

HTML and CSS

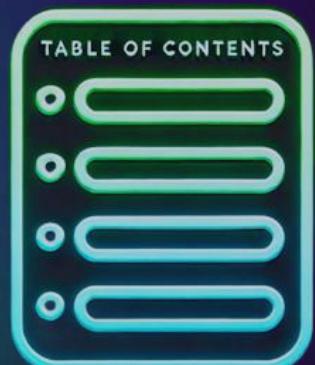
HTML, Structure, Tags, CSS, Styling,
Layout Systems, Tables, Forms, JS



Svetlin Nakov, PhD
Co-founder @ SoftUni

Agenda

1. **Intro to HTML**: basic tags, page structure
2. **More HTML tags**: hyperlinks, lists, images, emojis
3. **Intro to CSS**: inline, internal, external CSS, styling page elements
4. **Browser Dev Tools**: inspecting page content in the Web browser
5. **CSS selectors and basic styling**: styling by class, id, nested styling
6. **CSS box model, inline and block elements**
7. **CSS layouts**: flexbox, grid and responsive layouts
8. **HTML tables**: tables, rows, columns, merged cells
9. **HTML forms**: form, action, data controls, submit
10. **Integrating JavaScript in HTML, DOM Interaction**



Sli.do Code

#Soft-Tech-AI

Join at

slido.com

#Soft-Tech-AI



Breaks

20:00 / 21:00



Welcome to HTML

HTML Page Structure, Simple Tags

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>Page name</title>
    <meta name="description" content="content">
    <meta name="keywords" content="list of keywords">
  </head>
  <body>
    <div class="feature-item">
      <header class="navbar-header">
        <div class="container">
```

HTML and CSS

- **HTML** and **CSS** are languages for creating Web sites
 - Display text, images, tables, forms, etc. in the Web browser
 - **HTML** – used to describe Web content, e. g. Web page
 - **CSS** – styling and formatting rules for HTML code

```
<h1>Languages</h1>
<ul>
  <li>JS</li>
  <li>Python</li>
  <li>PHP</li>
</ul>
```

```
h1 { color: #2E4AA7 }
li {
  display: inline;
  padding: 5px 10px;
  background: #CCC;
}
```

Languages

JS Python PHP

What is HTML?

- **HTML** – simple language for describing Web content
 - Used to create Web pages, Web sites, Web apps
 - Displayed by Web browsers
 - Used together with **CSS** for styling
- HTML uses simple **text** with **tags**:

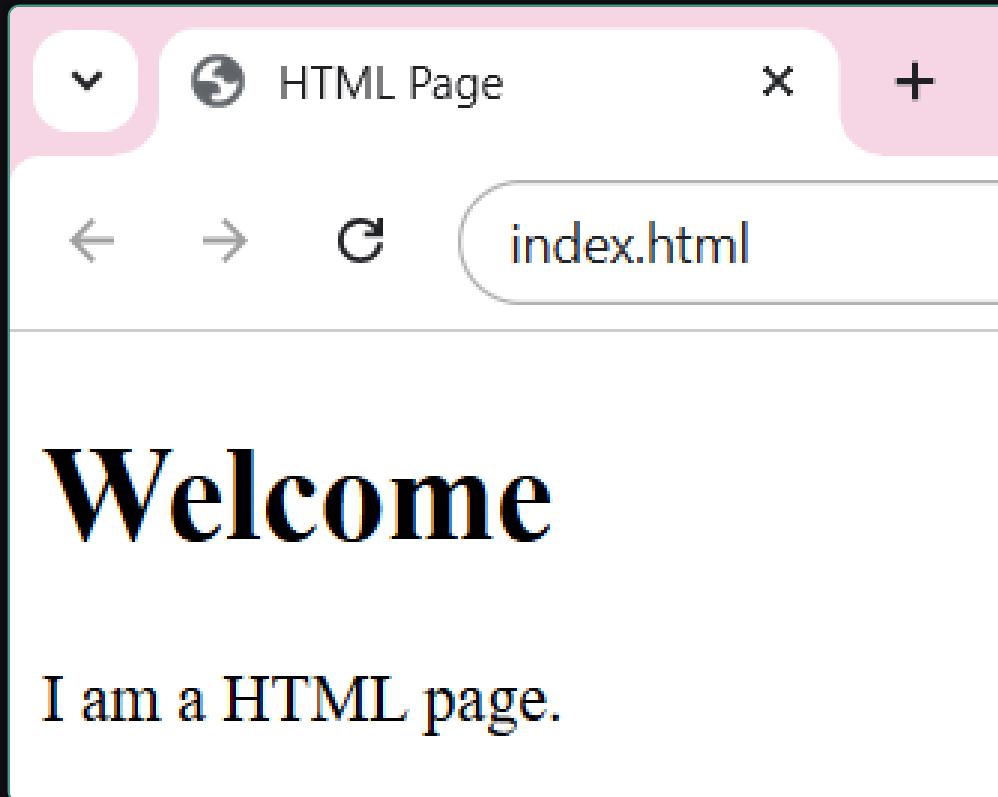
```
<h1>I am a Title</h1>
```

```
<p>I am a paragraph</p>
```



HTML Structure

- HTML page structure:
<html>, <head>, <body>



index.html

<html>

<head>

<title>HTML Page</title>

</head>

<body>

<h1>Welcome</h1>

<p>I am a HTML page.</p>

</body>

</html>

Headings and Paragraphs

- Headings: <h1> ... <h6>

```
<h1>Creating Web Sites</h1>
```

```
<h2>Intro to HTML</h2>
```

```
<h2>Intro to CSS</h2>
```

Creating Web Sites

Intro to HTML

Intro to CSS

- Paragraphs: <p> ... </p>

```
<p>HTML describes Web content ...</p>
```

```
<p>CSS styles Web content ...</p>
```

HTML describes Web content ...

CSS styles Web content ...

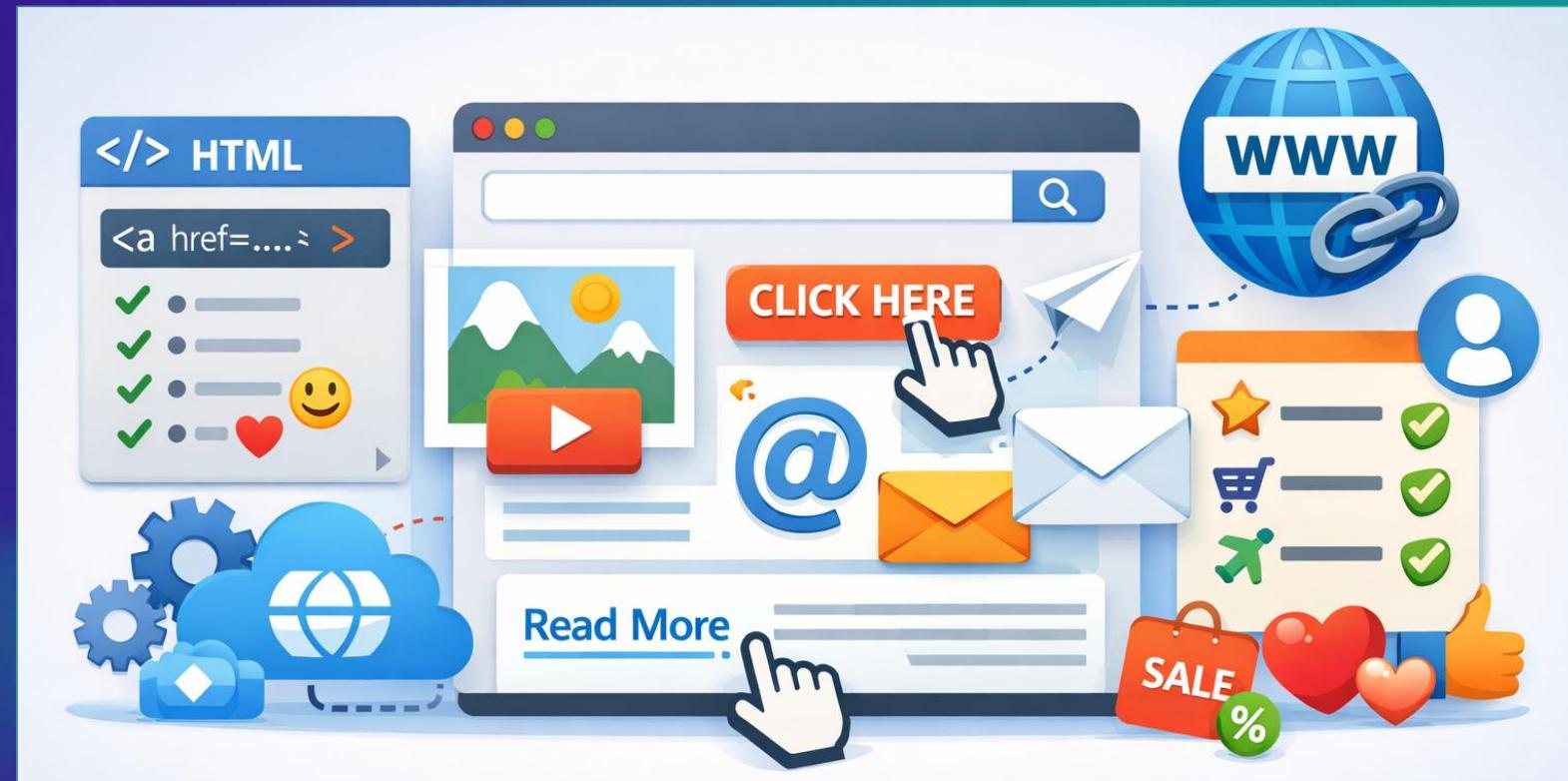


HTML: Live Demo

Creating a Simple HTML
Page: Head, Body, H1, P

More HTML Tags

Hyperlinks, Lists, Images, Emojis



Hyperlinks

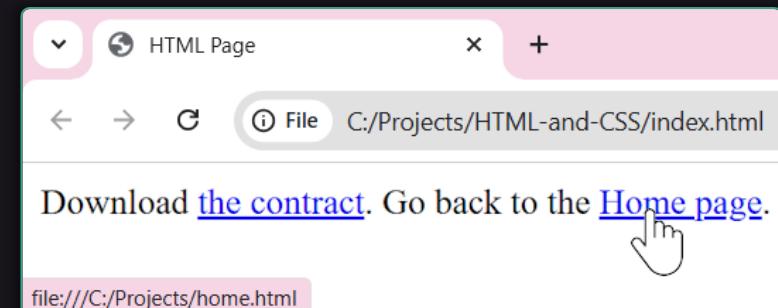
- Hyperlinks: `link text<a>`
- External hyperlink:

Browse the modern AI educational programs at the [SoftUni](#) web site.

Browse the modern AI educational programs at the
`SoftUni` web site.

- Relative hyperlink (path + file):

Download `the contract`.



Go back to the `Home page`.

Lists, Bold, Italic, Line Break

- Bullet lists: ...
 - List items: ...
- Ordered lists: ...
- Bold, italic, new line:

```
This is <b>important</b>.  
<br /> <!-- line break --&gt;<br/>This is <i>special</i>.
```

This is **important**.
This is *special*.

List of bullets:

```
<ul>  
  <li>First</li>  
  <li>Second</li>  
  <li>Third</li>  
</ul>
```

Numbered list:

```
<ol>  
  <li>One</li>  
  <li>Two</li>  
  <li>Three</li>  
</ol>
```

List of bullets:

- First
- Second
- Third

Numbered list:

1. One
2. Two
3. Three

Exercise: Very Simple CV

- Create a HTML page, holding a **very simple CV**:

Create a **very simple CV**:

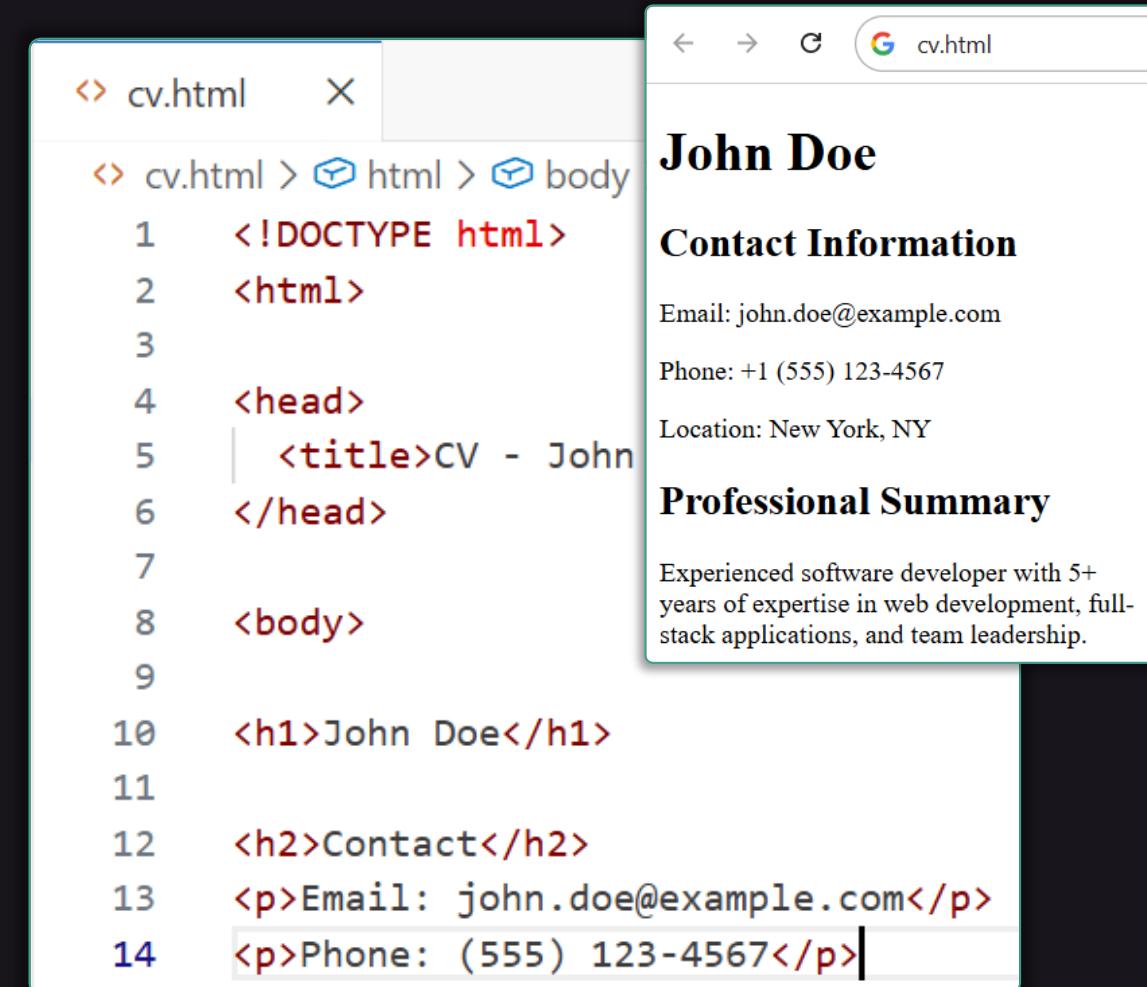
- HTML only: `cv.html`
- No CSS styles
- Only use headings, paragraphs, links, lists

Add Context...

Create a very simple CV:

- HTML only: `cv.html`
- No CSS styles
- Only use headings, paragraphs, links, lists

Agent ▾ Claude Haiku 4.5 ▾ ⚙ ➔ ➕



```
cv.html
cv.html > html > body
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5    <title>CV - John Doe</title>
6  </head>
7
8  <body>
9
10 <h1>John Doe</h1>
11
12 <h2>Contact</h2>
13 <p>Email: john.doe@example.com</p>
14 <p>Phone: (555) 123-4567</p>
```

John Doe

Contact Information

Email: john.doe@example.com
Phone: +1 (555) 123-4567
Location: New York, NY

Professional Summary

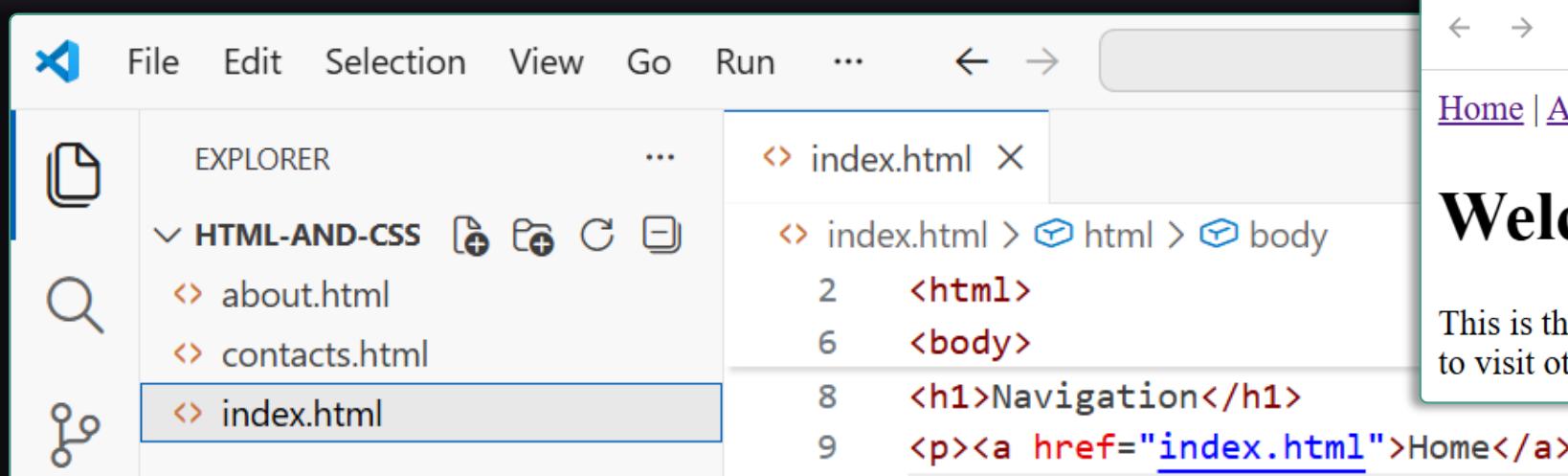
Experienced software developer with 5+ years of expertise in web development, full-stack applications, and team leadership.

Exercise: Multi-Page Site

- Create a **multi-page Web site** with navigation:

Create a **3-page Web site**: `index.html`, `about.html`, `contacts.html`

- Top navigation menu: Home | About | Contacts
- Hyperlinks between the pages, simple HTML, no CSS



The code editor shows three files in the Explorer panel: `index.html`, `about.html`, and `contacts.html`. The `index.html` file is currently selected and open in the main editor area.

