



CSS基础

选择器进阶+背景属性+元素显示模式+三大特性



黑马程序员
www.itheima.com

传智教育旗下
高端IT教育品牌



学习目标

Learning Objectives

- ◆ 能够使用 **复合选择器** 在 HTML 中定位元素
- ◆ 能够使用 **Emmet 语法** 提高编码速度
- ◆ 能够使用 **hover伪类选择器** 设置鼠标悬停状态时元素的样式
- ◆ 能够使用 **背景相关属性** 装饰元素样式
- ◆ 能够使用并控制 **背景图片** 的显示和位置
- ◆ 能够认识 **元素显示模式**，并通过代码实现元素显示模式的转换
- ◆ 能够实现行内元素和块级元素 **水平居中**



目录

Contents

- ◆ 选择器进阶
- ◆ 背景相关属性
- ◆ 元素显示模式
- ◆ CSS 三大特性
- ◆ 综合案例



目录

Contents

- ◆ 选择器进阶
- ◆ 背景相关属性
- ◆ 元素显示模式
- ◆ CSS 三大特性
- ◆ 综合案例

一、选择器进阶

目标：能够理解复合选择器的规则，并使用复合选择器在 HTML 中定位元素

学习路径：

1. 复合选择器
2. 并集选择器
3. 交集选择器
4. Emmet语法
5. hover伪类选择器

一、选择器进阶

1.1 后代选择器：空格

- 作用：根据 HTML 标签的嵌套关系，选择父元素 **后代中** 满足条件的元素
- 选择器语法：**选择器1 选择器2 { css }**
- 结果：
 - 在选择器1所找到标签的后代（儿子、孙子、重孙子...）中，找到满足选择器2的标签，设置样式
- 注意点：
 1. 后代包括：儿子、孙子、重孙子.....
 2. 后代选择器中，选择器与选择器之前通过 **空格** 隔开

一、选择器进阶

1.2 子代选择器：>

- 作用：根据 HTML 标签的嵌套关系，选择父元素 **子代中** 满足条件的元素
- 选择器语法：**选择器1 > 选择器2 { css }**
- 结果：
 - 在选择器1所找到标签的子代（儿子）中，找到满足选择器2的标签，设置样式
- 注意点：
 1. 子代只包括：儿子
 2. 子代选择器中，选择器与选择器之前通过 > 隔开

一、选择器进阶

目标：能够理解复合选择器的规则，并使用复合选择器在 HTML 中定位元素

学习路径：

1. 复合选择器
2. 并集选择器
3. 交集选择器
4. Emmet语法
5. hover伪类选择器

一、选择器进阶

2.1 并集选择器：，

- 作用：同时选择多组标签，设置相同的样式
- 选择器语法：选择器1 ， 选择器2 { css }
- 结果：
 - 找到 选择器1 和 选择器2 选中的标签，设置样式
- 注意点：
 1. 并集选择器中的每组选择器之间通过 ， 分隔
 2. 并集选择器中的每组选择器可以是基础选择器或者复合选择器
 3. 并集选择器中的每组选择器通常一行写一个，提高代码的可读性

一、选择器进阶

目标：能够理解复合选择器的规则，并使用复合选择器在 HTML 中定位元素

学习路径：

1. 复合选择器
2. 并集选择器
3. 交集选择器
4. Emmet语法
5. hover伪类选择器

一、选择器进阶

3.1 交集选择器：紧挨着

- 作用：选中页面中 **同时满足** 多个选择器的标签
- 选择器语法：**选择器1选择器2 { css }**
- 结果：
 - （既又原则）找到页面中 **既** 能被选择器1选中，**又** 能被选择器2选中的标签，设置样式
- 注意点：
 1. 交集选择器中的选择器之间是紧挨着的，没有东西分隔
 2. 交集选择器中如果有标签选择器，标签选择器必须写在最前面

一、选择器进阶

目标：能够理解复合选择器的规则，并使用复合选择器在 HTML 中定位元素

学习路径：

1. 复合选择器
2. 并集选择器
3. 交集选择器
4. Emmet语法
5. hover伪类选择器

4.1 emmet语法

- 作用：通过简写语法，快速生成代码
- 语法：
 - 类似于刚刚学习的选择器的写法

记忆	示例	效果
标签名	div	<code><div></div></code>
类选择器	.red	<code><div class="red"></div></code>
id选择器	#one	<code><div id="one"></div></code>
交集选择器	p.red#one	<code><p class="red" id="one"></p></code>
子代选择器	ul>li	<code></code>
内部文本	ul>li{我是li的内容}	<code>我是li的内容</code>
创建多个	ul>li*3	<code></code>

