$CSS部分

CSS选择器有哪些

1. ***通用选择器**：选择所有元素，**不参与计算优先级**，兼容性IE6+
2. **#X id选择器**：选择id值为X的元素，兼容性：IE6+
3. **.X 类选择器**： 选择class包含X的元素，兼容性：IE6+
4. **X Y后代选择器**： 选择满足X选择器的后代节点中满足Y选择器的元素，兼容性：IE6+
5. **X 元素选择器**： 选择标所有签为X的元素，兼容性：IE6+
6. **:link，：visited，：focus，：hover，：active链接状态**： 选择特定状态的链接元素，顺序LoVe HAte，兼容性: IE4+
7. **X + Y直接兄弟选择器**：在**X之后第一个兄弟节点**中选择满足Y选择器的元素，兼容性： IE7+
8. **X > Y子选择器**： 选择X的子元素中满足Y选择器的元素，兼容性： IE7+
9. **X ~ Y兄弟**： 选择**X之后所有兄弟节点**中满足Y选择器的元素，兼容性：IE7+
10. **[attr]**：选择所有设置了attr属性的元素，兼容性IE7+
11. **[attr=value]**：选择属性值刚好为value的元素
12. **[attr~=value]**：选择属性值为空白符分隔，其中一个的值刚好是value的元素
13. **[attr|=value]**：选择属性值刚好为value或者value-开头的元素
14. **[attr^=value]**：选择属性值以value开头的元素
15. **[attr$=value]**：选择属性值以value结尾的元素
16. **[attr*=value]**：选择属性值中包含value的元素
17. **[:checked]**：选择单选框，复选框，下拉框中选中状态下的元素，兼容性：IE9+
18. **X:after, X::after**：after伪元素，选择元素虚拟子元素（元素的最后一个子元素），CSS3中::表示伪元素。兼容性:after为IE8+，::after为IE9+
19. **:hover**：鼠标移入状态的元素，兼容性a标签IE4+， 所有元素IE7+
20. **:not(selector)**：选择不符合selector的元素。**不参与计算优先级**，兼容性：IE9+
21. **::first-letter**：伪元素，选择块元素第一行的第一个字母，兼容性IE5.5+
22. **::first-line**：伪元素，选择块元素的第一行，兼容性IE5.5+

23. **:nth-child(an + b)**：伪类，选择前面有an + b - 1个兄弟节点的元素，其中n
    >= 0， 兼容性IE9+
24. **:nth-last-child(an + b)**：伪类，选择后面有an + b - 1个兄弟节点的元素其中n >= 0，兼容性IE9+
25. **X:nth-of-type(an+b)**：伪类，X为选择器，**解析得到元素标签**，选择**前面**有an + b - 1个**相同标签**兄弟节点的元素。兼容性IE9+
26. **X:nth-last-of-type(an+b)**：伪类，X为选择器，解析得到元素标签，选择**后面**有an+b-1个相同**标签**兄弟节点的元素。兼容性IE9+
27. **X:first-child**：伪类，选择满足X选择器的元素，且这个元素是其父节点的第一个子元素。兼容性IE7+
28. **X:last-child**：伪类，选择满足X选择器的元素，且这个元素是其父节点的最后一个子元素。兼容性IE9+
29. **X:only-child**：伪类，选择满足X选择器的元素，且这个元素是其父元素的唯一子元素。兼容性IE9+
30. **X:only-of-type**：伪类，选择X选择的元素，**解析得到元素标签**，如果该元素没有相同类型的兄弟节点时选中它。兼容性IE9+
31. **X:first-of-type**：伪类，选择X选择的元素，**解析得到元素标签**，如果该元素
    是此此类型元素的第一个兄弟。选中它。兼容性IE9+

css sprite是什么,有什么优缺点

概念：将多个小图片拼接到一个图片中。通过background-position和元素尺寸调节需要显示的背景图案。

优点：

1. 减少HTTP请求数，极大地提高页面加载速度
2. 增加图片信息重复度，提高压缩比，减少图片大小
3. 更换风格方便，只需在一张或几张图片上修改颜色或样式即可实现