```
<html>
<body>
<h1>Navigation</h1>
<p><a href="#">Home</a></p>
```

The browser window shows the rendered content of `index.html`:

[Home](#) | [About](#) | [Contacts](#)

Welcome to Our Website

This is the home page. Use the navigation menu above to visit other pages.

Exercise: Task List

- Create a simple HTML page, holding a **task list**

Create a minimalistic HTML page `tasks.html`
- No CSS, single HTML file
- Display a heading + list of tasks

Add Context...

Create a minimalistic HTML page `tasks.html`
- No CSS, single HTML file
- Display a heading + list of tasks

Agent ▾ Claude Haiku 4.5 ▾  

```
tasks.html
tasks.html > html > head
2   <html>
6   <body>
8     <h1>My Tasks</h1>
9
10    <ul>
11      <li>Complete project documentation</li>
12      <li>Review code changes</li>
13      <li>Update database schema</li>
14      <li>Test new features</li>
15      <li>Deploy to production</li>
16    </ul>
17
18  </body>
19  </html>
20
```

My Tasks

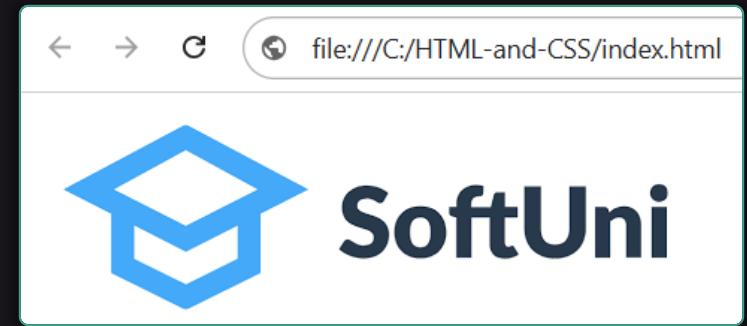
- Complete project documentation
- Review code changes
- Update database schema
- Test new features
- Deploy to production

Images

- Images: ``

```

```



- Image from external URL with alt text:

```
<img src=
"https://www.wikipedia.org/portal/wikipedia.org/
assets/img/Wikipedia-logo-v2@1.5x.png"
alt="Wikipedia logo" />
```



- Image from `data-url` (inline image):

```

```

Emojis

- **Emojis** are special Unicode characters → look like icons

<h2>Smileys & Emotions</h2>

```
<p> 😊 😋 😃 😄 😆 😁 😃 😍 😊 </p>
```

<h2>People & Body</h2>

```
<p> 🤝 🤗 🙌 🤘 🤝 🤞 🤝 🤝 </p>
```

<h2>Animals & Nature</h2>

```
<p> 😺 😱 😱 😱 😱 😱 😱 😱 </p>
```

Smileys & Emotions



People & Body



Animals & Nature



- In HTML **special characters** are encoded as **HTML entities**:

This is a <tag>
& it's encoded.

This is a <tag> & it's encoded.

Exercise: Party Invitation

- Create a **party invitation** HTML page (with poster and emojis)

Create a **party invitation**: `party.html`

- Insert an image `party-poster.jpg`
- Use emojis in the invitation text
- Add minimal styling to make it look well

```

<html lang="en">
<body>

  <div class="container">
    <h1>🎉 You're Invited! 🎉</h1>

    <h2>Join us for an amazing celebration! 🎉</h2>

    <div class="details">
      <p>📅 <strong>Date:</strong> Saturday, December 16th</p>
      <p>⌚ <strong>Time:</strong> 7:00 PM - 11:00 PM</p>
      <p>📍 <strong>Location:</strong> The Grand Ballroom</p>
    </div>

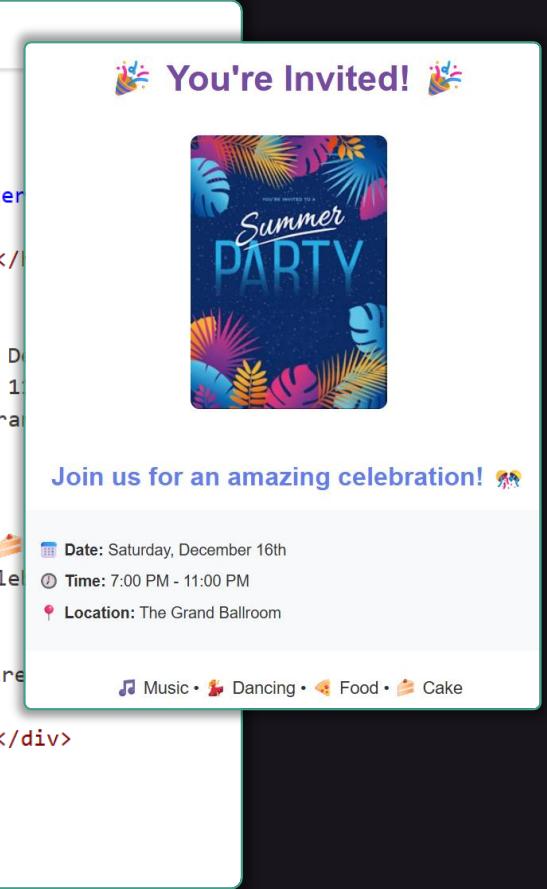
    <div class="highlights">
      <p>🎵 Music • 💃 Dancing • 🍔 Food • 🍰 Cake</p>
      <p>🎈 Balloons • 🎁 Surprises • 🎉 Celebration</p>
    </div>

    <p><strong>Dress code:</strong> Formal Attire</p>

    <div class="rsvp">➡ RSVP by December 10th</div>
  </div>

</body>
</html>

```



Intro to CSS

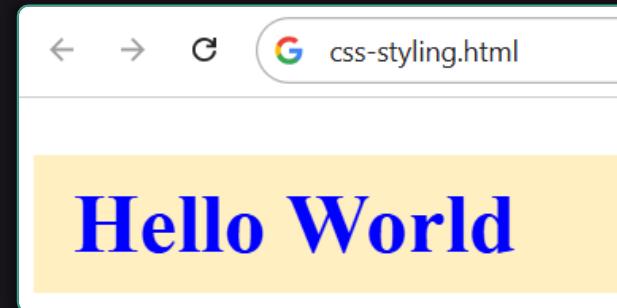
Inline, Internal, External CSS, Styling Page Elements

```
34  /* A reference to a type */
35  span.ts span.type-ref {
36      color: □rgb(175, 0, 219) !important;
37  }
38
39  /* Signature details */
40  div.signature > table {
41      border-collapse: collapse;
42      border: thin □darkgray solid;
43      width: 60%;
44  }
```

What is CSS?

- CSS (Cascading Style Sheets) defines the **styling** of HTML elements (fonts, colors, spacing, layout, positioning, etc.)
- CSS styling **rules**: `selector { property: value; }`
- CSS styling **example**:

```
h1 {  
    font-size: 32pt;  
    color: blue;  
    background: #ffefc1;  
    padding: 10px 20px;  
}
```

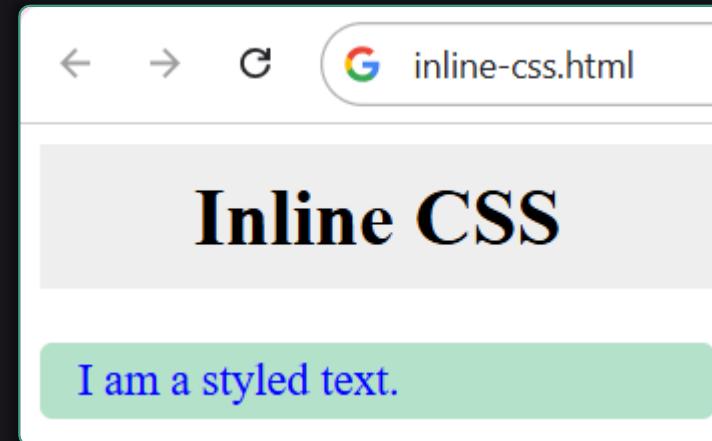


- Selector: **h1**
- Property: **padding**
- Value: **10px 20px**

Inline CSS

- **Inline CSS styles** are directly hardcoded in HTML elements

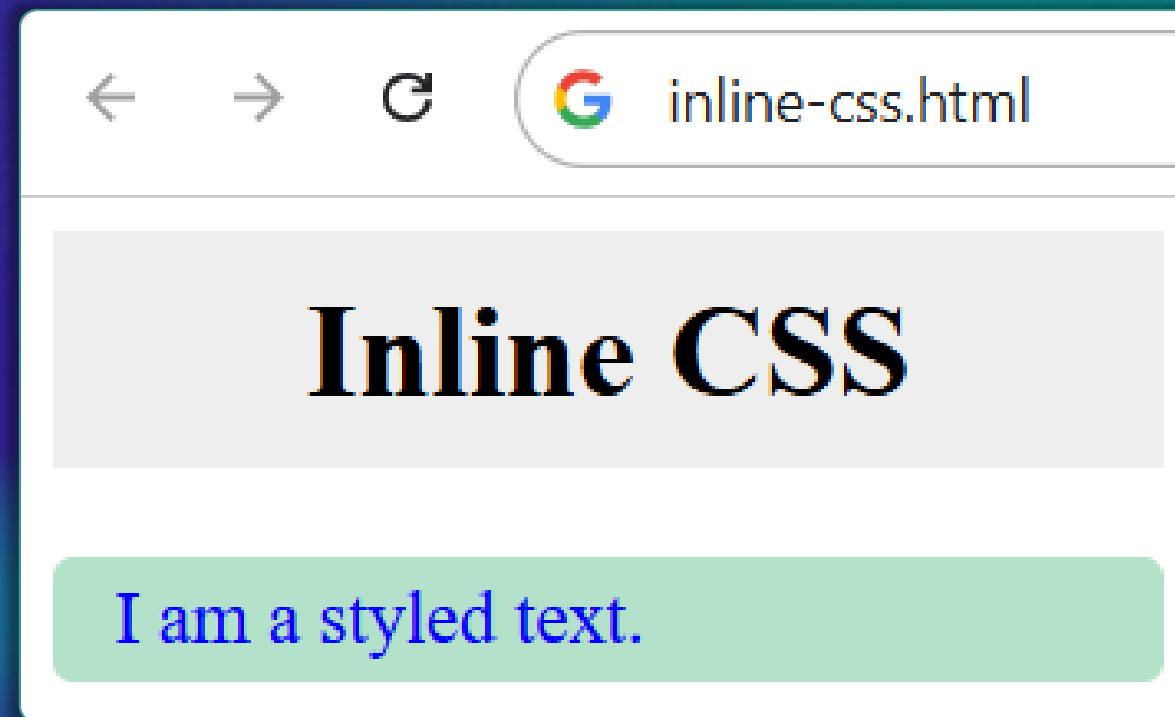
```
<h1 style="background:#EEE;  
padding:10px; text-align:  
center;">Inline CSS</h1>  
  
<p style="font-size: 24px;  
color: blue;  
background: #b3e1c9;  
padding: 5px 15px;  
border-radius: 5px;">  
I am a styled text.  
</p>
```



- Not recommended!
- Hard to maintain
- Cannot reuse repeating styles

Inline CSS

Live Demo

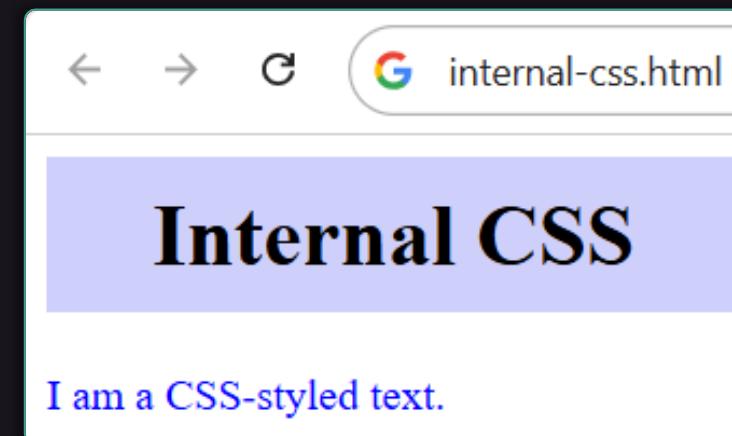


Internal CSS

- Internal CSS is CSS stylesheet inside an HTML page:

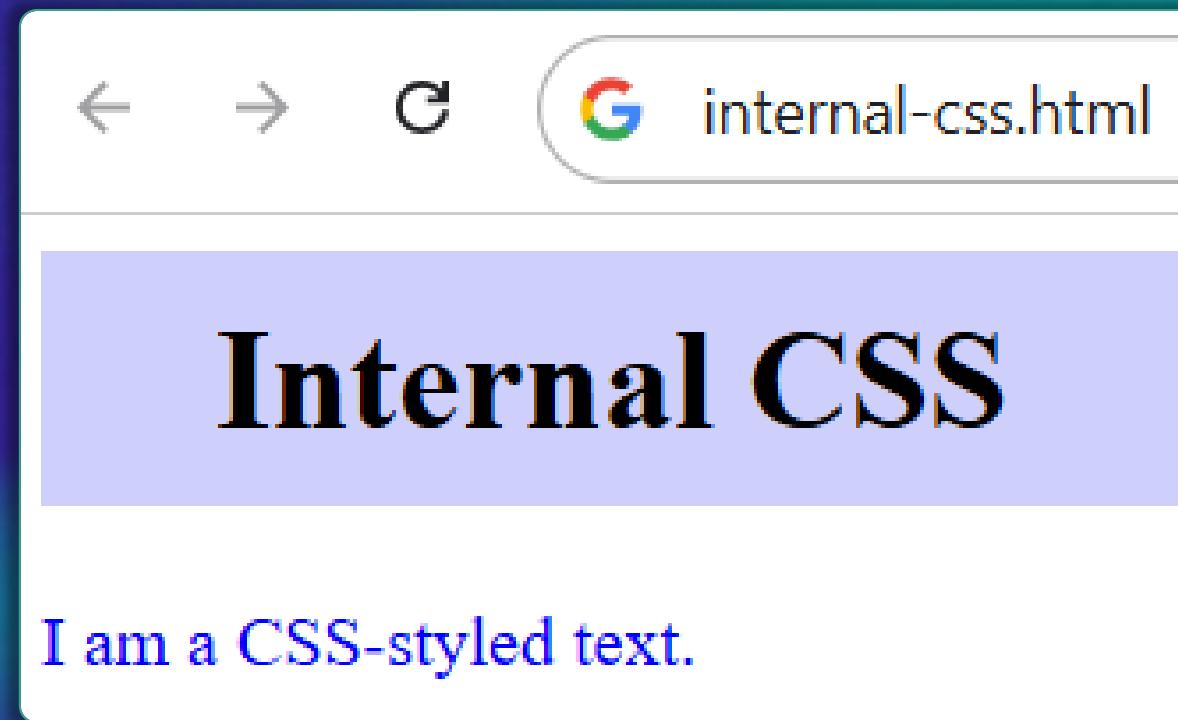
```
<html>
<head>
  <style>
    h1 {
      background: #ced0fb;
      padding: 10px;
      text-align: center;
    }
    p { color: blue; }
  </style>
</head>
```

