模块一　HTML5 基础

项目 1　制作个人信息展示页面

项目背景

在数字化时代，个人信息展示页面已成为线上社交、求职应聘、职业推广的核心载体。无论是领英等职业社交平台的个人主页，还是设计师、程序员的作品集首页，都需要通过简洁直观的页面呈现个人核心信息。这类页面不仅是个人形象的数字化延伸，更是快速建立第一印象的重要窗口。对于初学者而言，制作个人信息展示页面作为本书的第一个实践项目，是 Web 前端开发的入门阶梯，能帮助理解网页“结构 + 样式”的基本构成逻辑，掌握基础开发模式，为后续学习复杂页面打下认知基础，同时满足个人在网络空间展示自我的实际需求。

项目描述

本项目通过制作含个人照片与详细简介的静态网页，通过文字与图片组合展示姓名、职业、教育背景等核心信息，帮助访问者快速了解个人情况。项目涉及 HTML5 与 CSS3 基础核心知识：HTML5 方面需掌握文档结构、<meta> 标签配置、文本与图片标签及路径应用；CSS3 方面要运用基础语法与选择器，定义文字样式、段落排版及图片样式。制作分 3 个阶段：搭建 HTML 文档框架、填充个人信息内容、使用 CSS 样式修饰页面，最终形成布局清晰的个人信息展示页面。

学习目标

1. 知识目标

（1）掌握 HTML5 文档核心结构标签 <html>、<head> 和 <body> 的功能与嵌套逻辑。

（2）理解 <meta> 标签中 charset 与 viewport 配置的作用及应用场景。

（3）掌握 <p> 和 <h1>~<h6> 等文本标签的语义层级及使用规范。

（4）理解 <img> 标签路径（相对 / 绝对）配置原理及 alt 属性意义。

（5）了解 CSS 中字体、颜色、尺寸等样式属性的基础语法规则。

（6）了解 和 标签的语义化强化功能及使用区别。

2. 技能目标

（1）能独立搭建符合规范的 HTML 文档框架并编写基础结构代码。

（2）掌握插入图片并根据需求配置相对或绝对路径的方法。

（3）能运用 和 标签对关键文本进行语义化强化处理。

（4）会使用 CSS 定义字体、颜色、行高及首行缩进等文本样式。

3. 素质目标

（1）培养按规范编写代码的严谨性与良好开发习惯。

（2）提升对页面布局与视觉呈现的审美能力和设计思维。

（3）增强分析需求并转换为技术实现的问题解决能力。

任务 1.1　搭建 HTML 文档框架

任务描述

本任务将搭建 HTML 文档框架，HTML 文档框架是制作个人信息展示页面的第一个步骤，这将为后续内容填充和样式修饰提供基础结构。本任务分为 3 个子任务。首先，编写核心结构标签：以 <!DOCTYPE html> 声明为 HTML5 文档，再用 <html> 作为根标签，其内部依次嵌套 <head>（存放元信息）和 <body>（承载可见内容），形成完整架构。其次，配置 <meta> 标签：在 <head> 中，用 <meta charset="UTF-8"> 保证字符正常显示，<meta name="viewport" content="width=device-width, initial-scale=1.0"> 实现设备自适应。最后，添加标题与注释：用 <title> 定义页面标题，同时对核心标签添加 <!-- 注释 --> 说明结构，增强代码可读性。

1.1.1　编写 <html>、<head> 和 <body> 核心结构标签

Web 项目的创建过程

1. 目的

（1）理解 <html>、<head> 和 <body> 标签的定义与功能。

（2）掌握使用 <html>、<head> 和 <body> 标签构建完整文档结构的方法。

（3）理解 HTML 文档树状结构与标签层级关系。

（4）养成良好的编码习惯。

2. 实施步骤

本任务通过以下 5 个步骤完成 HTML 文档核心结构标签的编写，最终形成可被浏览器正确解析的基础文档框架。

1）新建 HTML 文件及相关文件夹

（1）打开 HBuilder 软件。选择“文件”→“新建”→“Web 项目”命令，菜单命令界面如图 1-1 所示。

注意：HBuilder 每个版本界面或者菜单会有差别，但功能是类似的，请读者自行分辨。

（2）在弹出的对话框中设置项目名称、保存路径，选择合适的模板。在“项目信息”的“项目名称”输入框中输入“项目 1”；在“位置”选项中单击“浏览 ...”按钮，选择合适的项目保存路径；在“选择模板”选项中选中“默认项目”。以上设置好以后，单击“完成”按钮，完成项目的创建。创建 Web 项目界面如图 1-2 所示。