一、选择器进阶

目标：能够理解复合选择器的规则，并使用复合选择器在 HTML 中定位元素

学习路径：

1. 复合选择器
2. 并集选择器
3. 交集选择器
4. Emmet语法
5. **hover伪类选择器**

一、选择器进阶

5.1 hover 伪类选择器

- 作用：选中鼠标悬停在元素上的状态，设置样式
- 选择器语法：选择器: hover { css }
- 注意点：
 1. 伪类选择器选中的元素的某种状态

一、选择器进阶

目标：能够理解复合选择器的规则，并使用复合选择器在 HTML 中定位元素

选择器	作用	格式	示例
后代选择器	找后代	选择器之间通过 空格 分隔	<code>.father .son { css }</code>
子代选择器	找儿子	选择器之间通过 > 分隔	<code>.father > .son { css }</code>
并集选择器	找到多类元素	选择器之间通过 , 分隔	<code>div,p,span { css }</code>
交集选择器	找同时满足多个选择器的元素	选择器之间 紧挨着	<code>p.red { css }</code>
hover伪类选择器	选中鼠标悬停在元素上的状态	:hover	<code>a:hover { css }</code>



目录

Contents

- ◆ 选择器进阶
- ◆ 背景相关属性
- ◆ 元素显示模式
- ◆ CSS 三大特性
- ◆ 综合案例

二、背景相关属性

目标：能够使用 **背景相关属性** 装饰元素样式

学习路径：

1. **背景颜色**
2. 背景图片
3. 背景平铺
4. 背景位置
5. 背景相关属性连写

1.1 背景颜色

- 属性名: **background-color** (bgc)
- 属性值:
 - 颜色取值: 关键字、rgb表示法、rgba表示法、十六进制.....
- 注意点:
 - 背景颜色默认值是透明: `rgba(0,0,0,0)`、`transparent`
 - 背景颜色不会影响盒子大小, 并且还能看清盒子的大小和位置, 一般在布局中会习惯先给盒子设置背景颜色



二、背景相关属性

1.2 小结

- 背景颜色属性的属性名是？
 - background-color
- 背景颜色属性的属性值默认是？
 - 透明：rgba(0,0,0,0)、transparent

二、背景相关属性

目标：能够使用 **背景相关属性** 装饰元素样式

学习路径：

1. 背景颜色
2. **背景图片**
3. 背景平铺
4. 背景位置
5. 背景相关属性连写

二、背景相关属性

2.1 背景图片

- 属性名: **background-image** (bgi)
- 属性值: `background-image: url('图片的路径');`
- 注意点:
 - 背景图片中url中可以省略引号
 - 背景图片默认是在水平和垂直方向平铺的
 - 背景图片仅仅是指给盒子起到装饰效果, 类似于背景颜色, 是不能撑开盒子的



2.2 小结

- 背景图片属性的属性名是?
 - background-image
- 背景图片属性的属性值格式是什么样的?

```
background-image: url('图片的路径');
```

二、背景相关属性

目标：能够使用 **背景相关属性** 装饰元素样式

学习路径：

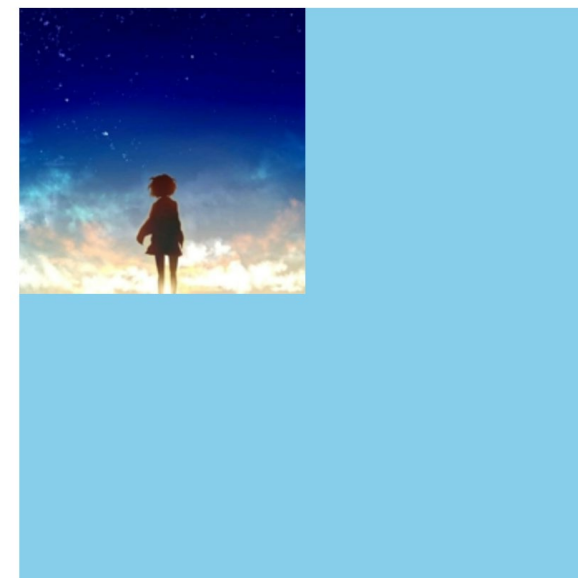
1. 背景颜色
2. 背景图片
3. **背景平铺**
4. 背景位置
5. 背景相关属性连写