```
<body>
  <h1>Internal CSS</h1>
  <p>I am a CSS-styled text.</p>
</body>
</html>
```



Internal CSS

Live Demo



External CSS

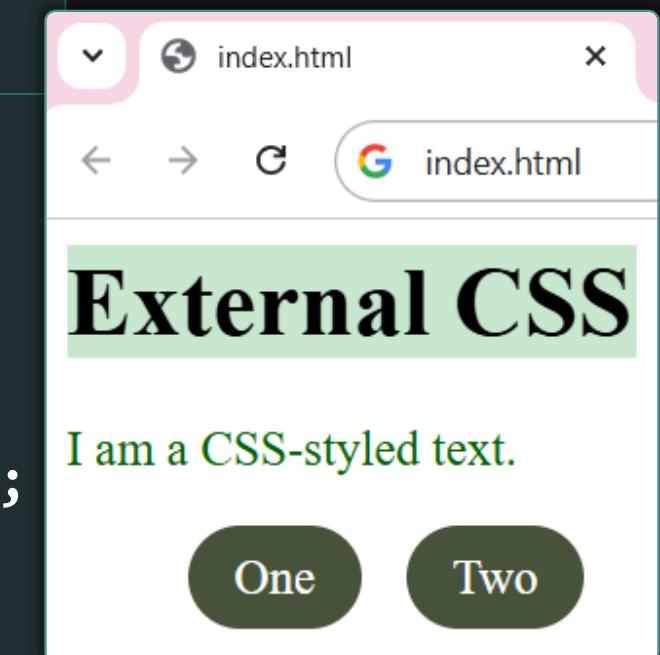
- **External CSS** is separate CSS file, linked to the HTML

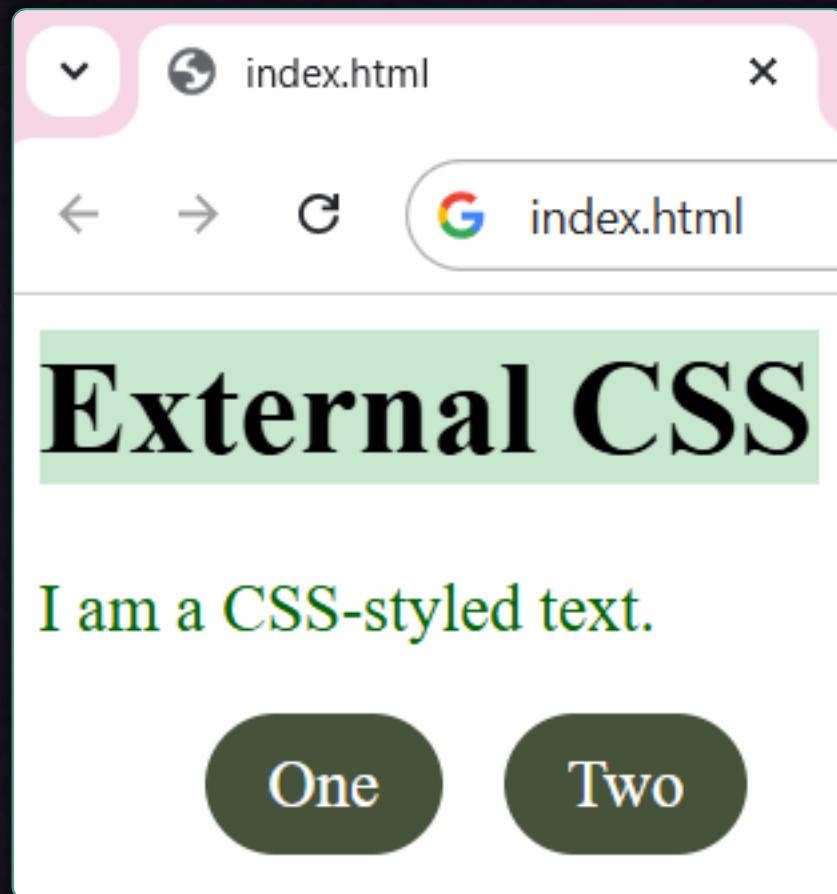
index.html

```
<html>
<head>
  <link rel="stylesheet"
        href="styles.css">
</head>
<body>
  <h1>External CSS</h1>
  <p>CSS-styled text</p>
  <ul><li>1</li><li>2</li>
    <li>3</li></ul>
</body>
</html>
```

styles.css

```
h1 {
  background: #c7e8cf;
}
p { color: darkgreen; }
li {
  display: inline-block;
  color: white;
  background: #46523a;
  margin-right: 15px;
  padding: 8px 15px;
  border-radius: 20px;
}
```





External CSS

Live Demo

Exercise: Styled CV

- Create a HTML page, holding a **simple CV**, with styling:

Create a **very simple CV**:

```
`styled-cv.html`
```

- Only use **headings, paragraphs, links, lists**
- Apply **simple CSS styling**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>John Doe - CV</title>
  <style>
    * {
      margin: 0;
      padding: 0;
      box-sizing: border-box;
    }

    body {
      font-family: Arial, sans-serif;
      line-height: 1.6;
      color: #333;
      background-color: #f4f4f4;
    }
  </style>
</head>
<body>
```

```
70   <body>
71     <div class="container">
72       <h1>John Doe</h1>
73       <p class="contact-info">Email: <a href="mailto:john@example.com">john@example.com</a> | Phone: +1 (555) 123-4567 | LinkedIn: <a href="https://www.linkedin.com/in/johndoe">Profile</a></p>
74
75       <h2>Professional Summary</h2>
76       <p>Experienced software developer with a passion for creating clean and efficient code. Proficient in multiple programming languages and frameworks with a strong background in full-stack development.</p>
77
78       <h2>Experience</h2>
79       <ul>
80         <li><strong>Senior Developer</strong> - Tech Company (2021 - Present)</li>
81         <li><strong>Junior Developer</strong> - Web Solutions Inc. (2019 - 2021)</li>
82         <li><strong>Intern</strong> - Digital Startup (2018 - 2019)</li>
83       </ul>
84
85       <h2>Skills</h2>
86       <ul>
87         <li>HTML, CSS, JavaScript</li>
88         <li>React, Vue.js</li>
89         <li>Python, Java</li>
90         <li>Database Design & SQL</li>
91         <li>Git & Version Control</li>
92       </ul>
</div>
</body>
```

John Doe

Email: john@example.com | Phone: +1 (555) 123-4567 | LinkedIn: [Profile](https://www.linkedin.com/in/johndoe)

Professional Summary

Experienced software developer with a passion for creating clean and efficient code. Proficient in multiple programming languages and frameworks with a strong background in full-stack development.

Experience

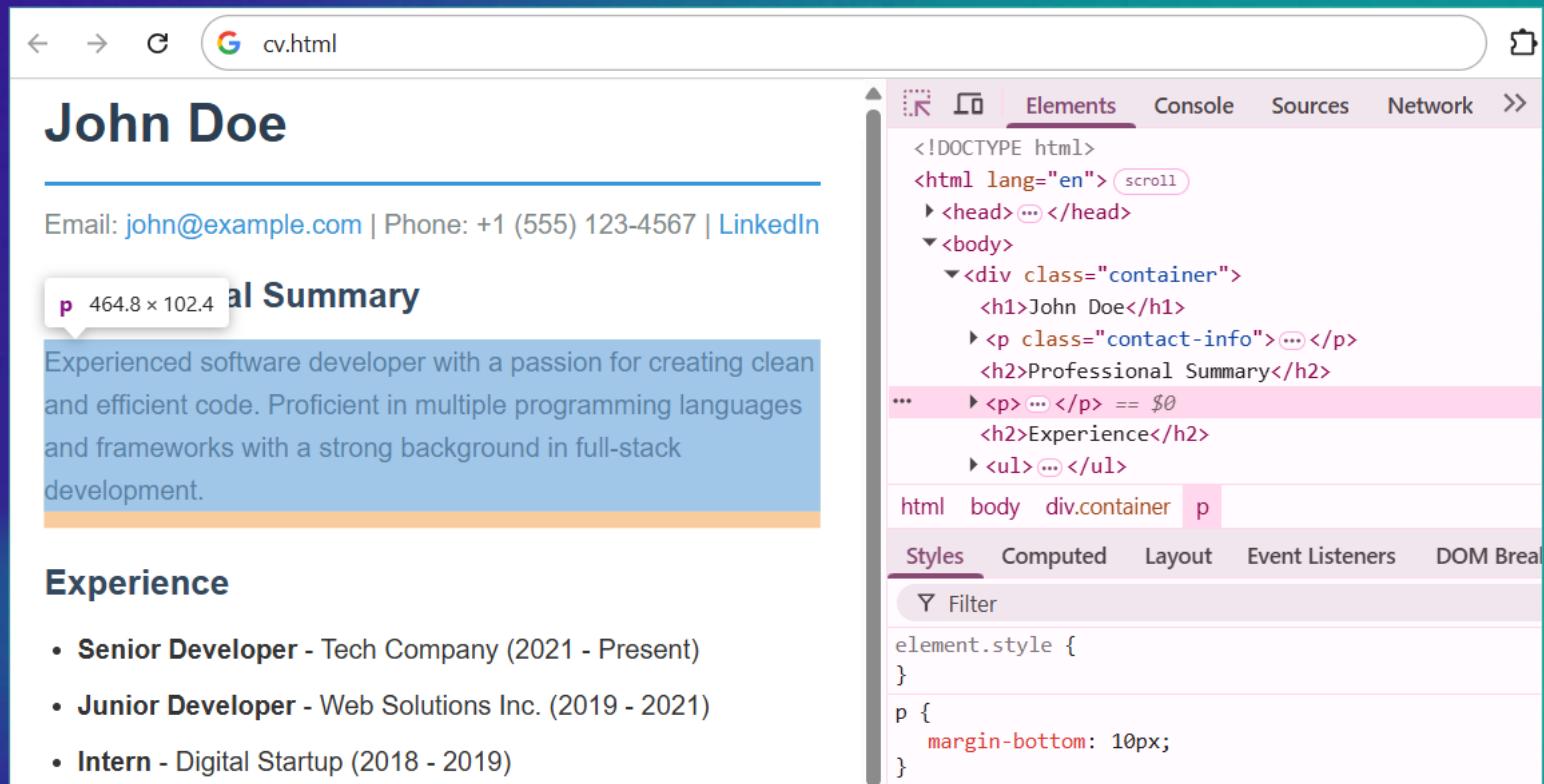
- Senior Developer - Tech Company (2021 - Present)
- Junior Developer - Web Solutions Inc. (2019 - 2021)
- Intern - Digital Startup (2018 - 2019)

Skills

- HTML, CSS, JavaScript
- React, Vue.js
- Python, Java
- Database Design & SQL
- Git & Version Control

Browser Dev Tools

Inspecting Page Content in the Web Browser



Browser Dev Tools

- **Browser Dev Tools** == built-in Web browser tools for **inspecting** the page content (HTML, CSS, JavaScript)
 - **View / inspect / edit** page structure, **debug** and **experiment** with the currently loaded document, styles, scripts
- Open the **Dev Tools** sidebar in Windows:
 - Press **[F12]** key / right click at the page → **[Inspect]**
 - **[F12]** works on **Chrome / Firefox**
- On Mac, use **[Cmd ⌘ + Alt ⌥ + I]**
 - **Safari:** View → Developer → Developer Tools

Inspecting Page and Styles



Page DOM inspector

```
Elements Console Sources

▼<div class="container">
  <h1>John Doe</h1>
  ▶<p class="contact-info">...</p>
  <h2>Professional Summary</h2>
  ▶<p>...</p>
  <h2>Experience</h2>
...
  ▶<ul> == $0
    ▶<li>...</li>
    ▶<li>...</li>
    ▶<li>...</li>
  </ul>
  <h2>Skills</h2>
  ▶<ul>...</ul>
  <h2>Education</h2>
```

CSS styles inspector

```
html body div.container ul
Styles Computed Layout

Y Filter

element.style {
}

ul {
    background: skyblue;
    margin-left: skyblue
    margin-bottom: deepskyblue
}
* {
    margin: ▶ 0;
    padding: ▶ 0;
    box-sizing: border-box;
}
```

Element size inspector

A detailed box model diagram for an `h2` element, showing its layout structure. The outermost box has a dashed border and is labeled with a width of 464.800 and a height of 33.275. Inside, there is a solid black box representing the content area, which is 464.800 wide and 33.275 high. This is surrounded by a padding box with a width of 0 and a height of 0. The padding box is further surrounded by a border box with a width of 0 and a height of 0. The entire structure is enclosed within a margin box with a width of 20 and a height of 10.

margin	20				
border	0				
padding	0				
0	0	464.800×33.275	0	0	0
0	0	0	0	0	0
10					

Browser Dev Tools

Live Demo

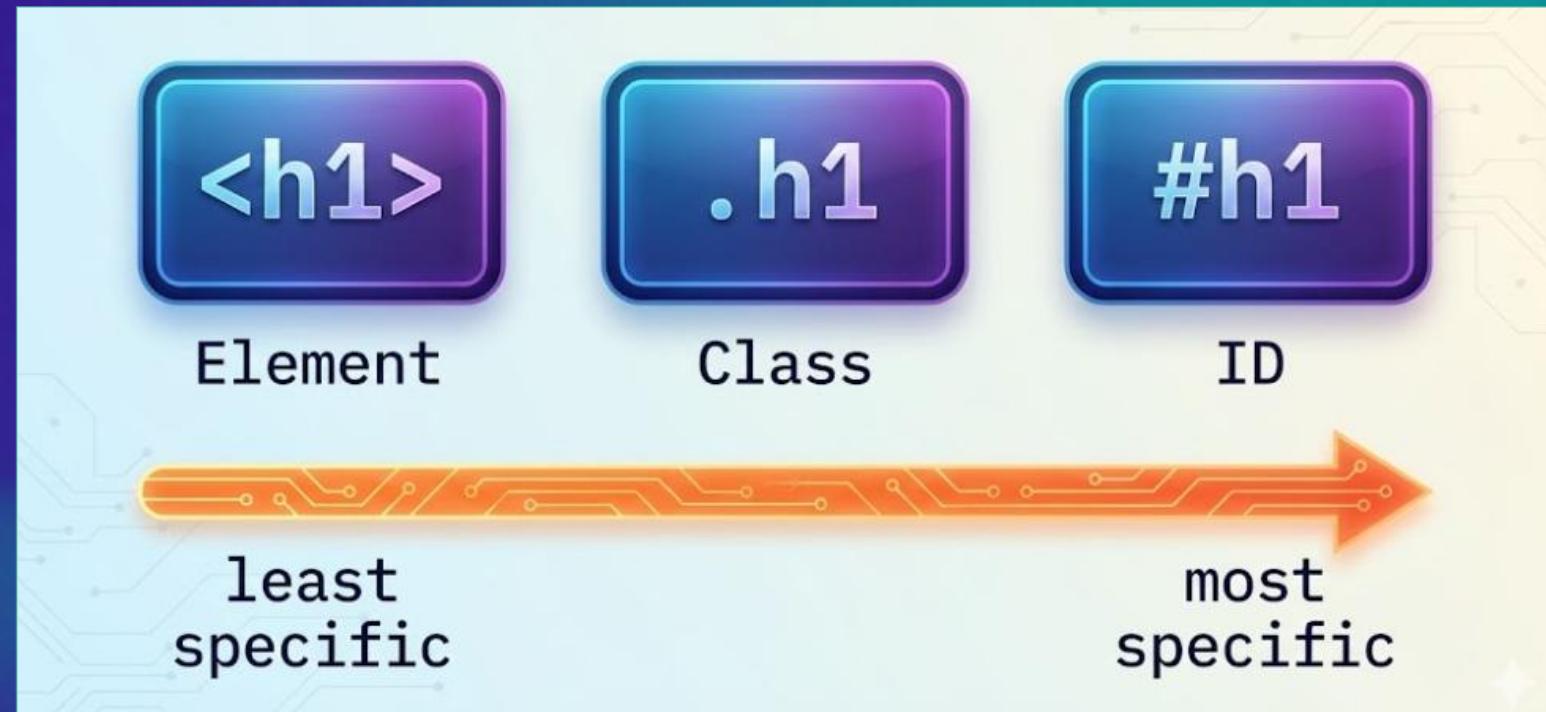
Inspect / Edit Page Elements and Styles

The screenshot shows a resume page for "John Doe" with various sections like Professional Summary, Experience, and Skills. A tooltip highlights the "Experience" section. The browser's developer tools are open, specifically the "Elements" tab, which displays the DOM structure and styles for the page. An H2 element under the "Experience" section is selected, and its style properties are shown in the "Styles" panel:

```
element.style {  
}  
h2 {  
    color: #34495e;  
    margin-top: 20px;  
    margin-bottom: 10px;  
    font-size: 1.3em;  
}
```

CSS Selectors and Basic Styles

Selectors, Rules, Class, Id, Nested Styling,



CSS Selectors

- Selecting elements by **tag name**

```
<h1>Page Title</h1>
```



```
h1 { color: blue; }
```

- Selecting elements by **class**

```
<p class="odd">Text</p>
```



```
.odd { font-size: 10px; }
```

- Selecting elements by **id**

```
<span id="login">Go</span>
```



```
#login { width: 150px; }
```

- Selecting **nested elements**

```
<div id="news"><p>Some  
news</p></div>
```



```
#news p { color: green; }
```

Styling Color, Font, Margin, Padding

- **Color / background:**
(named colors, #RRGGBB)

```
h1 { color: skyblue;  
background: #46523a; }
```

First Heading

- **Font name & size:**
(list of names, size in pt,
px, %, em, rem, ...)

```
p { font-family:  
monospace; font-size:  
18px; }
```

This is a
paragraph
of text.

