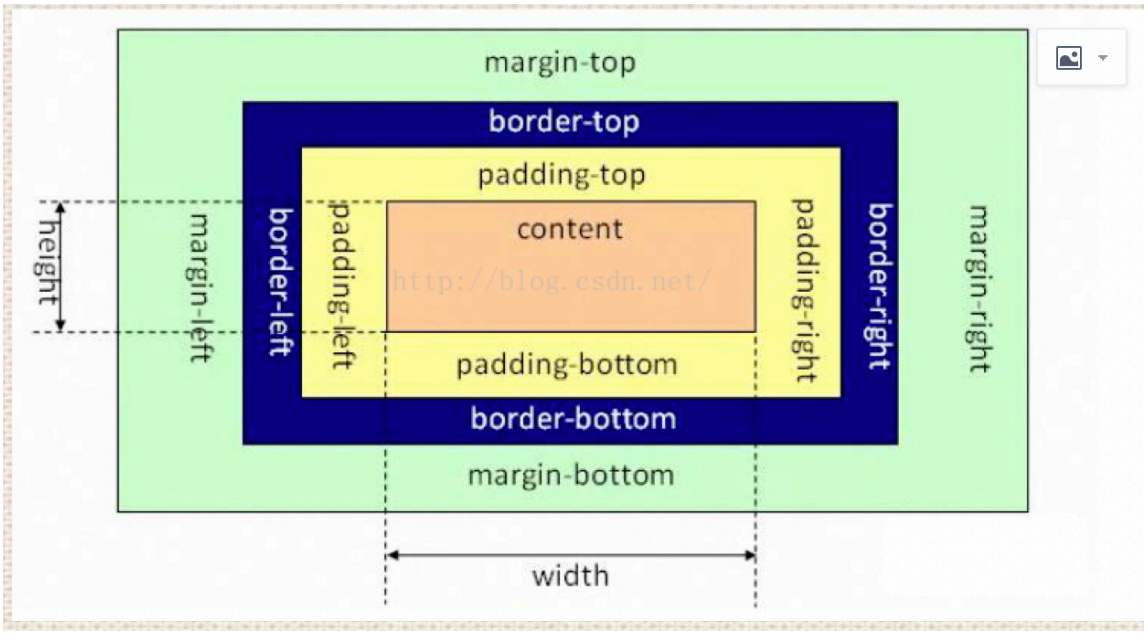
Frontend Interview keypoints

CSS

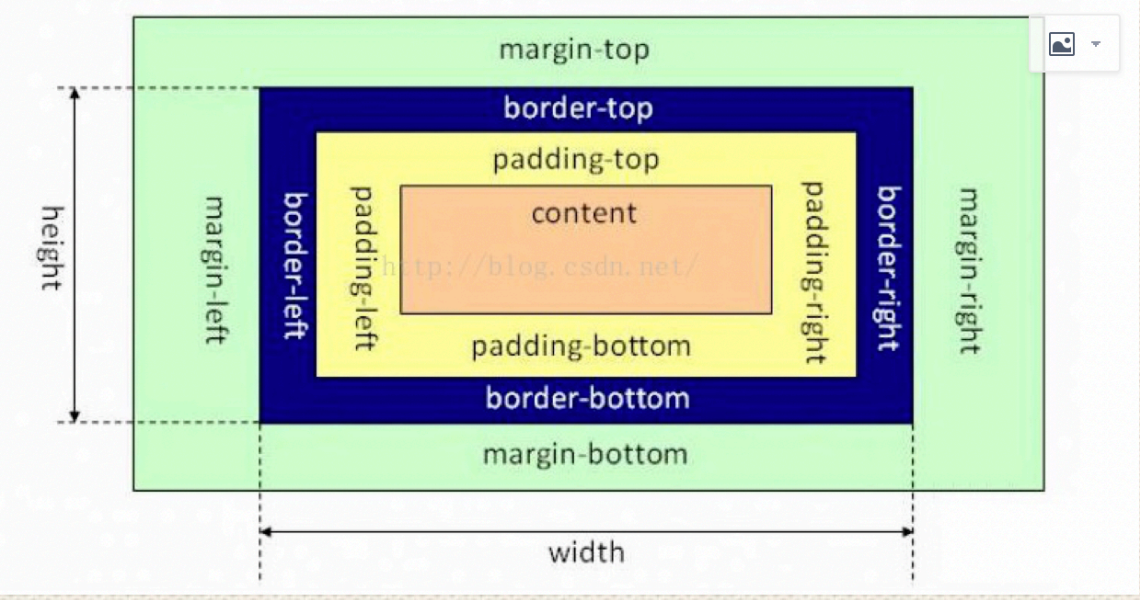
1. 说一下css盒模型

简介：就是用来装页面上的元素的矩形区域。CSS中的盒子模型包括IE盒子模型和标准的W3C盒子模型。box-sizing(有3个值哦)：border-box,padding-box,content-box.

标准盒子模型：



IE盒子模型：



区别：从图中我们可以看出，这两种盒子模型最主要的区别就是width的包含范围，在标准的盒子模型中，width指content部分的宽度，在IE盒子模型中，width表示content+padding+border这三个部分的宽度，故这使得在计算整个盒子的宽度时存在着差异：

标准盒子模型的盒子宽度：左右border+左右padding+width

IE盒子模型的盒子宽度：width

在CSS3中引入了box-sizing属性，box-sizing:content-box;表示标准的盒子模型，box-sizing:border-box表示的是IE盒子模型

最后，前面我们还提到了，box-sizing:padding-box,这个属性值的宽度包含了左右padding+width

也很好理解性记忆，包含什么，width就从什么开始算起。

CSS盒模型本质上是一个盒子，封装周围的HTML元素，它包括：边距，边框，填充，和实际内容。

- 标准盒模型：一个块的总宽度=width+margin(左右)+padding(左右)+border(左右)
- 怪异盒模型：一个块的总宽度=width+margin（左右）（既width已经包含了padding和border值）
- 设置盒模型：box-sizing:border-box

2. 画一条0.5px的线

采用meta viewport的方式:

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

3. link标签和import标签的区别

link属于html标签，而@import是css提供的。

页面被加载时，link会同时被加载，而@import引用的css会等到页面加载结束后加载。

link是html标签，因此没有兼容性，而@import只有IE5以上才能识别。

link方式样式的权重高于@import的。

4. transition和animation的区别

