HTML 与 CSS 讲义

中文版

Contents

1 HTML 基础 **3**
- 1.1 Your first page . . . 3
- 1.2 Headings & paragraphs . . . 3
- 1.3 Text formatting . . . 4

2 列表、链接与图片 **5**
- 2.1 Lists . . . 5
- 2.2 Links . . . 5
- 2.3 Images . . . 6

3 页面结构 **7**
- 3.1 一个完整的页面 . . . 7
- 3.2 注释与嵌套 . . . 8

4 表格与表单 **9**
- 4.1 Tables . . . 9
- 4.2 Forms . . . 10

5 盒子与分区 **11**
- 5.1 div、span 与语义分区 . . . 11

6 CSS 基础 **12**
- 6.1 添加 CSS . . . 12
- 6.2 选择器:class 与 id . . . 12

7 颜色与文字 **14**
- 7.1 Colours & backgrounds . . . 14
- 7.2 Styling text . . . 14

8 盒模型 **15**
- 8.1 外边距与内边距 . . . 15
- 8.2 display 显示 . . . 16

9 布局 **17**
- 9.1 宽度与居中 . . . 17
- 9.2 Flexbox . . . 17
- 9.3 小项目: 个人名片 . . . 18

10 更进一步 **19**
- 10.1 Hover & pseudo-classes . . . 19
- 10.2 Positioning . . . 20
- 10.3 Media queries & responsive design . . . 20

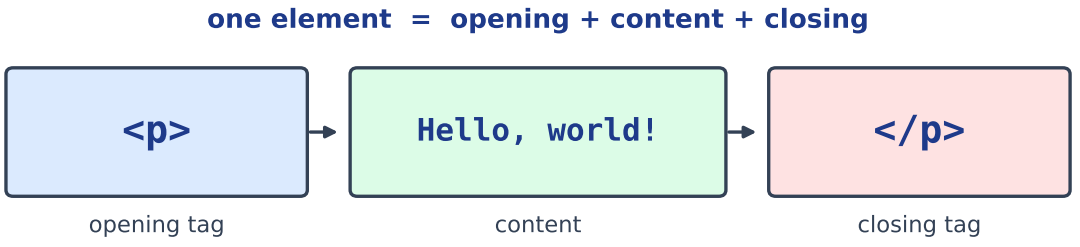
1 HTML 基础

1.1 ■ Your first page

HTML(超文本标记语言) 用标签 (tag) 来描述一个网页。标签是尖括号里的一个词, 比如 。大多数标签成对出现 — 一个开始标签 和一个结束标签 把内容包起来。一个标签加上它的内容就是一个元素 (element)。

```
<p>Hello, world!</p>
<p>This is my first web page.</p>
```

- 浏览器会画出标签之间的内容。
- 忘记闭合或拼错的标签, 可能显示不正确。



Exam tip: most tags come in pairs — forgetting breaks layout below

$$\textit{One element} = \textit{opening tag} + \textit{content} + \textit{closing tag}$$

1.2 ■ Headings & paragraphs

标题标记六个等级，从 <h1>（最高级）到 <h6>（最低级）；CSS 控制其字号。段落标签为 。页面主标题应仅使用一个 <h1>。

```
<h1>My Blog</h1>
<h2>About me</h2>
<p>I am learning to build web pages.</p>
```

1.3 ■ Text formatting

使用 `<strong>` 表示强调（通常加粗），使用 `<em>` 表示着重（通常斜体）。CSS 可更改其外观。换行符为 `<br>` — 它没有闭合标签。

```
<p><strong>Warning:</strong> read this carefully.</p>
<p>First line<br>second line</p>
<p>You can <em>emphasise</em> any word.</p>
```

