Webprogrammierung
Teil 1: Html

Theorie

Requests im Browser

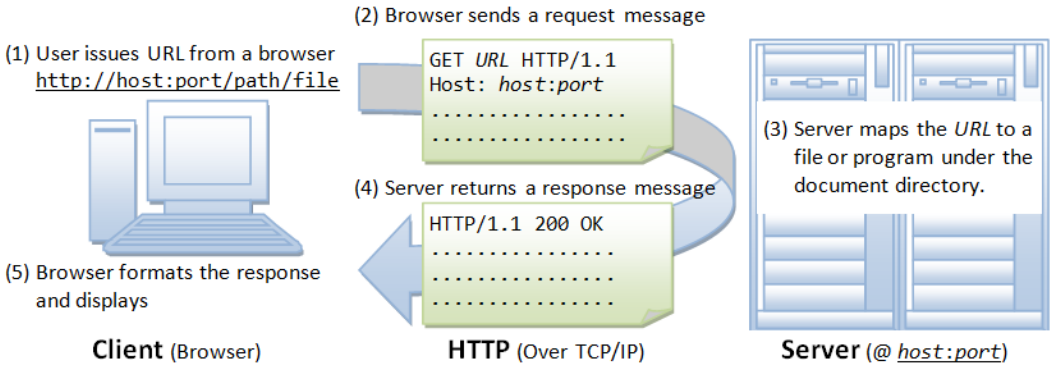
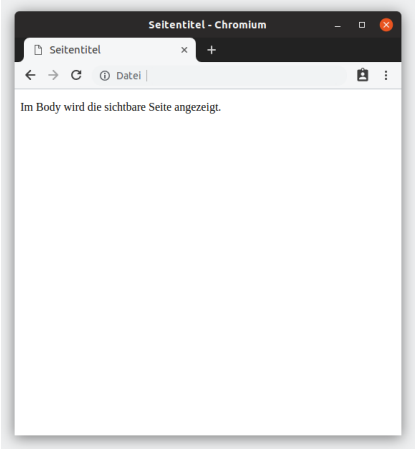


Abbildung 1: Ablauf von Requests im Browser, `https://www.ntu.edu.sg/home/ehchua/programming/webprogramming/images/HTTP_Steps.png`

Html

Minimalbeispiel Html

```
<!DOCTYPE html>
<html>
<head>
    <title>Seitentitel</title>
</head>
<body>
<!-- Ein Kommentar -->
<p>Im Body wird die sichtbare Seite
    angezeigt.</p>
</body>
</html>
```



Html

- HyperText Markup Language
- Auszeichnungssprache für Hypertext (Text, der mit Querverweisen auf andere Texte verweist)
- Verschiedene Versionen, unter anderem
  - `HTML 4.01` (Lange Zeit Standard, bis 2014)
  - `XHTML` (Verwendet XML-Syntax)
  - Aktuell: `HTML5` (Basis 4.01, Überarbeitung und Erweiterung)
- Semantische Strukturierung eines Textes, keine Formatierung
- Trennung von Design und Layout

Aufbau eines Dokuments

- Angabe Dokumenttyp:
  `<!DOCTYPE html>`
- Wurzelelement `<html>`
  - Seitenkop `<head>`
  - Seitenrumpf `<body>`

```
<!DOCTYPE html>
<html>
<head>
    <!-- Seitenkopf -->
</head>
<body>
    <!-- Seitenrumpf -->
</body>
</html>
```

Seitenkopf: <head>

Der Seitenkopf ist im Browser (mit der Ausnahme: Titel der Seite) nicht sichtbar (das heißt, er wird nicht gerendert). Er gibt allgemeine Informationen zum Dokument an wie

- Titel des Dokuments
- Sprache des Dokuments
- Codierung (UTF-8, ISO, ...)
- Andere Metainformation, zum Beispiel für Suchmaschinen
- Verlinkung CSS in `<link>`- oder `<style>`-Tag
- Verlinkung/Definition Javascript in `<script>`-Tag

