Webprogrammierung und Mobile Computing

Skript für die 2AHIF · Schuljahr 2026/27

Dieses Skript begleitet dich durch HTML, CSS und JavaScript. Es ersetzt das bisherige HTML-Basics-Skript. Alle Erklärungen sind auf Deutsch; wichtige englische Fachbegriffe werden bei ihrer Einführung mitgenannt. HTML-Namen und andere Bestandteile des Quelltexts bleiben in ihrer vorgeschriebenen Schreibweise.

Du brauchst zu Beginn keine HTML-Kenntnisse. Wir lernen die Werkzeuge und die Schreibweise gemeinsam kennen. Im Unterricht wechseln wir gelegentlich zwischen den drei Teilen. Die Kapitelnummern helfen dir, die passende Erklärung wiederzufinden.

Bearbeitungsstand 17.09.2026: Die Kapitel 1 bis 5 bilden den ausgearbeiteten Einstieg. Die Kapitel 6 bis 23 sind die Gliederung für die weitere Ausarbeitung; ihre Inhalte folgen im Laufe des Schuljahres. Die Gliederung ist keine Vorgabe, alles in dieser Reihenfolge zu unterrichten.

Inhaltsübersicht

Teil 1: HTML

1. Webseiten und unsere Werkzeuge

1.1 Was gehört zu einer Webseite?

Eine Webseite kann Texte, Bilder, Links und Eingabefelder enthalten. Drei Techniken übernehmen unterschiedliche Aufgaben:

Technik	Aufgabe	Beispiel
HTML	Struktur und Bedeutung von Inhalten beschreiben	Dieser Text ist eine Überschrift.
CSS	Das Aussehen gestalten	Die Überschrift ist blau.
JavaScript	Verhalten programmieren	Nach einem Klick wird ein Ergebnis berechnet.

HTML bedeutet **Hypertext Markup Language**, auf Deutsch etwa „Auszeichnungssprache für verknüpfte Texte“. Mit Auszeichnungen kennzeichnen wir, welche Bedeutung ein Inhalt hat. Für unsere erste Webseite verwenden wir nur HTML.

Im Web fordert ein Browser Dateien von einem Webserver an. Ein Webserver stellt diese Dateien bereit. Heute öffnen wir eine HTML-Datei direkt vom eigenen Rechner. Dafür benötigen wir keinen Webserver.

1.2 Editor und Browser

Ein **Code-Editor** ist ein Programm zum Schreiben und Bearbeiten von Quelltext (*source code*). Unsere HTML-Datei enthält normalen Text mit HTML-Auszeichnungen. Im Editor sehen und verändern wir diesen Quelltext.

Ein **Browser** liest den HTML-Code und stellt die Webseite dar. Beispiele sind Firefox, Chrome, Edge und Safari. Editor und Browser zeigen somit zwei unterschiedliche Ansichten derselben Datei: den Quelltext und die daraus dargestellte Seite.

1.3 Unser Editor: Visual Studio Code

Wir verwenden **Visual Studio Code**, kurz **VS Code**. Der Editor ist für Windows, macOS und Linux verfügbar. Er unterstützt uns unter anderem durch farbliche Hervorhebung des Quelltexts, Vorschläge beim Tippen und automatisch ergänzte End-Tags. Erweiterungen (*extensions*) können weitere Funktionen hinzufügen.

Für den Anfang brauchst du nur drei Bereiche:

Bereich	Was machst du dort?
Dateiexplorer (*Explorer*)	Dateien und Unterordner deines geöffneten Projektordners finden
Dateireiter (*editor tab*)	Zwischen geöffneten Dateien wechseln
Bearbeitungsbereich (*editor*)	Den Quelltext der ausgewählten Datei bearbeiten

Öffne den gesamten Projektordner über **Datei → Ordner öffnen …** (*File → Open Folder …*). So siehst du später auch die Bilder, die zu deiner Seite gehören. Öffne anschließend `index.html` im Dateiexplorer von VS Code.

1.4 Deinen Projektordner vorbereiten

1. Lege einen Ordner namens `wmc-pinkafeld` an.
2. Kopiere den **Inhalt** des bereitgestellten Startordners hinein.
3. Öffne `wmc-pinkafeld` in VS Code.
4. Öffne dort `index.html` zum Bearbeiten.
5. Öffne dieselbe Datei zusätzlich im Browser, beispielsweise über „Öffnen mit …“ im Dateimanager des Betriebssystems.
6. Ordne Editor und Browser nebeneinander an.

