伪类 && 伪元素

伪类：伪类是**根据元素的状态特征**在一些选择器上来对元素添加特殊效果，以 **一个冒号** : 为前缀

- **常见：**a链接中的**link、visited**状态；元素的**hover、active**状态；列表中的**偶数项/奇数项** nth-child(2n)
- **注意：**a链接中遵循**LVHA顺序**：**:link -> :visited -> :hover -> :active**
- **用途**：添加简单的动画、交互样式，例如滑入滑出等。

伪元素：伪元素是**创造DOM之外的对象**，并且可以为这些不存在的内容分配样式

- （就是说如果不通过伪元素，它可能不存在，或者是你没有办法去选中它为它设置特定的样式~）
- **CSS3中 :: CSS2中 :**
- **常见：**
  - **::after ::before** 在对应元素下创建第一/最后一个子元素，默认**行内标签**，需要为其指定**content**属性（否则无效）
  - ::first-letter 选中**某块级元素第一行的第一个字母**
  - :first-line 选中**某块级元素的第一行**应用样式
- **用途**：清除浮动、写一些图标：三角形、箭头、叉；写一个tip框

伪类、伪元素的区别

- 伪类的受体是**文档树中已有的元素**，而伪元素则**创立了一个DOM外的元素**
- 伪类用于添加**元素特定状态下的特殊效果（交互）**，而伪元素则是**添加元素的内容/选中一类原本不能选中**的
- 伪类使用**一个冒号**，css3标准伪元素使用**两个冒号**，**css2中伪元素也可以使用一个冒号。**
- **伪类更常用于一些简单的动画或交互的样式，例如滑入滑出等，而伪元素更常用于字体图标、清除浮动等**

position

- **静态定位 static** ：默认值，即**没有定位**，遵循正常的文档流对象
- **相对定位 relative**：**相对其正常文本流中的位置**进行定位，不脱离文档流**（仍然占据原本的空间）**
- **绝对定位 absolute**：相对于**最近的已定位父元素**（除了static之外）来对元素进行定位，**如果没有则相对于**
- **固定定位 fixed**：相对于**浏览器窗口**来固定位置，即使窗口滚动也不会移动
- **粘性定位 sticky**
  - 元素根据正常文档流进行定位，相对它的**最近可滚动块级祖先**，包括table-related元素，基于`top`，`right`，`bottom`，和 `left`的值进行偏移。
  - **依赖于用户的滚动**，在 **position:relative** 与 **position:fixed** 定位之间切换。

fixed和absolute定位区别

- **相对于谁定位：**
  - absolute：相对于**最近的已定位父元素**（除了static之外），如果没有已定位的父元素，则相对于`<html>`
  - fixed：相对于**浏览器窗口**来固定位置
- **滚动条：**
  - 如果没有滚动条时，都相对于各自的定位元素定位
  - 如果有滚动条时，absolute元素会随着滚动而滚动，fixed元素则固定在屏幕的某一处

弹性布局 flex

采用 Flex 布局的元素，称为**容器**，它的**所有子元素自动成为容器成员**，称为**项目（flex item）**。

flex可以用来解决一些比较复杂的布局（以前用float、position）

容器属性：

- flex-direction：决定主轴的方向
- flex-wrap：如果主轴排不下，如何换行。【默认不换行，发生挤压】
- flex-flow：简写属性 < flex-direction> || < flex-wrap>
- justify-content：flex-start | flex-end | center | space-between | space-around | initial | inherit
- align-items：定义项目在交叉轴上如何对齐
- align-content：定义了多根轴线的对齐方式

项目属性：

- order：小的排在前
- flex-grow：扩大比例，默认为0
- flex-shrink：缩小比例，默认为1
- flex-basis：定义了在**分配多余空间**之前，项目占据的主轴空间（main size）
- flex：flex-grow flex-shrink flex-basis | auto | initial | inherit;
- align-self：允许单个项目有与其他项目不一样的交叉轴对齐方式，可覆盖`align-items`属性。