- **Margin and padding:**
(in pt, px, %, em, rem, ...)

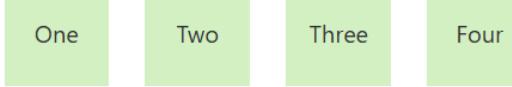
```
li { margin: 5px;  
padding: 5px 10px;  
display: inline-block;  
background: #d8e4cc; }
```

First Second
Third Fourth

Styling

- **Width and height:**
(in pt, px, %, em, rem, ...)
- **Border:**
(width, style, size)
- **Border-radius:**
(in pt, px, %, em, rem, ...)
- **Text-align:**
(left, center, justify)

```
div.tile { width: 70px;  
height: 70px; }
```



```
p { border-top: 4px  
solid olive; border:  
2px solid #ddd;  
padding: 5px; }
```

This is a paragraph
of text.

```
h1 { background: #ddd;  
border-radius: 5px;  
padding: 7px 15px; {  
text-align: center }
```

First Heading

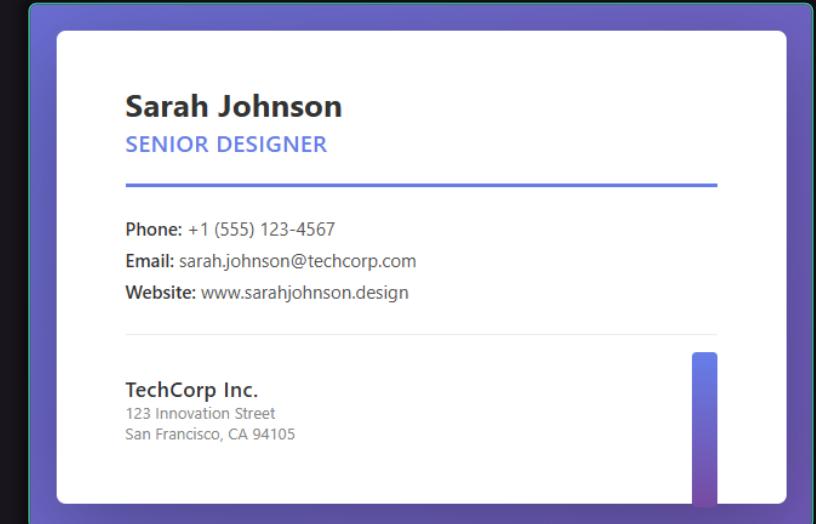
Exercise: Business Card

- Create a **business card** and style it with simple CSS:

Create a **business card**: `card.html`

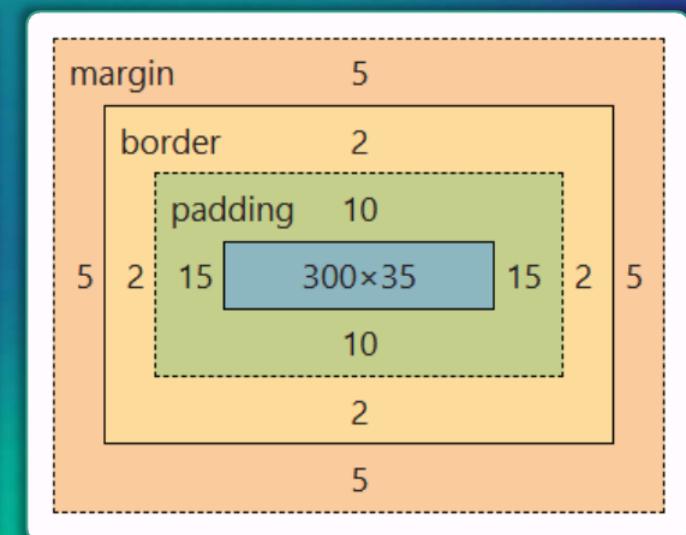
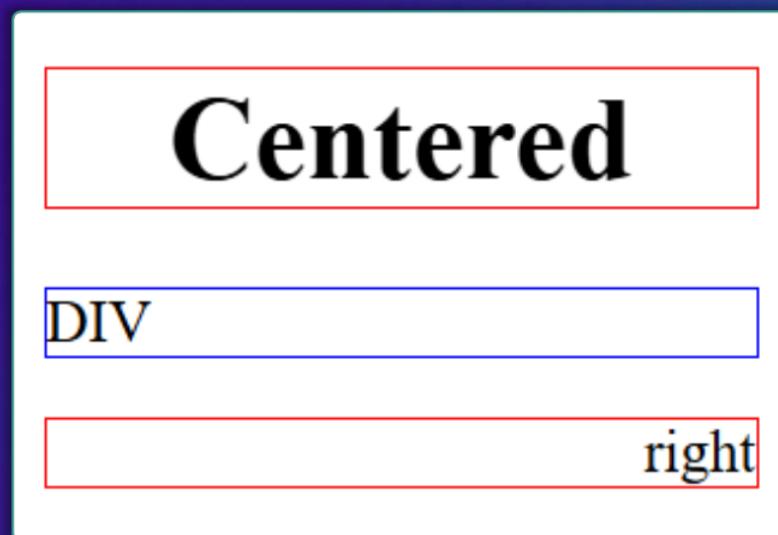
- Generate info: person name, position, phone, email, site, company name, address

- Card size: 85 mm x 55 mm
- Use simple elements: `h1`, `h2`, `div`, `p`
- Style with CSS selectors: by tag, by class, by id, nested selectors, simple styles only



Box Model, Inline, Block

CSS Box Model, Inline and Block Elements



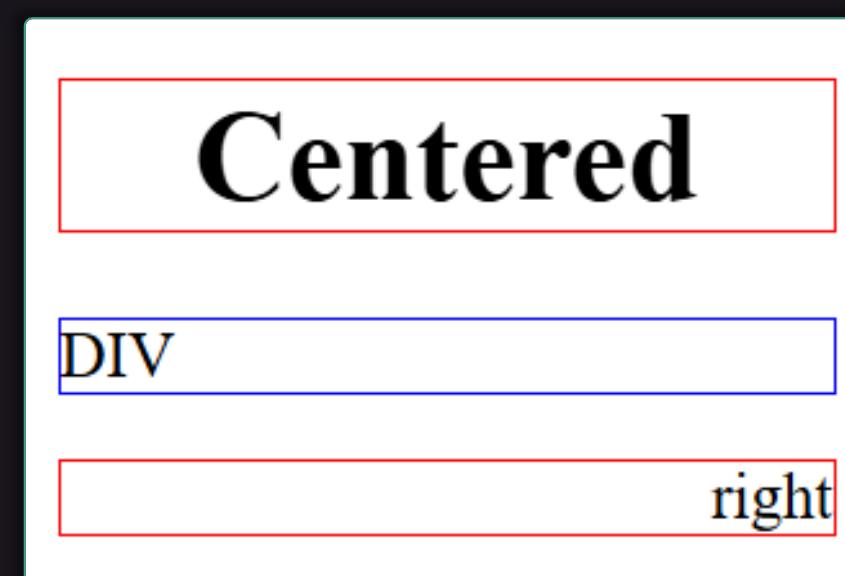
Block Elements

- **Block-level elements** are **rectangles**, which:
 - Start at a **new line**, take the **whole width** (by default)
- Examples: <**div**>, <**p**>, <**h1**>, <**h2**>, ... , <**ul**>, <**li**>

```
<h1 style="border:1px solid red;  
text-align:center">Centered</h1>
```

```
<div style="border:1px solid  
blue">DIV</div>
```

```
<p style="border:1px solid red;  
text-align:right">right</p>
```



Centered

DIV

right

Inline Elements

- **Inline-level elements** behave like text snippets:
 - Start at the current text position, flow left to right, and **wrap to the next line** when space runs out
 - Examples: ****, **<a>**, ****, ****, ****, **<label>**

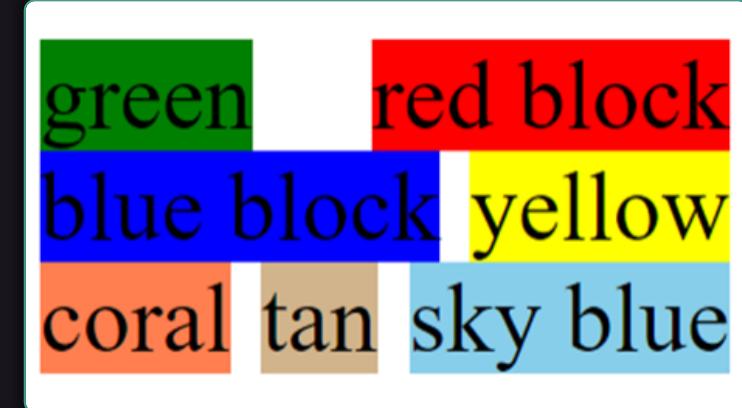
```
<p style="text-align:justify">Welcome  
<span style="color:white;  
background:blue;">to the Software University  
(SoftUni) in Sofia (Bulgaria)</span>  
, good luck!</p>
```

Welcome to the Software
University (SoftUni) in
Sofia (Bulgaria), good luck!

Inline-Block Elements

- **Inline-block** elements are **rectangles**, which flow like **words** in a text, without wrapping

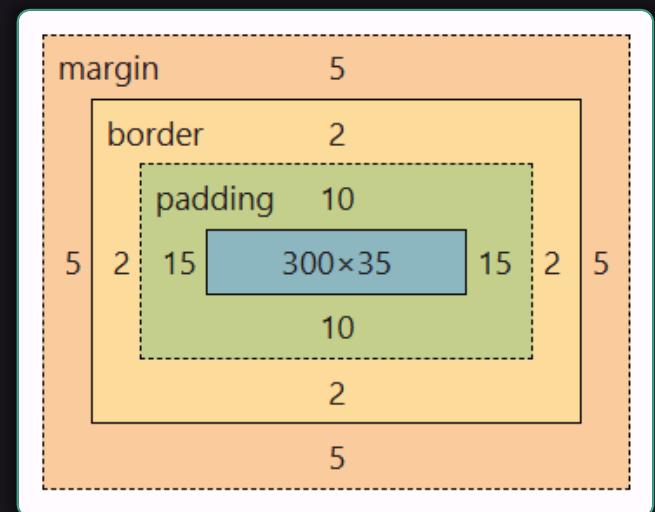
```
<div style="text-align:justify;">  
  <div style="display:inline-block;  
background:green">green</div>  
  <div style="display:inline-block;  
background:red">red block</div>  
  ...  
</div>
```



CSS Box Model

- The **CSS box model** describes the **element size** and **spacing** around it:
 - Element size = **content** + **padding** + **border**
 - Margin** = spacing outside the element

```
<h1 style="padding: 10px 15px; border: 2px solid green; margin: 5px">  
I am a Title</h1>
```

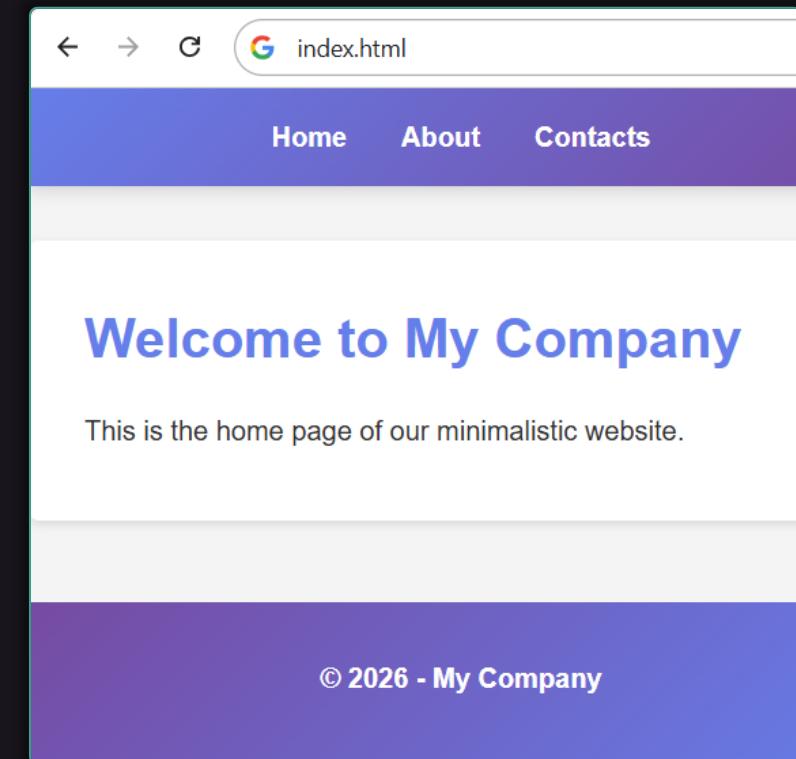


Exercise: Minimalistic Web Site

- Create a **minimalistic Web site**: Home | About | Contacts

Create a **minimalistic Web site**:

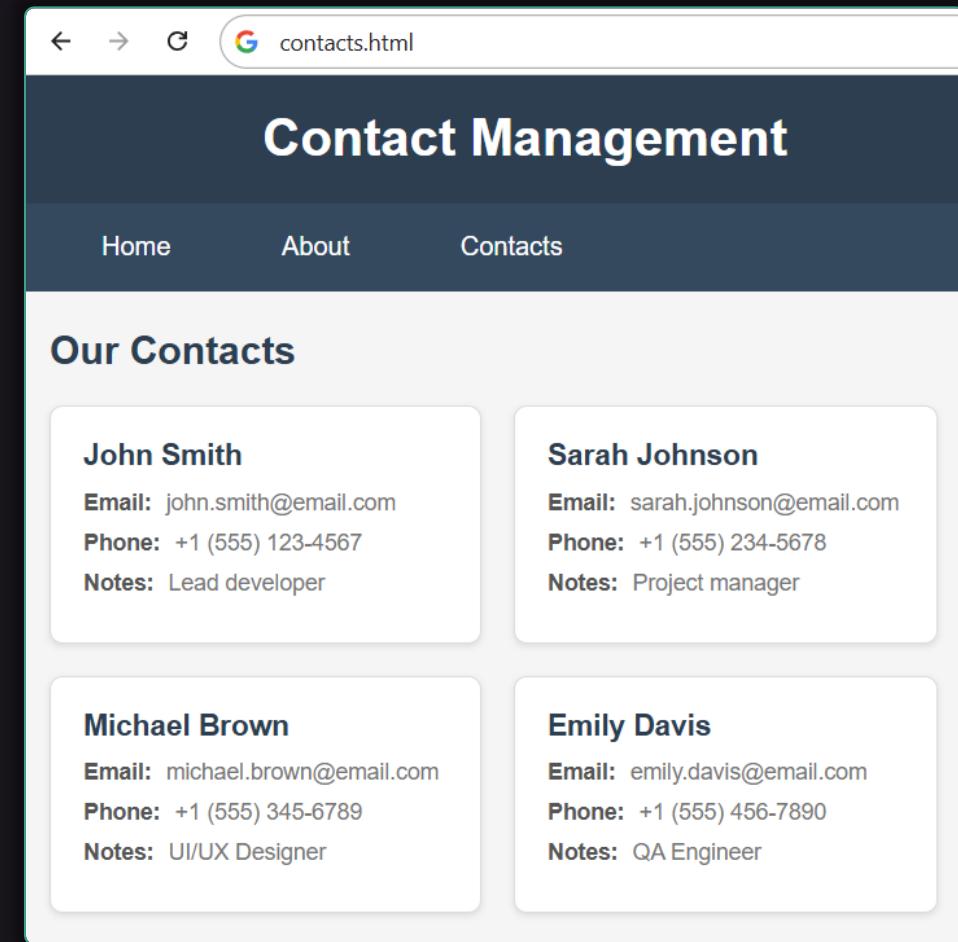
- index.html, about.html, contacts.html
- Top navigation menu: Home | About | Contacts
- Footer: (c) 2026 - My Company
- Add very simple CSS styling: styles.css