Dein Ordner sieht so aus:

```
wmc-pinkafeld/
├── index.html
└── bilder/
    └── lernort.svg
```

Prüfe die Dateiendung: Die Datei heißt `index.html`, nicht `index.html.txt`. Die Endung `.html` kennzeichnet sie als HTML-Dokument. `index.html` ist ein üblicher Name für eine Einstiegsseite.

1.5 Bewusst ausprobieren: Ändern → Speichern → Browser neu laden

Führe diesen Ablauf zuerst ohne automatische Live-Vorschau durch:

1. Suche in der Startdatei die Zeile mit `<title>`. Ändere nur den Text zwischen `<title>` und `</title>` auf `Meine erste Webseite`.
2. Wechsle zum Browser und lade die Seite neu, **bevor** du speicherst. Bei ausgeschaltetem automatischem Speichern siehst du noch den bisherigen Titel im Browser-Tab.
3. Speichere in VS Code: **Strg+S** unter Windows/Linux oder **Cmd+S** unter macOS.
4. Wechsle zum Browser und lade die Seite mit der Schaltfläche „Neu laden“ erneut.
5. Beobachte den geänderten Titel im Browser-Tab.

Der Editor bearbeitet zunächst einen Arbeitsstand. Durch Speichern landet dieser auf der Festplatte. Beim Neuladen liest der Browser die gespeicherte Datei erneut. Falls automatisches Speichern aktiviert ist, übernimmt VS Code den Speicherschritt bereits selbst; für diese erste Demonstration schalten wir es kurz aus.

Merksatz: Ändern → Speichern → Browser neu laden → Ergebnis prüfen.

1.6 Danach: Die Live-Vorschau

Eine **Live-Vorschau** aktualisiert die angezeigte Webseite automatisch beim Bearbeiten oder Speichern, abhängig von der verwendeten Funktion und ihren Einstellungen. Das erleichtert später die Arbeit, wenn Quelltext und Ergebnis nebeneinander sichtbar sind.

Aktuelle VS-Code-Versionen bieten eine integrierte Browser-Vorschau. Je nach installierter Version kann auch eine Erweiterung verwendet werden. Die Lehrperson zeigt die auf den Unterrichtsrechnern verfügbare Variante. Ein Vorschaufeld innerhalb von VS Code und ein separates Browserfenster sind unterschiedliche Ansichten; beide stellen HTML dar.

Auch mit Live-Vorschau speicherst du deine Dateien. Für die Abgabe zählt der gespeicherte Projektordner.

Kurz prüfen: Wozu brauchst du den Editor? Wozu den Browser? Warum kann der Browser noch einen alten Stand zeigen?

2. Die Sprache HTML: Elemente, Tags und Attribute

2.1 Was ist ein Tag?

Ein **Tag** ist eine Markierung in spitzen Klammern. Damit zeichnen wir Inhalte aus. Unser erstes Beispiel beschreibt einen Absatz:

```
<p>Ich lerne HTML.</p>
```

Bestandteil	Bezeichnung	Bedeutung
`<p>`	Start-Tag (*start tag* oder *opening tag*)	Hier beginnt ein Absatz.
`Ich lerne HTML.`	Inhalt (*content*)	Dieser Text gehört zum Absatz.
`</p>`	End-Tag (*end tag* oder *closing tag*)	Hier endet der Absatz.

Der Schrägstrich / steht beim End-Tag **vor** dem Namen. Start-Tag, Inhalt und End-Tag bilden hier gemeinsam ein **HTML-Element** (*HTML element*). Ein Tag und ein Element sind also nicht dasselbe.

Wir schreiben die Tag-Namen klein. Zwischen < und dem Namen steht kein Leerzeichen. Im Browser erscheinen normalerweise die Inhalte; die Tags steuern deren Bedeutung und Darstellung.

2.2 Elemente verschachteln

Ein Element kann andere Elemente enthalten. Das nennt man **Verschachtelung** (*nesting*).

```
<body>
  <p>Ich lerne HTML.</p>
</body>
```

Der Absatz liegt innerhalb des Seiteninhalts. Wir schließen zuerst das innere Element und danach das äußere. Einrückungen helfen uns beim Lesen; sie erzeugen keine entsprechenden Abstände auf der Webseite.