图 1-1　菜单命令界面

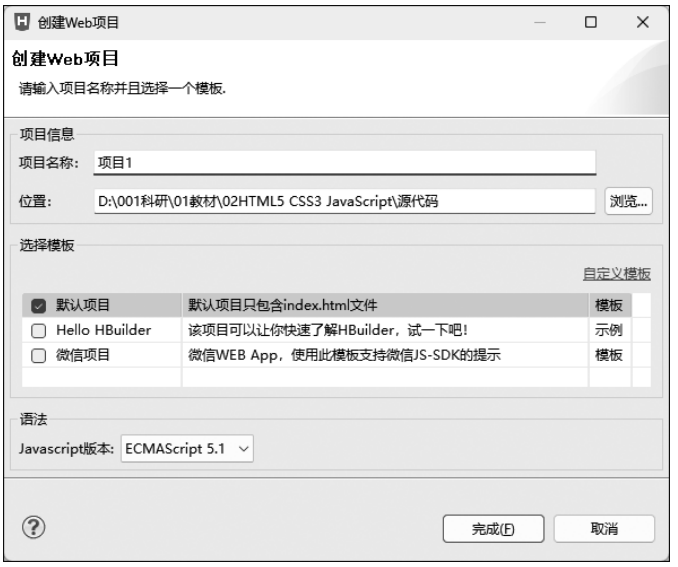

图 1-2　创建 Web 项目界面

注意：HBuilder 会自动生成基础文件结构，分别是 css 文件夹（用于存放 CSS 代码文件）、img 文件夹（用于保存图片文件）、js 文件夹（用于保存 JavaScript 代码文件），并生成一个名称为 index.html 的网页文件，文件结构界面如图 1-3 所示。

图 1-3　文件结构界面

2）编写文档类型声明

在 HBuilder 自动生成的 index.html 文件中，代码第一行默认已包含 HTML5 的文档类型声明：<!DOCTYPE html>。若未显示，需手动输入。该声明不属于 HTML 标签，其作用是告知浏览器当前文档使用 HTML5 标准，确保浏览器以标准模式解析文档，避免因解析模式差异导致页面显示异常。

3）添加 <html> 标签

在文档类型声明下方，HBuilder 通常会自动生成 <html> 标签对（即 <html></html>）。若未生成，需手动编写。<html> 标签是 HTML 文档的根标签，所有其他标签必须嵌套在其内部，作为整个文档结构的“容器”。为增强语义化，可在 <html> 标签中添加 lang 属性，如中文页面设置为 <html lang="zh-CN">，英文页面设置为 <html lang="en">，该属性有助于搜索引擎识别页面语言，同时便于屏幕阅读器等辅助工具正确处理内容。代码界面如图 1-4 所示。

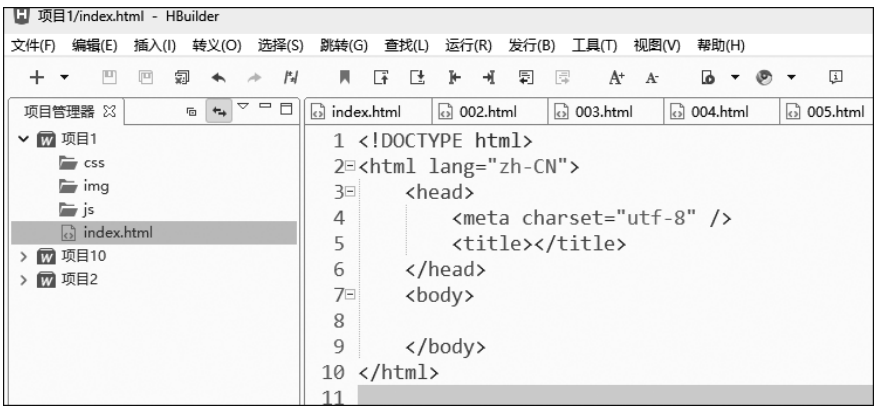

图 1-4　代码界面

4）嵌套 <head> 和 <body> 标签

在 <html> 标签内部，HBuilder 会自动按固定顺序生成 <head> 和 <body> 标签对（即 <head></head> 和 <body></body>）。若未生成，需手动编写，且必须保证顺序正确（<head>

