1.

overflow 属性规定当内容溢出元素框时发生的事情。

|  |  |
| --- | --- |
| **值** | **描述** |
| visible | 默认值。内容不会被修剪，会呈现在元素框之外。 |
| hidden | 内容会被修剪，并且其余内容是不可见的。 |
| scroll | 内容会被修剪，但是浏览器会显示滚动条以便查看其余的内容。 |
| auto | 如果内容被修剪，则浏览器会显示滚动条以便查看其余的内容。 |
| inherit | 规定应该从父元素继承 overflow 属性的值。 |

2.

transform 属性

transform 属性向元素应用 2D 或 3D 转换。该属性允许我们对元素进行旋转、缩放、移动或倾斜。

|  |  |
| --- | --- |
| **值** | **描述** |
| none | 定义不进行转换。 |
| matrix(n,n,n,n,n,n) | 定义 2D 转换，使用六个值的矩阵。 |
| matrix3d(n,n,n,n,n,n,n,n,n,n,n,n,n,n,n,n) | 定义 3D 转换，使用 16 个值的 4x4 矩阵。 |
| translate(x,y) | 定义 2D 转换。第一个参数是X轴，第二个参数是Y轴。 |
| translate3d(x,y,z) | 定义 3D 转换。 |
| translateX(x) | 定义转换，只是用 X 轴的值。 |
| translateY(y) | 定义转换，只是用 Y 轴的值。 |
| translateZ(z) | 定义 3D 转换，只是用 Z 轴的值。 |
| scale(x,y) | 定义 2D 缩放转换。放大或者缩小 |
| scale3d(x,y,z) | 定义 3D 缩放转换。 |
| scaleX(x) | 通过设置 X 轴的值来定义缩放转换。 |
| scaleY(y) | 通过设置 Y 轴的值来定义缩放转换。 |
| scaleZ(z) | 通过设置 Z 轴的值来定义 3D 缩放转换。 |
| rotate(angle) | 定义 2D 旋转，在参数中规定角度。 |
| rotate3d(x,y,z,angle) | 定义 3D 旋转。 |
| rotateX(angle) | 定义沿着 X 轴的 3D 旋转。 |
| rotateY(angle) | 定义沿着 Y 轴的 3D 旋转。 |
| rotateZ(angle) | 定义沿着 Z 轴的 3D 旋转。 |
| skew(x-angle,y-angle) | 定义沿着 X 和 Y 轴的 2D 倾斜转换。 |
| skewX(angle) | 定义沿着 X 轴的 2D 倾斜转换。 |
| skewY(angle) | 定义沿着 Y 轴的 2D 倾斜转换。 |
| perspective(n) | 为 3D 转换元素定义透视视图。 |

P. s translate (-50%，-50%) 函数是css3的新特性.在不知道自身元素宽高的情况下，可以利用百分比的写法，使它根据自身的宽和高（未知大小）来进行水平方向以及垂直方向的居中。这句话的作用是，向上自动元素自身的宽50%, 向右移动元素自身的50%.

所以结合left:50%, top:50%, translate (-50%，-50%) 就可以将元素居中。如果没有translate(-50%，-50%)那么只是元素的左上角起点这个点水平垂直居中，而不是整个元素块居中。

P.s transform-origin, 设置放大的时候以图片的哪个点作为起点

3.

static：无特殊定位，对象遵循正常文档流。top，right，bottom，left等属性不会被应用。

relative：对象遵循正常文档流，但将依据top，right，bottom，left等属性在正常文档流中偏移位置。而其层叠通过z-index属性定义。

absolute：对象脱离正常文档流，使用top，right，bottom，left等属性进行绝对定位。而其层叠通过z-index属性定义。

fixed：对象脱离正常文档流，使用top，right，bottom，left等属性以窗口为参考点进行定位，当出现滚动条时，对象不会随着滚动。而其层叠通过z-index属性定义。

🡪只有三种情况会使得元素脱离文档流，分别是：浮动、绝对定位和固定定位。

4.

# CSS3的transition属性。Transition属性用于规定当Html元素发生样式发生变化的时候，如果改变。例如当宽度变化，背景图变化，我们使用transition属性可以设置变化的快慢等。如果不设置，则改变宽高，背景图片的时候，是直接改变，一步到位，而没有动画效果。

# 5.

# CSS伪类：

# ::after和::before的使用很简单，可以认为其所在元素上存在一前一后的两个的元素，这两个元素默认是内联元素，但我们可以为其增添样式。::after和::before使用的时候一定要注意，必须设置content，否则这两个伪元素是无法显示出来的。而content属性，会作为这两个伪元素的内容嵌入他们中

# 6.

# position: relative, left, right, top, bottom的赋值。

# left相对于元素的左侧位置，可以理解为屏幕的左边界。正数代表增加多少距离，负数代表减少多少距离。

# right相对于元素的右侧位置，可以理解为屏幕的右边界。正数代表增加多少距离，负数代表减少多少距离。

# top相对于元素的上方位置，可以理解为屏幕的上边界。正数代表增加多少距离，负数代表减少多少距离。

# bottom相对于元素的下方位置，可以理解为屏幕的下边界。正数代表增加多少距离，负数代表减少多少距离。

# 所以position:relative. Left:20px 将让元素向右方移动20px. Position:relative. Right: -20px也能达到一样效果。

# 所以position:relative. Top:20px 将让元素向下方移动20px. Position:relative. bottom: -20px也能达到一样效果。

# 7. CSS3是CSS的第三号版本(2011年发布) ，在此之前还有CSS1(1996年发布), CSS2(1998年发布) 。在每一代CSS版本上提出不一样的特性，例如margin属性便在CSS1便提出, CSS3提供的动画特性。并且不同浏览器的版本对CSS的支持度不一样，现今版本的浏览器对CSS1, CSS2的特性支持度非常高，对CSS3基本支持。