Beispiel: Seitenkopf

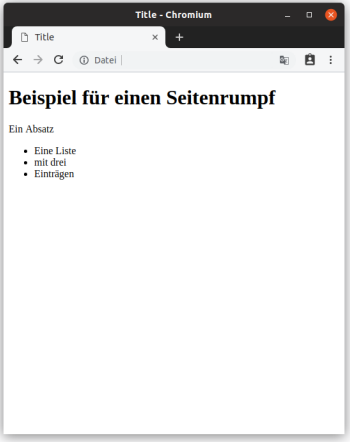
```
<head>
    <!-- Angabe des Seitentitels -->
    <title>Hier steht der Seitentitel</title>

    <!-- Verlinkung eines CSS-Dokuments -->
    <link rel="stylesheet" href="mystyle.css">

    <!-- Angabe von Metadaten, u. a. für Suchmaschinen -->
    <meta charset="UTF-8">
    <meta name="description" content="Beispiel Seitenkopf">

    <!-- Verlinkung eines JS-Dokuments -->
    <script type="text/javascript" src="script.js"></script>
</head>
```

Seitenrumpf: <body>

- Seitenkörper ist alles innerhalb des <body>-Tags
- Seitenrumpf ist die eigentliche Seite, die sichtbar ist
- Durch Verschachtelung von Elementen kann (wie bei XML, siehe später) komplexe Hierarchie erzeugt werden
- Durch Einsatz von CSS kann ein modernes Layout erreicht werden

Beispiel: Seitenrumpf

```
<body>
<h1>Beispiel f&uuml;r einen
    Seitenrumpf</h1>
<p>Ein Absatz</p>
<ul>
    <li>Eine Liste</li>
    <li>mit drei</li>
    <li>Eintr&auml;gen</li>
</ul>
</body>
```



Tags in Html

Tags werden in HTML in spitzen Klammern erstellt und bestehen (normalerweise) aus einem öffnenden Tag (mit den Attributen) und einem schließenden Tag und dem Taginhalt:

<*tagname attribute*>*inhalt*</*tagname*>

In HTML sind alle möglichen Tags vorgegeben. Laut Spezifikation müssen nicht alle Tags geschlossen werden, es empfiehlt sich aber aufgrund besserer Lesbarkeit und schnellerer Übersicht.

Einige Tags bringen Funktionalität mit, zum Beipsiel

- Zeilenumbrüche
- vergrößerter Text
- maschinenlesbare Informationen
- ...

Attribute in Html

Die möglichen Attribute in HTML sind vordefiniert, alles andere wird vom Browser ignoriert.

Es gibt Universalattribute, die für alle Elemente (im Body) möglich sind:

- `id`: Gibt dem Element eine eindeutige ID, kann bei CSS benutzt werden
- `class`: Wird für CSS benutzt
- `hidden`: Element wird ausgeblendet
- `style`: Inline-Definition von CSS (sollte nur in Ausnahmefällen benutzt werden)
- `title`: Beschreibung des Elements, wird normalerweise bei „Mousehover“ angezeigt

Besprechung der jeweils relevanten Attribute später im Kontext der Elemente

Umlaute und Sonderzeichen

Umlaute wie ä, ö und ü und andere Sonderzeichen sind im ASCII-Zeichensatz nicht vorhanden.

Daher sollten diese Zeichen in einer „Escapesequenz“ in HTML eingegeben werden:

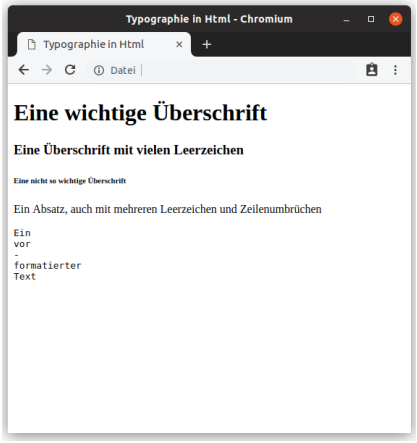
Sonderzeichen	Schreibweise	Sonderzeichen	Schreibweise
Ä / ä	`&Auml; / &auml;`	&	`&`
Ö / ö	`&Ouml; / &auml;`	€	`&euro;`
Ü / ü	`&Uuml; / &uuml;`	®	`&reg;`
™	`&trade;`	©	`&copy;`
< / >	`< / >`	§	`&seg;`
ß	`&szlig;`	Anführungszeichen	`&quot;`