2.3 Was ist ein Attribut?

Ein **Attribut** (*attribute*) ergänzt eine Eigenschaft eines Elements. Es steht im Start-Tag hinter dem Elementnamen.

```
<html lang="de">
```

Hier ist `lang` der Attributname und `de` der Wert. Dazwischen steht ein Gleichheitszeichen. Wir schreiben Attributwerte in doppelte Anführungszeichen. Vor dem Attribut steht ein Leerzeichen.

`lang` kommt von *language* und bezeichnet die Sprache. `de` steht für Deutsch. Attributnamen und festgelegte Werte werden nicht ins Deutsche übersetzt.

Browser und Übersetzungsdienste können diese Sprachangabe auch als Hinweis für die automatische Übersetzung verwenden. Eine falsche Angabe wie `lang="en"` bei einem deutschen Text kann zu einer unpassenden Übersetzung beitragen. So könnte aus einem Navigationspunkt „Menu“ bei einer

Übersetzung aus dem Englischen ins Deutsche plötzlich „Speisekarte“ werden – obwohl das Navigationsmenü gemeint ist. Das hängt auch vom Kontext und vom Übersetzungsdienst ab: Browser können die Sprache zusätzlich anhand des Textes erkennen. Die richtige Sprachangabe hilft, garantiert aber keine fehlerfreie Übersetzung.

2.4 Elemente ohne End-Tag

Manche Elemente haben keinen Inhalt zwischen zwei Tags. Sie heißen **leere Elemente** (*void elements*). Dazu gehören `meta` und `img`.

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

Für diese Elemente gibt es in HTML keinen End-Tag: Wir schreiben beispielsweise kein `</meta>` und kein `</img>`. Das ist eine Eigenschaft dieser Elemente, keine allgemeine Abkürzung für andere Tags.

Kurz prüfen: Zeige Start-Tag, Inhalt und End-Tag in `<p>Hallo!</p>`. Welche Aufgabe hat der Schrägstrich?

3. Das HTML-Grundgerüst

3.1 Ein vollständiges kleines Dokument

```
<!DOCTYPE html>
<html lang="de">
<head>
  <meta charset="UTF-8">
  <title>Meine erste Webseite</title>
</head>
<body>
  <h1>Pinkafeld entdecken</h1>
  <p>Hier entsteht meine erste Webseite.</p>
</body>
</html>
```

Dieses Beispiel ist ein **HTML-Dokument mit einem Grundgerüst**. Ein Formular zum Eingeben von Daten lernen wir erst in Kapitel 8 kennen.

3.2 Dokumenttyp und Wurzelelement

`<!DOCTYPE html>` ist die **Dokumenttyp-Deklaration** (*document type declaration*). Sie steht am Anfang und sorgt dafür, dass der Browser das Dokument im Standardmodus verarbeitet. Sie ist kein gewöhnliches HTML-Element und besitzt keinen End-Tag.

`<html>` ist das **Wurzelelement** (*root element*). Es umfasst den Kopf und den Inhalt des Dokuments. `</html>` beendet dieses Element.

`lang="de"` gibt die Hauptsprache des Dokuments an. Diese Information hilft beispielsweise Vorleseprogrammen bei der Aussprache. Sie übersetzt die Seite nicht automatisch.

3.3 Der Dokumentkopf: head, charset und title

`head` bedeutet „Kopf“. Der **Dokumentkopf** enthält Informationen über die Seite. Seine Angaben erscheinen nicht als gewöhnlicher Seiteninhalt.

`<meta charset="UTF-8">` legt die Zeichenkodierung fest. `charset` steht für *character set* („Zeichensatz“). Die Datei soll ebenfalls als UTF-8 gespeichert sein. So können auch Umlaute und viele weitere Schriftzeichen richtig gelesen werden. `meta` kennzeichnet Metadaten, also Informationen über das Dokument.

`<title>Meine erste Webseite</title>` bestimmt den Dokumenttitel. `title` bedeutet „Titel“. Diesen Text zeigt der Browser im Browser-Tab an.

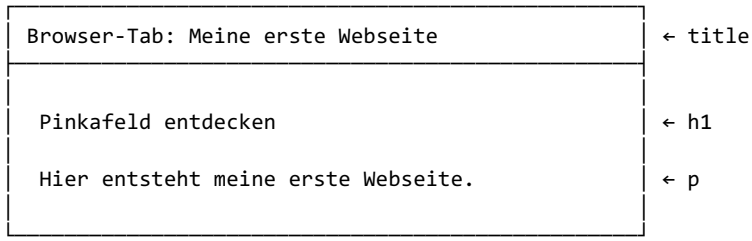
3.4 Der Seiteninhalt: body

body bedeutet „Körper“. Innerhalb von `<body>` und `</body>` stehen die Inhalte der Webseite, beispielsweise Überschriften, Absätze, Links und Bilder.