二、背景相关属性

3.1 背景平铺

- 属性名: **background-repeat** (bgr)
- 属性值:

取值	效果
repeat	(默认值) 水平和垂直方向都平铺
no-repeat	不平铺
repeat-x	沿着水平方向 (x轴) 平铺
repeat-y	沿着垂直方向 (y轴) 平铺



3.2 小结

- 背景平铺属性的属性名是？
 - background-repeat
- 背景平铺属性的属性值常见的有哪些？

取值	效果
repeat	(默认值) 水平和垂直方向都平铺
no-repeat	不平铺
repeat-x	沿着水平方向 (x轴) 平铺
repeat-y	沿着垂直方向 (y轴) 平铺

二、背景相关属性

目标：能够使用 **背景相关属性** 装饰元素样式

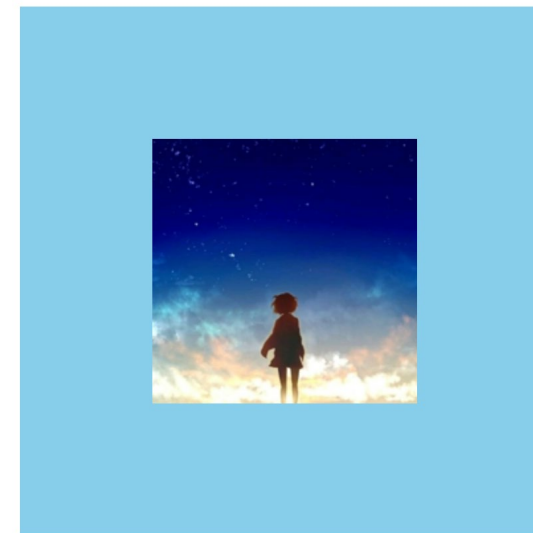
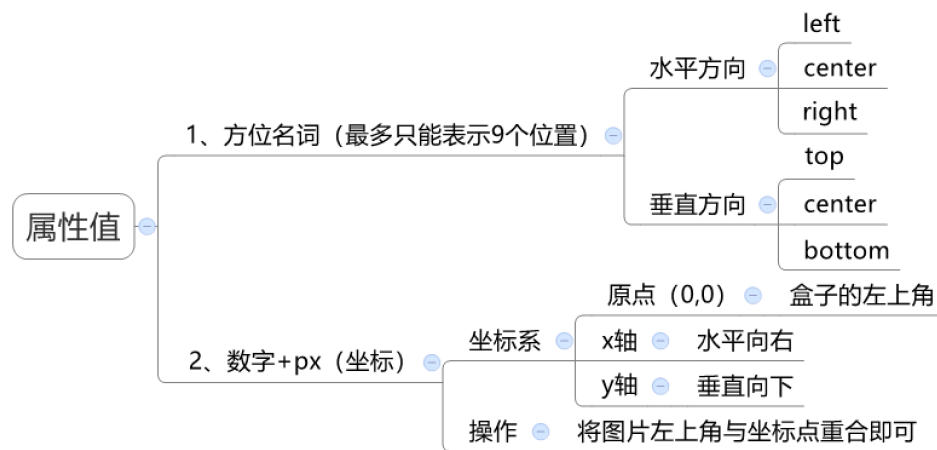
学习路径：

1. 背景颜色
2. 背景图片
3. 背景平铺
4. **背景位置**
5. 背景相关属性连写

二、背景相关属性

4.1 背景位置

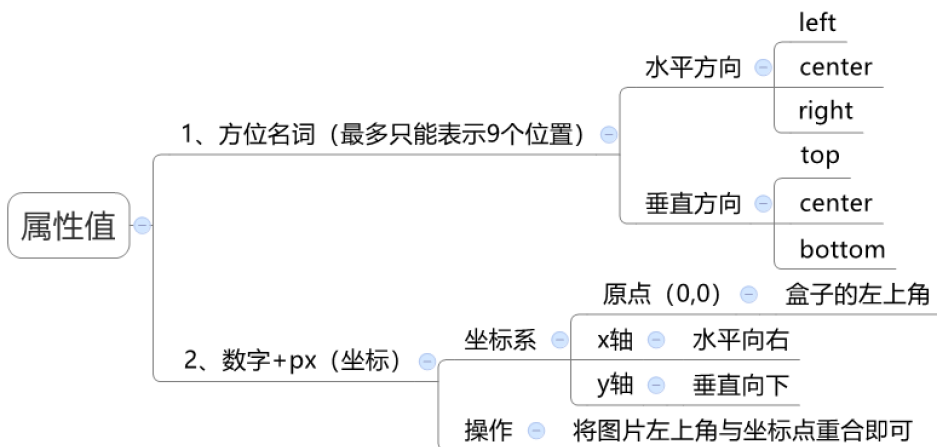
- 属性名: **background-position** (bgp)
- 属性值: **background-position**: 水平方向位置 垂直方向位置;