常见错误

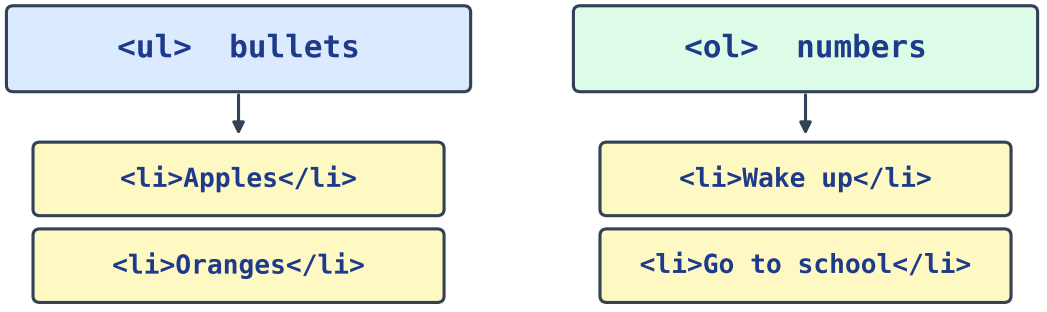
- 大多数元素需成对使用开始和结束标签。错误的嵌套可能导致浏览器以意外方式修复文档。
- 标签写在尖括号里; 你看到的文字写在开始和结束标签之间。
- 标题等级从 1 (`<h1>`) 到 6 (`<h6>`) — 不要仅因字号大小而随意选择标题等级。

2 列表、链接与图片

2.1 ■ Lists

无序列表 (unordered list)`<ul>` 显示圆点; 有序列表 (ordered list)`<ol>` 显示数字。每个列表项 (list item) 是一个 `<li>`。

```
<ul>
  <li>Apples</li>
  <li>Oranges</li>
</ul>
<ol>
  <li>Wake up</li>
  <li>Go to school</li>
</ol>
```



Exam tip: every item is <li>; wrap them in <ul> (bullets) or <ol> (numbers)

ul 用于无序列表，*ol* 用于有序列表；每个项目项为 *li*

2.2 ■ Links

链接 (link) 是一个锚 `<a>`。它的 `href` 属性 (attribute) 保存它指向的地址。

```
<a href="https://example.com">Visit Example</a>
<br>
<a href="page2.html">Go to the next page</a>
```

- 加上 `target="_blank"` 可以在新标签页打开链接。

2.3 ■ Images

图片 (image) 是 `<img>`— 它没有结束标签。`src` 是图片的地址;`alt` 是当图片无法加载时显示的文本 (也会读给视障用户听)。

```
<img src=」
↪  "https://upload.wikimedia.org/wikipedia/commons/thumb/6/6b/Bitmap_VS_SVG.svg/
     alt="bitmap versus vector" width="120">
<p>If the picture is missing, the alt text shows instead.</p>
```

常见错误

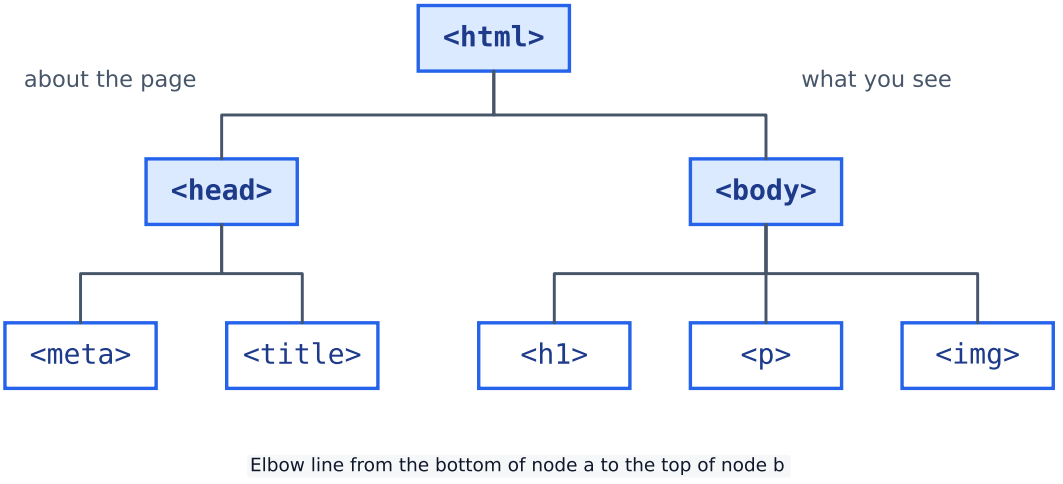
- 一张信息性图片需要替代文本来说明其含义。一张纯装饰性图片使用 `alt=""`; 省略 `alt` 则不同。一个链接需要 `href`。
- `<img>` 没有单独的闭合标签。
- 一个损坏的 `src` 路径可以显示替代文本而不是图片 — 检查文件名和文件夹。