`h1` bezeichnet eine Überschrift der ersten Ebene (*heading level 1*). `p` steht für *paragraph*, also Absatz. Diese Elemente schauen wir uns in Kapitel 4 genauer an.

3.5 Wo erscheinen title, h1 und p?

Die folgende schematische Abbildung zeigt die Zuordnung aus dem Beispiel:



Der Titel im Browser-Tab und die Hauptüberschrift auf der Seite dürfen unterschiedlich lauten. Der Browser stellt `h1` normalerweise groß und fett dar. Entscheidend ist aber seine Bedeutung als Hauptüberschrift. Das genaue Aussehen gestalten wir später mit CSS.

Ausprobieren: Ändere zuerst nur den Text in `title`, dann nur den Text in `h1`. Speichere und lade jeweils neu. Beobachte, welcher Bereich sich verändert.

3.6 Zusatz in unserer Startdatei: viewport

Die bereitgestellte Startdatei enthält außerdem diese Angabe im Dokumentkopf:

```
<meta name="viewport"
      content="width=device-width, initial-scale=1">
```

Sie hilft bei der Darstellung auf schmalen Bildschirmen. `name="viewport"` benennt die Einstellung für den sichtbaren Darstellungsbereich. `content` enthält ihre Werte: `width=device-width` verwendet die Gerätebreite als Ausgangspunkt; `initial-scale=1` legt den anfänglichen Zoomfaktor fest. Heute lässt du diese Zeile unverändert. Wir vertiefen sie bei der responsiven Gestaltung in Kapitel 14.

4. Texte, Links und Bilder

4.1 Überschriften und Absätze

```
<h1>Pinkafeld entdecken</h1>
<h2>Mein Lernort</h2>
<p>Ich besuche die HTL Pinkafeld. Hier lerne ich Webentwicklung.</p>
<h2>Das möchte ich lernen</h2>
<p>Ich möchte eigene Webseiten erstellen. Heute beginne ich mit HTML.</p>
```

Name	Englische Bedeutung	Verwendung heute
h1	heading level 1	Eine Hauptüberschrift für die Seite

Name	Englische Bedeutung	Verwendung heute
h2	heading level 2	Überschrift eines Abschnitts unterhalb der Hauptüberschrift
p	paragraph	Ein zusammengehöriger Textabsatz

HTML kennt die Überschriftenebenen h1 bis h6. Die Zahl bezeichnet die Gliederungsebene. Wähle sie nach der inhaltlichen Struktur, nicht danach, welche Schriftgröße dir gefällt.

Ein Zeilenumbruch beim Schreiben des Quelltexts erzeugt normalerweise keinen neuen Absatz im Browser. Für zwei Absätze verwendest du zwei p-Elemente.

4.2 Ein Link

Ein **Hyperlink**, kurz Link, verweist auf ein Ziel, beispielsweise eine andere Webseite.

```
<a href="https://www.htlpinkafeld.at/">HTL Pinkafeld</a>
```

• a steht für *anchor* („Anker“) und kennzeichnet hier einen Link.

• href steht für *hypertext reference* und enthält die Zieladresse.

• HTL Pinkafeld ist der sichtbare und anklickbare Linktext.

Der End-Tag lautet . Das Attribut steht nur im Start-Tag. Für den Aufruf dieser externen Seite brauchst du eine Internetverbindung; deine eigene lokale Webseite lässt sich auch ohne Internet anzeigen.

4.3 Ein Bild

```
<img src="bilder/lernort.svg"
     alt="Symbolbild eines Schulgebäudes"
     width="360">
```

img steht für *image* („Bild“). Es ist ein leeres Element ohne End-Tag. Drei Attribute bestimmen unser Beispiel:

Attribut	Englische Bedeutung	Aufgabe
src	source	Pfad zur Bilddatei
alt	alternative text	Textalternative, etwa für Vorleseprogramme oder wenn das Bild nicht geladen wird
width	width	Darstellungsbreite, hier 360 CSS-Pixel

Der Wert von width wird hier als Zahl ohne px geschrieben. Die Bilddatei selbst wird dadurch nicht verkleinert. Ohne weitere Vorgaben passt der Browser die Höhe entsprechend dem Seitenverhältnis an.

