HTML && CSS

HTML5 新特性、语义化

1. **概念**:

   HTML5的语义化指的是合理正确的使用语义化的标签来创建页面结构。【正确的标签做正确的事】

2. **语义化标签**:

   header nav main article section aside footer

3. **语义化的优点**:

   - 在没CSS样式的情况下，页面整体也会呈现很好的结构效果
   - 代码结构清晰，易于阅读，
   - 利于开发和维护 方便其他设备解析（如屏幕阅读器）根据语义渲染网页。
   - 有利于搜索引擎优化（SEO），搜索引擎爬虫会根据不同的标签来赋予不同的权重

HTML5新特性有哪些

- 语义化标签
- 音视频处理API(audio,video)
- canvas / webGL
- 拖拽释放(Drag and drop) API
- history API
- requestAnimationFrame
- 地理位置(Geolocation)API
- webSocket
- web存储 localStorage、SessionStorage
- 表单控件，calendar、date、time、email、url、search

CSS 选择器及优先级

选择器

- id选择器(#myid)
- 类选择器(.myclass)
- 属性选择器(a[rel="external"])
- 伪类选择器(a:hover, li:nth-child)
- 标签选择器(div, h1,p)
- 相邻选择器（h1 + p）
- 子选择器(ul > li)
- 后代选择器(li a)
- 通配符选择器(*)

优先级:

- `!important`
- 内联样式（1000）

- ID选择器（0100）
- 类选择器/属性选择器/伪类选择器（0010）
- 元素选择器/伪元素选择器（0001）
- 关系选择器/通配符选择器（0000）

带!important 标记的样式属性优先级最高； 样式表的来源相同时： `!important` > 行内样式>ID选择器 > 类选择器 > 标签 > 通配符 > 继承 > 浏览器默认属性

渐进增强与优雅降级的理解及区别

渐进增强（Progressive Enhancement）： 一开始就针对低版本浏览器进行构建页面，完成基本的功能，然后再针对高级浏览器进行效果、交互、追加功能达到更好的体验。

优雅降级（Graceful Degradation）： 一开始就构建站点的完整功能，然后针对浏览器测试和修复。比如一开始使用 CSS3 的特性构建了一个应用，然后逐步针对各大浏览器进行 hack 使其可以在低版本浏览器上正常浏览。 **两者区别** 1、广义： 其实要定义一个基准线，在此之上的增强叫做渐进增强，在此之下的兼容叫优雅降级 2、狭义： 渐进增强一般说的是使用CSS3技术，在不影响老浏览器的正常显示与使用情形下来增强体验，而优雅降级则是体现html标签的语义，以便在js/css的加载失败/被禁用时，也不影响用户的相应功能。

```
/* 例子 */
.transition { /*渐进增强写法*/
  -webkit-transition: all .5s;
     -moz-transition: all .5s;
       -o-transition: all .5s;
          transition: all .5s;
}
.transition { /*优雅降级写法*/
          transition: all .5s;
       -o-transition: all .5s;
     -moz-transition: all .5s;
  -webkit-transition: all .5s;
}
```

常见的兼容性问题

1. 不同浏览器的标签默认的margin和padding不一样。*{margin:0;padding:0;}
2. IE6双边距bug：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。hack：display:inline;将其转化为行内属性。
3. 设置较小高度标签（一般小于10px），在IE6，IE7中高度超出自己设置高度。hack：给超出高度的标签设置overflow:hidden;或者设置行高line-height 小于你设置的高度。
4. Chrome 中文界面下默认会将小于 12px 的文本强制按照 12px 显示,可通过加入 CSS 属性 -webkit-text-size-adjust: none; 解决。
5. 超链接访问过后hover样式就不出现了，被点击访问过的超链接样式不再具有hover和active了。解决方法是改变CSS属性的排列顺序:L-V-H-A (love hate): a:link {} a:visited {} a:hover {} a:active {}

CSS3新特性

- 过渡

```
/*所有属性从原始值到制定值的一个过渡，运动曲线ease,运动时间0.5秒*/
transition: all,.5s
```

- 动画

```
//animation：动画名称，一个周期花费时间，运动曲线（默认ease），动画延迟（默认0），播放次数（默认1），是否反向播放动画（默认normal），是否暂停动画（默认running）
/*执行一次logo2-line动画，运动时间2秒，运动曲线为 linear*/
animation: logo2-line 2s linear;
```

- 形状转换

```
//transform:适用于2D或3D转换的元素
//transform-origin：转换元素的位置（围绕那个点进行转换）。默认(x,y,z)：(50%,50%,0)
transform:translate(30px,30px);
transform:rotate(30deg);
transform:scale(.8);
```