3 页面结构

3.1 ■ 一个完整的页面

真正的页面有固定的骨架 (skeleton)。`<!DOCTYPE html>` 表示它是 HTML5。所有内容都放在 `<html>` 里面。`<head>` 放关于页面的信息;`<body>` 放你看得见的内容。

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My Page</title>
  </head>
  <body>
    <h1>Welcome</h1>
    <p>This is a complete page.</p>
  </body>
</html>
```



Elbow line from the bottom of node a to the top of node b

页面是一棵树:*head* 描述页面,*body* 放你看到的内容

- `<title>` 显示在浏览器标签页上。
- `<meta charset="utf-8">` 让页面能显示任何语言。

3.2 ■ 注释与嵌套

注释 (comment)`<!-- ... -->` 是浏览器会忽略的备注。标签彼此嵌套 (nest); 每一层都缩进 (indent), 让结构清楚。按打开的相反顺序关闭标签。

```
<!-- a simple navigation bar -->
<nav>
  <ul>
    <li><a href="#">Home</a></li>
    <li><a href="#">About</a></li>
  </ul>
</nav>
```

常见错误

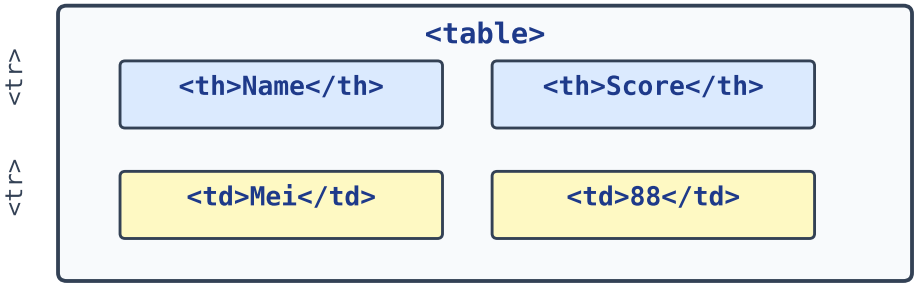
- 标签要嵌套, 不能交叉:`<b><i>text</i></b>`, 绝不能 `<b><i>text</b></i>`。
- 一个页面需要 `<html>`、一个 `<head>`(放标题和链接) 和一个 `<body>`(放看得见的内容)。
- 注释是 `<!-- ... -->`; 浏览器会忽略它。

4 表格与表单

4.1 ■ Tables

表格 (table)<table> 由行 <tr> 构成。表头单元格 (cell) 是 <th>(加粗、居中); 普通数据单元格是 <td>。

```
<style>
  table { border-collapse: collapse; }
  th, td { border: 1px solid; padding: 6px; }
</style>
<table>
  <tr><th>Name</th><th>Score</th></tr>
  <tr><td>Mei</td><td>88</td></tr>
  <tr><td>Sam</td><td>71</td></tr>
</table>
```



Exam tip: each <tr> is a row; use <th> for headers and <td> for data cells

table → tr rows → th/td cells

4.2 ■ Forms

表单 (form)<form> 收集用户的输入。<input> 是一个输入框;<label> 给它一个名字;<button> 用来提交 (submit)。