- 注意点:
 - 方位名词取值和坐标取值可以混使用, 第一个取值表示水平, 第二个取值表示垂直

4.2 小结

- 背景位置属性的属性名是？
 - background-position
- 背景位置属性的取值有哪些？



二、背景相关属性

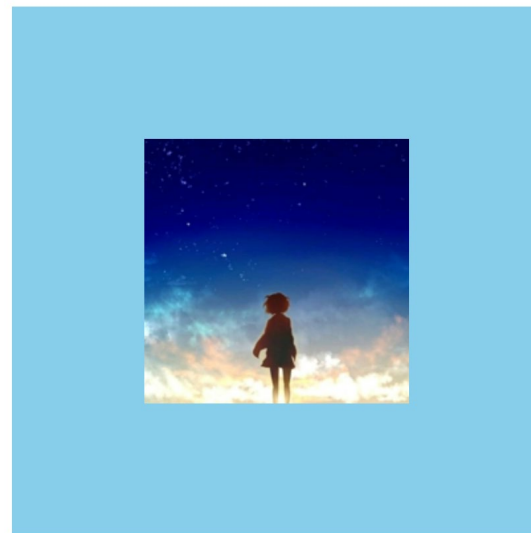
目标：能够使用 **背景相关属性** 装饰元素样式

学习路径：

1. 背景颜色
2. 背景图片
3. 背景平铺
4. 背景位置
5. **背景相关属性连写**

5.1 背景相关属性的连写形式


- 属性名: **background** (bg)
- 属性值:
 - 单个属性值的合写, 取值之间以空格隔开
- 书写顺序:
 - 推荐: background: color image repeat position
- 省略问题:
 - 可以按照需求省略
 - 特殊情况: 在pc端, 如果盒子大小和背景图片大小一样, 此时可以直接写 background: url()
- 注意点
 - 如果需要设置单独的样式和连写
 - ① 要么把单独的样式写在连写的下面
 - ② 要么把单独的样式写在连写的里面



5.2 小结

- 背景相关属性连写的属性名是?
 - background
- 背景相关属性连写的取值的写法是?
 - 单个属性值的合写，取值之间以空格隔开
 - 推荐：background: color image repeat position

6.1 拓展 `img` 标签和背景图片的区别

- 需求：需要在网页中展示一张图片的效果？
- 方法一：直接写上标签即可
 - `img` 标签是一个标签，不设置宽高默认会以原尺寸显示
- 方法二：`div` 标签 + 背景图片
 - 需要设置

div

的宽高，因为背景图片只是装饰的CSS样式，不能撑开

div

标签



目录

Contents

- ◆ 选择器进阶
- ◆ 背景相关属性
- ◆ 元素显示模式
- ◆ CSS 三大特性
- ◆ 综合案例

三、元素显示模式

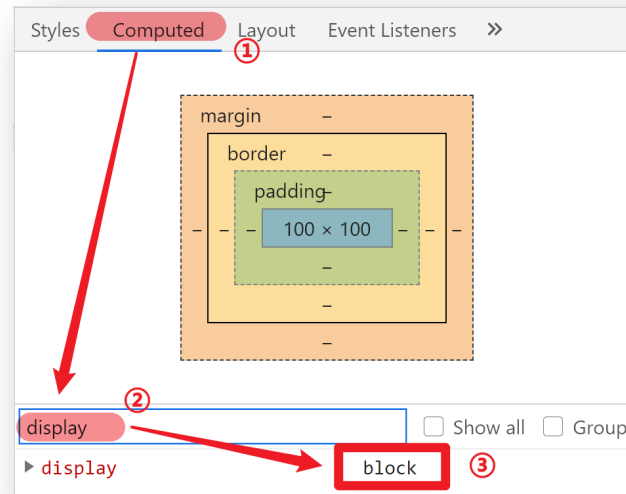
目标：能够认识**元素显示模式**，并通过代码实现元素显示模式的转换

学习路径：

1. 块级元素
2. 行内元素
3. 行内块元素
4. 元素显示模式转换

1.1 块级元素

- 属性: `display: block`
- 显示特点:
 1. 独占一行 (一行只能显示一个)
 2. 宽度默认是父元素的宽度, 高度默认由内容撑开
 3. 可以设置宽高
- 代表标签:
 - `div`、`p`、`h`系列、`ul`、`li`、`dl`、`dt`、`dd`、`form`、`header`、`nav`、`footer`.....



