# CSS面试题

### 1 display: none; 与 visibility: hidden; 的区别

（1）如果给一个元素设置了display: none，那么该元素以及它的所有后代元素都会隐藏，

给元素设置visibility: hidden也可以隐藏这个元素，但是隐藏元素仍需占用与未隐藏时一样的空间，也就是说虽然元素不可见了，但是仍然会影响页面布局

（2）display: none;是非继承属性，子孙节点消失由于元素从渲染树消失造成，通过修改子孙节点属性无法显示；visibility:hidden;是继承属性，子孙节点消失由于继承了hidden，通过设置visibility: visible;可以让子孙节点显式

（3） 修改常规流中元素的display通常会造成文档重排，修改visibility属性只会造成本元素的重绘

**重绘、重排指的是什么**

**重绘（repaint或redraw）：**

重绘是指当 DOM 元素的属性发生变化 (如 color) 时, 浏览器会通知render 重新描绘相应的元素, 此过程称为重绘。

**重排（重构）：**

重排是指某些元素变化涉及元素布局 (如width), 浏览器则抛弃原有属性, 重新计算，此过程称为重排。（重排一定会重绘，重绘不一定重排）

**回流（reflow）：**

重排好的结果，传递给render以重新描绘页面元素, 此过程称为回流

如何优化重排重绘

1.元素的多次样式修改合并成一次修改；

2.如需进行对DOM节点进行多次操作，先将其脱离文本流之后再进行多次操作；

3.table布局的渲染与普通DOM节点的操作相比，性能消耗更大，如果可以，尽量减少table布局的使用；

4.缓存常用的布局信息；

### **2 link 与 @import 的区别**

（1）link 是HTML方式， @import 是CSS方式

（2）link最大限度支持并行下载，@import 过多嵌套导致串行下载，出现FOUC

（3）link 可以通过 rel="alternate stylesheet" 指定候选样式

（4）浏览器对 link 支持早于@import ，可以使用 @import 对老浏览器隐藏样式

（5）@import 必须在样式规则之前，可以在css文件中引用其他文件

总体来说：link优于@import

### 3 外边距折叠及解决

块级元素的上下外边距(margin)在某些情况下会合并（折叠）起来，合并之后的大小为较大的margin。注意浮动和绝对定位的元素的margin从不折叠。

（1）垂直方向上相邻的两个元素  
这种情况很好理解，就是上面的例子中给出的情况。这里要注意一点，所谓相邻指的是渲染之后的元素位置相邻，而不是dom结构上的相邻

（2）父元素和第一个／最后一个子元素之间

**2 外边距折叠后的大小**

1. 两个相同大小的正数：取某个外边距的值。即30px与30px发生折叠，折叠后的值为30px。  
   2.两个不同大小的正数：取较大的外边距的值。即30px与20px发生折叠，折叠后的值为30px。  
   3.一正一负：取正数与负数的和。即30px与-20px发生折叠，折叠后的值为10px。  
   4.相同大小的负数：同相同大小正数。-30px与-30px折叠，折叠后为-30px。  
   5.不同大小负数： 取绝对值较大的负数。-30px与-20px折叠，折叠后为-30px。

**3 解决**

1. 直接设置margin-bottom:50px;而不用30+20
2. 加浮动或定位
3. BFC

### 4 介绍一下标准的CSS的盒子模型？低版本IE的盒子模型有什么不同的？如何切换？

1 有两种， IE 盒子模型、W3C 盒子模型；

2 盒模型： 内容(content)、填充(padding)、边界(margin)、 边框(border)；

3 区 别： IE的content部分把 border 和 padding计算了进去;

4 切换：通过box-sizing：

box-sizing: content-box; // 默认的标准(W3C)盒模型元素效果，border和padding不计算入width之内

box-sizing: border-box; // 触发怪异(IE)盒模型元素的效果，border和padding计算入width之内

box-sizing: inherit; // 继承父元素 box-sizing 属性的值

### 5 CSS优先级算法如何计算？

优先级就近原则，同权重情况下样式定义最近者为准

载入样式以最后载入的定位为准

优先级为: !important > id > class > tag ，important 比 内联优先级高

选择器优先级: 行内样式[1000] > id[100] > class[10] > Tag[1]