缺点：

1. 图片合并麻烦
2. 维护麻烦，修改一个图片可能需要从新布局整个图片，样式

`display: none;`与`visibility: hidden;`的区别

联系：它们都能让元素不可见

区别：

1. display:none;会让元素完全从渲染树中消失，渲染的时候不占据任何空间；visibility: hidden;不会让元素从渲染树消失，渲染师元素继续占据空间，只是内容不可见
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。
4. 读屏器不会读取display: none;元素内容；会读取visibility: hidden;元素内容

css hack原理及常用hack

原理：利用**不同浏览器对CSS的支持和解析结果不一样**编写针对特定浏览器样式。常见的hack有1）属性hack。2）选择器hack。3）IE条件注释

- IE条件注释：适用于[IE5, IE9]常见格式如下

```
<!--[if IE 6]>
Special instructions for IE 6 here
<![endif]-->
```

- 选择器hack：不同浏览器对选择器的支持不一样

```
/***** Selector Hacks ******/

/* IE6 and below */
* html #uno  { color: red }

/* IE7 */
*:first-child+html #dos { color: red }

/* IE7, FF, Saf, Opera  */
html>body #tres { color: red }

/* IE8, FF, Saf, Opera (Everything but IE 6,7) */
html>/**/body #cuatro { color: red }

/* Opera 9.27 and below, safari 2 */
html:first-child #cinco { color: red }

/* Safari 2-3 */
html[xmlns*=""] body:last-child #seis { color: red }

/* safari 3+, chrome 1+, opera9+, ff 3.5+ */
body:nth-of-type(1) #siete { color: red }

/* safari 3+, chrome 1+, opera9+, ff 3.5+ */
body:first-of-type #ocho {  color: red }

/* saf3+, chrome1+ */
media screen and (-webkit-min-device-pixel-ratio:0) {
 #diez  { color: red  }
}

/* iPhone / mobile webkit */
@media screen and (max-device-width: 480px) {
 #veintiseis { color: red  }
}

/* Safari 2 - 3.1 */
html[xmlns*=""]:root #trece  { color: red  }

/* Safari 2 - 3.1, Opera 9.25 */
*|html[xmlns*=""] #catorce { color: red  }
```

```
42.
43. /* Everything but IE6-8 */
44. :root *> #quince { color: red  }
45.
46. /* IE7 */
47. *+html #dieciocho {  color: red }
48.
49. /* Firefox only. 1+ */
50. #veinticuatro,  x:-moz-any-link  { color: red }
51.
52. /* Firefox 3.0+ */
53. #veinticinco,  x:-moz-any-link, x:default  { color: red
      }
```

- 属性hack：不同浏览器解析bug或方法

```
1. /* IE6 */
2. #once { _color: blue }
3.
4. /* IE6, IE7 */
5. #doce { *color: blue; /* or #color: blue */ }
6.
7. /* Everything but IE6 */
8. #diecisiete { color/**/: blue }
9.
10. /* IE6, IE7, IE8 */
11. #diecinueve { color: blue\9; }
12.
13. /* IE7, IE8 */
14. #veinte { color/*\**/: blue\9; }
15.
16. /* IE6, IE7 -- acts as an !important */
17. #veintesiete { color: blue !ie; } /* string after ! can
     be anything */
```

specified value,computed value,used value计算方法

- specified value: 计算方法如下：

    1. 如果样式表设置了一个值，使用这个值
    2. 如果没有设置值，这个属性是继承属性，从父元素继承
    3. 如果没设置，并且不是继承属性，使用css规范指定的初始值

- computed value: 以specified value根据规范定义的行为进行计算，通常将相对值计算为绝对值，例如em根据font-size进行计算。一些使用百分数并且需要布局来决定最终值的属性，如width，margin。百分数就直接作为computed value。line-height的无单位值也直接作为computed value。这些值将在计算used value时得到绝对值。**computed value的主要作用是用于继承**
- used value：属性计算后的最终值，对于大多数属性可以通过window.getComputedStyle获得，尺寸值单位为像素。以下属性依赖于布局，