```
<form>
  <label>Name: <input type="text" name="name"></label>
  <br><br>
  <label>Age: <input type="number" name="age"></label>
  <br><br>
  <button type="submit">Send</button>
</form>
```

- name 属性是提交时随控件值一同发送的键名；未命名的文本输入框不会被提交。
- 常见的 <input> type 值:text、number、email、password、checkbox。

常见错误

- 表格单元格放在行里面: 一个 <tr> 装若干 <td> 单元格。
- 每个表单输入都应有对应的 <label>, 这样更清楚也更易用。
- 表单里的按钮默认会提交表单 — 不想这样就设 type="button"。

5 盒子与分区

5.1 ■ div、span 与语义分区

 是一个通用的块级 (block) 盒子, 把内容成块地组合起来。 是一行文字里通用的行内 (inline) 盒子。用它们把内容分组, 方便设置样式或定位。

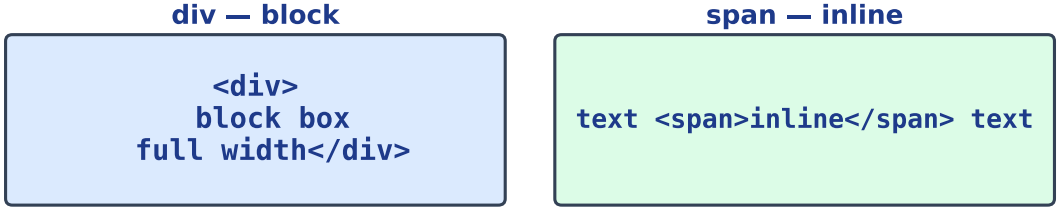
```
<div>
  <span>Price:</span> <span>$9.99</span>
</div>
<div>
  <span>Stock:</span> <span>in stock</span>
</div>
```

语义 (semantic) 标签描述自己的角色, 有助于屏幕阅读器和搜索引擎。在合适的地方优先用它们, 而不是普通的 :<header>、<nav>、<main>、<section>、<article>、<footer>。

```
<header><h1>My Site</h1></header>
<main>
  <section>
    <h2>News</h2>
    <p>Today we launched the site.</p>
  </section>
</main>
<footer>© 2026 My Site</footer>
```

常见错误

- 是块级 (另起一行); 是行内 (留在文字中)。
- 在合适处用 <header>、<nav>、<footer> 等语义标签, 别什么都用 。
- 不要只为了标签的默认外观去用它 — 外观交给 CSS。



Exam tip: div groups sections; span styles a word inside a line; prefer semantic tags when possible

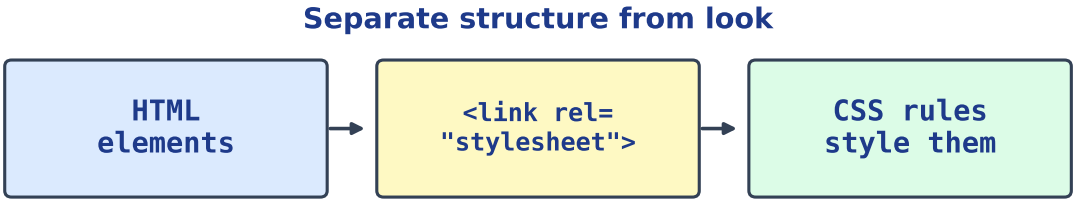
div 是块级元素； *span* 包裹行内文本

6 CSS 基础

6.1 ■ 添加 CSS

CSS(层叠样式表) 给 HTML 设置样式。最简单的方式是在 `<head>` 里放一个 `<style>` 块 (更大的站点会用单独的 `.css` 文件配合 `<link>`)。一条 CSS 规则 (rule) 有一个选择器 (selector), 后面是 `{ property: value; }` 这样的对, 叫作声明 (declaration)。

```
<style>
  p  { color: blue; font-size: 18px; }
  h1 { color: darkred; }
</style>
<h1>Styled heading</h1>
<p>This paragraph is blue.</p>
```



Exam tip: put CSS in a .css file and link it in <head>; class/id target elements

Link a stylesheet so CSS rules style HTML elements

6.2 ■ 选择器:class 与 id

类 (class) 给许多元素设置样式: 在 HTML 里加 `class="..."`, 在 CSS 里写 `.name`。id(标识符) 只给一个元素设置样式:`id="..."` 和 `#name`。