Animation和transition大部分属性是相同的，他们都是随时间改变元素的属性值，他们的主要区别是transition需要触发一个事件才能改变属性，而animation不需要触发任何事件的情况下才会随时间改变属性值，并且transition为2帧，从from .... to，而animation可以一帧一帧的。

5. Flex布局

Flex是Flexible Box的缩写，意为"弹性布局"，用来为盒状模型提供最大的灵活性。

布局的传统解决方案，基于盒状模型，依赖display属性 + position属性 + float属性。它对于那些特殊布局非常不方便，比如，垂直居中就不容易实现。

简单的分为容器属性和元素属性

容器的属性：

```
flex-direction：决定主轴的方向（即子item的排列方法）
.box {
flex-direction: row | row-reverse | column | column-reverse;
}
```

```
flex-wrap：决定换行规则
.box{
flex-wrap: nowrap | wrap | wrap-reverse;
}
```

```
flex-flow：
.box {
flex-flow: <flex-direction> || <flex-wrap>;
}
```

justify-content：对其方式，水平主轴对齐方式

align-items：对齐方式，竖直轴线方向

项目的属性（元素的属性）：

order属性：定义项目的排列顺序，顺序越小，排列越靠前，默认为0

flex-grow属性：定义项目的放大比例，即使存在空间，也不会放大

flex-shrink属性：定义了项目的缩小比例，当空间不足的情况下会等比例的缩小，如果定义个item的flow-shrink为0，则为不缩小

flex-basis属性：定义了在分配多余的空间，项目占据的空间。

flex：是flex-grow和flex-shrink、flex-basis的简写，默认值为0 1 auto。

align-self：允许单个项目与其他项目不一样的对齐方式，可以覆盖align-items，默认属性为auto，表示继承父元素的align-items

比如说，用flex实现圣杯布局

6. BFC（块级格式化上下文，用于清楚浮动，防止margin重叠等）

- 直译成：块级格式化上下文，是一个独立的渲染区域，并且有一定的布局规则。
- BFC区域不会与float box重叠
- BFC是页面上的一个独立容器，子元素不会影响到外面
- 计算BFC的高度时，浮动元素也会参与计算