- 选择器:nth-of-type()
- 阴影 文字阴影: text-shadow: 2px 2px 2px #000;(水平阴影，垂直阴影，模糊距离，阴影颜色) 盒子阴影: box-shadow: 10px 10px 5px #999
- 边框 border-image: url(border.png);
- 背景
- 文字
- 渐变
- Filter（滤镜）
- 弹性布局、栅格布局、多列布局
- 媒体查询

position 属性的值有哪些及其区别

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。

box-sizing属性

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。【标准盒子模型】

border-box：为元素设定的宽度和高度决定了元素的边框盒。【IE 盒子模型】

inherit：继承父元素的 box-sizing 值。

CSS 盒子模型

CSS 盒模型本质上是一个盒子，它包括：边距，边框，填充和实际内容。CSS 中的盒子模型包括 IE 盒子模型和标准的 W3C 盒子模型。
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`。

故在计算盒子的宽度时存在差异：

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）

BFC（块级格式上下文）

BFC的概念

`BFC` 是 `Block Formatting Context` 的缩写，即块级格式化上下文。`BFC`是CSS布局的一个概念，是一个独立的渲染区域，规定了内部box如何布局， 并且这个区域的子元素不会影响到外面的元素，其中比较重要的布局规则有内部 box 垂直放置，计算 BFC 的高度的时候，浮动元素也参与计算。

BFC的原理布局规则

- 内部的Box会在垂直方向，一个接一个地放置
- Box垂直方向的距离由margin决定。属于同一个BFC的两个相邻Box的margin会发生重叠
- 每个元素的margin box的左边， 与包含块border box的左边相接触(对于从左往右的格式化，否则相反

- BFC的区域不会与float box重叠
- BFC是一个独立容器，容器里面的子元素不会影响到外面的元素
- 计算BFC的高度时，浮动元素也参与计算高度
- 元素的类型和display属性，决定了这个Box的类型。不同类型的Box会参与不同的Formatting Context。

如何创建BFC?

- 根元素，即HTML元素
- float的值不为none
- position为absolute或fixed
- display的值为inline-block、table-cell、table-caption
- overflow的值不为visible

BFC的使用场景

- 去除边距重叠现象
- 清除浮动（让父元素的高度包含子浮动元素）
- 避免某元素被浮动元素覆盖
- 避免多列布局由于宽度计算四舍五入而自动换行

让一个元素水平垂直居中

- **水平居中**
    - 对于 行内元素 : text-align: center;
    - 对于确定宽度的块级元素：

      （1）width和margin实现。margin: 0 auto;

      （2）绝对定位和margin-left: margin-left: (父width - 子width）/2, 前提是父元素position: relative
    - 对于宽度未知的块级元素

      （1）table标签配合margin左右auto实现水平居中。使用table标签（或直接将块级元素设值为display:table），再通过给该标签添加左右margin为auto。

      （2）inline-block实现水平居中方法。display：inline-block和text-align:center实现水平居中。

      （3）绝对定位+transform，translateX可以移动本身元素的50%。

      （4）flex布局使用justify-content:center
- **垂直居中**
    1. 利用 line-height 实现居中，这种方法适合纯文字类
    2. 通过设置父容器 相对定位 ，子级设置 绝对定位，标签通过margin实现自适应居中
    3. 弹性布局 flex :父级设置display: flex; 子级设置margin为auto实现自适应居中
    4. 父级设置相对定位，子级设置绝对定位，并且通过位移 transform 实现
    5. table 布局，父级通过转换成表格形式，然后子级设置 vertical-align 实现。（需要注意的是：vertical-align: middle使用的前提条件是内联元素以及display值为table-cell的元素）。

传送门 ☞ # 图解CSS水平垂直居中常见面试方法

隐藏页面中某个元素的方法

1.`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

2.`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，隐藏对应元素，在文档布局中仍保留原来的空间（重绘）

3.`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。 不显示对应的元素，在文档布局中不再分配空间（回流+重绘）

> 该问题会引出 回流和重绘

用CSS实现三角符号

```
/*记忆口诀：盒子宽高均为零，三面边框皆透明。 */
div:after{
    position: absolute;
    width: 0px;
    height: 0px;
    content: " ";
    border-right: 100px solid transparent;
    border-top: 100px solid #ff0;
    border-left: 100px solid transparent;
    border-bottom: 100px solid transparent;
}
```

页面布局

1.Flex 布局

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。

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。指定容器 display: flex 即可。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）flex-direction: row | row-reverse | column | column-reverse;
- flex-wrap：决定换行规则 flex-wrap: nowrap | wrap | wrap-reverse;
- flex-flow： .box { flex-flow: || ; }
- 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
- align-content