    - background-position
    - bottom, left, right, top
    - height, width
    - margin-bottom, margin-left, margin-right, margin-top
    - min-height, min-width
    - padding-bottom, padding-left, padding-right, padding-top
    - text-indent

`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
6. 总体来说：**link优于@import**

`display: block;`和`display: inline;`的区别

`block`元素特点：

1.处于常规流中时，如果`width`没有设置，会自动填充满父容器
2.可以应用`margin/padding`
3.在没有设置高度的情况下会扩展高度以包含常规流中的子元素
4.处于常规流中时布局时在前后元素位置之间（独占一个水平空间）
5.忽略`vertical-align`

`inline`元素特点

1.水平方向上根据`direction`依次布局
2.不会在元素前后进行换行
3.受`white-space`控制
4.`margin/padding`在竖直方向上无效，水平方向上有效
5.`width/height`属性对非替换行内元素无效，宽度由元素内容决定
6.非替换行内元素的行框高由`line-height`确定，替换行内元素的行框高由`height`,`margin`,`padding`,`border`决定
6.浮动或绝对定位时会转换为`block`
7.`vertical-align`属性生效

PNG,GIF,JPG的区别及如何选

参考资料： 选择正确的图片格式
GIF:

1. 8位像素，256色
2. 无损压缩
3. 支持简单动画
4. 支持boolean透明
5. 适合简单动画

JPEG:

1. 颜色限于256
2. 有损压缩
3. 可控制压缩质量
4. 不支持透明
5. 适合照片

PNG:

1. 有PNG8和truecolor PNG
2. PNG8类似GIF颜色上限为256，文件小，支持alpha透明度，无动画
3. 适合图标、背景、按钮

CSS有哪些继承属性

- 关于文字排版的属性如：
  - font
  - word-break
  - letter-spacing
  - text-align
  - text-rendering
  - word-spacing
  - white-space
  - text-indent
  - text-transform
  - text-shadow

- line-height
- color
- visibility
- cursor

IE6浏览器有哪些常见的bug,缺陷或者与标准不一致的地方,如何解决

- IE6不支持min-height，解决办法使用css hack：

```
.target {
    min-height: 100px;
    height: auto !important;
    height: 100px;   // IE6下内容高度超过会自动扩展高度
}
```

- `ol`内`li`的序号全为1，不递增。解决方法：为li设置样式`display: list-item;`
- 未定位父元素`overflow: auto;`，包含`position: relative;`子元素，子元素高于父元素时会溢出。解决办法：1）子元素去掉`position: relative;`；2）不能为子元素去掉定位时，父元素`position: relative;`

```
<style type="text/css">
.outer {
    width: 215px;
    height: 100px;
    border: 1px solid red;
    overflow: auto;
    position: relative;  /* 修复bug */
}
.inner {
    width: 100px;
    height: 200px;
    background-color: purple;
    position: relative;
}
</style>

<div class="outer">
    <div class="inner"></div>
</div>
```

- IE6只支持`a`标签的`:hover`伪类，解决方法：使用js为元素监听mouseenter，mouseleave事件，添加类实现效果：

```
<style type="text/css">
.p:hover,
.hover {
    background: purple;
}
</style>

<p class="p" id="target">aaaa
bbbbb<span>DDDDDDDDDDDd</span> aaaa lkjlkjdf j</p>

<script type="text/javascript">
function addClass(elem, cls) {
    if (elem.className) {
        elem.className += ' ' + cls;
    } else {
        elem.className = cls;
    }
}
function removeClass(elem, cls) {
    var className = ' ' + elem.className + ' ';
    var reg = new RegExp(' +' + cls + ' +', 'g');
    elem.className = className.replace(reg, ' ').replac
e(/^ +| +$/, '');
}

var target = document.getElementById('target');
if (target.attachEvent) {
    target.attachEvent('onmouseenter', function () {
        addClass(target, 'hover');
    });
    target.attachEvent('onmouseleave', function () {
        removeClass(target, 'hover');
    })
}
</script>
```