1.2 小结

- 块级元素的display属性的属性值是?
 - display: block
- 块级元素的显示特点有哪些?
 1. 独占一行（一行只能显示一个）
 2. 宽度默认是父元素的宽度，高度默认由内容撑开
 3. 可以设置宽高
- 块级元素的代表标签有哪些?
 - div、p、h系列

三、元素显示模式

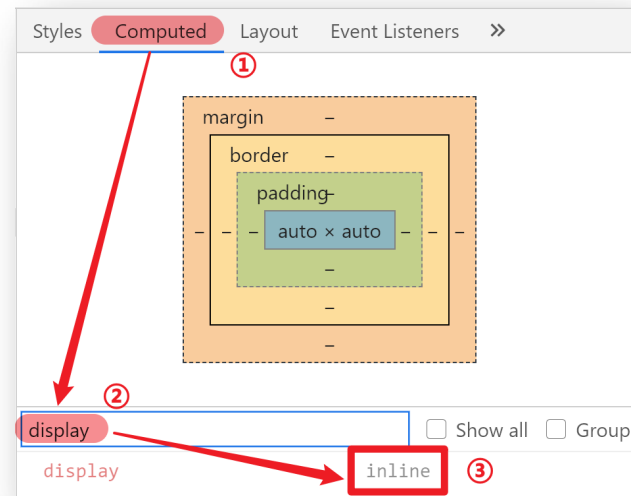
目标：能够认识**元素显示模式**，并通过代码实现元素显示模式的转换

学习路径：

1. 块级元素
2. **行内元素**
3. 行内块元素
4. 元素显示模式转换

2.1 行内元素

- 属性: display: inline
- 显示特点:
 1. 一行可以显示多个
 2. 宽度和高度默认由内容撑开
 3. 不可以设置宽高
- 代表标签:
 - a、span、b、u、i、s、strong、ins、em、del.....



2.2 小结

- 行内元素的display属性的属性值是?
 - display: inline
- 行内元素的显示特点有哪些?
 1. 一行可以显示多个
 2. 宽度和高度默认由内容撑开
 3. 不可以设置宽高
- 行内元素的代表标签有哪些?
 - a、span

三、元素显示模式

目标：能够认识**元素显示模式**，并通过代码实现元素显示模式的转换

学习路径：

1. 块级元素
2. 行内元素
3. **行内块元素**
4. 元素显示模式转换

3.1 行内块元素

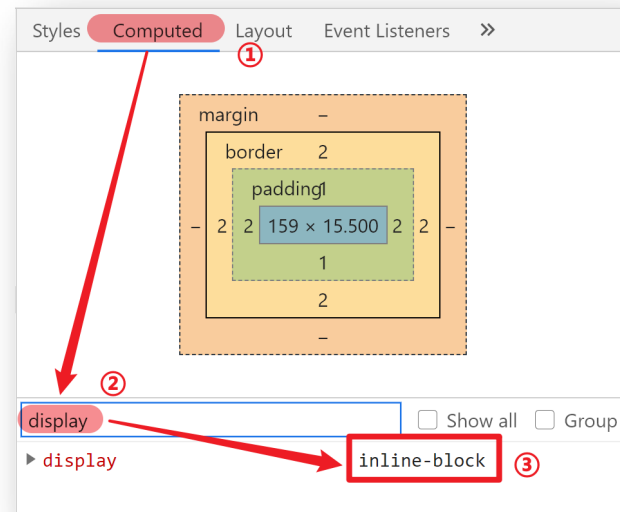
➤ 属性: display: **inline-block**

➤ 显示特点:

1. 一行可以显示多个
2. 可以设置宽高

➤ 代表标签:

- input、textarea、button、select.....
- 特殊情况: img标签有行内块元素特点, 但是Chrome调试工具中显示结果是inline