子元素的margin百分比相对于谁

不论是margin-top/right/bottom/left 一律相对于**父元素的宽度**

- 如果父元素中没有内容，则会一直向上追溯，最终以浏览器视口为参考点

绝对定位的基准点

- `position: absolute;` 相对于relative容器的content
- `position: absolute; top: 0; left: 0;` 相对于border以内，padding的外侧

css3中可直接影响JS事件的属性

- **pointer-events: none;** 表示该元素永远**不会成为鼠标事件的target**【其它事件仍然可以触发】。

    - 同时鼠标事件会**“穿透”该元素**并且指定该元素下pointer-events不为none的元素。
    - **应用**：**防止多次点击、重复提交。**点击“发送验证码”按钮后，为按钮添加 pointer-events: none;，
- **touch-action: none 禁用**元素上的所有**手势**，**不响应用户操作**（比如浏览器自带的划动、缩放等），以使用自己提供的拖放和缩放行为。

CSS选择器种类

- id选择器(#myid)
- 类选择器(.myclassname)
- 元素选择器(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（*）
- 属性选择器（a[rel=”external”]）
- 伪类选择器（a:hover, li:nth-child）

CSS样式优先级算法

- 在同一组属性设置中标有“**!important**”规则的优先级最大
- **计算选择器权重**，权重大的优先：**元素选择器 1；类选择器 10；id选择器100； 内联样式 1000**
- 权重相等则看样式位置：**内联样式优先级最大；内部style样式跟外部引用样式，后出现的覆盖前面的**
- 指定的CSS 样式 > 继承的CSS 样式
- 开发者的CSS样式 > 浏览器的CSS样式

Vue组件CSS样式中的scoped

scoped 属性的效果：编译打包后，在当前**组件的标签**中统一添加一个随机的属性（如data-v-97a9747e)

- 编译的css也会对于加上那个随机属性
- **样式穿透、深度选择器**：在需要穿透的选择器前边添加 >>> 或者 /deep/ 或者 ::v-deep

```
.el-form-item >>>.el-form-item__error{
    margin-left: 120px;
}
```

@import & link 加载CSS

区别：

- **类型**：link属于html标签，而@import是css提供的
- **加载时间**：页面被加载时，遇见link就会去加载CSS，而@import引用的css会等到页面加载结束后加载。

- **兼容性**：link是html标签，因此没有兼容性，而@import只有IE5以上才能识别。
- **权重**：link方式样式的权重高于@import的。

盒模型

盒模型是什么?

将**所有元素**表示为一个个矩形的盒子

盒模型是由： **内容(content)、内边距(padding)、边框(border)、外边距(margin)** 组成的。

标准盒模型 & 怪异盒模型

- 标准模型的宽高是指的content区域的宽高；**增加内外边距、边框时会将元素框撑大**
  - 元素框的宽=width+padding+border+margin
- 怪异盒模型的宽高是指的content+padding+border的宽高**（更容易地设定元素宽高）**增加内外边距、边框时会挤压内容区域，元素框大小不变
  - 元素框的宽=width+margin

> 在MDN中的解释：
>
> - 元素框由 **内容(content)、内边距(padding)、边框(border)、外边距(margin)**这四部分组成

JS如何设置获取盒模型宽高

- **dom.style.width/height**：**只能取出内联样式的宽度和高度**
- **dom.currentStyle.width/height：** 获取即时计算的样式，**只有IE支持**
- **window.getComputedStyle(dom).width/height;**：获取即时计算的样式，**支持其他浏览器**，兼容性好
- **dom.getBoundingClientRect().width/height：** 计算盒模型在页面中的绝对位置，比较少用
  - 返回一个矩形对象，包含属性：left、top、right和bottom（分别表示元素各边与页面上边和左边的距离
- **dom.offsetWidth/offectHeight**：返回元素实际大小，包含边框，内边距和滚动条
  - 一般是块级(block)元素并且以设置了CSS大小的元素较为方便。如果是内联元素(inline)或者没有设置大小的元素就尤为麻烦，所以，建议使用的时候注意。

BFC

BFC是什么

BFC是页面中的一块**渲染区域**，可以理解为是**决定如何渲染元素的容器**，并且是一个**隔离的**独立容器，容器里面的子元素不会影响到外面的元素。

> 块格式化上下文对**浮动定位与清除浮动**、**外边距折叠**都很重要。

> - **浮动定位和清除浮动**时只会**应用于同一个BFC内的元素。**浮动不会影响其它BFC中元素的布局，而清除浮动只能清除同一BFC中在它前面的元素的浮动。
> - **外边距折叠也只会发生在属于同一BFC的块级元素之间。**

如何触发BFC

- **浮动元素、绝对定位元素**，
- 'display' 特性为 **"inline-block"**，"table-cell"， "table-caption" 的元素
- **'overflow' 不是 "visible"** 的元素【常用】