Exercise: Contacts Page

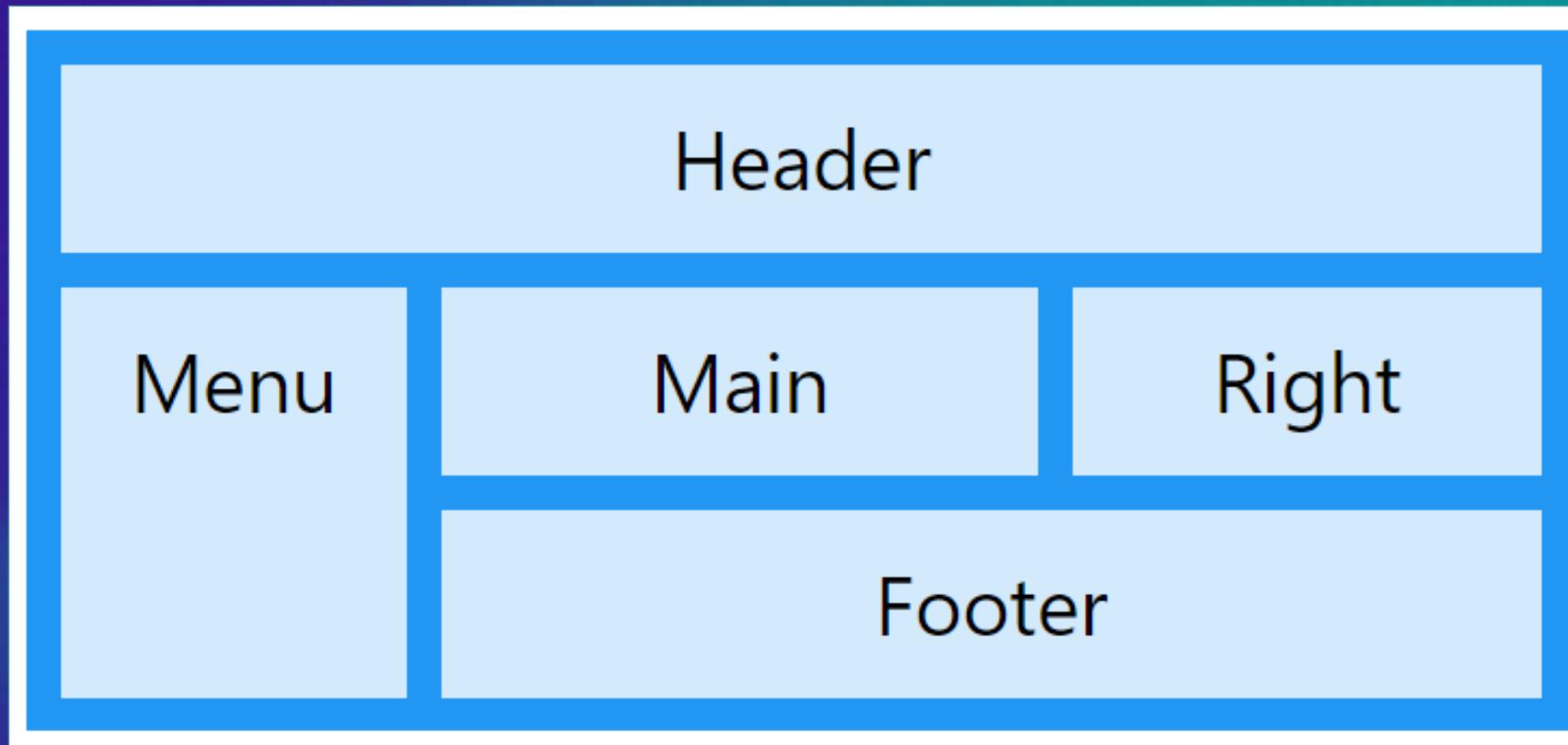
- Create a **contacts page**: name + email + phone + notes

- Create a simple web page
`**contacts.html**` (HTML + CSS)
 - Page header with nav menu:
Home | About | Contacts
 - Main page: list of contacts
(name + email + phone + notes), styled as boxes
 - Page footer
 - Use simple CSS styling, without layout system



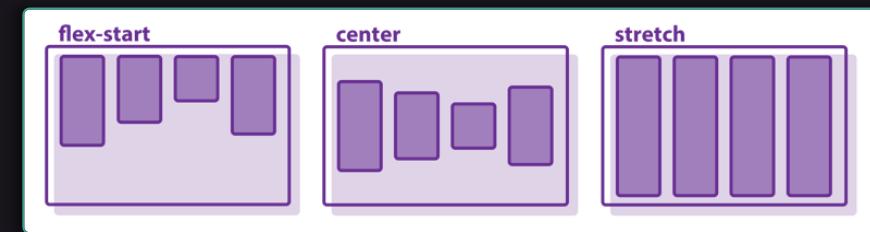
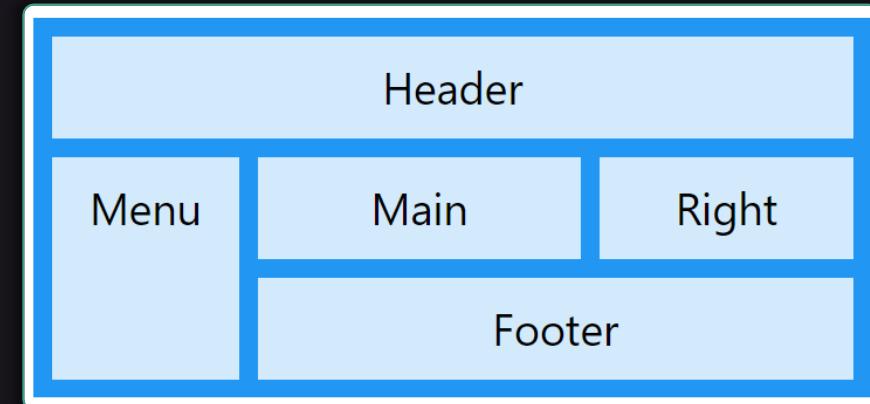
CSS Layouts

Flexbox, Grid and Responsive Layouts



CSS Layouts

- CSS layouts define how elements are arranged on a page
- Standard page layout: **block** elements (vertical flow) and **inline** elements (horizontal flow)
- **Flexbox** – one-dimensional layout (rows or columns)
- **Grid** – two-dimensional layout (rows and columns)

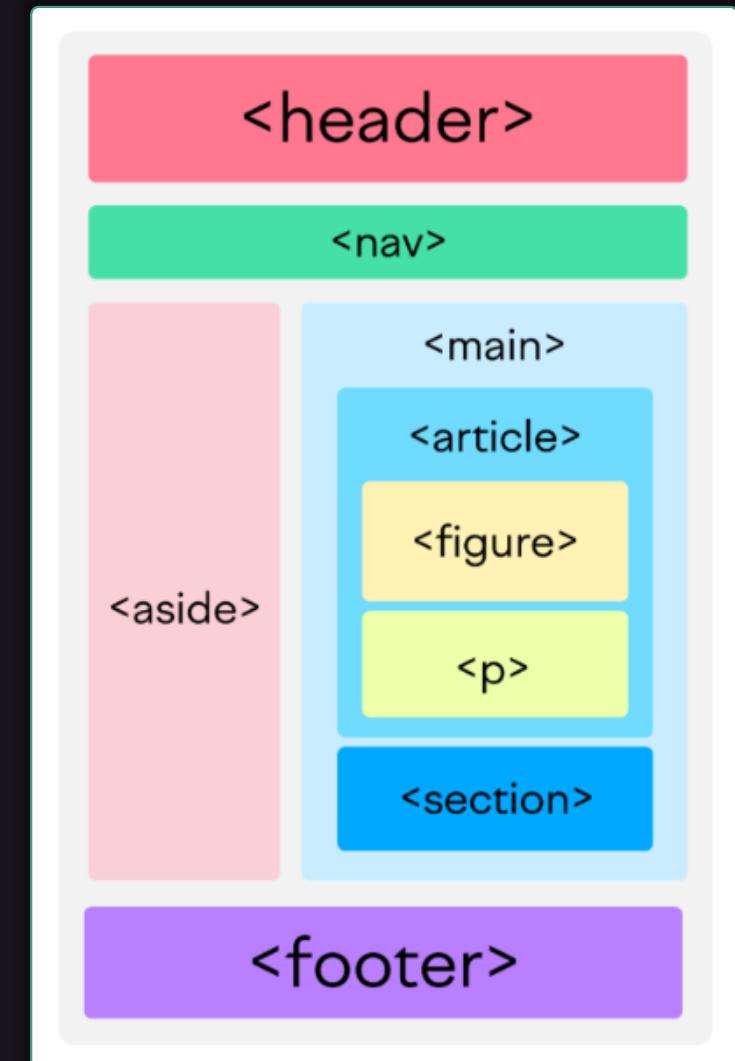


HTML Semantic Tags

- HTML **semantic tags** give semantics to page elements
 - **<header>, <nav>, <aside>, <main>, <footer>, <section>, <article>, <figure>**

```
<header>
  <h1>Home</h1>
  <nav>
    <ul>...</ul>
  </nav>
</header>
<aside>...</aside>
```

```
<main>
  <section>
    <h1>...</h1>
    <article>...</article>
  </section>
</main>
<footer>...</footer>
```

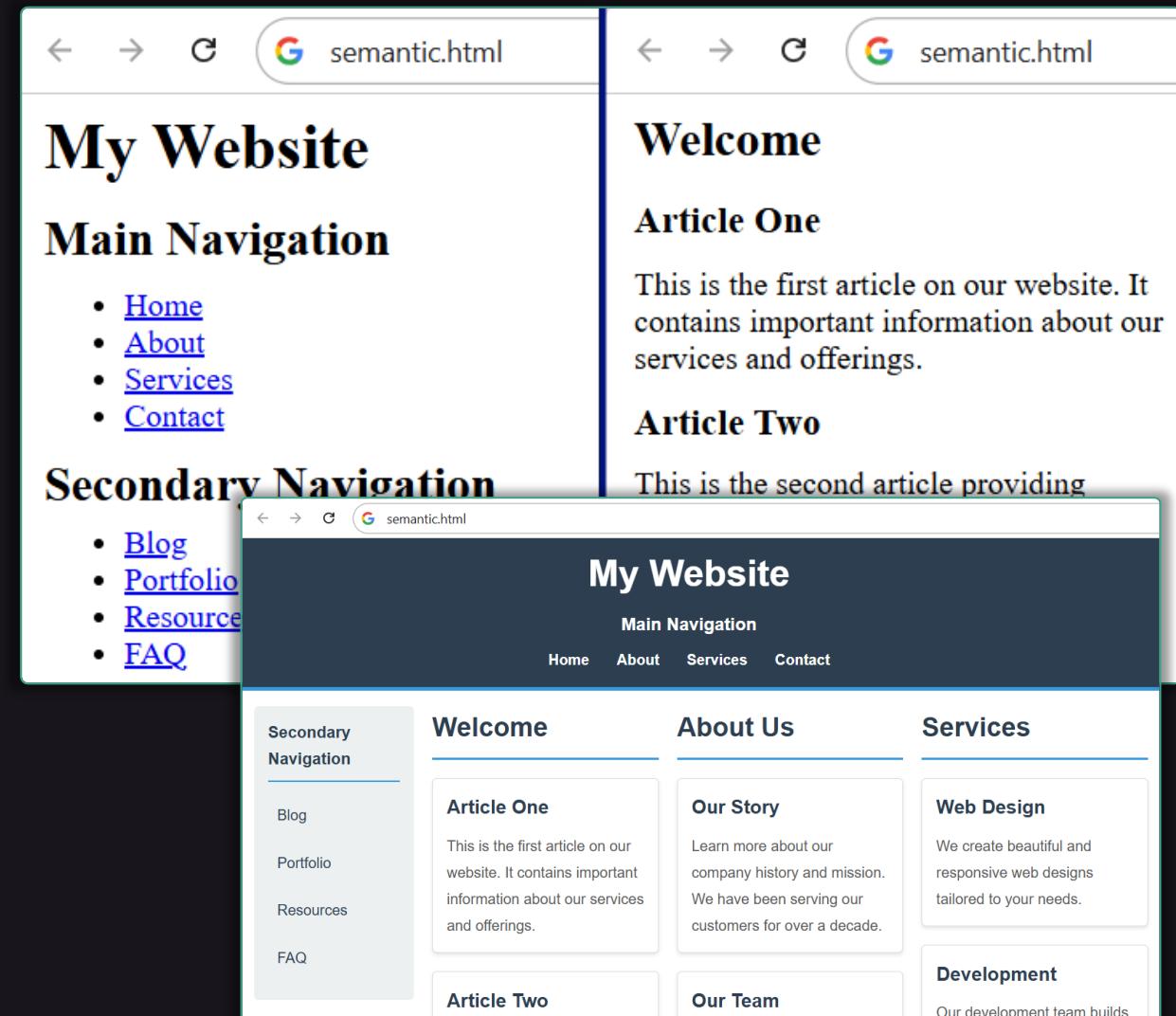


HTML Semantic Tags – Example

- Example of HTML page with **semantic tags**:

Generate a **semantic** HTML page: header with nav, sidebar with nav, main with sections and articles, footer. **No CSS!**

Implement standard layout for the page with **CSS**.



The figure displays three screenshots of a semantic HTML page, illustrating its structure and layout.

- Screenshot 1:** A browser window showing the full page. The title bar says "semantic.html". The page has a header with "My Website" and "Main Navigation" (links: Home, About, Services, Contact). Below it is "Secondary Navigation" (links: Blog, Portfolio, Resource, FAQ). The main content area contains two articles: "Article One" (text: "This is the first article on our website. It contains important information about our services and offerings.") and "Article Two" (text: "This is the second article providing").
- Screenshot 2:** A screenshot of the "My Website" header. It shows the "Main Navigation" (Home, About, Services, Contact) and a "Secondary Navigation" sidebar (Blog, Portfolio, Resources, FAQ).
- Screenshot 3:** A detailed view of the main content area. It shows the "Welcome" section (Article One), the "About Us" section ("Our Story"), and the "Services" section ("Web Design"). Each section contains descriptive text and links to other pages.

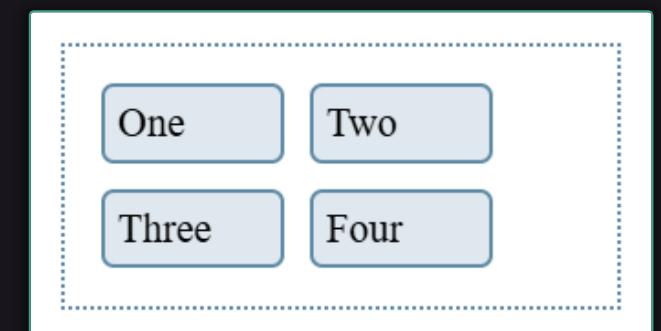
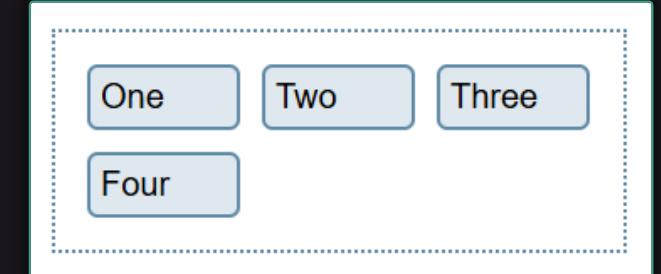
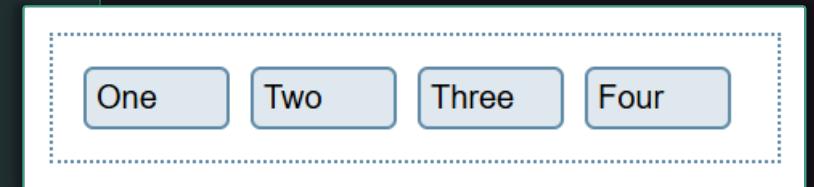
CSS Flex Layout: 1-Dimensional Flow

- CSS **flex layout** arranges elements in a **row** or **column**

```
<div class="container">  
  <div>One</div> <div>Two</div>  
  <div>Three</div> <div>Four</div>  
</div>
```

```
.container {  
  display: flex;  
  flex-flow: row wrap;  
  width: 200px;  
  ...  
}
```

```
.container div {  
  width: 60px;  
  margin: 5px;  
  padding: 5px;  
  ...  
}
```



Exercise: Playing Cards

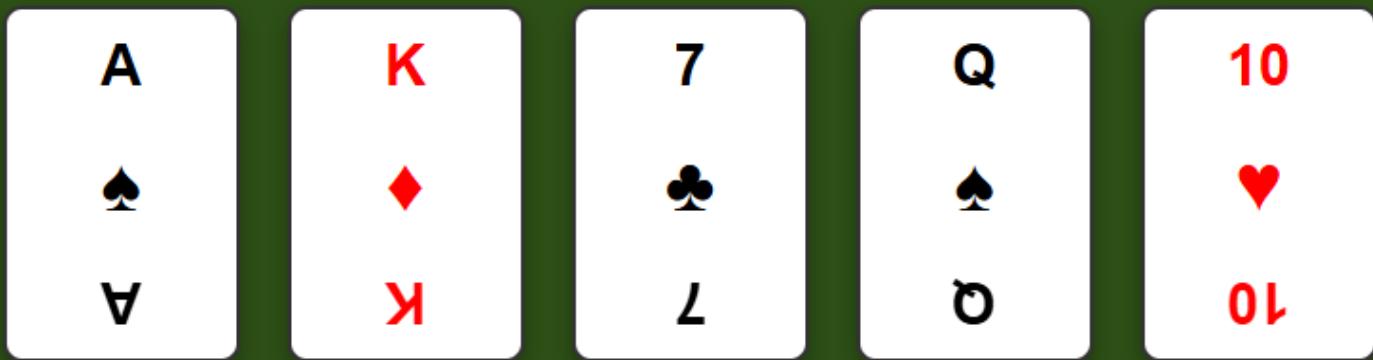
- Using **CSS flex layout**, visualize a **deck of cards**