important：无限高

行内样式：权重值为1000

id选择器：权重值为100

类、伪类、属性选择器：权重值为10

元素选择器：权重值为1

在同一组属性设置中，!important 优先级最高，高于行内样式

!important > 行内样式>ID选择器 > 类选择器/属性/伪类 > 标签 > 通配符 > 继承 > 浏览器默认属性

### 6 CSS3新增伪类有那些？

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。

:after 在元素之前添加内容,也可以用来做清除浮动。

:before 在元素之后添加内容

:enabled

:disabled 控制表单控件的禁用状态。

:checked 单选框或复选框被选中

### 7 CSS3有哪些新特性？

新增各种CSS选择器 （: not(.input)：所有 class 不是“input”的节点）

圆角 （border-radius:8px）

多列布局 （multi-column layout）

阴影和反射 （Shadow\Reflect）

文字特效 （text-shadow、）

文字渲染 （Text-decoration）

线性渐变 （gradient）

旋转 （transform）

增加了旋转,缩放,定位,倾斜,动画

### 8 用纯CSS创建一个三角形的原理是什么？

// 把上、左、右三条边隐藏掉（颜色设为 transparent）

#demo {

width: 0;

height: 0;

border-width: 20px;

border-style: solid;

border-color: transparent transparent red transparent;

}

### 9 经常遇到的浏览器的兼容性有哪些？原因，解决方法是什么，常用hack的技巧 ？

浏览器默认的margin和padding不同。解决方案是加一个全局的\*{margin:0;padding:0;}来统一

IE下,可以使用获取常规属性的方法来获取自定义属性,也可以使用getAttribute()获取自定义属性;Firefox下,只能使用getAttribute()获取自定义属性。

解决方法:统一通过getAttribute()获取自定义属性

浏览器兼容问题二：块属性标签float后，又有横行的margin情况下，在IE6显示margin比设置的大

问题症状:常见症状是IE6中后面的一块被顶到下一行 碰到频率：90%（稍微复杂点的页面都会碰到，float布局最常见的浏览器兼容问题）

解决方案：在float的标签样式控制中加入 display:inline;将其转化为行内属性

浏览器兼容问题五：图片默认有间距

问题症状：几个img标签放在一起的时候，有些浏览器会有默认的间距，加了问题一中提到的通配符也不起作用。 碰到几率：20%

解决方案：使用float属性为img布局 备注：因为img标签是行内属性标签，所以只要不超出容器宽度，img标签都会排在一行里，但是部分浏览器的img标签之间会有个间距。去掉这个间距使用float是正道。（我的一个学生使用负margin，虽然能解决，但负margin本身就是容易引起浏览器兼容问题的用法，所以我禁止他们使用）

### 10 display:inline-block什么时候会显示间隙？

display-inline-block是让元素在一行显示，但是这些元素在html里面是上下行排列的，所以中间有换行符，于是并排显示就有了换行符带来的空隙。

**方案一：**将html标签要display:inline-block 的元素写在一行。缺点：代码可读性差。

**方案二：**给父元素设置font-size:0,缺点是子元素如果里面有文字，文字会消失不见，所以又要给子元素重新设置font-size,增加了代码量。

**方案三：**给元素设置float:left,缺点高度塌陷，要清楚浮动。

**方案四：**设置子元素的margin-left为负值，但是元素之间的间隙大小是根据上下文的字体大小确定的，而每个浏览器的换行空隙大小不同，如果font-szie：16px,chrome空隙为8px,火狐空隙为4px.所以这个方法不通用。

**方案五：**设置父元素 display:table;word-spacing:-1em;目前这个方法可以完美解决，且兼容其他浏览器。

### 11 浮动及清除浮动

（1）浮动元素会脱离文档流并向左/向右浮动，直到碰到父元素或者另一个浮动元素

（2）清除浮动主要是为了解决父级元素因为子级浮动引起的内部高度为0的问题。

清除方法：1 给父盒子定义一个高度

2双伪元素清除浮动：给父盒子添加一个clearfix类

.clearfix:before,

.clearfix:after {

display: table;

content: ""; /\*不用有内容也可以\*/

}

.clearfix:after {

clear: both;

}

.clearfix {

\*zoom: 1;