  那些元素会生成BFC：

  - 根元素
  - float不为none的元素
  - position为fixed和absolute的元素
  - display为inline-block、table-cell、table-caption，flex，inline-flex的元素
  - overflow不为visible的元素

7. 垂直居中的方法

(1)margin:auto法

```
css:

div{
width: 400px;
height: 400px;
position: relative;
border: 1px solid #465468;
}

img{
position: absolute;
margin: auto;
top: 0;
left: 0;
right: 0;
bottom: 0;
}

html:

<div>
<img src="mm.jpg">
</div>
```

定位为上下左右为0，margin：0可以实现脱离文档流的居中.

(2)margin负值法

```
.container{
width: 500px;
height: 400px;
border: 2px solid #379;
position: relative;
}

.inner{
width: 480px;
height: 380px;
background-color: #746;
position: absolute;
top: 50%;
left: 50%;
```

margin-top: -190px; /*height的一半*/

margin-left: -240px; /*width的一半*/

}

补充：其实这里也可以将marin-top和margin-left负值替换成， transform：translateX(-50%)和transform：translateY(-50%)

(3)table-cell（未脱离文档流的）

设置父元素的display:table-cell,并且vertical-align:middle，这样子元素可以实现垂直居中。

```
css:

div{
width: 300px;
height: 300px;
border: 3px solid #555;
display: table-cell;
vertical-align: middle;
text-align: center;
}

img{
vertical-align: middle;
}
```

(4)利用flex

将父元素设置为display:flex，并且设置align-items:center;justify-content:center;

```
css:

.container{
width: 300px;
height: 200px;
border: 3px solid #546461;
display: -webkit-flex;
display: flex;
-webkit-align-items: center;
align-items: center;
-webkit-justify-content: center;
justify-content: center;
}

.inner{
border: 3px solid #458761;
padding: 20px;
}
```

如何实现图片在某个容器中居中的?

- 父元素固定宽高，利用定位及设置子元素margin值为自身的一半。
- 父元素固定宽高，子元素设置position: absolute，margin：auto平均分配margin
- css3属性transform。子元素设置position: absolute; left: 50%; top: 50%;transform: translate(-50%,-50%);即可。
- 将父元素设置成display: table, 子元素设置为单元格 display: table-cell。
- 弹性布局display: flex。设置align-items: center; justify-content: center

如何实现元素的垂直居中

法一：父元素display:flex,align-items:center;

法二：元素绝对定位，top:50%，margin-top：-（高度/2）

法三：高度不确定用transform：translateY（-50%）

法四：父元素table布局，子元素设置vertical-align:center;

8. 关于js动画和css3动画的差异性

渲染线程分为main thread和compositor thread，如果css动画只改变transform和opacity，这时整个CSS动画得以在compositor trhead完成（而js动画则会在main thread执行，然后出发compositor thread进行下一步操作），特别注意的是如果改变transform和opacity是不会layout或者paint的。

区别：

- 功能涵盖面，js比css大
- 实现/重构难度不一，CSS3比js更加简单，性能跳优方向固定
- 对帧速表现不好的低版本浏览器，css3可以做到自然降级
- css动画有天然事件支持
- css3有兼容性问题

CSS动画

- 创建动画序列，需要使用animation属性或其子属性，该属性允许配置动画时间、时长以及其他动画细节，但该属性不能配置动画的实际表现，动画的实际表现是由 @keyframes规则实现，具体情况参见使用keyframes定义动画序列小节部分。
- transition也可实现动画。transition强调过渡，是元素的一个或多个属性发生变化时产生的过渡效果，同