3.2 小结

- 行内块元素的display属性的属性值是?
 - display: inline-block
- 行内块元素的显示特点有哪些?
 1. 一行可以显示多个
 2. 可以设置宽高
- 行内块元素的代表标签有哪些?
 - input、textarea

三、元素显示模式

目标：能够认识**元素显示模式**，并通过代码实现元素显示模式的转换

学习路径：

1. 块级元素
2. 行内元素
3. 行内块元素
4. **元素显示模式转换**

4.1 元素显示模式转换

- 目的：改变元素默认的显示特点，让元素符合布局要求
- 语法：

属性	效果	使用频率
<code>display: block</code>	转换成块级元素	较多
<code>display: inline-block</code>	转换成行内块元素	较多
<code>display: inline</code>	转换成行内元素	极少

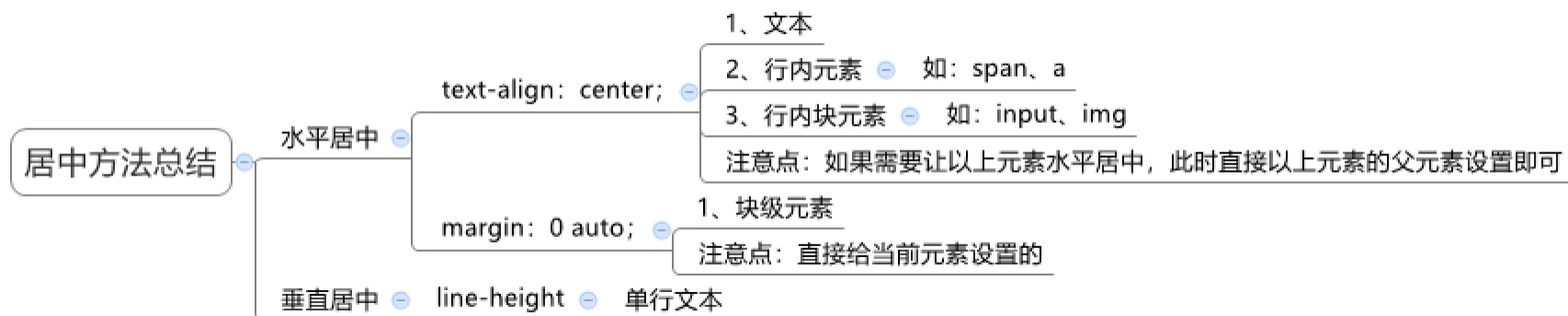
4.2 小结

- 转换成块级元素的属性是什么?
 - display: block
- 转换成行内块元素的属性是什么?
 - display: inline-block
- 转换成行内元素的属性是什么?
 - display: inline

拓展1：HTML嵌套规范注意点

1. 块级元素一般作为大容器，可以嵌套：文本、块级元素、行内元素、行内块元素等等.....
 - 但是：p标签中不要嵌套div、p、h等块级元素
2. a标签内部可以嵌套任意元素
 - 但是：a标签不能嵌套a标签

拓展2：居中方法总结





目录

Contents

- ◆ 选择器进阶
- ◆ 背景相关属性
- ◆ 元素显示模式
- ◆ CSS 三大特性
- ◆ 综合案例

四、CSS 三大特性

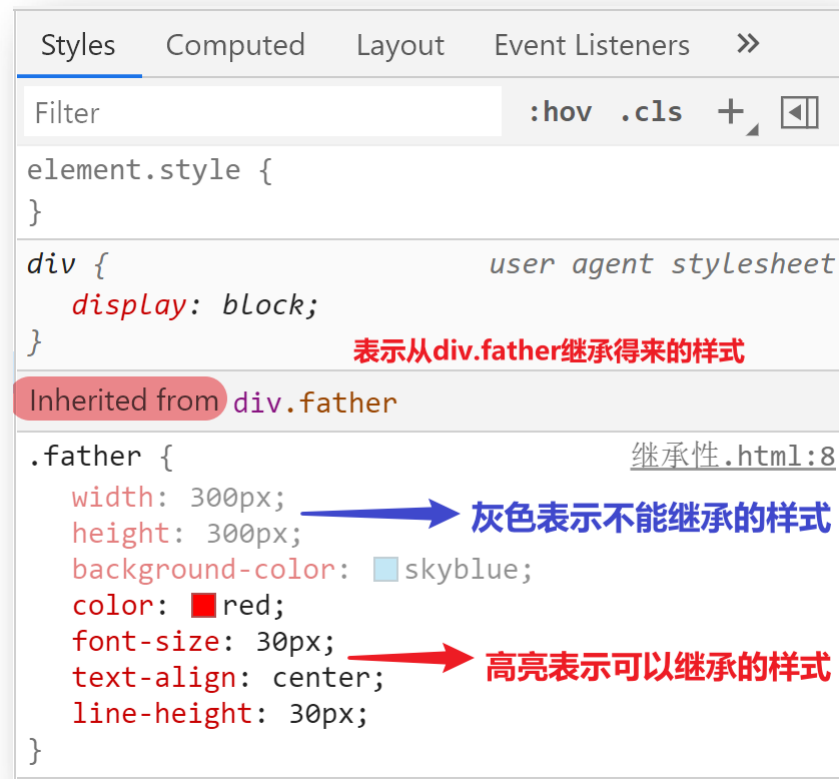
目标：能够认识 CSS 的 **继承** 和 **层叠** 特性，会计算 CSS 的优先级权重的比较

