1. CSS背景样式

- `background-color` ： 背景颜色
- `background-image` ： 背景图片
  - `background-image:url(背景地址)` ，默认：会水平垂直都铺满
- `background-repeat` ： 背景图片的平铺方式
  - `repeat-x` ： `x` 轴平铺
  - `repeat-y` ： `y` 轴平铺
  - `repeat` ： `x` ，`y` 都进行平铺，默认值
  - `no-repeat` ： 都不平铺
- `background-position` ： 背景图片的位置
  - `background-position: x,y`
    - `number` (`px` 、 `%`)
    - 单词， `x` ： `left center right` ， `y` ： `top center bottom`
- `background-attachment` ： 背景图随滚动条的移动方式
  - `scroll` ： 默认值 ， 背景位置是按照当前元素进行偏移
  - `fixed` ： 背景位置是按照浏览器进行偏移

2. CSS边框样式

- `border-style` : 边框样式
  - `solid` ： 实线
  - `dashed` ： 虚线
  - `dotted` ： 点线
- `border-width` : 边框大小(边框宽度)
- `border-color` : 边框颜色
  - 格式： `red` `#f0000` `rgb(255,255,255)`
    - 透明颜色： `transparent`

- 边框扩展:
  - `border: 1px solid black;`
  - `border-width: 1px 0;` 这样也可以，上下有边框，左右没有。
  - `border-width: 1px(上) 0px(左/右) 1px(下);`
- 边框也可以针对某一条边进行单独设置： `border-left-color` 、 `border-top-color` ...
- 透明颜色： `transparent`
- 扩展： `outline` （轮廓）是绘制于元素周围的一条线，位于边框边缘的外围，可起到突出元素的作用。
  - 用法和 `border` 相同，但不可以独立设置一条边。

3. CSS文字样式

- `font-family`：字体类型
  - 英文、中文，每个中文字体都有英文昵称，宋体：simsun
  - 字体分为衬线体和非衬线体
    - 衬线体棱角多，如宋体； 非衬线体比较圆润，如微软雅黑。
  - 添加多字体：

    ```
    font-family:"Source Sans Pro", Helvetica, Arial, sans-serif;
    ```

    - 字体有空格使用`""`包裹，多字体使用`,`隔开
- `font-size`：字体大小
  - 默认大小： `16px`
    - 注：字体大小一般为偶数
- `font-weight`：字体粗细
  - 单词： `normal`、 `bold`(加粗)
  - 数值：100、200...900，100到500都是正常的，600-900都是加粗
- `font-style`：字体样式
  - 单词： `normal`、 `italic`(斜体)
  - `oblique`也是表示斜体，用的比较少，一般了解即可。
    - `italic` 带有倾斜属性的字体才可以设置倾斜操作。
    - `oblique` 没有倾斜属性的字体也可以设置倾斜操作。
- `color`：字体颜色

4. CSS段落样式

- `text-decoration`：文本修饰
  - `text-decoration: underline dashed #cccccc;`
    - 下划线： `underline`，删除线： `line-through`，上划线： `overline`，不添加任何装饰：none
    - 添加多个文本修饰用空格隔开：`text-decoration:line-through overline underline`

- `text-transform`：文本大小写 （针对英文段落）
  - 小写： `lowercase`，大写： `uppercase`，只针对首字母： `capitalize`

- `text-indent`：文本缩进(首行缩进)
  - `em`单位：相对单位， `1em`永远都是跟字体大小相同

- `text-align`：文本对齐方式
  - 左对齐： `left` ，居中`center` ，右对齐`right` ，两端点对齐`justify` （两端对齐，中间间距自动调整）

- `line-height`：定义行高
  - 什么是行高，（文字上边距、文字下边距、文字大小是行高的组成部分）一行文字的高度，上边距和下边距的等价关系。
  - 默认行高不是固定值，而是根据当前字体大小进行变化（可以理解为：默认文字上下边距不变）
  - 取值：`number`（`px`）| `scale`（比例值，跟文字大小成比例）

- `letter-spacing`：字之间的间距
  - `spacing`是外部距离的意思，`letter`是文字的意思，同样还有单元格外边距`cellspacing`
    `cell`是单人牢房的意思
- `word-spacing`：单词之间的间距 （针对英文段落）

- 英文和数字不自动折行的问题（连贯性的英文和数字在行尾是不会自动折行的）
  1. `word-break : break-all;`： 非常强烈的折行)
  2. `word-wrap : break-word;`： 不是那么强烈的折行,会产生一些空白区域
  - `white-space : nowrap;`：不让内容折行

5. CSS复合样式

- 复合样式的写法是通过空格的方式实现的。复合写法有的是不需要关心顺序的，例如`background`、 `border`，有的是需要关心顺序的，例如`font`