一个元素通过两个不同的途径获取样式，而第二个途径当某种改变发生（例如hover）时才能获取样式，这样就会产生过渡动画。

9. 说一下块元素和行元素

- 块元素：独占一行，并且有自动填满父元素，可以设置margin和pading以及高度和宽度
- 行元素：不会独占一行，width和height会失效，并且在垂直方向的padding和margin会失 效。

10. 多行元素的文本省略号

```
display: -webkit-box
-webkit-box-orient:vertical
-webkit-line-clamp:3
overflow:hidden
```

11. visibility=hidden, opacity=0，display:none

opacity=0，该元素隐藏起来了，但不会改变页面布局，并且，如果该元素已经绑定一些事件，如click事件，那么点击该区域，也能触发点击事件的visibility=hidden，该元素隐藏起来了，但不会改变页面布局，但是不会触发该元素已经绑定的事件display=none，把元素隐藏起来，并且会改变页面布局，可以理解成在页面中把该元素删除掉一样。

12. 双边距重叠问题（外边距折叠）

多个相邻（兄弟或者父子关系）普通流的块元素垂直方向marigin会重叠 折叠的结果为：

- 两个相邻的外边距都是正数时，折叠结果是它们两者之间较大的值。
- 两个相邻的外边距都是负数时，折叠结果是两者绝对值的较大值。
- 两个外边距一正一负时，折叠结果是两者的相加的和。

13. position属性 比较

固定定位fixed： 元素的位置相对于浏览器窗口是固定位置，即使窗口是滚动的它也不会移动。Fixed定位使元素的位置与文档流无关，因此不占据空间。 Fixed定位的元素和其他元素重叠。

相对定位relative： 如果对一个元素进行相对定位，它将出现在它所在的位置上。然后，可以通过设置垂直或水平位置，让这个元素“相对于”它的起点进行移动。 在使用相对定位时，无论是否进行移动，元素仍然占据原来的空间。因此，移动元素会导致它覆盖其它框。

绝对定位absolute： 绝对定位的元素的位置相对于最近的已定位父元素，如果元素没有已定位的父元素，那么它的位置相对于。 absolute 定位使元素的位置与文档流无关，因此不占据空间。 absolute 定位的元素和

其他元素重叠。

粘性定位sticky： 元素先按照普通文档流定位，然后相对于该元素在流中的flow root（BFC）和 containing block（最近的块级祖先元素）定位。而后，元素定位表现为在跨越特定阈值前为相对定位，之后为固定定位。

默认定位Static： 默认值。没有定位，元素出现在正常的流中（忽略top, bottom, left, right 或者 z-index 声明）。

inherit: 规定应该从父元素继承position 属性的值。

14. 浮动清除

方法一：使用带clear属性的空元素

在浮动元素后使用一个空元素如

```
<div class="clear"></div>
```

并在CSS中赋予

```
.clear{clear:both;}
```

属性即可清理浮动。

方法二：使用CSS的overflow属性

给浮动元素的容器添加overflow:hidden;或overflow:auto;可以清除浮动，另外在 IE6 中还需要触发 hasLayout ，例如为父元素设置容器宽高或设置 zoom:1。

在添加overflow属性后，浮动元素又回到了容器层，把容器高度撑起，达到了清理浮动的效果。

方法三：给浮动的元素的容器添加浮动

给浮动元素的容器也添加上浮动属性即可清除内部浮动，但是这样会使其整体浮动，影响布局，不推荐使用。

方法四：使用邻接元素处理

什么都不做，给浮动元素后面的元素添加clear属性。

方法五：使用CSS的:after伪元素

结合:after 伪元素（注意这不是伪类，而是伪元素，代表一个元素之后最近的元素）和 IEhack ，可以完美兼容当前主流的各大浏览器，这里的 IEhack 指的是触发 hasLayout。