- IE5-8不支持`opacity`，解决办法：

```
.opacity {
    opacity: 0.4
    filter: alpha(opacity=60); /* for IE5-7 */
    -ms-filter: "progid:DXImageTransform.Microsoft.Alpha(Opacity=60)"; /* for IE 8*/
}
```

- IE6在设置`height`小于`font-size`时高度值为`font-size`，解决办法：`font-size: 0;`
- IE6不支持PNG透明背景，解决办法: **IE6下使用gif图片**
- IE6-7不支持`display: inline-block`解决办法：设置inline并触发hasLayout

```
    display: inline-block;
    *display: inline;
    *zoom: 1;
```

- IE6下浮动元素在浮动方向上与父元素边界接触元素的外边距会加倍。解决办法：
  1）使用padding控制间距。
  2）浮动元素`display: inline;`这样解决问题且无任何副作用：css标准规定浮动元素display:inline会自动调整为block
- 通过为块级元素设置宽度和左右margin为auto时，IE6不能实现水平居中，解决方法：为父元素设置`text-align: center;`

容器包含若干浮动元素时如何清理(包含)浮动

1. 容器元素闭合标签前添加额外元素并设置`clear: both`
2. 父元素触发块级格式化上下文(见块级可视化上下文部分)
3. 设置容器元素伪元素进行清理推荐的清理浮动方法

```
/**
* 在标准浏览器下使用
* 1 content内容为空格用于修复opera下文档中出现
*   contenteditable属性时在清理浮动元素上下的空白
* 2 使用display使用table而不是block：可以防止容器和
*   子元素top-margin折叠,这样能使清理效果与BFC，IE6/7
*   zoom: 1;一致
**/

.clearfix:before,
.clearfix:after {
    content: " "; /* 1 */
    display: table; /* 2 */
}

.clearfix:after {
    clear: both;
}

/**
* IE 6/7下使用
* 通过触发hasLayout实现包含浮动
**/
.clearfix {
    *zoom: 1;
}
```

什么是FOUC?如何避免

Flash Of Unstyled Content：用户定义样式表加载之前浏览器使用默认样式显示文档，用户样式加载渲染之后再从新显示文档，造成页面闪烁。**解决方法**：把样式表放到文档的`head`

如何创建块级格式化上下文(block formatting context),BFC有什么用

创建规则：

1. 根元素
2. 浮动元素（`float`不是`none`）
3. 绝对定位元素（`position`取值为`absolute`或`fixed`）
4. `display`取值为`inline-block`,`table-cell`,`table-caption`,`flex`,`inline-flex`之一的元素
5. `overflow`不是`visible`的元素

作用:

1. 可以包含浮动元素
2. 不被浮动元素覆盖
3. 阻止父子元素的margin折叠

display,float,position的关系

1. 如果`display`为none，那么position和float都不起作用，这种情况下元素不产生框
2. 否则，如果position值为absolute或者fixed，框就是绝对定位的，float的计算值为none，display根据下面的表格进行调整。
3. 否则，如果float不是none，框是浮动的，display根据下表进行调整
4. 否则，如果元素是根元素，display根据下表进行调整
5. 其他情况下display的值为指定值

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。规则如下:

1. 两个或多个毗邻的普通流中的块元素垂直方向上的margin会折叠
2. 浮动元素/inline-block元素/绝对定位元素的margin不会和垂直方向上的其他元素的margin折叠
3. 创建了块级格式化上下文的元素，不会和它的子元素发生margin折叠
4. 元素自身的margin-bottom和margin-top相邻时也会折叠

如何确定一个元素的包含块(containing block)