Die Textalternative beschreibt den für den Zusammenhang wichtigen Bildinhalt. Unser mitgeliefertes Bild ist ein Symbolbild eines Schulgebäudes, kein Foto der HTL. Es ist daher passend, dies auch im Text auszudrücken.

Ausprobieren: Ändere width="360" auf width="240". Speichere und lade neu. Was verändert sich? Stelle anschließend wieder 360 ein.

4.4 Was bedeutet der Bildpfad?

Der **relative Pfad** `bilder/lernort.svg` startet hier im Ordner der HTML-Datei:

1. Gehe in den Unterordner `bilder`.

2. Verwende dort die Datei `lernort.svg`.

Schreibe Pfade in HTML mit Schrägstrichen `/`. Dateiname, Endung und Groß-/Kleinschreibung müssen zur tatsächlichen Datei passen. Gib später den gesamten Projektordner weiter, damit das Bild mitkommt.

Eine vollständige Webadresse wie `https://www.htlpinkafeld.at/` ist dagegen eine absolute Adresse. Relative Links auf eine zweite eigene HTML-Datei probierst du in der Zusatzaufgabe aus.

4.5 Die wichtigsten HTML-Tags im Überblick

Diese Übersicht hilft dir beim Nachschlagen während der Praxis. Die englischen Bezeichnungen erklären, wofür die Tag-Namen stehen.

Tag-Name	Bezeichnung	Beschreibung mit Beispiel
`html`	HTML-Wurzelelement	Umschließt das gesamte Dokument. Beispiel: `<html lang="de">…</html>`
`head`	Head – Dokumentkopf	Enthält Informationen über das Dokument, beispielsweise Zeichensatz und Titel.
`meta`	Metadata – Metadaten	Gibt zusätzliche Informationen an. Beispiel: `<meta charset="UTF-8">`. Kein End-Tag.
`title`	Title – Dokumenttitel	Bestimmt den Text im Browser-Tab. Beispiel: `<title>Meine Webseite</title>`
`body`	Body – Dokumentinhalt	Enthält die Inhalte der Webseite, etwa Überschriften, Absätze und Bilder.
`h1`	Heading Level 1 – Hauptüberschrift	Kennzeichnet die wichtigste Überschrift der Seite. Beispiel: `<h1>Pinkafeld entdecken</h1>`
`h2`	Heading Level 2 – Unterüberschrift	Kennzeichnet einen Abschnitt unterhalb der Hauptüberschrift. Beispiel: `<h2>Mein Lernort</h2>`
`p`	Paragraph – Absatz	Kennzeichnet einen Textabsatz. Browser stellen Absätze normalerweise mit Abstand dar. Beispiel: `<p>Das ist ein Absatz.</p>`
`a`	Anchor – Anker/Link	Erstellt mit `href` einen Link. Beispiel: `<a href="https://www.htlpinkafeld.at/">Unsere Schule</a>`
`img`	Image – Bild	Bindet ein Bild ein. Beispiel: `<img src="bilder/lernort.svg" alt="Schulgebäude" width="360">`. Kein End-Tag.

Nicht verwechseln: `<!DOCTYPE html>` ist eine Dokumenttyp-Deklaration und kein HTML-Tag. `lang`, `charset`, `href`, `src`, `alt` und `width` sind Attribute, keine Tags.

5. Praxis: Meine erste Webseite

5.1 Dein Auftrag

Erstelle eine kleine Webseite „Pinkafeld entdecken“. Nutze den vorbereiteten Projektordner und das Grundgerüst. Alle Texte schreibst du selbst auf Deutsch; eine Recherche ist nicht nötig. CSS und JavaScript brauchst du heute noch nicht.

5.2 Teil A: Struktur und Inhalt

- Speichere deine Seite als `index.html`.
- Behalte das erklärte Grundgerüst einschließlich `lang="de"` und UTF-8 bei.
- Gib dem Browser-Tab einen passenden Titel.
- Verwende eine `h1` als Hauptüberschrift.
- Ergänze die Abschnitte „Mein Lernort“ und „Das möchte ich lernen“, jeweils mit einer `h2` und einem `p`.
- Schreibe pro Absatz mindestens zwei eigene kurze Sätze.
- Führe eine Änderung mit dem Ablauf **Ändern → Speichern → Browser neu laden** vor.