Create a HTML page `cards.html` to visualize a **deck of 5 cards**:

- A♠, K♦, 7♣, Q♠, 10♥

Use **CSS flex layout**.

```
.deck {  
    display: flex;  
    flex-flow: row wrap;  
    gap: 20px;  
    justify-content: center;  
    align-items: center;  
}
```

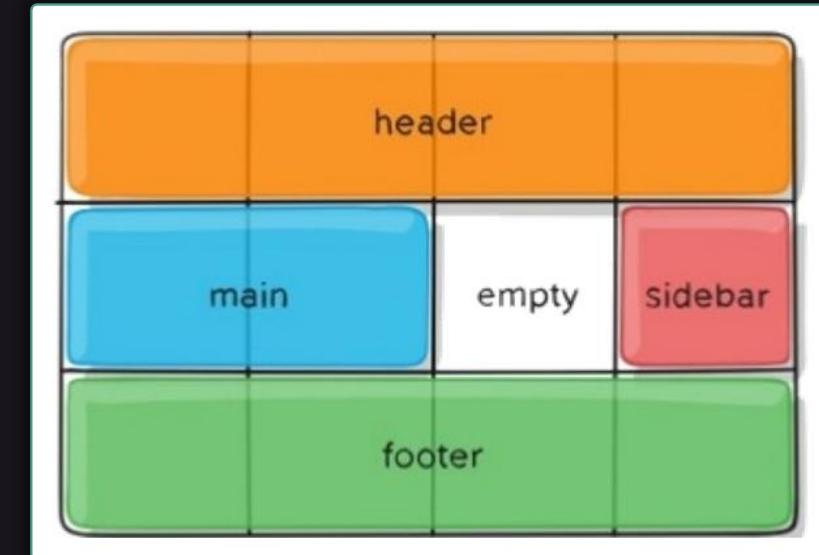


```
.card {  
    width: 100px;  
    height: 150px;  
    display: flex;  
    flex-direction: column;  
    justify-content: space-between;  
    align-items: center;
```

CSS Grid Layout: 2-Dimensional

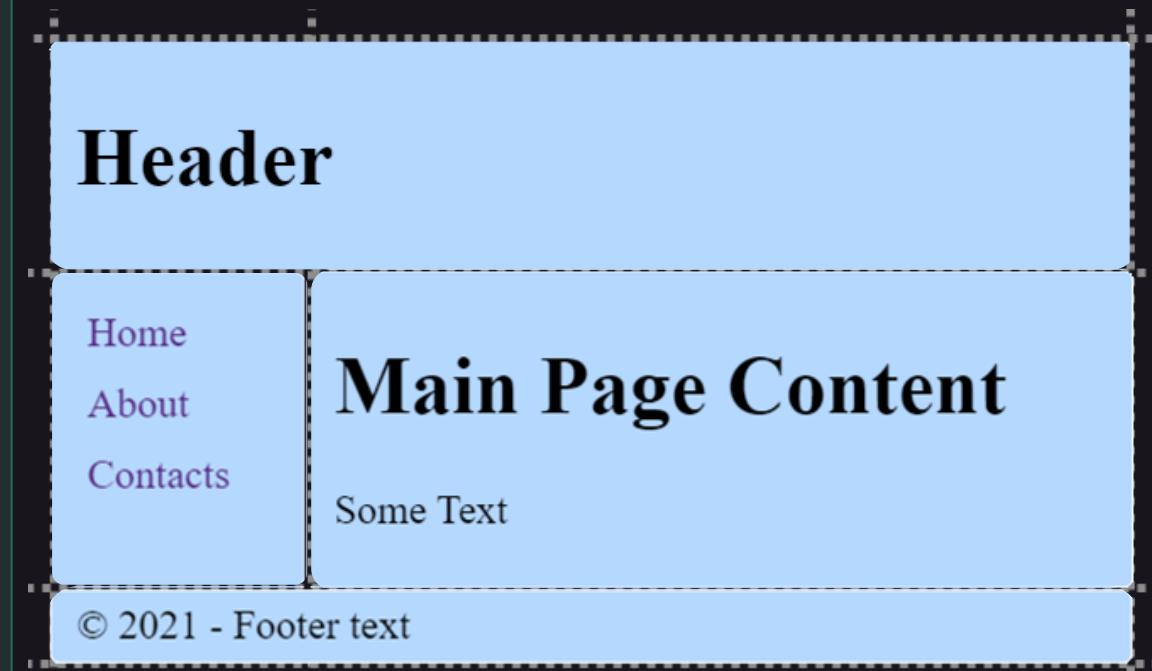
- **CSS grid** is universal 2D layout system for modern Web
 - Splits the container into grid system: **rows** and **columns**
 - **Grid areas** define the layout visually

```
body {  
    grid-template-areas:  
        "header header header header"  
        "main   main   empty  sidebar"  
        "footer footer footer footer";  
}
```



CSS Grid Layout – Example

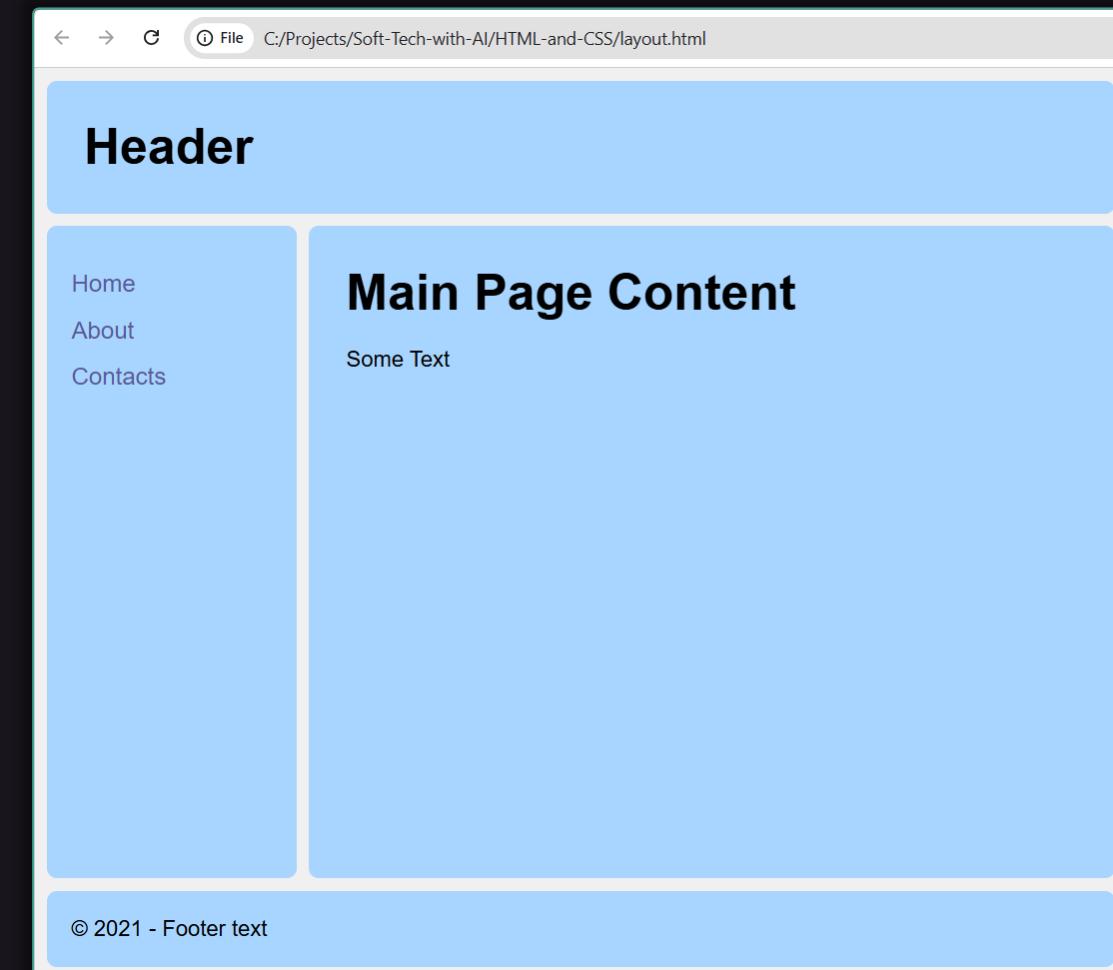
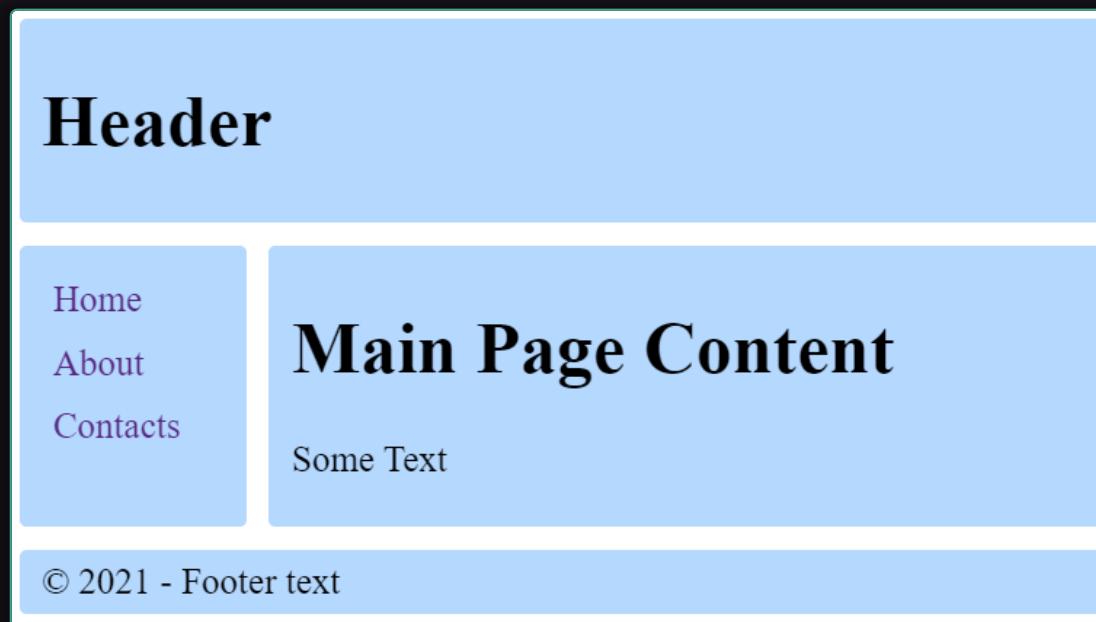
```
body { display: grid;  
      grid-template-areas:  
        "header header"  
        "aside main"  
        "footer footer";  
      grid-template-columns:  
        100px auto;  
    }  
  
header { grid-area: header; }  
aside { grid-area: aside; }  
main { grid-area: main; }  
footer { grid-area: footer; }
```



Exercise: Simple Site Layout

- Create a **simple CSS grid layout**, like at the screenshot

Create a Web page layout
`**layout.html**` , like at the
screenshot. Use **CSS grid**.



Responsive Design

Adaptive Design for Different Screen Sizes



0-480

*Smaller
smartphones*



481-768

*Tablets & larger
smartphones*



769-1279

*Laptops, larger tablets
in landscape, and small
desktops*



1280+

*Larger desktops
and monitors*

Responsive Design

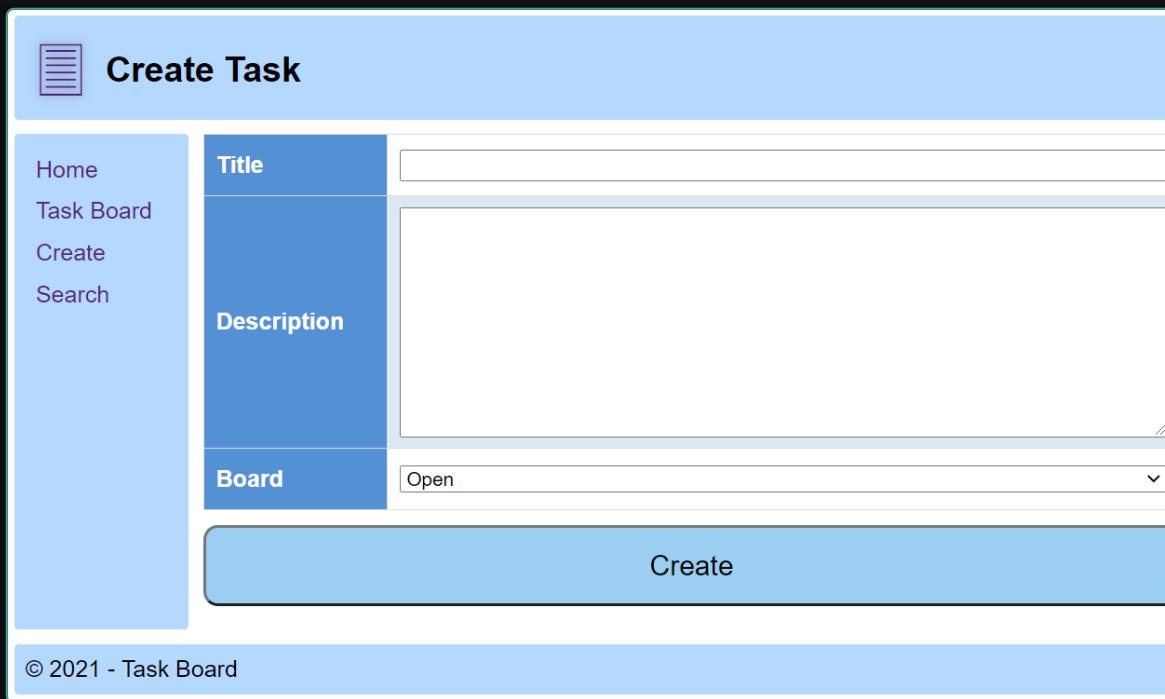
- **Responsive design** (adaptive design, adaptive layout)
 - Site auto **re-arranges its layout** at different **screen sizes**
 - **Desktop screens, tablet screens, mobile screens**
 - CSS may define custom styles at different **breakpoints**



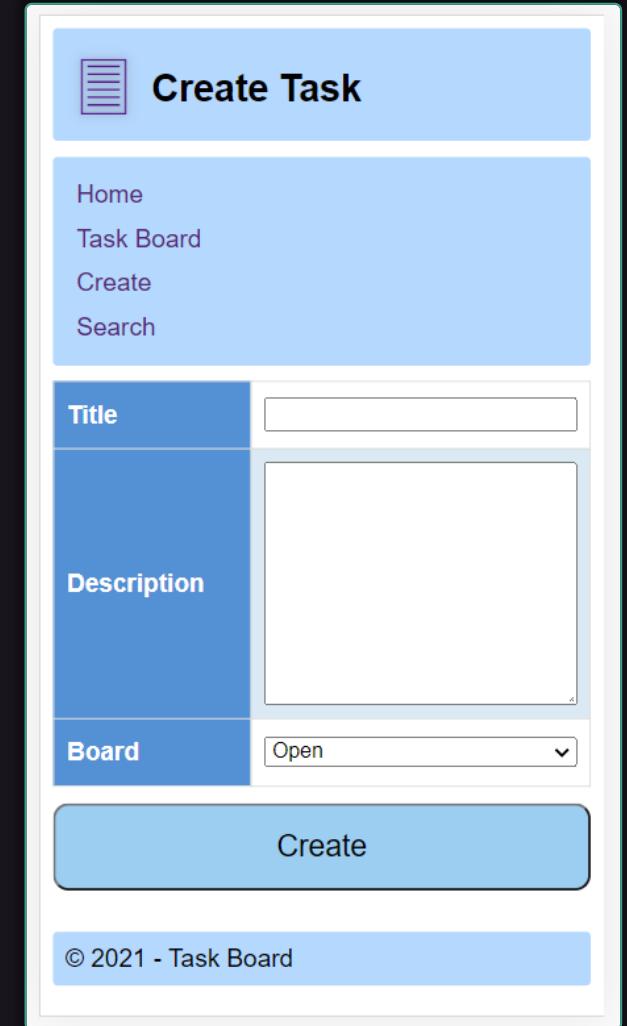
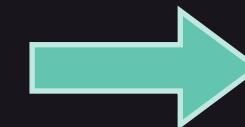
```
@media (max-width: 480px) {  
    nav {  
        display: none;  
    }  
}
```

Responsive Design – Example

- Example of **responsive design**:
 - Web form, rendered on **desktop**
 - The same form, rendered on **mobile**