在前，<body> 在后）。

（1）<head> 标签用于存放文档的元数据（如字符集、视口设置等）、页面标题、样式表链接等不直接显示在页面中的内容。

（2）<body> 标签用于存放页面中所有可见内容，包括文本、图片、按钮等元素，是用户在浏览器中实际能看到的部分。

5）检查结构完整性

完成上述步骤后，使用 HBuilder 的代码校验功能检查标签结构完整性。

确认所有标签成对出现（每个开始标签都有对应的结束标签），HBuilder 中未闭合的标签会以红色高亮提示，需补全。

检查嵌套关系是否正确（<head> 和 <body> 必须是 <html> 的直接子标签，且无交叉嵌套）。

通过 HBuilder 的“格式化文档”功能自动调整代码缩进，使结构层级更清晰，便于后续维护。

以下是完成后的基础结构代码，在代码中加入了 HTML 的注释，以方便读者更好地理解代码。代码界面如图 1-5 所示。

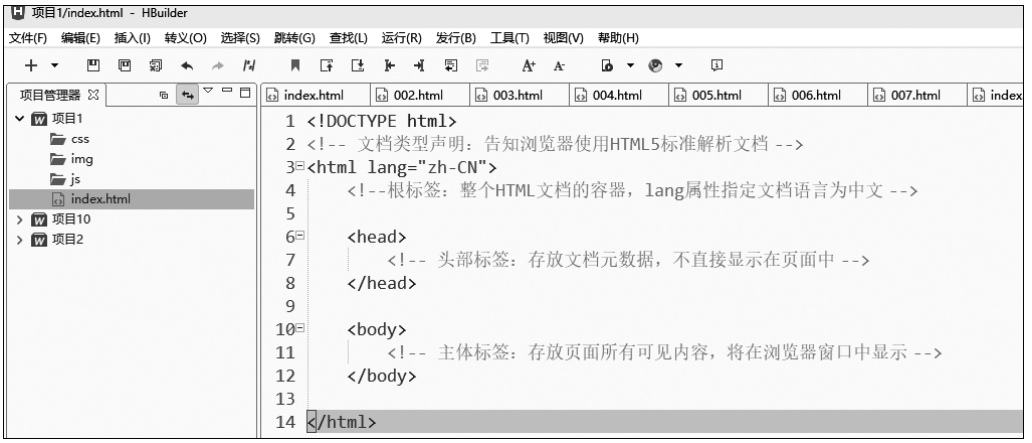

图 1-5　代码界面

3. 背景知识

1）HTML

HTML（hypertext markup language，超文本标记语言）用于创建网页的标准标记语言，其文档的基本结构遵循“容器—内容”的逻辑，由一系列标签组成。这些标签如同建筑的框架，决定了网页的基础形态。其中，<html>、<head> 和 <body> 是构成 HTML 文档的三大核心标签，它们共同形成了网页的“骨架”，所有其他内容（如文本、图片、样式等）都必须嵌套在这个骨架中才能被浏览器正确解析和显示。

2）<!DOCTYPE html> 声明

<!DOCTYPE html> 是 HTML5 的文档类型声明，位于 HTML 文档的第一行，且必须单独成行。它的主要作用是：告知浏览器当前文档使用的 HTML 版本规范，使浏览器能够按照对应的标准解析文档中的标签和属性。

3）<html> 标签的语义与属性

<html> 标签是 HTML 文档的根标签，所有其他标签都必须包含在 <html> 标签内部，形成“父子”嵌套关系。其基本语法为 <html></html>，其中开始标签 <html> 可以添加 lang 属性，用于指定文档的主要语言。lang 属性虽然不影响页面的视觉显示，但对搜索引擎优化（search engine optimization，SEO）和无障碍访问（如屏幕阅读器）具有重要意义。搜索引擎可根据 lang 属性判断文档语言，提高搜索结果的相关性；屏幕阅读器可根据语言设置调整发音规则，提升用户体验。