# 在CSS3以后，CSS组织决定今后发展不以CSS4，CSS5进行迭代。而是将CSS3进行模块化发展，升级模块。

# 8.

# CSS中的! important 标记

# CSS中样式来源有用户代理（user-agent declaration），用户（user declaration），开发者申明 (author declaration). 用户代理只浏览器所产生的样式， 用户指用户浏览器选项设置或通过开发人员调试工具修改， 开发者便是developer编写的样式。他们的权重优先级为：

# 用户代理声明（译注：如浏览器默认样式）

# 用户声明（译注：如用户浏览器选项设置或通过开发人员调试工具修改）

# 开发者声明（译注：如页面中引用的CSS）

# 开发者带有!important的声明

# 用户带有!important的声明

# 使用!important的例子：

# #example {

# font-size: 14px! important;

# }

# #container #example {

# font-size: 10px;

# }

# 如果不使用!important，第二个样式声明的代码块很自然地比第一个的权重要大，原因有二：在样式表中第二个代码块要比第一个出现的晚（即，它位列第二）；第二个代码块有更大的权重（是由两个id，#container #example组合而成，而不是只有一个id，#example。但是因为第一个代码块里面包含了! important，所以对于字号设置来说它有更大的权重。

# 🡪如果!important被用于一个简写的样式属性，那么这条简写的样式属性所代表的子属性都会被作用上! important。例如：

.e{margin: 0 !important;}

将等效于

.e{

margin-top: 0! important;

margin-right: 0! important;

margin-bottom: 0! important;

margin-left: 0! important;

}

# 🡪不要滥用!important, 因为! important就扰乱了原本的顺序，让更多的权重给了正常情况下本不应该获得这么多权重的样式

# 9.

# CSS权重： 权重的级别根据选择器被划分为四个分类：行内样式（权重1000），id(权重100) ，类与属性（权重10），以及元素（权重1）。

# 如果两个选择器作用在同一元素上，则权重高者生效。

# 如果两个选择器权重值相同，则最后定义的规则被计算到权重中

# 10.

# CSS3标准的nth-child(N) 选择器，代表选择属于其父元素的第N个子元素。N一般可以写为一个常数，或者是2n+x

# 🡪子元素由1开始计数，没有第0个子元素。N可以取0,1,2,3,4….

# 11.CSS3的border-radius属性

# 使用 CSS3 border-radius 属性，你可以给任何元素制作 "圆角"。

# border-radius可以简写也可以显示申明四个值，指定四个角的圆角角度（顺序为左上，右上，右下，左下，顺时针方向）。

# 12. CSS的单位px, em, rem, vw, vh

# px: px就是pixel像素的缩写，相对长度单位，网页设计常用的基本单位。像素px是相对于显示器屏幕分辨率而言的。（电脑像素是1024x768的，表示的是水平方向是1024个像素点，垂直方向是768个像素点。）

# em: em是相对长度单位。相对于当前对象内文本的字体尺寸（参考物是父元素的font-size）（父元素font-size：16px，当前元素2em=32px）

rem: rem是CSS3新增的一个相对单位，rem是相对于HTML根元素的字体大小（font-size）来计算的长度单位

如果你没有设置html的字体大小，就会以浏览器默认字体大小，一般是16px

# vw, vh:

# vw是相对视口（viewport）的宽度而定的，长度等于视口宽度的1/100，假如浏览器的宽度为200px，那么1vw就等于2px（200px/100）。

# vh是相对视口（viewport）的高度而定的，长度等于视口高度的1/100，假如浏览器的高度为500px，那么1vh就等于5px（500px/100）。

# vmin, vmax:

vmin和vmax是相对于视口的高度和宽度两者之间的最小值和最大值。

如果浏览器的高为300px、宽为500px，那么1vmin就是3px，1vmax就是5px；如果浏览器的高为800px，宽为1080px，那么1vmin也是8px，1vmax也是10.8px

# 13.

# rgba与opacity设置透明度的区别

# 首先rbga的语法为rgba（r,g,b,a）

# - r为红值， 正整数 | 百分数

# - g为绿色值，正整数 | 百分数

# - b为蓝色值，正整数 | 百分数

# - a为alpha（透明度），值为0 ~ 1之间的小数， 0.0 （完全透明）到 1.0（完全不透明）

# - 上面的正整数为十进制0 ~ 255之间的任意值，百分数为0% ~ 100%之间的任意值

# opacity的语法为从 0.0 （完全透明）到 1.0（完全不透明）

# 不同之处在于有opacity属性的所有后代元素都会继承 opacity 属性，而RGBA后代元素不会继承不透明属性

# 14.

# table-layout属性用来显示表格单元格、行、列的算法规则。

# 固定表格布局(fixed)：

# 固定表格布局与自动表格布局相比，允许浏览器更快地对表格进行布局。

# 在固定表格布局中，水平布局仅取决于表格宽度、列宽度、表格边框宽度、单元格间距，而与单元格的内容无关。

# 通过使用固定表格布局，用户代理在接收到第一行后就可以显示表格。

# 自动表格布局(automatic,默认值)：

# 在自动表格布局中，列的宽度是由列单元格中没有折行的最宽的内容设定的。

# 此算法有时会较慢，这是由于它需要在确定最终的布局之前访问表格中所有的内容。

|  |  |
| --- | --- |
| 值 | 描述 |
| automatic | 默认。列宽度由单元格内容设定。 |
| fixed | 列宽由表格宽度和列宽度设定。 |
| inherit | 规定应该从父元素继承 table-layout 属性的值。 |

# 15.