1. `background : red url() repeat 0 0;`
2. `border : 1px dotted #颜色`
3. `font` ：
   - 注：`font`最少要有两个值 （先写）`size` （后写）`family`
     - `font: weight style size family` √
     - `font: style weight size family` √
     - `weight style size/line-height family` √
       - `font:italic bold 30px/40px 宋体;`
       - `size`和`family`要写在最后，而且`size`要在前

- 尽量不要混写，如果非要混写，那么一定要先写复合样式，再写单一样式，这样样式才不会被覆盖掉。

6. CSS标签选择器

1. `ID`选择器

```
#id{}
```

   1. `ID`是唯一值，在一个页面中只能出现一次，出现多次是不符合规范的。

2. 命名的规范，由字母、下划线、中划线、数字（并且第一个不能是数字）
3. 驼峰写法： `searchButton`(小驼峰) `Searchbutton`(大驼峰)
    - 短线写法： `search-small-button`
    - 下划线写法： `search_small_button`

2. `class`选择器

```
.class{}
```

1. `class`选择器可以复用。
2. （标签）可以添加多个`class`样式。
3. 多个样式的时候，样式的优先级根据`CSS`决定，而不是`class`属性中的顺序。(具有覆盖性)
4. 标签+类的写法。
    - 只套用`div`标签中`class="box"` `div.box{}`

```
<div class="box"> <p class="box">
```

3. 标签选择器(`TAG`选择器)

```
div{}
```

- 使用的场景：
    1. 去掉某些标签的默认样式
    2. 复杂的选择器当中，如层次选择器

4. 群组选择器(分组选择器)
    - 可以通过逗号的方式，给多个不同的选择器添加统一的CSS样式，来达到代码的复用。

```
p,#text,.title{background: red;}
```

5. 通配选择器

```
*{ } -> div,ul,li,p,h1,h2......{}
```

- 使用的场景：去掉所有标签的默认样式时

6. 层次选择器

名称	代码	描述
后代	`M N { }`	`ul li{ }` 、 `#list ul li{ }` 在`ul`里面去寻找`li`
父子	`M > N { }`	`M`的孩子N会继承`M`的样式，但`N`的孩子不会继承样式
兄弟	`M ~ N { }`	当前`M` 下方 的所有`N`标签 `div~h2{ }` 会找到`div`后面的所有`h2`标签添加样式)
相邻	`M + N { }`	当前`M`下面相邻的`N`标签 `div+p{ }` 如果`P`标签在`div`正下方，则会被选中，否则不会被选中

7. 属性选择器

写法	代码	描述
`M[属性]{ }`	`div[class]{ }`	所有带有`class`属性的`div`标签
`M[属性][属性2]{ }`	`div[class][id]{ }`	同时带有`class`和`id`属性的`div`标签
`M[属性=值]{ }`	`div[class=search]{ }`	`=` ： 完全匹配，名称完全相同
`M[属性*=值]{ }`	`div[class*=search]{ }`	`*=` ： 部分匹配，部分名称相同
`M[属性^=值]{ }`	`div[class^=search]{ }`	`^=` : 起始匹配，起始名称相同
`M[属性$=值]{ }`	`div[class$=search]{ }`	`$=` ： 结束匹配，结束名称相同

8. 伪类选择器
   - 常用伪类

伪类	描述
`:link`	访问前的样式，只能添加给`a`链接
`:visited`	访问后的样式，只能添加给`a`链接
`:hover`	鼠标移入时的样式，可以添加给所有标签
`:active`	鼠标按下时的样式，可以添加给所有标签

     - 一般网站都只设置 `a{}` (`link` `visited` `active`) 和 `a:hover`
   - 其他伪类
     - `:after` 、 `:before` ： 通过伪类的方式给元素添加一段文本内容，使用`content` `CSS`属性。
       - `:after`和`:before`属于伪元素，在`h5`中要加两个`:`，为了兼容低版本浏览器可以加一个冒号。
   - `::selection` ： 选中文本的样式 伪元素
   - `:checked` 、 `:disabled` 、 `:focus` 都是针对表单元素的。
     - `:checked` 是（单选、多选都可以）选中后的样式
     - `:disabled` 是不可选中表单样式，要给表单预设不可选中状态
     - `:focus` 是获取光标后样式，文本框常见

   - 结构性伪类选择器
     - `:nth-of-type(1)` `:nth-child(n)`
       - **角标是从`1`开始的**， `n`值 表示从`1`到无穷大
     - `:nth-of-type()` ： 根据括号里面的值选