4）<head> 标签的功能与包含元素

<head> 标签位于 <html> 标签内部，是文档的“头部”，用于存放描述文档的元数据（meta data），这些数据不会直接显示在浏览器的页面内容区域，但会影响文档的解析、表现和行为。<head> 标签中通常包含以下类型的内容。

（1）文档标题（<title> 标签）：定义浏览器标签页显示的标题，也是搜索引擎结果中显示的主要标题。

（2）字符集设置（<meta> 标签的 charset 属性）：指定文档的字符编码方式，如 UTF-8，确保文档中的文字（尤其是中文、日文等非英文字符）能被正确显示。

（3）视口设置（<meta> 标签的 viewport 属性）：用于控制移动设备上页面的显示方式，是响应式网页设计的基础。

（4）样式信息（<style> 标签或 <link> 标签引用外部 CSS 文件）：定义页面的样式规则。

（5）脚本信息（<script> 标签）：引入 JavaScript 代码，实现页面的交互功能。

需要注意的是，<head> 标签中的内容主要面向浏览器和搜索引擎，而非直接呈现给用户，因此其内容的正确性直接影响文档的解析效率和功能性。

5）<body> 标签的作用与内容类型

<body> 标签同样位于 <html> 标签内部，且必须跟在 <head> 标签之后，是文档的“主体”，用于存放所有需要在浏览器页面中直接显示的内容。内容如下。

（1）文本内容：如段落（<p> 标签）、标题（<h1>~<h6> 标签）等。

（2）多媒体内容：如图片（<img> 标签）、音频（<audio> 标签）、视频（<video> 标签）等。

（3）交互元素：如按钮（<button> 标签）、链接（<a> 标签）、表单（<form> 标签）等。

（4）其他结构化标签：如 、 等，用于对内容进行分组和布局。

浏览器会将 <body> 标签中的内容按照标签的语义和样式规则渲染后，显示在页面的主区域，因此 <body> 标签是用户直接感知的内容载体，其结构和内容的合理性直接影响用户体验。

6）文档树结构与嵌套规则

HTML 文档的标签嵌套关系形成了一棵“文档树”（document tree），其中 <html> 标签是根节点，<head> 和 <body> 是 <html> 的直接子节点，而 <head> 和 <body> 内部的标签则是它们的子节点，以此类推。这种树状结构遵循严格的嵌套规则。

（1）标签必须正确闭合：每个开始标签（如 <html>）都必须有对应的结束标签（如 </html>），空标签（如 <meta>、<img>）除外。

（2）标签不能交叉嵌套：如 <head><body></head></body> 是错误的，正确的应为 <head></head><body></body>。

（3）层级关系明确：子标签必须完全包含在父标签内部，形成清晰的“父子”“兄弟”关系。

遵循这些规则是保证 HTML 文档结构正确的基础，也是后续学习 CSS 布局和 JavaScript DOM（document object model，文档对象模型）操作的前提。DOM 正是以文档树为基础，将标签转换为可操作的对象。

7）语义化结构的重要性

<html>、<head> 和 <body> 的划分体现了 HTML 的语义化思想——标签的名称应反映其包含内容的含义和功能。语义化结构的优势在于以下方面。

（1）提高代码的可读性和可维护性：开发者能通过标签名称快速理解文档结构和内容用途，便于后续修改和扩展。

（2）有利于搜索引擎优化：搜索引擎的爬虫依赖标签的语义来分析文档内容的重要性和关联性，语义化的结构能使页面在搜索结果中获得更好的排名。

（3）提升无障碍访问性：屏幕阅读器等辅助工具可通过标签的语义解析内容，帮助残障用户更好地理解页面信息。

因此，正确编写 <html>、<head> 和 <body> 核心结构标签，不仅是完成页面搭建的基础操作，更是培养良好开发习惯、理解 Web 标准的重要起点。

1.1.2 使用 <meta> 标签设置字符集（charset）与视口（viewport）信息

1. 目的

（1）掌握使用 <meta> 标签设置 UTF-8 字符集的方法。

（2）理解视口的概念及其对移动设备显示的影响。

（3）能独立编写格式规范的字符集与视口设置代码。

2. 实施步骤

（1）新建 HTML 文件。启动 HBuilder，打开已创建的“项目 1”文件夹。右击“项目 1”，选择“新建”→“HTML 文件”命令，将新建文件命名为 1.1.2.html。一般来说，在 HBuilder 中新建 HTML 文件的代码如下：

```
<!DOCTYPE html>
<html>
  <head>
    <!-- 此处将添加 <meta> 标签 -->
    <title> 个人信息展示页 </title>
  </head>

  <body>
    页面内容将在后续步骤添加
  </body>
</html>
```