}

### 12 box-sizing 常用的属性有哪些？分别有什么作用？

box-sizing: content-box; // 默认的标准(W3C)盒模型元素效果

box-sizing: border-box; // 触发怪异(IE)盒模型元素的效果

box-sizing: inherit; // 继承父元素 box-sizing 属性的值

### 13 请列举几种隐藏元素的方法

（1）visibility: hidden; 简单的隐藏某个元素，但是元素占用的空间任然存在

（2）position: absolute; 设置一个很大的 left 负值定位，使元素定位在可见区域之外

（3）display: none; 元素会变得不可见，并且不会再占用文档的空间。

（4）height: 0; 将元素高度设为 0 ，并消除边框

### 14 rgba() 和 opacity 的透明效果有什么不同？

rgba() 只作用于元素自身的颜色或其背景色，子元素不会继承透明效果

opacity 作用于元素以及元素内的所有内容（包括文字）的透明度

### 15  CSS3 的 Flexbox及应用场景

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，Flex布局，可以简便、完整、响应式地实现各种页面布局

使用场景主要是屏幕自适应布局和取代浮动布局。（传统的父盒子A包含B,C两个子盒子，B向左靠齐，C向右靠齐......）

### 16 li与li之间有看不见的空白间隔是什么原因引起的？有什么**解决办法？**

li排列受到中间空白(回车/空格)等的影响，因为空白也属于字符，会被应用样式占据空间，产生间隔

解决办法：在ul设置设置font-size=0,在li上设置需要的文字大小

### 17 px、em 、rem在表现上有什么区别？

px 相对于显示器屏幕分辨率，无法用浏览器字体放大功能

em 值并不是固定的，em为相对长度单位，它是用来设置文本的字体尺寸的，它是相对于当前对象内文本的字体尺寸；会继承父级的字体大小： em = 像素值 / 父级font-size，一般浏览器默认1em=16px

rem是css3新增的一个相对长度单位，它的出现是为了解决em的缺点，em可以说是相对于父级元素的字体大小，当父级元素字体大小改变时，又得重新计算。rem出现就可以解决这样的问题，rem只相对于根目录，即HTML元素。所以只要在html标签上设置字体大小，文档中的字体大小都会以此为参照标准，一般用于自适应布局

### 18 CSS优化、提高性能的方法有哪些？

多个css合并，尽量减少HTTP请求

将css文件放在页面最上面

移除空的css规则

避免使用CSS表达式

选择器优化嵌套，尽量避免层级过深

充分利用css继承属性，减少代码量

抽象提取公共样式，减少代码量

属性值为0时，不加单位

属性值为小于1的小数时，省略小数点前面的0

css雪碧图

### 19 说一说你使用过的CSS布局

gird布局，layout布局，flex布局，双飞翼，圣杯布局等

### 20 rem和em的区别

在css中单位长度用的最多的是px、em、rem，这三个的区别是：

　　px是固定的像素，一旦设置了就无法因为适应页面大小而改变。

　　em和rem相对于px更具有灵活性，他们是相对长度单位，意思是长度不是定死了的，更适用于响应式布局。

对于em和rem的区别一句话概括：**em相对于父元素，rem相对于根元素**。

rem中的r意思是root（根源），这也就不难理解了。

### 21 Position 值的有哪些？分别相对于什么定位？

absolute :生成绝对定位的元素， 相对于最近一级的 定位不是 static 的父元素来进行定位。

fixed （老IE不支持）生成绝对定位的元素，通常相对于浏览器窗口或 frame 进行定位。

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。

static 默认值。没有定位，元素出现在正常的流中

sticky 生成粘性定位的元素，容器的位置根据正常文档流计算得出

### 22 行内元素和块级元素区别：

1.行内元素与块级元素直观上的区别二、行内元素与块级元素的三个区别

    行内元素会在一条直线上排列（默认宽度只与内容有关），都是同一行的，水平方向排列。

    块级元素各占据一行（默认宽度是它本身父容器的100%（和父元素的宽度一致），与内容无关），垂直方向排列。块级元素从新行开始，结束接着一个断行。

2.块级元素可以包含行内元素和块级元素。行内元素不能包含块级元素，只能包含文本或者其它行内元素。

3.行内元素与块级元素属性的不同，主要是盒模型属性上：行内元素设置width无效，height无效(可以设置line-height)，margin上下无效，padding上下无效