```
<style>
  .note  { color: green; }
  #title { font-size: 24px; }
</style>
<h1 id="title">Welcome</h1>
<p class="note">First note.</p>
<p class="note">Second note.</p>
```

最常见的选择器, 集中在一张表里:

选择器	选中
`p`	每个 `<p>` 元素
`.note`	所有 `class="note"` 的元素
`#title`	那个 `id="title"` 的元素
`h1, h2`	同时选中多个选择器
`nav a`	`<nav>` 里的每个 `<a>`
`.btn:hover`	元素的一种状态 (见主题 10)

常见错误

- 类选择器以 `.` 开头,id 选择器以 `#` 开头。
- 在 `<head>` 里链接样式表, 或把规则放进 `<style>` 块。
- 一个类可以用在许多元素上; 一个 id 每页只应出现一次。

7 颜色与文字

7.1 ■ Colours & backgrounds

用 color 设置文字颜色, 用 background-color 设置背景 (background)。颜色可以是名字 (red)、十六进制 (hex) 代码 (#ff0000), 或 rgb(255, 0, 0)。

```
<style>
  body { background-color: #f0f4ff; }
  .card { background-color: white; color: #1e3a8a; padding: 12px; }
</style>
<div class="card">A card with its own colours.</div>
```



Exam tip: colour can be a name, #hex, or rgb(); background-color paints the box

CSS colours: named, hex, and rgb()

7.2 ■ Styling text

常用文本属性: font-size、font-weight (如 bold)、font-family 字体、text-align 对齐方式 (left / center / right), 以及用于间距的 line-height。

```
<style>
  h1 { font-family: Arial, sans-serif; text-align: center; }
  p  { font-size: 18px; line-height: 1.6; }
</style>
<h1>Centered title</h1>
<p>Readable paragraph text with comfortable line spacing.</p>
```

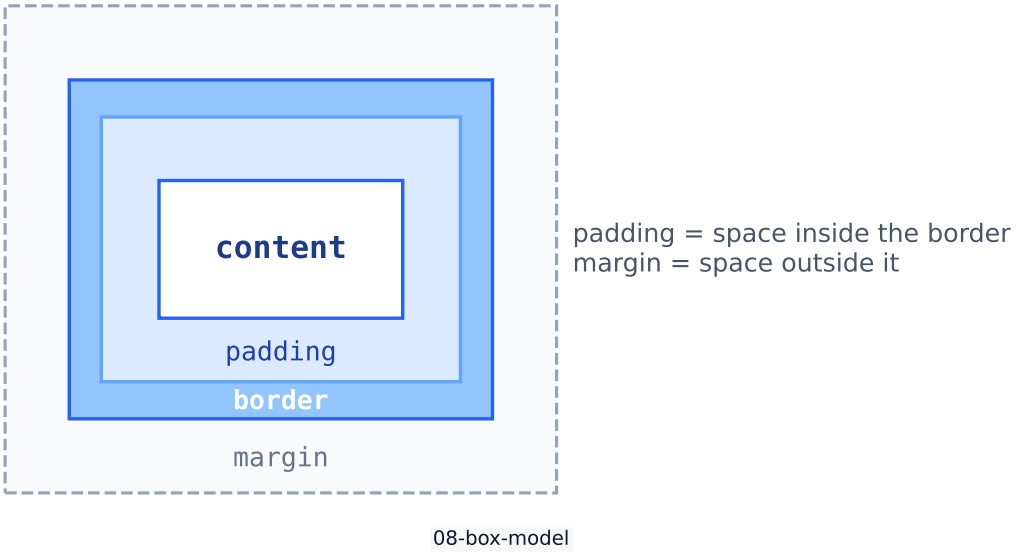
常见错误

- color 设置文字颜色;background-color 设置背景色。
- 颜色可以是名字、#rrggbb 十六进制码, 或 rgb(...)。
- 大括号里每条声明 (declaration) 都以分号 ; 结尾 — 分号用来分隔 property: value 对 (最后一条声明后的分号可选)。规则本身以右大括号 } 结尾。

8 盒模型

每个元素都是一个盒子, 从里到外有四层: 内容 (content)、内边距 (padding)(边框内侧的空间)、边框 (border), 以及外边距 (margin)(边框外侧的空间)。