5.3 Teil B: Link und Bild

- Ergänze einen Link auf `https://www.htlpinkafeld.at/` mit dem Linktext „HTL Pinkafeld“.
- Binde `bilder/lernort.svg` mit passender Textalternative und `width="360"` ein.
- Prüfe die Darstellung. Teste den Link, wenn Internet verfügbar ist; andernfalls kontrolliere die Zieladresse im Code.

5.4 Partnerprüfung

Öffnet die Seite aus dem Projektordner und prüft gemeinsam:

☐ Browser-Titel und Hauptüberschrift stehen an den richtigen Stellen.

☐ Zwei Zwischenüberschriften und zwei persönliche Absätze sind vorhanden.

☐ Der Link besitzt das richtige Ziel und verständlichen Text.

☐ Das Bild wird angezeigt; `alt` und `width` sind sinnvoll gesetzt.

☐ Die Autorin oder der Autor erklärt Start-Tag und End-Tag an einem Absatz.

☐ Die Autorin oder der Autor erklärt `src` und führt eine gespeicherte Textänderung im Browser vor.

Gebt einen konkreten Verbesserungshinweis und überarbeitet anschließend eure eigene Seite.

5.5 Fehler gezielt suchen

Beobachtung	**Das prüfst du zuerst**
Eine Änderung fehlt.	Ist gespeichert? Wurde die richtige Datei im richtigen Browser-Tab neu geladen?
Das Bild fehlt.	Stimmen `src`, Unterordner, Dateiname und Endung?

Beobachtung	Das prüfst du zuerst
Der Link funktioniert nicht.	Stimmen `href` und Zieladresse? Ist für das externe Ziel Internet verfügbar?
Text steht im falschen Bereich.	Steht der Inhalt im `body`? Sind Elemente richtig verschachtelt?
Umlaute werden falsch dargestellt.	Ist UTF-8 angegeben und wurde die Datei als UTF-8 gespeichert?

5.6 Zusatzaufgabe

Erstelle freiwillig `mehr.html` im selben Ordner mit einem vollständigen Grundgerüst. Verlinke von `index.html` mit `href="mehr.html"` dorthin. Ergänze auf der zweiten Seite einen Rücklink mit `href="index.html"`. Prüfe beide Richtungen.

5.7 Wissenscheck ohne Geräte

1. Welche Aufgaben haben Editor und Browser?
2. Markiere Start-Tag, Inhalt und End-Tag: `<p>Hallo Pinkafeld!</p>`.
3. Wo erscheinen `title` und `h1`?
4. Ergänze den Attributnamen: `<a ____="https://www.htlpinkafeld.at/">Schule</a>`.
5. Welche Aufgaben haben `src`, `alt` und `width`? Hat `img` einen End-Tag?
6. Warum genügt es bei unserer Seite nicht, nur `index.html` weiterzugeben?

5.8 Abgabe und Fortsetzung

Gib den gesamten Projektordner als `Nachname_Vorname_HTML01.zip` im von der Lehrperson genannten Lernsystem ab. Er enthält `index.html` und den Unterordner `bilder` mit `lernort.svg`; bei der Zusatzaufgabe auch `mehr.html`. Ein Screenshot ersetzt diese Dateien nicht.

Bis Montag, 21.09.2026: Verbessere die Pflichtseite anhand des Feedbacks und bereite dich darauf vor, drei eigene Codezeilen zu erklären. Arbeite höchstens 15 Minuten. Stoppe danach und notiere offene Fragen. Die zweite Seite bleibt freiwillig.

6. Listen und Tabellen

Geplant: Geordnete und ungeordnete Listen; Tabellen für tabellarische Daten mit Überschriften und sinnvoller Struktur. Tabellen werden nicht als Werkzeug für das Seitenlayout eingesetzt.

7. Semantische Seitenstruktur

Geplant: Inhalte mit `header`, `nav`, `main`, `section`, `article` und `footer` sinnvoll gliedern; Bedeutung gegenüber bloßem Aussehen unterscheiden; zugängliche Struktur und nachvollziehbare Überschriftenhierarchie.

8. HTML-Formulare

Geplant: `form`, `label`, `input`, `textarea`, `select` und `button`; Eingabetypen, Zuordnung von Beschriftungen, `name`, `value`, Pflichtfelder und grundlegende HTML-Prüfung. Formularziel und Übertragungsart einordnen. Für die clientseitige Auswertung wechseln wir später zu Kapitel 21.