### 23 利用flex布局实现左右固定，中间自适应？

<div class="content">

<div class="item-left"></div>

<div class="item-center"></div>

<div class="item-right"></div>

</div>  
.content {

display: flex;

align-items: center;

.item-left {

width: 94px;

flex: 0 0 94px;//不放大 不缩小 固定宽度94px

}

.item-center {

flex: 1 1 auto;//可放大 可缩小

}

.item-right {

width: 16px;

flex: 0 0 16px;//不放大 不缩小 固定宽度94px

}

}

### 24 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。

### 25 CSS模块化优点

1. 浏览器原生支持

2. 方便删除某个模块

3. 便于管理各种日常需求

4. 方便合作开发

5. 方便多人协作

### 26 CSS3动画（重点）

CSS3 的动画的三个属性transform、transition、animation（属性，举个例子）。transform我们可以理解为元素的几何变形，它是有规律可寻的，这种变形并不会产生动画效果仅仅是原有形状的改变；transition和animation它们很像 flash 中的补间动画和逐帧动画；transition是从一个状态变化到另外一种状态，当变化有了平滑的效果后就产生了动画，它是一个公式化的变化，在比较规则的动画效果中我们可以使用，例如：旋转的风车、行驶的汽车、颜色的渐变等等；animation的动画效果更加灵活，可以实现像影片一样的复杂无规则的动画。

transition和animation的区别

1、transition需要去触发比如：点击事件、鼠标移入事件；而animation则是在页面加载后就触发

2、transition触发一次播放一次；而animation则是可以设置播放次数可以一直播放

3、transition关注的是样式属性的变化，属性值和时间的关系是一个三次贝塞尔曲线；而animation作用于元素本身而不是样式属性，可以使用关键帧的概念，可以实现更自由的动画效果。

### 27 CSS3动画和js动画区别

1 js实现的是帧动画

2 css实现的是补间动画

**帧动画：使用定时器，每隔一段时间，更改当前的元素  
补间动画：过渡（加过渡只要状态发生改变产生动画）动画(多个节点来控制动画)性能会更好**

### 28 position中的absolute和relative的区别

1 relative 不脱离文档流，absolute 脱离文档流。

2 absolute是绝对定位，绝对定位就是相对于父元素的定位，不受父元素内其他子元素的影响；而relative是相对定位，相对定位是相对于同级元素的定位，也就是上一个同级元素。

3relative相对于其在正常文档流中的位置偏移，相当于块级元素，默认情况宽度为父元素宽度。 relative只能设置left和top，设置right和bottom无效。

absolute把其父元素看作一个盒子的话，它默认会被放在盒子的左上角，但绝对定位的元素相当于行级元素，宽度根据内容定，注意有一点小区别，可以设置宽高，相当于inline-block。absolute可以设置left，right，top,bottom,并且可以同时设置，宽高会自动计算出来。

### 29 水平垂直居中

**居中元素知道宽高：**

1. absolute + 负margin

<div class="wp">

<div class="box size">123123</div>

</div>

*/\* 公共代码 \*/*

.wp {

border: 1px solid red;

width: 300px;

height: 300px;

}

.box {

background: green;

}

.box.size{

width: 100px;

height: 100px;

}*/\* 公共代码 \*/*

.wp {

position: relative;

}

.box {

position: absolute;;

top: 50%;

left: 50%;

margin-left: -50px;

margin-top: -50px;

}

1. absolute + margin auto

.wp {

position: relative;

}.box {

position: absolute;;

top: 0;

left: 0;

right: 0;

bottom: 0;

margin: auto;

}

1. absolute + calc（css3新增计算属性）

.wp {

position: relative;

}.box {

position: absolute;;

top: calc(50% - 50px);

left: calc(50% - 50px);

}

**居中元素不知道宽高：**

1. absolute + transform

.wp {

position: relative;

}.box {

position: absolute;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

}

1. Flex

.wp {

display: flex;

justify-content: center;

align-items: center;

}

### 30 浏览器最小显示的字体大小 如果想要显示更小怎么办

最小12px

可以利用css中的transform属性来缩小：

**transform:scale(0.5,1)**

第一个参数是X轴比例，第二个参数是Y轴比例，第二个参数可省略，若缺省则等于第一个参数。

**注意：**

①transform只有块级元素有这属性，一开始我把这个属性设置在span上，怎么都不出效果，换div就有效果了

②如果你是因为字体太大，放div中会自动换行，想用这方法来压缩字体，去除换行，那就省省吧，我刚踩过这个坑。这种方式并不是在压缩字体，而是在压缩整个div。整个div变窄了。

最后我的方法是，把div宽度加长，右margin设为负数，再进行压缩

### 31 CSS选择器

标签选择器

类选择器：.