给浮动元素的容器添加一个clearfix的class，然后给这个class添加一个:after伪元素实现元素末尾添加一个看不见的块元素（Block element）清理浮动。

15. css3新特性

开放题。

- CSS3边框如border-radius，box-shadow等；CSS3背景如background-size，background-origin等；CSS3 2D，3D转换如transform等；CSS3动画如animation等。
- CSS3的新特性中，在布局方面新增了flex布局，在选择器方面新增了例如first-of-type,nth-child等选择器，在盒模型方面添加了box-sizing来改变盒模型，在动画方面增加了animation，2d变换，3d变换等，在颜色方面添加透明，rbga等，在字体方面允许嵌入字体和设置字体阴影，最后还有媒体查讯等

16. CSS选择器有哪些，优先级呢

- id 选择器，class 选择器，标签选择器，伪元素选择器，伪类选择器等 同一元素引用了多个样式时，排在后面的样式属性的优先级高；
- 样式选择器的类型不同时，优先级顺序为：id 选择器 > class 选择器 > 标签选择器；
- 标签之间存在层级包含关系时，后代元素会继承祖先元素的样式。如果后代元素定义了与祖先元素相同的样式，则祖先元素的相同的样式属性会被覆盖。继承的样式的优先级比较低，至少比标签选择器的优先级低；
- 带有!important 标记的样式属性的优先级最高；
- 样式表的来源不同时，优先级顺序为：内联样式> 内部样式 > 外部样式 > 浏览器用户自定义样式 > 浏览器默认样式

17. 怎么样让一个元素消失

```
display:none; visibility:hidden; opacity: 0;
position移到外部，z-index涂层遮盖等等
```

18. CSS3中对溢出的处理

text-overflow属性，值为clip是修剪文本；ellipsis为显示省略符号来表被修剪的文本；string为使用给定的字符串来代表被修剪的文本。

19. float的元素，display是什么

display为block

20. 三栏布局的实现方式，尽可能多写，浮动布局时，三个div的生成顺序有没有影响

三列布局又分为两种，两列定宽一列自适应，以及两侧定宽中间自适应 两列定宽一列自适应：

1、使用float+margin：

给div设置float：left，left的div添加属性margin-right：left和center的间隔px,right的div添加属性margin-left：left和center的宽度之和加上间隔

2、使用float+overflow：

给div设置float：left，再给right的div设置overflow:hidden。这样子两个盒子浮动，另一个盒子触发bfc达到自适应

3、使用position：

父级div设置position：relative，三个子级div设置position：absolute，这个要计算好盒子的宽度和间隔去设置位置，兼容性比较好，

4、使用table实现：

父级div设置display：table，设置border-spacing：10px//设置间距，取值随意,子级div设置display:table-cell，这种方法兼容性好，适用于高度宽度未知的情况，但是margin失效，设计间隔比较麻烦，

5、flex实现：

parent的div设置display：flex；left和center的div设置margin-right；然后right 的div设置flex：1；这样子right自适应，但是flex的兼容性不好

6、grid实现：

parent的div设置display：grid，设置grid-template-columns属性，固定第一列第二列宽度，第三列auto，

对于两侧定宽中间自适应的布局，对于这种布局需要把center放在前面，可以采用双飞翼布局：圣杯布局，来实现，也可以使用上述方法中的grid，table，flex，position实现

21. calc属性

Calc用户动态计算长度值，任何长度值都可以使用calc()函数计算，需要注意的是，运算符前后都需要保留一个空格，例如：width: calc(100% - 10px)；

22. 有一个width300，height300，怎么实现在屏幕上垂直水平居中

对于行内块级元素

1、父级元素设置text-alig：center，然后设置line-height和vertical-align使其垂直居中，最后设置font-size：0消除近似居中的bug

2、父级元素设置display：table-cell，vertical-align：middle达到水平垂直居中

3、采用绝对定位，原理是子绝父相，父元素设置position：relative，子元素设置position：absolute，然后通过transform或margin组合使用达到垂直居中效果，设置top：50%，left：50%，transform：translate（-50%，-50%）

