**meta标签有哪些属性？**

<meta charset="UTF-8"> 定义页面编码

<meta name="description" content=""> 定义页面描述

<meta name="keywords" content="HTML,CSS,XML,JavaScript"> 定义文档关键词

<meta name="author" content=""> 定义页面作者

**display:none和visibility:hidden的区别？**

display:none 隐藏对应的元素，在文档布局中不再给它分配空间，它各边的元素会合拢

visibility:hidden 隐藏对应的元素，但是在文档布局中仍保留原来的空间。

**CSS中 link 和@import 的区别？**

1 .link属于HTML标签，而@import是CSS提供的;

1. 页面被加载的时，link会同时被加载，而@import引用的CSS会等到页面被加载完再加载
2. import只在IE5以上才能识别，而link是HTML标签，无兼容问题;
3. link支持使用[Javascript](https://baike.sogou.com/v2484.htm?ch=ww.wap.chain.baike)控制DOM去改变样式；而@import不支持。

**position的absolute与fixed共同点与不同点**

共同点：

1. 改变行内元素的呈现方式，display被置为inline-block；
2. 让元素脱离普通流，不占据空间；
3. 默认会覆盖到非定位元素上

不同点：

1. absolute的根元素是可以设置的，而fixed的根元素固定为浏览器窗口。
2. 当你滚动网页，fixed元素与浏览器窗口之间的距离是不变的。

**介绍一下CSS的盒子模型？**

有两种， IE 盒子模型、标准 W3C 盒子模型；IE的content部分包含了 border 和 padding;

**CSS 选择符有哪些？哪些属性可以继承？优先级算法如何计算？ CSS3新增伪类有那些？**

CSS 选择符：

1.id选择器(# myid) 2.类选择器(.myclassname) 3.标签选择器(div, h1, p) 4.相邻选择器(h1 + p) 5.子选择器(ul > li)

6.后代选择器(li a) 7.通配符选择器( \* ) 8.属性选择器(a[rel = "external"]) 9.伪类选择器(a: hover, li:nth-child)

可继承的样式：

1.font-size 2.font-family 3.color 4.text-indent

不可继承的样式：

1.border 2.padding 3.margin 4.width 5.height

优先级算法：

1.优先级就近原则，同权重情况下样式定义最近者为准;

2.载入样式以最后载入的定位为准;

3.!important >  id > class > tag

4.important 比 内联优先级高，但内联比 id 要高

**CSS3新增伪类举例：**

p:first-of-type 选择属于其父元素的首个 <p> 元素的每个 <p> 元素。

p:last-of-type  选择属于其父元素的最后 <p> 元素的每个 <p> 元素。

p:only-of-type  选择属于其父元素唯一的 <p> 元素的每个 <p> 元素。

p:only-child    选择属于其父元素的唯一子元素的每个 <p> 元素。

p:nth-child(2)  选择属于其父元素的第二个子元素的每个 <p> 元素。

:enabled :disabled 控制表单控件的禁用状态。

:checked        单选框或复选框被选中。

**列出display的值，说明他们的作用**

display 的值的作用：

1.block 象块类型元素一样显示。

2.inline 缺省值。象行内元素类型一样显示。

3.inline-block 象行内元素一样显示，但其内容象块类型元素一样显示。

4.list-item 象块类型元素一样显示，并添加样式列表标记。

**position的值的定位区别：**

1.absolute 生成绝对定位的元素，相对于 static 定位以外的第一个祖先元素进行定位。

2.fixed 生成固定定位的元素，相对于浏览器窗口进行定位（老IE不支持）。

3.relative 生成相对定位的元素，相对于其在普通流中的位置进行定位。

4.static 默认值。没有定位，元素出现在正常的流中（忽略 top, bottom, left, right z-index 声明）。

5.inherit 规定从父元素继承 position 属性的值。

**CSS3有哪些新特性？**

1. CSS3实现圆角（border-radius），阴影（box-shadow），

2. 对文字加特效（text-shadow、），线性渐变（gradient），旋转（transform）

3. transform:rotate(9deg) scale(0.85,0.90) translate(0px,-30px) skew(-9deg,0deg);// 旋转,缩放,定位,倾斜

4. 增加了更多的CSS选择器 多背景 rgba

5. 在CSS3中唯一引入的伪类是 ::selection.

6. 媒体查询，多栏布局

7. border-image

**为什么要初始化CSS样式？**

因为浏览器的兼容问题，不同浏览器对有些标签的默认值是不同的，如果没对CSS初始化往往会出现浏览器之间的页面显示差异。

当然，初始化样式会对SEO有一定的影响，但鱼和熊掌不可兼得，但力求影响最小的情况下初始化。

\*最简单的初始化方法就是（不建议）： { padding:0; margin:0 }

**box-sizing的应用**

box-sizing 属性允许您以特定的方式定义匹配某个区域的特定元素。（个人认为可以理解为指定盒模型的类型，即上述两种类型）

box-sizing值为content-box时：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。（即 标准盒模型）

box-sizing值为border-box时：为元素设定的宽度和高度决定了元素的边框盒，就是说，为元素指定的任何内边距和边框都将在已设定的宽度和高度内进行绘制，通过从已设定的宽度和高度分别减去边框和内边距才能得到内容的宽度和高度。（即 怪异盒模型）