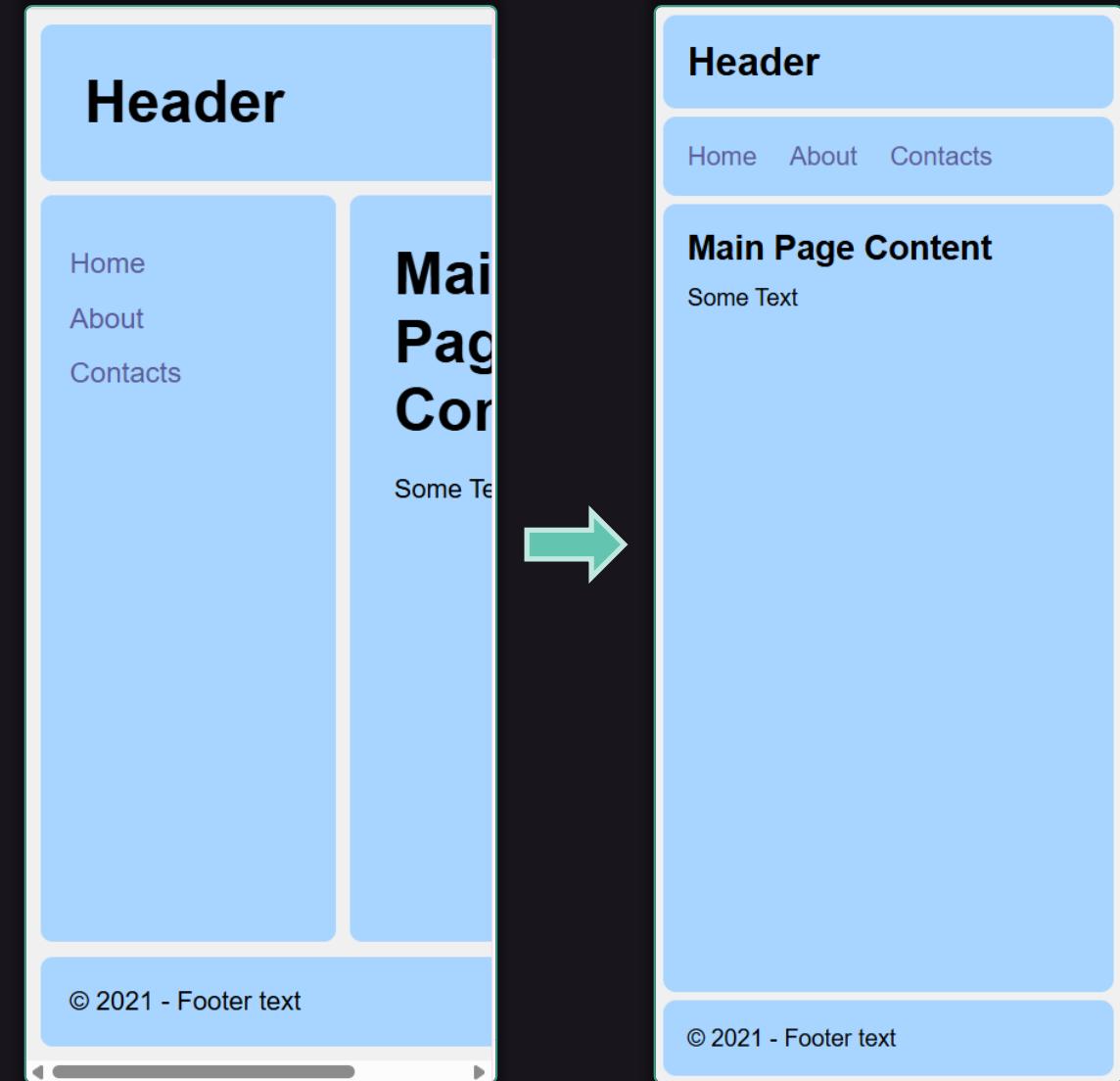
The desktop version of the 'Create Task' form is displayed. It features a sidebar on the left with links: Home, Task Board, Create, and Search. The main area contains fields for 'Title' (with a placeholder 'Enter title...'), 'Description' (with a large text area), and 'Board' (with a dropdown menu showing 'Open'). A prominent blue 'Create' button is at the bottom. The entire form is set against a light blue background.



The mobile version of the 'Create Task' form is shown. The sidebar and navigation links from the desktop version are present. The main area has been reorganized: the 'Title' field is at the top, followed by the 'Description' field, and then the 'Board' field with its dropdown menu. The 'Create' button is now a large, rounded rectangle at the bottom. The overall layout is more compact and suitable for a smaller screen. The background is also light blue.

How to Make Design Responsive?

- How to implement **responsive design** for mobile devices?
 - Just tell the AI dev agent:
"*implement responsive layout*"



- It will add CSS **breakpoints** for different screen resolutions

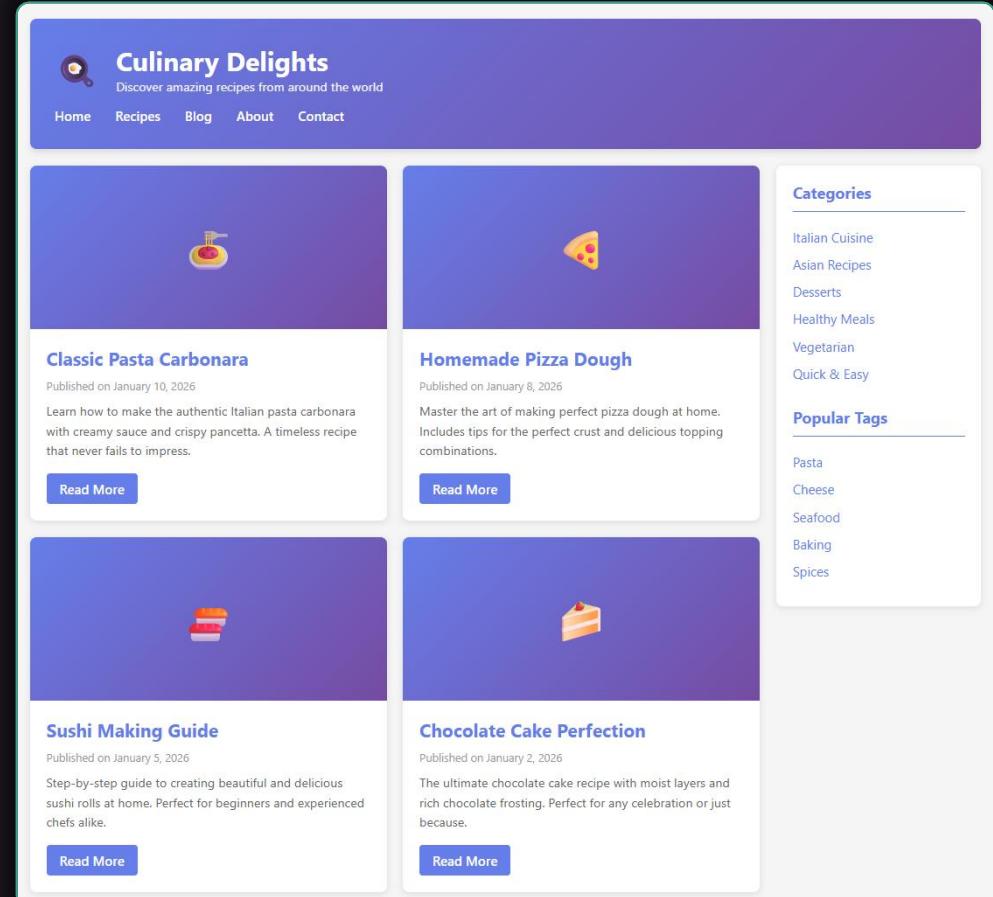
Exercise: Blog Responsive Layout

- Create a **responsive Web layout**, for a real-world web site

Create a **responsive CSS layout** for a blog page: blog.html

- Choose a topic of your choice, e.g. a training course, a travel blog, a cooking recipe site, etc.

- Site header: logo + title + nav
- Site main: sections, holding articles
- Site sidebar: categories, tags, links
- Site footer: copyright text + navigation



The screenshot displays a responsive web layout for a blog titled "Culinary Delights". The header includes the logo, title, and a navigation bar with links to Home, Recipes, Blog, About, and Contact. The main content area features four cards, each representing a recipe:

- Classic Pasta Carbonara**: Published on January 10, 2026. Description: Learn how to make the authentic Italian pasta carbonara with creamy sauce and crispy pancetta. A timeless recipe that never fails to impress. [Read More](#)
- Homemade Pizza Dough**: Published on January 8, 2026. Description: Master the art of making perfect pizza dough at home. Includes tips for the perfect crust and delicious topping combinations. [Read More](#)
- Sushi Making Guide**: Published on January 5, 2026. Description: Step-by-step guide to creating beautiful and delicious sushi rolls at home. Perfect for beginners and experienced chefs alike. [Read More](#)
- Chocolate Cake Perfection**: Published on January 2, 2026. Description: The ultimate chocolate cake recipe with moist layers and rich chocolate frosting. Perfect for any celebration or just because. [Read More](#)

The sidebar on the right contains two sections: "Categories" and "Popular Tags".

- Categories**: Italian Cuisine, Asian Recipes, Desserts, Healthy Meals, Vegetarian, Quick & Easy.
- Popular Tags**: Pasta, Cheese, Seafood, Baking, Spices.

HTML Tables

Tables, Rows, Columns, Merged Cells

Category	Product	Price (EUR)
Fruits	Apples (1 kg)	€2.99
Fruits	Bananas (1 kg)	€1.49
Fruits	Oranges (1 kg)	€3.29
Vegetables	Carrots (1 kg)	€1.29

HTML Tables

- **HTML tables** visualize tabular data
 - **Tables** hold **rows** and each row holds **columns**
 - Tags: **<table>**, **<tr>**, **<td>**, ...

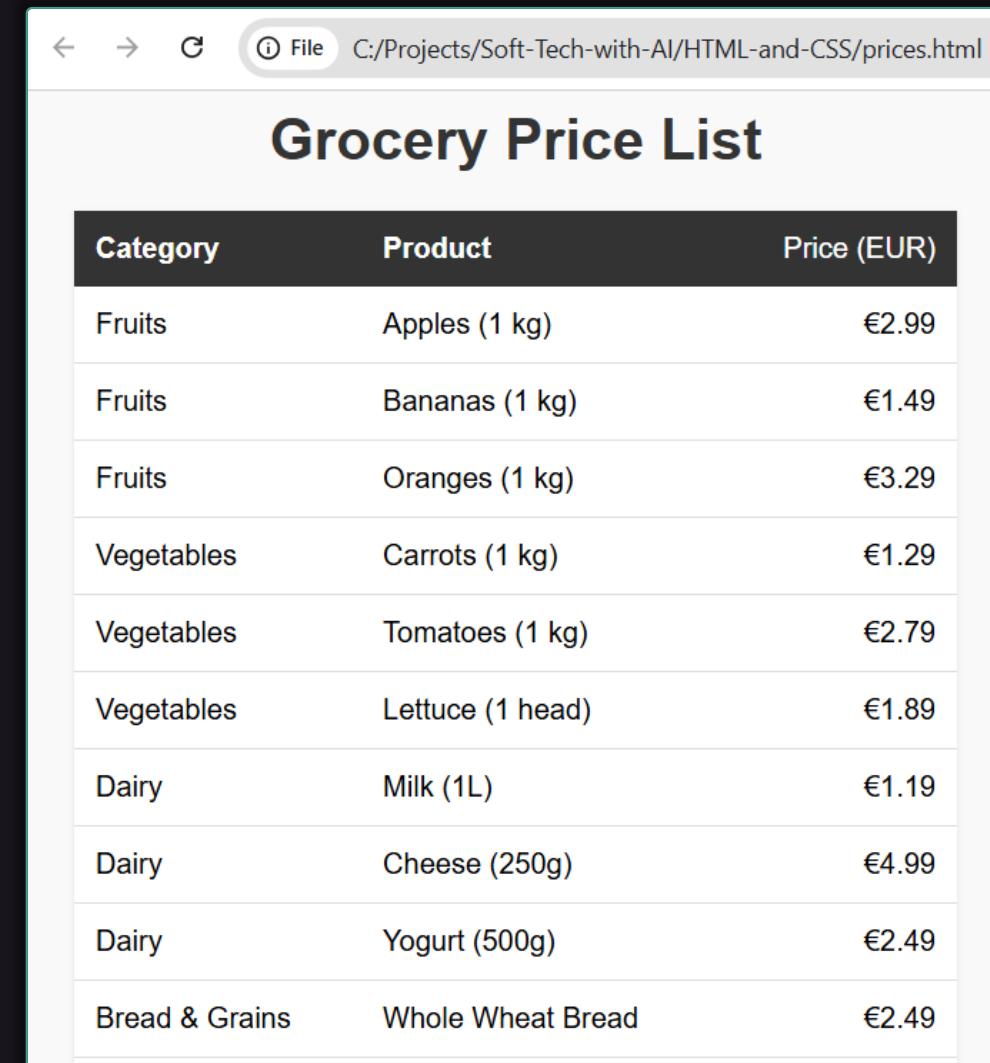
```
<table border="1">
  <tr><th>Country</th><th>Town</th></tr>
  <tr><td>Germany</td><td>Munich</td></tr>
  <tr><td>France</td><td>Paris</td></tr>
  <tr><td>Bulgaria</td><td>Sofia</td></tr>
</table>
```

Country	Town
Germany	Munich
France	Paris
Bulgaria	Sofia

Exercise: Product Comparison Table

- Create a product comparison Web page with **HTML table**

Create a price list HTML table: `prices.html`
- List grocery products
- Category | Product | Price (in EUR)
- Use an **HTML table** with minimalistic CSS styling



A screenshot of a web browser displaying a grocery price list. The title is "Grocery Price List". The table has three columns: Category, Product, and Price (EUR). The data includes various items like Apples, Bananas, Oranges, Carrots, Tomatoes, Lettuce, Milk, Cheese, Yogurt, and Whole Wheat Bread, categorized by Fruits, Vegetables, Dairy, and Bread & Grains.

Category	Product	Price (EUR)
Fruits	Apples (1 kg)	€2.99
Fruits	Bananas (1 kg)	€1.49
Fruits	Oranges (1 kg)	€3.29
Vegetables	Carrots (1 kg)	€1.29
Vegetables	Tomatoes (1 kg)	€2.79
Vegetables	Lettuce (1 head)	€1.89
Dairy	Milk (1L)	€1.19
Dairy	Cheese (250g)	€4.99
Dairy	Yogurt (500g)	€2.49
Bread & Grains	Whole Wheat Bread	€2.49

Exercise: Weekly Schedule Table

Create a school weekly schedule timetable: timetable.html

- Table with columns: Monday, Tuesday, ..., Sunday
- Each column holds timeslots: 8:00 - 9:00, 9:00 - 10:00
- Each timeslot each day holds a lesson, e.g. Math, English, Coding, Art
- The Lunch Break slot (12:00 - 13:00) stretches across Monday through Friday
- Saturday and Sunday stretch across all timeslots and hold "holiday"

Time	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
8:00 - 9:00	Math	English	Coding	Art	Math	Holiday	Holiday
9:00 - 10:00	English	Coding	Art	Math	English		
10:00 - 11:00	Coding	Art	Math	English	Coding		
11:00 - 12:00	Art	Math	English	Coding	Art		
12:00 - 13:00	Lunch Break						
13:00 - 14:00	Math	English	Coding	Art	Math		
14:00 - 15:00	English	Coding	Art	Math	English		
15:00 - 16:00	Coding	Art	Math	English	Coding		

HTML Forms

Form, Action, Form Controls, Submit

HTML Form

First Name :

Date of Birth :

Last Name :

Email :

SUBMIT

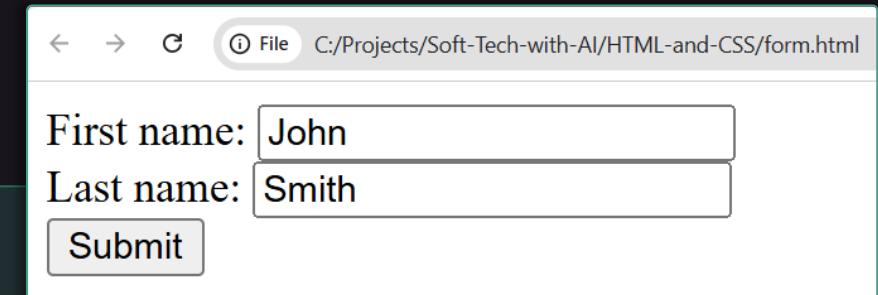
RESET

HTML Forms

- **HTML forms** collect user-entered data

- Hold form **fields** + **submit** button

```
<form>
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname" value="John" />
  <br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname" value="Smith" />
  <br>
  <input type="submit" value="Submit" />
</form>
```