学习路径：

1. **继承性**
2. 层叠性
3. 优先级

1.1 继承性的介绍

- 特性：子元素有默认继承父元素样式的特点（子承父业）
- 可以继承的常见属性：
 1. color
 2. font-style、font-weight、font-size、font-family
 3. text-indent、text-align
 4. line-height
 5.
- 注意点：
 - 可以通过调试工具判断样式是否可以继承



1.2 小结

- 继承性的特性是什么?
 - 子元素有默认继承父元素样式的特点
- 有哪些常见属性可以继承?
 1. color
 2. font-style、font-weight、font-size、font-family
 3. text-indent、text-align
 4. line-height

拓展 继承的应用

- 好处：可以在一定程度上减少代码
- 常见应用场景：
 1. 可以直接给ul设置 `list-style:none` 属性，从而去除列表默认的小圆点样式
 2. 直接给body标签设置统一的font-size，从而统一不同浏览器默认文字大小

拓展 继承失效的特殊情况

- 如果元素有浏览器默认样式，此时继承性依然存在，但是优先显示浏览器的默认样式
- 1. a标签的color会继承失效
 - 其实color属性继承下来了，但是被浏览器默认设置的样式给覆盖掉了
- 2. h系列标签的font-size会继承失效
 - 其实font-size属性继承下来了，但是被浏览器默认设置的样式给覆盖掉了
- 3. div的高度不能继承，但是宽度有类似于继承的效果
 - 宽度属性不能基继承，但是div有独占一行的特性

四、CSS 三大特性

目标：能够认识 CSS 的 **层叠** 和 **继承** 特性，会计算 CSS 的优先级权重计算

学习路径：

1. 继承性
2. **层叠性**
3. 优先级

2.1 层叠性的介绍

➤ 特性：

1. 给同一个标签设置不同的样式 → 此时样式会层叠叠加 → 会共同作用在标签上
2. 给同一个标签设置相同的样式 → 此时样式会层叠覆盖 → 最终写在最后的样式会生效

➤ 注意点：

1. 当样式冲突时，只有当选择器优先级相同时，才能通过层叠性判断结果

2.2 小结

- 如果给同一个标签设置了相同的属性，此时样式会？
 - 会层层覆盖，写在最后的会生效
- 如果给同一个标签设置了不同的样式，此时样式会？
 - 会层层叠加，共同作用在标签上

四、CSS 三大特性

目标：能够认识 CSS 的 **层叠** 和 **继承** 特性，会计算 CSS 的优先级权重计算

学习路径：

1. 继承性
2. 层叠性
3. **优先级**

3.1 优先级的介绍

- 特性：不同选择器具有不同的优先级，优先级高的选择器样式会覆盖优先级低选择器样式
- 优先级公式：
 - 继承 < 通配符选择器 < 标签选择器 < 类选择器 < id选择器 < 行内样式 < !important
- 注意点：
 1. !important写在属性值的后面，分号的前面！
 2. !important不能提升继承的优先级，**只要是继承优先级最低！**
 3. 实际开发中不建议使用 !important，推荐使用：后代 + 类选择器的形式更为便捷

3.2 权重叠加计算

➤ 场景：如果是复合选择器，此时需要通过权重叠加计算方法，判断最终哪个选择器优先级最高会生效

➤ 权重叠加计算公式：（每一级之间不存在进位）

1. 第一级数字：行内样式的个数
2. 第二级数字：id选择器的个数
3. 第三级数字：类选择器的个数
4. 第四级数字：标签选择器的个数

➤ 比较规则：

1. 先比较第一级数字，如果比较出来了，之后的统统不看
2. 如果第一级数字相同，此时再去比较第二级数字，如果比较出来了，之后的统统不看
3.
4. 如果最终所有数字都相同，表示优先级相同，则比较层叠性（谁写在下面，谁说了算!）