BFC布局规则

- 内部的Box会在垂直方向，一个接一个地放置。
- **垂直方向上**属于**同一个BFC**的两个相邻Box的**margin会发生重叠** 【应用：防止垂直margin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)。**即使存在浮动也是如此。
- **BFC的区域不会与float box重叠。** 【应用：清除浮动】
- BFC就是页面上的一个隔离的独立容器，容器里面的子元素不会影响到外面的元素。反之也如此。
- 计算BFC的高度时，**浮动元素**也参与计算 【应用：解决高度塌陷】

BFC应用

- **防止垂直margin重叠**：触发其一生成BFC
  - 原理：外边距折叠也只会发生在属于同一BFC的块级元素之间
- **清除浮动**，可以做**分栏布局**：将会被浮动影响的那部分触发生成BFC
  - 原理：BFC的区域不会与float box（也是一个BFC）重叠
- **解决高度塌陷**：在浮动元素的父元素包裹一个BFC，则计算高度时会计算浮动元素高度
  - 原理：计算BFC的高度时，**浮动元素**也参与计算

圣杯布局 & 双飞翼布局

相同点：

- 都是为了实现**右侧布局**： 
- 在**HTML标签中的顺序**是这样的：main、left、right，目的是为了先加载主要的部分**main**
- 利用了神奇的**负外边距**来解决三栏问题

不同点：在考虑main部分内容不被遮挡上思想不同

- **圣杯布局：**给包裹main、left、right这三个元素的content添加**左右padding**，来同时挤压三个元素，然后给left、right进行**相对定位**，把它们挪到正确的位置上，这样子看上去就只挤压了main
- **双飞翼布局**：在main里面**再添加一个div**，并给他**设置左右margin**

圣杯布局

顺序中左右、全float、左右调整负外边距、中间padding、左右相对布局（也可以用transform百分比）

双飞翼布局

顺序中左右、全float、左右调整负外边距、中间包裹元素设margin（包裹元素width不必100%）

元素居中系列

垂直水平居中

- **绝对定位 + margin自适应**：position:absolute; **+ top: 0；bottom: 0；left:0；right: 0**；**+** margin:auto
  - 父元素高度 = 元素高度 + top + bottom + margin-top + margin-bottom
- **绝对定位 + translate（百分比）**：transform: translate(-50%,-50%);
- **flex + margin自适应**
- **flex**：justify-content: center; + align-items: **center**;

垂直居中

单行文本垂直居中

- **设置行高** line-height=height；

多行文本垂直居中（也可单行）

- **表格布局** 父元素 display: table，子元素 display: table-cell；vertical-align: middle;

块级元素（高度不确定）

- **绝对定位 + margin自适应**
- **绝对定位 + translate（百分比）**
- **flex + margin自适应**
- **flex + align-items**

水平居中

行内元素

- **text-align:center**

确定宽度的块级元素

- **width+margin**：margin: 0 auto;

宽度未知的块级元素

- **inline-block**：父元素设置text-align
- **外边距 + translate（百分比）** ---- 因为外边距是相对于父元素的
- **绝对定位 + margin自适应**
- **绝对定位 + translate（百分比）**
- **flex + margin自适应**
- **flex + justify-content**f
- **table+margin（不好用）**：**给要居中显示的元素**，设置display:table，margin:0 auto

分栏系列

三栏布局

> **要求：左右定宽，中间自适应**

- **绝对定位**：顺序中左右（可灵活）；左右绝对定位；中间宽度100%，设两边margin（或用BFC）
- **浮动布局+BFC：**顺序左右中；左边左浮动，右边有浮动；中间宽度100%，设两边margin（或用BFC）
- **双飞翼布局**：顺序中左右、全float、左右负外边距、中间包裹元素设margin（包裹元素width不必100%）
- **圣杯布局**：顺序中左右、全float、左右负外边距、中间padding、左右相对布局（也可用transform%）
- **flex布局**：左右定宽，中间flex:1

两栏布局

> **要求：侧边栏定宽，主栏自适应**

- **绝对定位**
- **浮动布局+BFC**
- **双飞翼布局**
- **圣杯布局**
- **flex布局**

清除浮动系列

- 伪元素+clear:both
- BFC

用CSS画的小东西

实心三角形

```css
.icon-triangle{
  display: inline-block;
  width: 0;
  height: 0;
  border: 100px solid transparent;
  border-top: 100px solid #f00;
  /* transform: rotate(90deg); */
}
```