Überschriften und Absätze

- Mit den Tags `<h1>`, `<h2>`, ..., `<h6>` („Heading“) kann eine Überschrift erstellt werden
- Mit dem Tag <p> („Paragraph“) wird ein Absatz erstellt
- Überschriften und Absätze ignorieren mehrere Leerzeichen und Zeilenumbrüche, daher müssen vorformatierte Texte mit dem Tag `<pre>` („Pre-Formatted“) erstellt werden

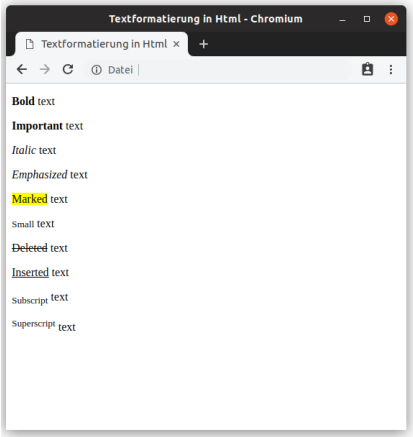
Beispiele für Typographie

```
<h1>Eine wichtige &Uuml;berschrift</h1>
<h3>Eine      &Uuml;berschrift
    mit      vielen    Leerzeichen</h3>
<h6>Eine nicht so wichtige
    &Uuml;berschrift</h6>
<p>Ein Absatz, auch mit mehreren
    Leerzeichen
    und Zeilenumbr&uuml;chen</p>
<pre>Ein
vor
-
formatierter
Text</pre>
```



Textformatierung

```
<p><b>Bold</b> text</p>
<p><strong>Important</strong> text</p>
<p><i>Italic</i> text</p>
<p><em>Emphasized</em> text</p>
<p><mark>Marked</mark> text</p>
<p><small>Small</small> text</p>
<p><del>Deleted</del> text</p>
<p><ins>Inserted</ins> text</p>
<p><sub>Subscript</sub> text</p>
<p><sup>Superscript</sup> text</p>
```



Hyperlinks

Links auf (andere) Seiten werden mit dem <a>-Tag erstellt. Der Linkinhalt kann Text, ein Bild oder ein beliebiges Element sein.

```
<a href="http://www.w3schools.com/html/"
   target="_blank">Ein HTML-Tutorial</a>
```

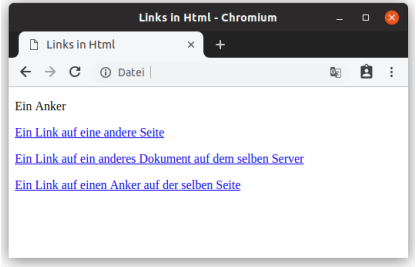
Links kennen zwei wichtige Attribute:

- `href` gibt die zu öffnende URL an
- `target` gibt (optional) an, in welchem Tab der Link geöffnet wird
  - `_blank`: Neues Fenster / neuer Tab
  - `_self`: Selbes Fenster / Tab (Standard)

Hyperlinks: href

Gibt das Ziel des Links an. Mögliche Werte sind:

- Andere Internetseite
- Anderes Dokument auf selben Server
- Anker auf selber Seite




```
<p id="myBookmark">Ein Anker</p>
<p><a href="https://de.wikipedia.org/wiki/Hyperlink">Ein Link
    auf eine andere Seite</a></p>
<p><a href="ein/anderes/document.html">Ein Link auf ein
    anderes Dokument auf dem selben Server</a></p>
<p><a href="#myBookmark">Ein Link auf einen Anker auf der
    selben Seite</a></p>
```