Id选择器:#

子选择器：以>隔开父子级元素，(模块名>模块名，修饰>前模块内的子模块)

包含选择器：以空格隔开包含关系的元素

兄弟选择器：以~隔开兄弟关系的元素

相邻选择器：以+隔开相邻关系的元素

全局选择器：\*

群选择器：以，分隔

属性选择器

伪类选择器：first-child{}，last-child{}，nth-child{} 等

### 32 CSS选择器的匹配过程

CSS选择器的读取顺序是从右向左。

例子： **.mod-nav h3 span{font-size:16px;}**

先找到所有的span，沿着span的父元素查找h3，中途找到了符合匹配规则的节点就加入结果集；如果直到根元素html都没有匹配，则不再遍历这条路径，从下一个span开始重复这个过程（如果有多个最右节点为span的话）（索引树）

33 CSS精灵图（雪碧图）  
 css sprites直译过来就是CSS精灵。通常被解释为“CSS图像拼合”或“CSS贴图定位”。其实就是通过将多个图片融合到一张图里面，然后通过CSS background背景定位技术技巧布局网页背景。这样做的好处也是显而易见的，因为图片多的话，会增加http的请求，无疑促使了网站性能的减低，特别是图片特别多的网站，如果能用css sprites降低图片数量，带来的将是速度的提升。

css sprites是什么通俗解释：CSS Sprites其实就是把网页中一些背景图片整合拼合成一张图片中，再利用DIV 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在布局盒子对象位置。

### 34 CSS精灵图和字体图标区别

字体图标：可以缩放的矢量图标。你可以使用CSS对它们进行修改:大小，颜色，阴影等。体积特别的小。

| **对比角度** | **CSS雪碧图** | **字体图标** |
| --- | --- | --- |
| 做动画 | 图片做动画较合适 | 做动画的话，不如图片 |
| 高分辨率 | 图片在高分辨率下会模糊 | 字体图标是文字，受分辨率影响小 |
| 多色展示 | 图片多色图片展示很好实现 | 多色图标目前来看比较难实现 |
| 图标变换颜色或者其他效果 | 需要换一个图片 | 改变一下文字的color就好 |
| 文件体积 | 图片体积较大 | 字体图标体积较小 |
| 实现简单icon | 体积大、变化不灵活 | 体积小、方便添加效果 |

CSS雪碧图和字体图标因为本质上的不同（一个是图片，一个是文字），造成了其所适用的场景不同。

在一些对分辨率要求不高、展示的图标色彩丰富、图标变化不多的情况下，可以选用雪碧图。而对于那种多场景多分辨率下展示、单图标颜色单一、图标的颜色变化较多的情况下，比较适合使用字体图标。

### 35什么是BFC以及什么情况下产生BFC

块格式化上下文（Block Formatting Context）是一个独立的渲染区域，简单来说就是一个独立的盒子，并且这个盒子里的布局不受外部影响，也不会影响到外部元素。

**触发BFC：**

· 根元素（html就是根元素，整个html就是一个独立的BFC）

· float属性不为none（如：left | right）

· overflow的值不为visible（如：hidden | auto | scroll）

· display属性值为inline-block | flex | inline-flex | table-cell | table-caption

· position为absolute或fixed  
**应用BFC：**

· **解决margin塌陷问题**也就是解决，在一个标准流body（body元素就是一个BFC）中相邻盒子之间垂直margin重叠的问题。  
方法：触发其中一个盒子的BFC，成为一个独立的容器，根据BFC规则，这个盒子的布局不受外部元素影响。

·  **解决高度塌陷问题**当一个标准流中的盒子中所有的子元素都进行了浮动，并且没有给盒子设置高度时，那么这个盒子的整个高度就会塌陷，浮动的子元素高度不计算在父元素内，父元素高度就为0。  
方法：触发这个盒子生成BFC，根据规则计算BFC的高度时，浮动元素也参与计算。  
例如：1设置父元素浮动;2使用clearfix;3在浮动元素后加个div设置clear: both; height:0,overflow:hidden