➤ 注意点：**!important**如果不是继承，则权重最高，天下第一！



3.3 小结

- 单个选择器优先级比较公式是怎样的？
 - 继承 < 通配符选择器 < 标签选择器 < 类选择器 < id选择器 < 行内样式 < !important
- 复合选择器权重叠加计算的公式是怎样的？



拓展 权重叠加计算案例

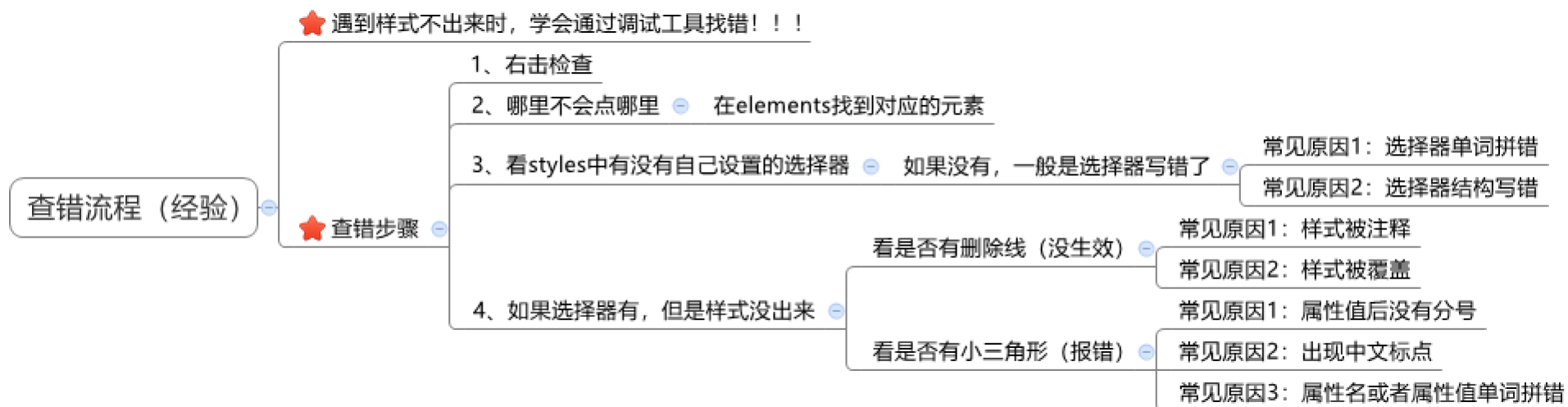
➤ 权重计算题解题步骤：

1. 先判断选择器 **是否能直接选中标签**，如果不能直接选中 → 是继承，优先级最低 → **直接pass**
2. 通过权重计算公式，判断谁权重最高

➤ 注意点：

- 实际开发中选择选择标签需要精准，尽量避免多个选择器同时选中一个标签的情况，不要自己难为自己

拓展 查错流程 （遇到样式出不来，要学会通过调试工具找错）





目录

Contents

- ◆ 选择器进阶
- ◆ 背景相关属性
- ◆ 元素显示模式
- ◆ 三大特性
- ◆ 综合案例

五、综合案例

目标：通过今天学习的CSS相关属性，完成综合案例

学习路径：

1. 综合案例1-普通导航
2. 综合案例2-五彩导航

1. 普通导航-效果图



五、综合案例

目标：通过今天学习的CSS相关属性，完成综合案例

学习路径：

1. 综合案例1-普通导航
2. 综合案例2-五彩导航

2. 五彩导航-效果图





总结

- ◆ 能够使用 **复合选择器** 在 HTML 中定位元素
- ◆ 能够使用 **Emmet 语法** 提高编码速度
- ◆ 能够使用 **hover 伪类选择器** 设置鼠标悬停状态时元素的样式
- ◆ 能够使用 **背景相关属性** 装饰元素样式
- ◆ 能够使用并控制 **背景图片** 的显示和位置
- ◆ 能够认识 **元素显示模式**，并通过代码实现元素显示模式的转换
- ◆ 能够实现行内元素和块级元素 **水平居中**



传智教育旗下高端IT教育品牌