1. 根元素的包含块叫做初始包含块，在连续媒体中他的尺寸与viewport相同并且anchored at the canvas origin；对于paged media，它的尺寸等于

page area。初始包含块的direction属性与根元素相同。
2. `position`为`relative`或者`static`的元素，它的包含块由最近的块级（`display`为`block`,`list-item`，`table`）祖先元素的**内容框**组成
3. 如果元素`position`为`fixed`。对于连续媒体，它的包含块为viewport；对于paged media，包含块为page area
4. 如果元素`position`为`absolute`，它的包含块由祖先元素中最近一个`position`为`relative`,`absolute`或者`fixed`的元素产生，规则如下：
    - 如果祖先元素为行内元素，the containing block is the bounding box around the **padding boxes** of the first and the last inline boxes generated for that element.
    - 其他情况下包含块由祖先节点的**padding edge**组成

```
1. 如果找不到定位的祖先元素，包含块为**初始包含块**
```

stacking context,布局规则

z轴上的默认层叠顺序如下（从下到上）：

1. 根元素的边界和背景
2. 常规流中的元素按照html中顺序
3. 浮动块
4. positioned元素按照html中出现顺序

如何创建stacking context：

1. 根元素
2. z-index不为auto的定位元素
3. a flex item with a z-index value other than ‘auto’
4. opacity小于1的元素
5. 在移动端webkit和chrome22+，z-index为auto，position: fixed也将创建新的stacking context

如何水平居中一个元素

- 如果需要居中的元素为**常规流中inline元素**，为父元素设置`text-align: center;`即可实现
- 如果需要居中的元素为**常规流中block元素**，1）为元素设置宽度，2）设置左右margin为auto。3）IE6下需在父元素上设置`text-align: center;`,再给子元素恢复需要的值

```
<body>
    <div class="content">
    aaaaaa aaaaaa a a a a a a a a
    </div>
</body>

<style>
    body {
        background: #DDD;
        text-align: center; /* 3 */
    }
    .content {
        width: 500px;       /* 1 */
        text-align: left;   /* 3 */
        margin: 0 auto;     /* 2 */

        background: purple;
    }
</style>
```

- 如果需要居中的元素为**浮动元素**，1）为元素设置宽度，2）`position: relative;`，3）浮动方向偏移量（left或者right）设置为50%，4）浮动方向上的margin设置为元素宽度一半乘以-1

```
<body>
    <div class="content">
    aaaaaa aaaaaa a a a a a a a a
    </div>
</body>

<style>
    body {
        background: #DDD;
    }
    .content {
        width: 500px;          /* 1 */
        float: left;

        position: relative;    /* 2 */
        left: 50%;             /* 3 */
        margin-left: -250px;   /* 4 */

        background-color: purple;
    }
</style>
```

- 如果需要居中的元素为**绝对定位元素**，1）为元素设置宽度，2）偏移量设置为50%，3）偏移方向外边距设置为元素宽度一半乘以-1

```
<body>
    <div class="content">
    aaaaaa aaaaaa a a a a a a a a
    </div>
</body>

<style>
    body {
        background: #DDD;
        position: relative;
    }
    .content {
        width: 800px;

        position: absolute;
        left: 50%;
        margin-left: -400px;

        background-color: purple;
    }
</style>
```

- 如果需要居中的元素为**绝对定位元素**，1）为元素设置宽度，2）设置左右偏移量都为0,3）设置左右外边距都为auto

```
<body>
    <div class="content">
    aaaaaa aaaaaa a a a a a a a
    </div>
</body>

<style>
    body {
        background: #DDD;
        position: relative;
    }
    .content {
        width: 800px;

        position: absolute;
        margin: 0 auto;
        left: 0;
        right: 0;

        background-color: purple;
    }
</style>
```

如何竖直居中一个元素

参考资料：6 Methods For Vertical Centering With CSS。 盘点8种CSS实现垂直居中

- 需要居中元素为**单行文本**，为包含文本的元素设置大于`font-size`的`line-height`：

```
<p class="text">center text</p>

<style>
.text {
    line-height: 200px;
}
</style>
```