### 36 CSS3 filter:drop-shadow滤镜与box-shadow区别应用

1 box-shadow兼容性比filter:drop-shadow更好

2 drop-shadow没有内阴影效果

3 drop-shadow不能阴影叠加

4 阴影 vs 盒阴影：比如用border做一个虚线盒子，box-shadow顾名思意“盒阴影”，只是盒子的阴影；你想啊，这盒子中间明明是透明的，结果，阴影的时候，居然光线没有穿透；但是drop-shadow就符合真实世界的投影，非透明的颜色，我就有投影；透明部分，光线穿过，没投影，而什么盒子不盒子的，跟我没有任何关系。

### 37 offset和style区别

1 offset可以得到任意样式表中的样式值，style只能得到行内样式表中的样式值

2 offset获得的数值是没有单位的，style.width是有单位的

3 offsetWidth包含padding+border+width，style.width不包含padding+border

4 offset只读，不能赋值，而style.width是可读写的

5 获取元素大小位置，用offset更合适，给元素更改值，用style

### 38 伪类和伪元素的区别：

伪元素主要是用来创建一些不存在原有dom结构树种的元素，例如：用::before和::after在一些存在的元素前后添加文字样式等，这些被添加的内容会以具体的UI显示出来，被用户所看到的，这些内容不会改变文档的内容，不会出现在DOM中，不可复制，仅仅是在CSS渲染层加入

伪类表示已存在的某个元素处于某种状态，但是通过dom树又无法表示这种状态，就可以通过伪类来为其添加样式。例如a元素的:hover, :active等

**伪元素与伪类的根本区别在于：操作的对象元素是否存在于原来的dom结构里。**

### 39 vh和vw单位是什么？

vw：视窗宽度的百分比（1vw 代表视窗的宽度为 1%）  
vh：视窗高度的百分比  
vmin：当前 vw 和 vh 中较小的一个值  
vmax：当前 vw 和 vh 中较大的一个值

### 40 CSS可继承的属性

不可继承的：**display、margin、border、padding、background、height、min-height、max- height、width、min-width、max-width、overflow、position、left、right、top、 bottom、z-index、float**

可继承的：**line-height、color、font、 font-family、font-size、font-style、font-variant、font-weight、text- decoration、list-style、list-style-type**

### 41 CSS 预处理器框架作用

1.常见的 css 预处理器

1.Less(Node.js)  2.Sass (Ruby 但有 Node 版本）

2.预处理器作用

1 帮助更好地组织 CSS 代码

2 提高代码复用率

3 提升可维护性

3.预处理器能力

1 嵌套  反映层级和约束

2 变量和计算  减少重复代码

3 Extend 和 Mixin 代码片段

4 循环  适用于复杂有规律的样式

1. mport CSS 文件模块化

### 42 css中兼容ie浏览器的前缀是什么

-ms代表【ie】内核识别码

-moz代表火狐【firefox】内核识别码

-webkit代表谷歌【chrome】/苹果【safari】内核识别码

-o代表欧朋【opera】内核识别码

### 43 SVG和canvas区别

**1 绘制的图片格式不同**：Canvas 的工具getContext 绘制出来的图形或传入的图片都依赖分辨率，能够以 .png 和 .jpg格式保存存储图像，可以说是位图

SVG 可以在H5中直接绘制，但绘制的是矢量图

**2 Canvas不支持事件处理器，SVG支持事件处理器**

**3 适用范围不同**：Canvas是逐像素进行渲染的，一旦图形绘制完成，就不会继续被浏览器关注。而SVG是通过DOM操作来显示的。

所以Canvas的文本渲染能力弱，而SVG最适合带有大型渲染区域的应用程序，比如地图。

而Canvas 最适合有许多对象要被频繁重绘的图形密集型游戏。

而SVG由于DOM操作 在复杂度高的游戏应用中 会减慢渲染速度。所以不适合在游戏应用。

**[使用 isPointInPath 函数来设置canvas事件](https://www.cnblogs.com/xlx123/p/7978495.html)**

### 44 getElementsByClassName 和 querySelectorAll 的区别

getElementsByClassName会动态查询，会随着dom结构的变化，得到的结点列表也会发生变化和

querySelectorAll得到的是一个静态列表，不会对DOM进行动态查询