格式	说明
`:nth-of-type(2)`	选取第`2`个标签
`:nth-of-type(+n+2)`	选取大于等于`2`的标签
`:nth-of-type(-n+2)`	选取小于等于`2`的标签
`:nth-of-type(2n)`	选取偶数标签，`2n`也可以是`even`
`:nth-of-type(2n-1)`	选取奇数标签，`2n-1`可以是`odd`
`:nth-of-type(3n+1)`	自定义选取标签，`3n+1`表示“隔二取一”

- `:nth-child()`和`:nth-of-type()`差不多，有一点区别
  - 区别：`type`只识别要套用的标签，`child`不识别
    - `type`：类型，`child`：孩子
- `:first-of-type`：第一个套用
- `:last-of-type`：最后一个套用
- `:nth-last-of-type(2)`：倒数第二个开始套用
- `:only-of-type`：唯一一个套用（其父容器里面只有唯一的标签值）

7. CSS样式继承

- **文字相关的样式可以被继承**(颜色、字体大小、行高、缩进等)
- **布局相关的样式不能被继承**（默认是不能继承的，但是可以设置继承属性`inherit`值）

8. CSS样式优先级

1. 相同样式优先级
   - 当设置相同样式时，**后面的优先级较高**，但不建议出现重复设置样式的情况。
2. 内部样式与外部样式
   - **内部样式与外部样式优先级相同**，如果设置了相同样式，那么后写的引入方式优先级高。
3. 单一样式优先级

   `style行间 > id > class > tag(标签) > * > 继承`
4. `!important`
   - 提升样式优先级，非规范方式，不建议使用。(不能针对继承的属性进行优先级提升)
5. 标签+类与单类
   - 标签+类（`div.box`） > 单类(`.box`)
6. 群组优先级
   - **群组选择器与单一选择器的优先级相同**，靠后写的优先级越高。
7. 层次优先级
   1. 权重比较

      `ul li .box p input{}` = `1 + 1 + 10 + 1 + 1`

`.hello span #elem` = `10 + 1 + 100`

2. 约分比较

   `ul li .box p input{}` = `li p input{}`

   `.hello span #elem` = `#elem`

9. CSS盒子模型

- 组成：`content` -> `padding` -> `border` -> `margin`
  - `content`：内容区域，由`width`和`height`组成
  - `padding`：内边距（内填充）
    - 只写一个值: `30px`（上下左右）
    - 只写两个值: `30px 40px`（上下、左右）
    - 写四个值: `30px 40px 50px 60px`（上、右、下、左）
      - `margin`同上
  - `border`：外部边框
  - `margin`：外边距（外填充）
  - `padding`、`border`、`margin`都可以单独设置
    - `margin-left`、`margin-right`、`margin-top`、`margin-bottom`

  - **背景颜色会填充到`border`以内的区域。**（`content`、`padding`）
  - **文字会在`content`区域。**
  - **`padding`不能出现负值，`margin`是可以出现负值的。**

  - `box-sizing`： 盒尺寸，可以改变盒子模型的展示形态
    - content-box：`content`，宽、高只分给`content`，默认值
    - border-box：`content` `padding` `border`（宽、高分给三个）
    - 使用的场景：
      1. 不用再去计算一些值
      2. 解决一些百分比的问题
- 盒子模型的一些问题
  1. `margin`**叠加问题**，出现在两个上下容器`margin`同时存在的时候。会取上下中值较大的作为叠加的值。
     - 解决方案：1. `BFC`规范；2. 想办法只给一个元素添加间距

  2. `margin`**传递问题**，出现在**嵌套结构中**，只针对`margin-top`
     - 解决方案: 1. `BFC`规范 2. 给父容器加边框 3. 把`margin`换成`padding`加给父容器
       - 浮动的元素不存在`margin`传递问题(`BFC`)
  - 扩展：`margin`左右自适应是可以的，但是上下自适应是不行的。
  - 当给**元素脱离文档流后**(`float`,`absolute`,`fixed`)，`margin: 0 auto;` **margin居中就不起作用了**

10. 标签分类

- 按类型
  - `block` ： 块 `div` 、 `p` 、 `ul` 、 `li` 、 `h1...h6` ...
    1. 独占一行。
    2. 支持所有样式。
    3. 不写宽的时候，跟父元素的宽相同。
    4. 所占区域是一个矩形

  - `inline` ： 内联， `span` 、 `a` 、 `em` 、 `strong` 、 `img` ...
    1. 挨在一起的
    2. 有些样式不支持，例如： `width` 、 `height`
       - `margin` **左右支持**，相邻的内联元素会隔开
       - `padding` **上下左右都支持**，只是视觉上撑开了元素，和其他元素排列时，会和其他元素区域重叠
    3. 宽由内容决定。
    4. 所占的区域不一定是矩形。
    5. 内联标签之间会有空隙，原因：代码中换行产生的
       - 两个 `span` 写在一行就不会有空隙、或者给父容器设置 `font-size : 0;` 后，给子容器单独设置字体大小
  - `inline-block` ：内联块 `input` 、 `select` ...
    1. 挨在一起，但是支持宽高。
  - **布局一般用块标签，修饰文本一般用内联标签**