```
<style>
  .box { background: #dbeafe; border: 3px solid #2563eb;
         padding: 16px; margin: 20px; }
</style>
<div class="box">content + padding + border + margin</div>
```



盒模型: 内容, 然后内边距, 然后边框, 然后外边距

8.1 ■ 外边距与内边距

padding 在边框**内侧**加空间, 把内容往里推。margin 在**外侧**加空间, 把其它盒子推开。可以给一个值 (四边相同), 或四个值 (上右下左)。

```
<style>
  .a { background: #fde68a; padding: 4px; }
  .b { background: #bbf7d0; padding: 20px; margin-top: 16px; }
</style>
<div class="a">tight padding</div>
<div class="b">roomy padding</div>
```

8.2 ■ display 显示

`display` 属性决定盒子如何排布。`block`(块级) 占一整行;`inline`(行内) 处在一行文字之中;`none` 把盒子完全移除;`flex` 把子元素排成一行 (下一主题)。

```
<style>
  .hidden { display: none; }
  .chip { display: inline; background: #e0e7ff; padding: 2px 8px;
↪   border-radius: 6px; }
</style>
<p>Visible text. <span class="hidden">(secret)</span> <span
↪   class="chip">a chip</span></p>
```

常见错误

- 内边距 (padding) 在边框内侧; 外边距 (margin) 是边框外侧的空间。
- `box-sizing: border-box` 让设定的宽度**包含**内边距和边框 — 通常正合你意。
- `display: none` 会彻底隐藏元素;`visibility: hidden` 会保留它的位置。

9 布局

9.1 ■ 宽度与居中

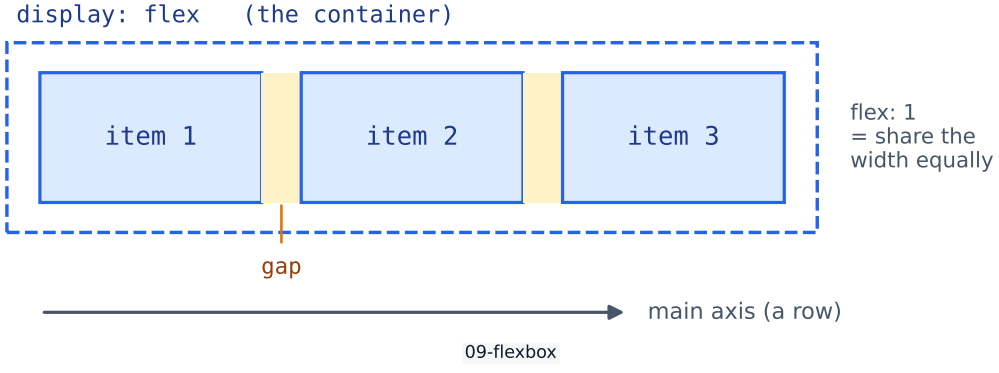
设置盒子的 `width`。要让一个块居中 (centre), 给它一个宽度, 再加 `margin: 0 auto`——`auto` 会把剩余空间在左右两边平分。

```
<style>
  .page { width: 300px; margin: 0 auto; background: #e0e7ff;
          padding: 16px; text-align: center; }
</style>
<div class="page">A centered, fixed-width page.</div>
```

9.2 ■ Flexbox

Flexbox(弹性布局) 把子元素排成一行 (或一列)。在父元素 — 容器 (container)— 上设置 `display: flex`。`gap` 在子元素之间加空隙;`flex: 1` 让它们平分宽度。

```
<style>
  .row { display: flex; gap: 10px; }
  .row div { background: #93c5fd; padding: 12px; flex: 1; text-align:
↪   center; }
</style>
<div class="row">
  <div>One</div>
  <div>Two</div>
  <div>Three</div>
</div>
```



弹性容器沿主轴摆放子项;*gap* 在它们之间加间距

9.3 ■ 小项目：个人名片

把盒子、颜色、文字和间距组合成一个小组件 (component)。