4、绝对居中，原理是当top,bottom为0时，margin-top&bottom设置auto的话会无限延伸沾满空间并平分，当left，right为0时,margin-left&right设置auto会无限延伸占满空间并平分，

5、采用flex，父元素设置display：flex，子元素设置margin：auto

6、视窗居中，vh为视口单位，50vh即是视口高度的50/100，设置margin：50vh auto 0，transform：translate(-50%)

23. display：table和本身的table有什么区别

Display:table和本身table是相对应的，区别在于，display：table的css声明能够让一个html元素和它的子节点像table元素一样，使用基于表格的css布局，是我们能够轻松定义一个单元格的边界，背景等样式，而不会产生因为使用了table那样的制表标签导致的语义化问题。

之所以现在逐渐淘汰了table系表格元素，是因为用div+css编写出来的文件比用table边写出来的文件小，而且table必须在页面完全加载后才显示，div则是逐行显示，table的嵌套性太多，没有div简洁

24. z-index的定位方法

z-index属性设置元素的堆叠顺序，拥有更好堆叠顺序的元素会处于较低顺序元素之前，z-index可以为负，且z-index只能在定位元素上奏效，该属性设置一个定位元素沿z轴的位置，如果为正数，离用户越近，为负数，离用户越远，它的属性值有auto，默认，堆叠顺序与父元素相等，number，inherit，从父元素继承z-index属性的值

25. 如果想要改变一个DOM元素的字体颜色，不在它本身上进行操作?

可以更改父元素的color

26. 用的最多的css属性是啥?

用的目前来说最多的是flex属性，灵活但是兼容性方面不强。

27. line-height和height的区别

line-height一般是指布局里面一段文字上下行之间的高度，是针对字体来设置的，height一般是指容器的整体

高度。

28. 设置一个元素的背景颜色，背景颜色会填充哪些区域?

background-color设置的背景颜色会填充元素的content、padding、border区域。

29. 知道属性选择器和伪类选择器的优先级吗

属性选择器和伪类选择器优先级相同

30. inline-block、inline和block的区别；为什么img是inline还可以设置宽高

Block是块级元素，其前后都会有换行符，能设置宽度，高度，margin/padding水平垂直方向都有效。

Inline：设置width和height无效，margin在竖直方向上无效，padding在水平方向垂直方向都有效，前后无换行符

Inline-block：能设置宽度高度，margin/padding水平垂直方向 都有效，前后无换行符

31. 用css实现一个硬币旋转的效果

```
#euro {
width: 150px;
height: 150px;
margin-left: -75px;
margin-top: -75px;
position: absolute;
top: 50%;
left: 50%;
transform-style: preserve-3d;
animation: spin 2.5s linear infinite;
}

.back {
background-image: url("/uploads/160101/backeuro.png");
width: 150px;
height: 150px;
}

.middle {
background-image: url("/uploads/160101/faceeuro.png");
width: 150px;
height: 150px;
transform: translateZ(1px);
position: absolute;
top: 0;
}

.front {
background-image: url("/uploads/160101/faceeuro.png");
height: 150px;
position: absolute;
top: 0;
transform: translateZ(10px);
width: 150px;
}

@keyframes spin {
0% {
transform: rotateY(0deg);
}

100% {
transform: rotateY(360deg);
}
}
```

32. 了解重绘和重排吗，知道怎么去减少重绘和重排吗，让文档脱离文档流有哪些方法

DOM的变化影响到了预算内宿的几何属性比如宽高，浏览器重新计算元素的几何属性，其他元素的几何属性也会受到影响，浏览器需要重新构造渲染书，这个过程称之为重排，浏览器将受到影响的部分重新绘制在屏幕上 的过程称为重绘，引起重排重绘的原因有： 添加或者删除可见的DOM元素，