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

2.Rem 布局

首先 Rem 相对于根(html)的 font-size 大小来计算。简单的说它就是一个相对单例 如:font-size:10px;,那么（1rem = 10px）了解计算原理后首先解决怎么在不同设备上设置 html 的 font-size 大小。其实 rem 布局的本质是等比缩放，一般是基于宽度。

优点：可以快速适用移动端布局，字体，图片高度

缺点：

①目前 ie 不支持，对 pc 页面来讲使用次数不多；
②数据量大：所有的图片，盒子都需要我们去给一个准确的值；才能保证不同机型的适配；
③在响应式布局中，必须通过 js 来动态控制根元素 font-size 的大小。也就是说 css 样式和 js 代码有一定的耦合性。且必须将改变 font-size 的代码放在 css 样式之前。

3.百分比布局

通过百分比单位 " % " 来实现响应式的效果。通过百分比单位可以使得浏览器中的组件的宽和高随着浏览器的变化而变化，从而实现响应式的效果。 直观的理解，我们可能会认为子元素的百分比完全相对于直接父元素，height 百分比相 对于 height，width 百分比相对于 width。 padding、border、margin 等等不论是垂直方向还是水平方向，都相对于直接父元素的 width。 除了 border-radius 外，还有比如 translate、background-size 等都是相对于自身的。

缺点：

（1）计算困难
（2）各个属性中如果使用百分比，相对父元素的属性并不是唯一的。造成我们使用百分比单位容易使布局问题变得复杂。

4.浮动布局

浮动布局:当元素浮动以后可以向左或向右移动，直到它的外边缘碰到包含它的框或者另外一个浮动元素的边框为止。元素浮动以后会脱离正常的文档流，所以文档的普通流中的框就变的好像浮动元素不存在一样。

优点

这样做的优点就是在图文混排的时候可以很好的使文字环绕在图片周围。另外当元素浮动了起来之后，它有着块级元素的一些性质例如可以设置宽高等，但它与inline-block还是有一些区别的，第一个就是关于横向排序的时候，float可以设置方向而inline-block方向是固定的；还有一个就是inline-block在使用时有时会有空白间隙的问题

缺点

最明显的缺点就是浮动元素一旦脱离了文档流，就无法撑起父元素，会造成父级元素高度塌陷。

如何使用rem或viewport进行移动端适配

rem适配原理：

改变了一个元素在不同设备上占据的css像素的个数

rem适配的优缺点

- 优点：没有破坏完美视口
- 缺点：px值转换rem太过于复杂(下面我们使用less来解决这个问题)

viewport适配的原理

viewport适配方案中，每一个元素在不同设备上占据的css像素的个数是一样的。但是css像素和物理像素的比例是不一样的，等比的

viewport适配的优缺点

- 在我们设计图上所量取的大小即为我们可以设置的像素大小，即所量即所设
- 缺点破坏完美视口

清除浮动的方式

- 添加额外标签

```
<div class="parent">
    //添加额外标签并且添加clear属性
    <div style="clear:both"></div>
    //也可以加一个br标签
</div>
```

- 父级添加overflow属性，或者设置高度
- 建立伪类选择器清除浮动

```
//在css中添加:after伪元素
.parent:after{
    /* 设置添加子元素的内容是空 */
    content: '';
    /* 设置添加子元素为块级元素 */
    display: block;
    /* 设置添加的子元素的高度0 */
    height: 0;
    /* 设置添加子元素看不见 */
    visibility: hidden;
    /* 设置clear: both */
```

```
    clear: both;
}
```

CSS预处理器Sass、Less、Stylus的区别

什么事CSS预处理器?

CSS预处理器是一种语言用来为CSS增加一些变成的特性，无需考虑浏览器兼容问题，例如你可以在CSS中使用变量，简单的程序逻辑、函数等在编程语言中的一些基本技巧，可以让CSS更加简洁，适应性更强，代码更直观等诸多好处 基本语法区别

Sass是以.sass为扩展名，Less是以.less为扩展名，Stylus是以.styl为扩展名 变量的区别

Sass 变量必须是以$开头的，然后变量和值之间使用冒号（：）隔开，和css属性是一样的。 Less 变量是以@开头的，其余sass都是一样的。 Stylus 对变量是没有任何设定的，可以是以$开头或者任意字符，而且变量之间可以冒号，空格隔开，但是在stylus中不能用@开头 三种预处理器都有：嵌套、运算符、颜色函数、导入、继承、混入。Stylus还有一些高级特性。例如循环、判断等