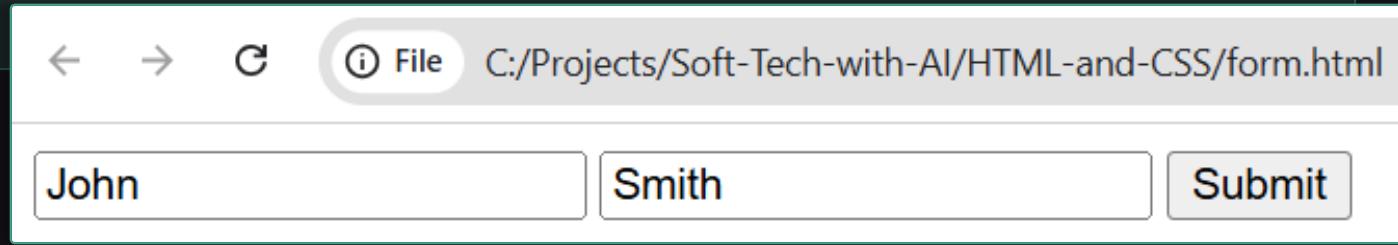


Form Structure and Submission

- Forms collect **fields** and **submit** them to certain **action** script

```
<form method="post" action="register.php">
  <input type="text" name="fname" value="John" />
  <input type="text" name="lname" value="Smith" />
  <input type="submit" value="Submit" />
</form>
```

submit

Form data sent to `register.php`
fname=John&lname=Smith

fname	John
lname	Smith

HTML Form Fields

- Text box (text, password, number, date, color), text area, drop-down, radio button, listbox, checkbox, submit button, labels, fieldset + legend

I agree to the Privacy Policy type="checkbox"

I want to receive offers and newsletters type="radio"

Your computer skills level 0/100 type="range"

Send us your request type="submit"

Free registration from here: type="button"

Attach your files: No file chosen type="file"

Exercise: Sign Up / Login Form

- Create user **sign up** and **login forms**

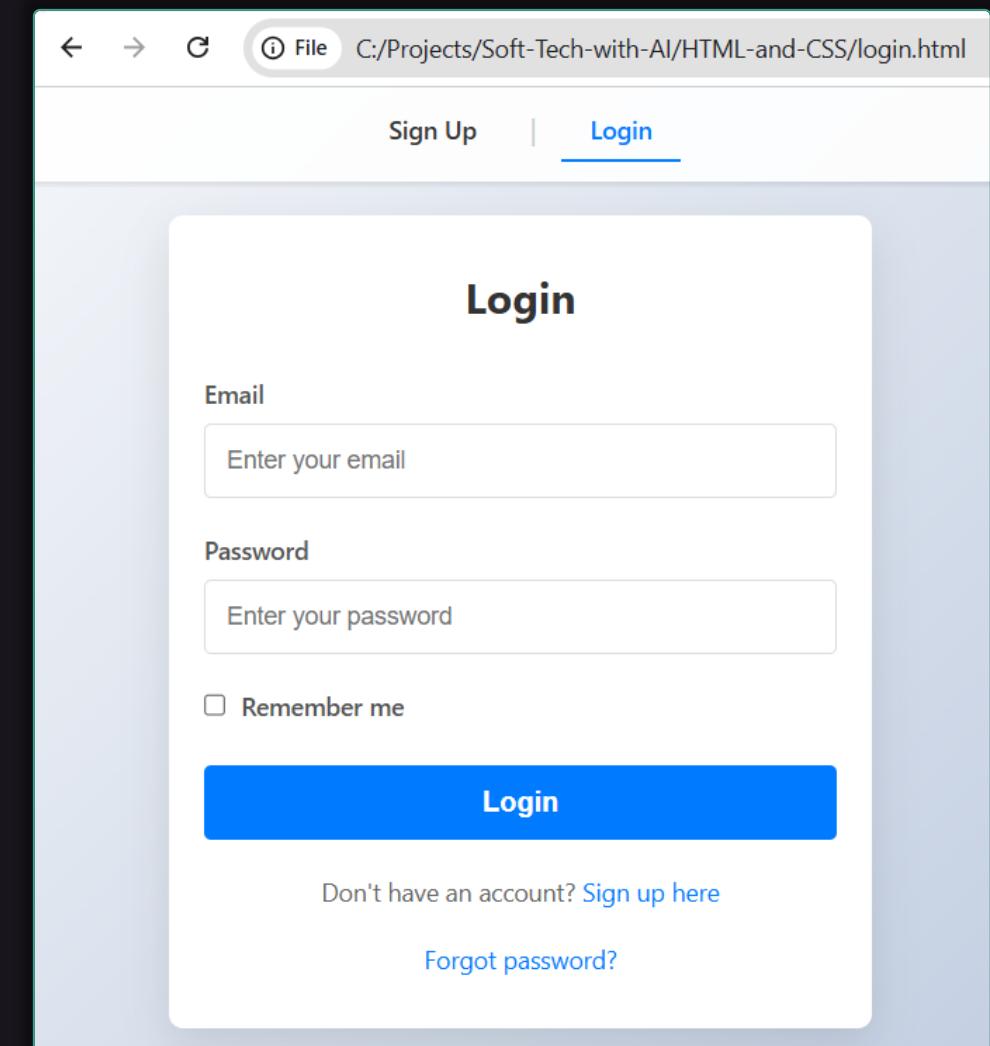
Create a sample user **sign up** and **login form**

- Define two HTML pages:
`signup.html`, `login.html`

- Show simple navigation:

[Sign Up](#) | [Login](#)

- Use **minimalistic CSS styling**



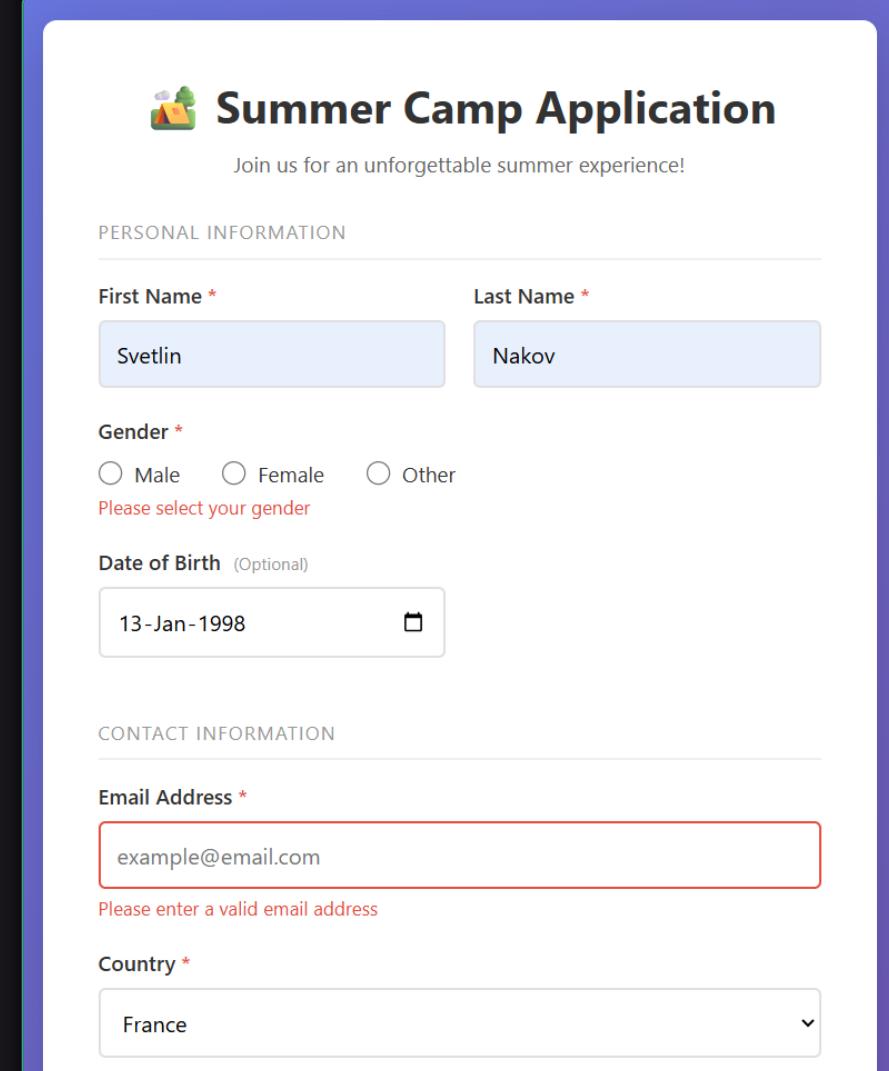
The screenshot shows a web browser window with the following details:

- Address Bar:** C:/Projects/Soft-Tech-with-AI/HTML-and-CSS/login.html
- Navigation:** Back, Forward, Stop, File menu.
- Buttons:** Sign Up (disabled), Login (active).
- Form Section:** Login
- Fields:** Email (placeholder: Enter your email), Password (placeholder: Enter your password).
- Checkboxes:** Remember me (unchecked).
- Buttons:** A large blue "Login" button.
- Links:** Don't have an account? [Sign up here](#), [Forgot password?](#).

Exercise: Application Form

Create an application form for summer camp: [apply-camp.html](#)

- First Name, Last Name
- Gender (radio buttons: Male / Female / Other)
- Date of birth (calendar, optional)
- Email (with validation)
- Country (drop down: Germany, France, Bulgaria, United States)
- Why you want to apply? (textarea)
- Comments (textarea, optional)
- Style the form with CSS
- Implement basic validation



The screenshot shows a web-based application form titled "Summer Camp Application". At the top, there is a small icon of a tent and the text "Join us for an unforgettable summer experience!". Below this is a section titled "PERSONAL INFORMATION" containing fields for "First Name *" (Svetlin) and "Last Name *" (Nakov). A "Gender *" section follows, with three radio buttons labeled "Male", "Female", and "Other", and a note "Please select your gender". An optional "Date of Birth" field contains the value "13-Jan-1998". The next section is "CONTACT INFORMATION", featuring an "Email Address *" field with the value "example@email.com" and an error message "Please enter a valid email address". Finally, a "Country *" dropdown menu is set to "France".

JavaScript in HTML

HTML, JavaScript and DOM Interaction

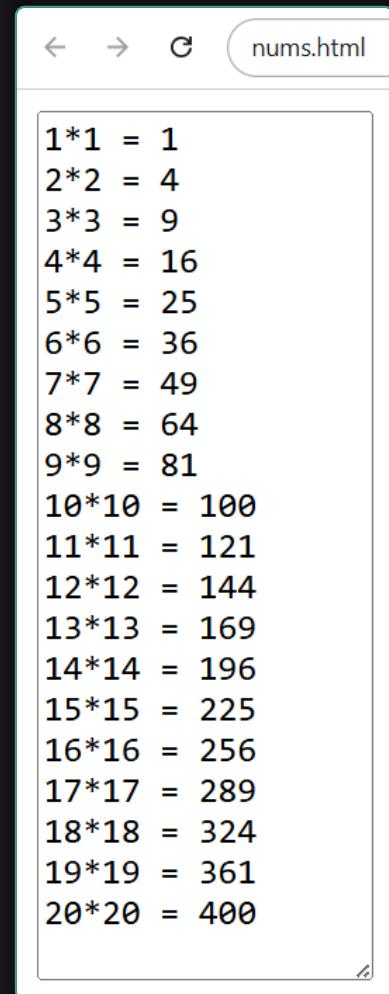
```
<textarea id="output" rows="21" cols="15">
</textarea>
<script>
    let result = "";
    for (let i = 1; i <= 20; i++)
        result += `${i}*${i} = ${i * i}\n`;
    document.getElementById('output').value = result;
</script>
```

JavaScript in HTML

- Inserting **JavaScript** code inside an HTML page:

```
<html>
  <textarea id="outputBox" rows="21" cols="30">
  </textarea>
</html>

<script>
  let result = "";
  for (let i = 1; i <= 20; i++)
    result += `${i}*${i} = ${i * i}\n`;
  let outputBox =
    document.getElementById('outputBox');
  outputBox.value = result;
</script>
```



The screenshot shows a browser window titled "nums.html". The page displays a list of multiplication results from 1*1 to 20*20, each on a new line. The results are as follows:

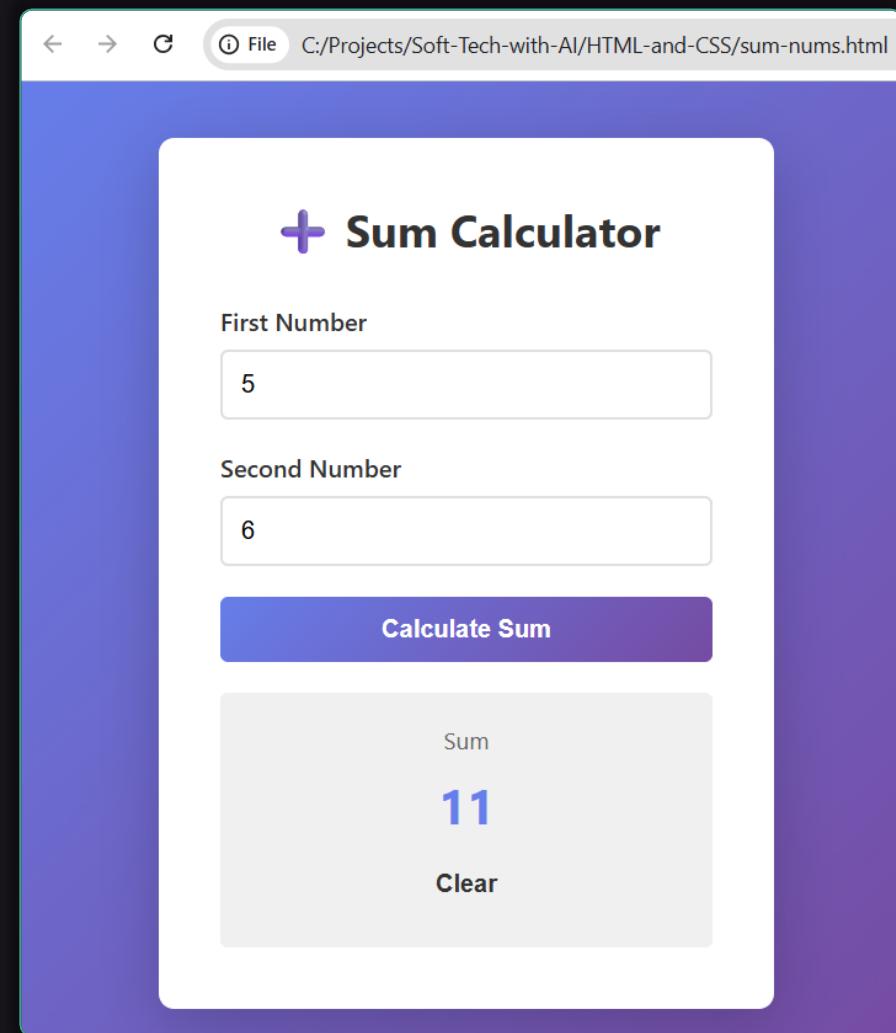
Product	Value
1*1	1
2*2	4
3*3	9
4*4	16
5*5	25
6*6	36
7*7	49
8*8	64
9*9	81
10*10	100
11*11	121
12*12	144
13*13	169
14*14	196
15*15	225
16*16	256
17*17	289
18*18	324
19*19	361
20*20	400

Exercise: Sum Two Numbers in JS

- Create a HTML page to sum two numbers

Create a HTML page `sum-nums.html` to sum two numbers:

- Two form fields (input numbers)
- Third form field (result)
- Button [Calculate]
- Inline JavaScript summator
- Keep it simple



Lesson Summary

- **HTML** defines Web page **content**, uses text + **tags**
- **CSS** defines content **styling**, uses **selectors** + **rules**

```
<h1>Languages</h1>
<ul>
  <li>JS</li>
  <li>Python</li>
  <li>PHP</li>
</ul>
```

```
h1 { color: #2E4AA7 }
li {
  display: inline;
  padding: 5px 10px;
  background: #CCC;
}
```

Languages

JS Python PHP

- CSS layout systems: **Flexbox** (1D flow), **Grid** (2D)
- HTML **tables** show tabular data: **<table>**, **<tr>**, **<td>**
- HTML **forms** collect user data in form **fields**

Questions?



Postbank – Exclusive Partner for SoftUni AI



- One of the leading **banking institutions** in Bulgaria
- Member of the Eurobank Group with € 103 billion assets
- Innovative trendsetter with next generation **beyond banking**, transforming today, empowering tomorrow
- Certified **Top Employer 2025** by the international Top Employers Institute
- Proven people care and **wellbeing initiatives**
- Benefits and unlimited access to professional, **personal and leadership trainings and programs**
- www.postbank.bg / careers.postbank.bg



Diamond Partners of Software University



Diamond Partners of SoftUni Digital



**SUPER
HOSTING
.BG**



Diamond Partners of SoftUni Digital



HUMAN

NETPEAK
DIGITAL GROWTH PARTNER



Marmalab | Е-комерс агенция

ETIEN YANEV
Break Your *Limits*. Live Your *Brand*.

1FORFIT



Diamond Partners of SoftUni Creative



Organization Partners of SoftUni Creative



THE
BUCKS
TOWN'S
WORK