元素尺寸位置的改变

浏览器页面初始化，

浏览器窗口大小发生改变，重排一定导致重绘，重绘不一定导致重排，

减少重绘重排的方法有：

不在布局信息改变时做DOM查询，

使用csstext,className一次性改变属性

使用fragment

对于多次重排的元素，比如说动画。使用绝对定位脱离文档流，使其不影响其他元素

33. CSS画正方体，三角形

画三角形

```
#triangle02{
width: 0;
height: 0;
border-top: 50px solid blue;
border-right: 50px solid red;
border-bottom: 50px solid green;
border-left: 50px solid yellow;
}
```

画正方体：

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>perspective</title>
<style>
.wrapper{
width: 50%;
```

```
float: left;
}
.cube{
font-size: 4em;
width: 2em;
margin: 1.5em auto;
transform-style:preserve-3d;
transform:rotateX(-35deg) rotateY(30deg);
}
.side{
position: absolute;
width: 2em;
height: 2em;
background: rgba(255,99,71,0.6);
border: 1px solid rgba(0,0,0,0.5);
color: white;
text-align: center;
line-height: 2em;
}
.front{
transform:translateZ(1em);
}
.bottom{
transform:rotateX(-90deg) translateZ(1em);
}
.top{
transform:rotateX(90deg) translateZ(1em);
}
.left{
transform:rotateY(-90deg) translateZ(1em);
}
.right{
transform:rotateY(90deg) translateZ(1em);
}
.back{
transform:translateZ(-1em);
}
</style>
</head>
<body>
<div class="wrapper w1">
<div class="cube">
<div class="side front">1</div>
<div class="side back">6</div>
<div class="side right">4</div>
<div class="side left">3</div>
<div class="side top">5</div>
<div class="side bottom">2</div>
```

```
</div>
</div>
<div class="wrapper w2">
<div class="cube">
<div class="side front">1</div>
<div class="side back">6</div>
<div class="side right">4</div>
<div class="side left">3</div>
<div class="side top">5</div>
<div class="side bottom">2</div>
</div>
</div>
</body>
</html>
```

34. overflow的原理

要讲清楚这个解决方案的原理，首先需要了解块格式化上下文，A block formatting context is a part of a visual CSS rendering of a Web page. It is the region in which the layout of block boxes occurs and in which floats interact with each other.翻译过来就是块格式化上下文是CSS可视化渲染的一部分，它是一块区域，规定了内部块盒 的渲染方式，以及浮动相互之间的影响关系

当元素设置了overflow样式且值部位visible时，该元素就构建了一个BFC，BFC在计算高度时，内部浮动元素的高度也要计算在内，也就是说技术BFC区域内只有一个浮动元素，BFC的高度也不会发生塌缩，所以达到了清除浮动的目的，

35. box-sizing的语法和基本用处

- box-sizing规定两个并排的带边框的框，语法为box-sizing：content-box/border-box/inherit
- content-box：宽度和高度分别应用到元素的内容框，在宽度和高度之外绘制元素的内边距和边框
- border-box：为元素设定的宽度和高度决定了元素的边框盒，
- inherit：继承父元素的box-sizing

36. 两个嵌套的div，position都是absolute，子div设置top属性，那么这个top是相对于父元素的哪个位置定位的。

margin的外边缘

37. block、inline、inline-block的区别。

block元素会独占一行，多个block元素会各自新起一行。默认情况下，block元素宽度自动填满其父元素宽度。

block元素可以设置width,height属性。块级元素即使设置了宽度,仍然是独占一行。

block元素可以设置margin和padding属性。

inline元素不会独占一行，多个相邻的行内元素会排列在同一行里，直到一行排列不下，才会新换一行，其宽度随元素的内容而变化。

inline元素设置width,height属性无效。

inline元素的margin和padding属性，水平方向的padding-left, padding-right, margin-left, margin-right都产生边距效果；但竖直方向的padding-top, padding-bottom, margin-top, margin-bottom不会产生边距效果。

inline-block：简单来说就是将对象呈现为inline对象，但是对象的内容作为block对象呈现。之后的内联对象会被排列在同一行内。比如我们可以给一个link（a元素）inline-block属性值，使其既具有block的宽度高度特性又具有inline的同行特性。