- 按内容划分标签

名称	**描述**
flow	流内容
Metadata	元数据
Sectioning	分区
Heading	标题
Phrasing	措辞
Embedded	嵌入式
Interactive	互动的

  - 具体访问http://www.w3.org/tr/html5/dom.html 在50%位置查看
- 按显示划分
  - 替换元素： 浏览器根据元素的标签和属性，来决定元素的具体显示内容。

        - `img`、`input` ...
    - 非替换元素： 将内容直接告诉浏览器，将其显示出来
        - `div`、`h1`、`p` ...

11. 显示框类型

- `display`:可以改变标签的类型
    - `display:block`、`disdisplay:inline`、`display:inline-block`、`display:none`
    - `display:none` ： 不占空间的隐藏
    - `visibility:hidden` ： 占空间的隐藏
- 扩展:
    - 内联元素居中：给父容器添加`text-align: center;`
    - 块元素居中：`margin: 0 auto;`
    - 内联块居中：给父容器添加`text-align: center;`

12. 标签嵌套规范

- 块标签可以嵌套内联标签

```
<div>
  <span></span>
  <a href="#"></a>
</div>
```

- 块嵌套块

```
<div>
  <div></div>
</div>
```

    - 特殊:

```
<!-- 错误的写法 -->
<p>
  <div></div>
</p>
```

- 内联不能嵌套块

```
<!-- 错误的写法 -->
<span>
  <div></div>
</span>
```

    - 特殊:

```
<!-- 正确的写法: -->
<a href="#">
    <div></div>
</a>
```

13. 溢出隐藏

- `overflow`
  - `visible` ： 默认
  - `hidden` ： 边框边界直接截断
  - `scroll` ： 边框右侧和下侧会出现滚动条，内容少的时候也会显示滚动条。
  - `auto` ： 溢出时右侧会出现滚动条，内容少的时候不显示滚动条。
- 针对两个轴分别设置： `overflow-x` 、 `overflow-y`

14. 透明度与手势

- `opacity` ： `0` (透明,占空间)~ `1` (不透明)
  - IE浏览器: `filter:alpha(opacity=50);`
  - 注：占空间、所有的内容也会透明
- `rgba()` ： `0 ~ 1`
  - `background:rgba(255,0,0,0.5)`
    - 可以让指定的样式透明，而不影响其他样式
- `cursor` : 手势
  - `default` ： 箭头
  - `pointer` : 手型
    - 要实现自定义手势:
      1. 准备图片: `.cur` 、 `.ico`
      2. `cursor : url(./img/cursor.ico),auto;`

15. 最大、最小宽高

- `min-width` 、 `max-width`
- `min-height` 、 `max-height`
  - `min-height:200px;` ： 最小为 `200px` ，如果内容溢出，则会根据内容变大
  - `max-height:200px;` ： 最大为 `200px` ，如果内容较少，则会根据内容变小
- `%` 单位：换算， **以所在文档流父容器的大小进行换算的**
- 一个容器怎么适应屏幕的高 ： 容器加 `height:100%;` `body:100%;` `html:100%;`

```
html,body{height:100%}
.Container{ height:100%}
```

16. CSS默认样式

- 没有默认样式的: `div` 、 `span`

- 有默认样式的:
  - `body` ： `margin:8px;`
  - `h1` ： `margin : 21.440px 0px;` `font-weight : bold;`
  - `p` ： `margin : 16px 0;`
  - `ul` ： `margin : 16px 0;` `padding-left : 40px;` `list-style : disc;`
  - `a` ： `text-decoration: underline;`
- `css reset` (样式重置) :
  - `*{ margin:0;padding:0;}`
    - 优点：不用考虑哪些标签有默认的 `magin` 和 `padding`
    - 缺点：稍微的影响性能
      - `body,p,h1,ul{ margin:0;padding:0;}`
  - `ul{ list-style:none;}`
  - `a{ text-decoration:none;color:#999;}`
  - `a:hover{ color:red;}`
  - `img{ display:block;}`
    - 问题的现象：图片跟容器底部有一些空隙。
      - 内联元素的对齐方式是按照文字基线对齐的，而不是文字底线对齐的。
      - `vertical-align : baseline;` 基线对齐方式，默认值
      - `img{ vertical-align : bottom;}` 底线对齐，解决方式是推荐的
- web前端助手下载：https://www.baidufe.com/fehelper