（2）配置字符集。在 <head> 标签内添加字符集 <meta> 标签，代码如下：

```
<meta charset="UTF-8">
```

<meta> 标签的 charset 属性指定字符编码方式。charset 属性的值 UTF-8 是通用编码，支持全球所有语言字符。

<meta charset="UTF-8"> 标签的写法是 HTML5 标准写法，等同于旧版的 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"> 写法。

（3）配置视口信息。在字符集 <meta> 标签下方添加视口配置，代码如下：

```
<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
```

该标签参数的作用描述和取值示例如表 1-1 所示。

表 1-1　<meta> 标签参数的作用描述和取值示例

参 数 名	作 用 描 述	取 值 示 例
width	设置视口宽度	device-width（设备宽度）
initial-scale	初始缩放比例	1.0（不缩放）
maximum-scale	允许的最大缩放比例	1.0（禁止放大）
user-scalable	是否允许用户手动缩放	no（禁用缩放）

设置好 <meta> 标签的页面完整代码示例如图 1-6 所示。

index.html | 002.html | 003.html | 004.html | 005.html | 006.html | 007.html | index.html | 项目2 | *index.html | 002.html | 项目1-001

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="UTF-8">
5          <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
6          <title>个人信息展示页</title>
7      </head>
8      <body>
9          <h1>我的个人介绍</h1>
10         <p>这里是个人信息内容...</p>
11     </body>
12 </html>
13
```

图 1-6　<meta> 标签的页面完整代码示例

（4）代码保存与预览。在 HBuilder 中选择“文件”→“保存”命令，以将编写的代码进行保存（快捷键 Ctrl+S）；然后在浏览器中运行该页面，在 HBuilder 中选择“运行”→“浏览器运行”→ Chrome 命令，此时的菜单命令界面如图 1-7 所示；在 Chrome 浏览器中查看网页运行效果，如图 1-8 所示。

图 1-7　菜单命令界面

图 1-8　运行效果

3. 背景知识

1）<meta> 标签

<meta> 标签是 HTML 元数据标签，提供关于 HTML 文档的信息；该标签不直接显示在页面中，但影响页面行为；<meta> 标签必须位于 <head> 标签内。

<meta> 标签常用于设置字符编码、视口、页面描述（用于 SEO）、关键词、自动刷新。

2）HTML 文档常用的字符集

HTML 文档常用的字符集如表 1-2 所示。

表 1-2　HTML 文档常用的字符集

字符集	支 持 语 言	字节数	应 用 场 景
ASCII	英文及符号	1	早期英文网站
GB2312	简体中文	1~2	国内早期网站
GBK	简体及繁体中文	1~2	中文 Windows 系统
UTF-8	全球所有语言	1~4	现代网站（推荐使用）

浏览器使用的字符集与网页编码不一致时，会显示乱码。例如，UTF-8 编码的页面被浏览器解析为 GBK 时，中文会显示为“”。

3）网页的视口（viewport）的分类

（1）布局视口（layout viewport）：网页布局的基准宽度，默认约 980px。

（2）视觉视口（visual viewport）：用户实际看到的区域，可通过缩放改变大小。

（3）理想视口（ideal viewport）：设备的物理屏幕宽度。

4）视口的配置参数

（1）width=device-width：将布局视口设置为设备宽度。

（2）initial-scale=1.0：初始不缩放，按 1∶1 显示。

（3）minimum-scale/maximum-scale：限制缩放范围。

（4）user-scalable=no：禁用用户手动缩放功能。

5）不同显示效果对比

不同显示效果对比如表 1-3 所示。

表 1-3　不同显示效果对比

配 置 代 码	显 示 效 果
无视口配置	移动端将按照桌面版页面尺寸等比例缩小，显示界面需横向滚动
width=device-width	页面内容适应设备屏幕宽度，文字可能过小
width=device-width, initial=1.0	内容正确显示，无须缩放和滚动