```
<style>
  .card { width: 240px; margin: 0 auto; border: 1px solid #cbd5e1;
          border-radius: 12px; padding: 16px; font-family: sans-serif; }
  .card h2 { margin: 0; color: #1e3a8a; }
  .card .role { color: #64748b; margin: 4px 0 8px; }
</style>
<div class="card">
  <h2>Mei Chen</h2>
  <p class="role">Student · Grade 11</p>
  <p>Loves maths and building web pages.</p>
</div>
```

常见错误

- 给设定了宽度的块用 `margin: 0 auto` 居中。
- Flexbox 要在**容器**上写 `display: flex`, 不是在子项上。
- 以 % 为单位的宽度是相对父元素的, 所以也要看父元素的宽度。

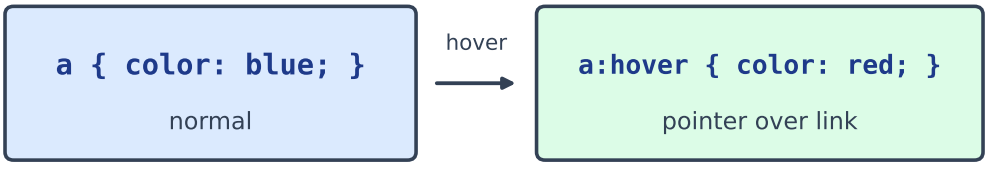
10 更进一步

10.1 ■ Hover & pseudo-classes

伪类 (pseudo-class) 给处于某种状态的元素设置样式。`:hover` 在鼠标悬停在元素上时生效。加上 `transition`(过渡), 变化就会平滑发生, 而不是突然跳变。

```
<style>
  .btn { background: #2563eb; color: white; padding: 10px 18px;
         border: none; border-radius: 8px; font-size: 16px;
         transition: background 0.3s; }
  .btn:hover { background: #16a34a; }
</style>
<button class="btn">Hover me</button>
```

- 其他有用的伪类: `:focus` (获得焦点的元素, 例如按 Tab 键或点击后) 和`:first-child`。



Exam tip: :hover, :focus, :active are pseudo-classes — write selector:pseudo

:hover 在指针悬停时样式化元素

10.2 ■ Positioning

定位定位将方框移出正常位置。`relative` 偏移方框（并将其作为子元素的锚点锚点）；`absolute` 从最近的定位祖先精确放置方框；`fixed` 通常在页面滚动时将其固定在视口上。变换过的祖先则建立其包含块。

```
<style>
  .card { position: relative; width: 260px; height: 90px;
          background: #dbeafe; border-radius: 10px; padding: 12px; }
  .badge { position: absolute; top: -10px; right: -10px;
           background: #dc2626; color: white; padding: 4px 10px;
           border-radius: 999px; font-size: 13px; }
</style>
<div class="card">
  A product card
  <span class="badge">NEW</span>
</div>
```

- `top`、`right`、`bottom`、`left` 指定移动多远 — 它们只对已定位的盒子生效。

10.3 ■ Media queries & responsive design

响应式 (responsive) 页面会适应屏幕大小。媒体查询 (media query) 只在屏幕满足某个条件时才应用它的规则, 例如 `max-width`。

```
<style>
  .banner { background: #2563eb; color: white; padding: 16px; }
  @media (max-width: 600px) {
    .banner { background: #16a34a; }   /* narrow screens turn green */
  }
</style>
<div class="banner">Resize the window: blue when wide, green when
↪  narrow.</div>
```

- 移动优先 (mobile-first) 的做法: 先写手机样式, 再用 `@media (min-width: ...)` 补充大屏样式。
- 百分比宽度和 `max-width` 让盒子能跟着屏幕缩小。

常见错误

- `:hover` 紧贴选择器, 中间没有空格: 是 `.btn:hover`, 不是 `.btn :hover`。
- `position: absolute` 从最近的**定位**祖先测量 — 需指定预期的包含祖先 `position: relative`。若无包含祖先，则使用初始包含块。
- 媒体查询要包住完整的规则:`@media (max-width: 600px) { .a { ... } }`。