box-sizing值为inherit时：规定应从父元素继承 box-sizing 属性的值。

**元素单位px、em、rem的区别**

px,相对单位，相对于屏幕分辨率而言

em，相对单位，相对于当前对象内文本的字体尺寸，如果未设置，则相对于浏览器的默认字体尺寸（16px）/ 继承父级元素的字体大小

rem，CSS3新增的相对单位，相对的是HTML根元素的大小。（em是相对于每个单独对象的字体），只修改根类就可以做到成比例地调整所有字体大小

如何选择

只需要适配少部分设备，且分辨率影响不大，使用px即可

需要适配各种设备，且分辨率差异较大，使用rem （单位的选择也是响应式设计的内容之一）

**解释下浮动和它的工作原理？清除浮动的技巧**

特点：浮动元素脱离文档流，不占据空间。浮动元素碰到包含它的边框或者浮动元素的边框时停留。

浮动元素引起的问题：

1. 父元素的高度无法被撑开，影响与父元素同级的元素

2. 与浮动元素同级的非浮动元素会跟随其后

3. 若非第一个元素浮动，则该元素之前的元素也需要浮动，否则会影响页面显示的结构

清除浮动的方法：

1. 使用空标签清除浮动。这种方法是在所有浮动标签后面添加一个空标签 定义css clear:both. 弊端就是增加了无意义标签。

#parent:after{

    content:".";  height:0;

    visibility:hidden; display:block;

   clear:both; }

2. 使用overflow。可以给父元素设置overflow：auto或者hidden。给包含浮动元素的父标签添加css属性 overflow:auto; zoom:1; zoom:1用于兼容IE6。

3. 使用after伪对象清除浮动。 该方法只适用于非IE浏览器。具体写法可参照以下示例。使用中需注意以下几点。一、该方法中必须为需要清除浮动元素的伪对象中设置 height:0，否则该元素会比实际高出若干像素；

4.浮动外部元素

**解释下 CSS sprites，以及你要如何在页面或网站中使用它**

CSS Sprites 其实就是把网页中一些背景图片整合到一张图片文件中，再利用 CSS 的"background-image"，"background-repeat"，"background-position" 的组合进行背景定位，background-position 可以用数字能精确的定位出背景图片的位置。这样可以减少很多图片请求的开销，因为请求耗时比较长；请求虽然可以并发，但是也有限制，一般浏览器都是6个。对于未来而言，就不需要这样做了，因为有了 http2。

**什么是js阻塞，什么是css阻塞？**

JS的阻塞特性：所有浏览器在下载 JS 的时候，会阻止一切其他活动，比如其他资源的下载，内容的呈现等等。直到 JS 下载、解析、执行完毕后才开始继续并行下载其他资源并呈现内容。为了提高用户体验，新一代浏览器都支持并行下载 JS，但是 JS 下载仍然会阻塞其它资源的下载（例如.图片，css文件等）。

由于浏览器为了防止出现 JS 修改 DOM 树，需要重新构建 DOM 树的情况，所以就会阻塞其他的下载和呈现。

嵌入 JS 会阻塞所有内容的呈现，而外部 JS 只会阻塞其后内容的显示，2 种方式都会阻塞其后资源的下载。也就是说外部样式不会阻塞外部脚本的加载，但会阻塞外部脚本的执行。

CSS 本来是可以并行下载的，在以下情况下会出现阻塞加载(在测试观察中，IE6 下 CSS 都是阻塞加载）

当 CSS 后面跟着嵌入的 JS 的时候，该 CSS 就会出现阻塞后面资源下载的情况。而当把嵌入 JS 放到 CSS 前面，就不会出现阻塞的情况了。

根本原因：因为浏览器会维持 html 中 css 和 js 的顺序，样式表必须在嵌入的 JS 执行前先加载、解析完。而嵌入的 JS 会阻塞后面的资源加载，所以就会出现上面 CSS 阻塞下载的情况。

嵌入JS应该放在什么位置？

1. 放在底部，虽然放在底部照样会阻塞所有呈现，但不会阻塞资源下载。

2. 如果嵌入JS放在head中，请把嵌入JS放在CSS头部。

3. 使用 defer（只支持IE）

4. 不要在嵌入的JS中调用运行时间较长的函数，如果一定要用，可以用 setTimeout 来调用

Javascript无阻塞加载具体方式：

1. 将脚本放在底部。<link>还是放在head中，用以保证在js加载前，能加载出正常显示的页面。<script>标签放在</body>前。

2. 阻塞脚本：由于每个<script>标签下载时阻塞页面解析过程，所以限制页面的<script>总数也可以改善性能。适用于内联脚本和外部脚本。

3. 非阻塞脚本：等页面完成加载后，再加载js代码。也就是，在 window.onload 事件发出后开始下载代码。

<script>

    var script=document.createElement("script");

    script.type="text/javascript";

    script.src="file.js";

   document.getElementsByTagName("head")[0].appendChild(script);

</script>

4. defer属性：支持IE4和fierfox3.5更高版本浏览器

5. 动态脚本元素：文档对象模型（DOM）允许你使用js动态创建HTML的几乎全部文档内容。代码如下：

此技术的重点在于：无论在何处启动下载，文件额下载和运行都不会阻塞其他页面处理过程，即使在head里（除了用于下载文件的 http 链接）。