Bilder

Bilder werden mit dem Tag **<img>** eingebungen

- Attribut **src** gibt Speicherpfad des Bildes an (funktioniert ähnlich wie **href** bei Links)
- Breite **width** und Höhe **height** geben Maße des Bildes an, verhindern „Flackern“ der Seite während des Ladens
- Attribut **alt** wird angezeigt, falls das Bild nicht existiert und daher nicht angezeigt werden kann

```
<img src="pfad/zu/bild.jpg" width="200" height="100"
     alt="Alternative Beschreibung, falls Bild nicht vorhanden">
```

Beispiel: Bilder

```
<!-- Bild auf externem Server -->
<img src="https://www.uni-wuerzburg.de/fileadmin/images/11neu/header_li
     width="164" height="77" alt="Logo Universität Würzburg">

<!-- Bild in selbem Server -->
<img src="images/seal_uni_wue.png" width="220" height="220"
     alt="Logo Universität Würzburg">

<!-- Bild als Link -->
<a href="https://www.uni-wuerzburg.de/startseite/">
    <img src="images/logo_uni_wue.png" width="220" height="88"
         alt="Logo Universität Würzburg">
</a>
```

Aufgabe: Hyperlinks und Bilder

Erstellen Sie eine HTML-Seite mit einem Bild, das gleichzeitig ein Link ist.

Als URL können Sie zum Beispiel einen passenden Wikipedia-Eintrag verwenden.

Zeitrahmen: 10 Minuten

Listen

Listen können durch die beiden Tags `<ul>` und `<ol>` erzeugt werden, einzelne Listenelementen per `<li>`

- `<ul>` erzeugt eine „ungeordente“ Liste
- `<ol>` eine „geordnete“ (nummerierte) Liste

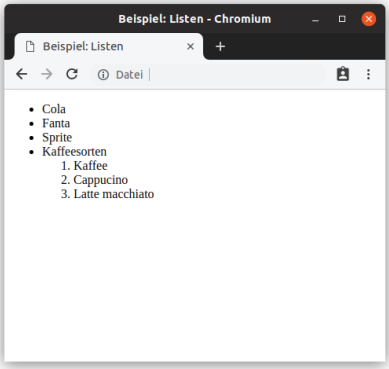
„Unterlisten“ können durch Verschachtelung erzeugt werden

Über das Attribut `type` kann der Aufzählungstyp von *nummerierten* Listen beeinflusst werden:

- „1“ für Zahlen (Standard),
- „a“ und „A“ für Klein- und Großbuchstaben,
- „i“ und „I“ für klein- und großgeschriebene römische Zahlen

Beispiel: Listen

```
<ul>
    <li>Cola</li>
    <li>Fanta</li>
    <li>Sprite</li>
    <li>Kaffeesorten
        <ol>
            <li>Kaffee</li>
            <li>Cappucino</li>
            <li>Latte macchiato</li>
        </ol>
    </li>
</ul>
```



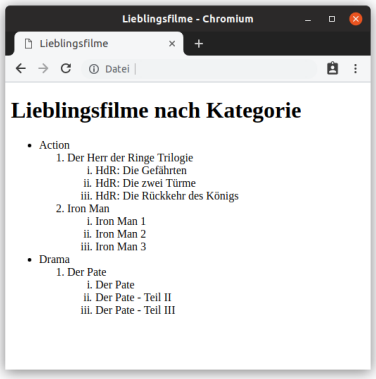
Aufgabe: Listen

Erstellen Sie eine Liste Ihrer Lieblingsfilme, geordnet nach

- Kategorie (Action, Komödie, ...) und
- Film

Beispiel: siehe rechts

Zeitrahmen: circa 10 min



Beispiellösung: Listen

```
<h1>Lieblingsfilme nach Kategorie</h1>
<ul>
    <li>Action
        <ol>
            <li>Der Herr der Ringe Trilogie
                <ol type="i">
                    <li>HdR: Die Gef&auml;hrten</li><li>HdR: Die zwei T&uuml;rm
                    <li>HdR: Die Rückkehr des Königs</li>
                </ol>
            </li><!-- Ende Der Herr der Ringe Trilogie -->
            <li>Iron Man
                <ol type="i">
                    <li>Iron Man 1</li><li>Iron Man 2</li><li>Iron Man 3</li>
                </ol>
            </li><!-- Ende Iron Man Trilogie -->
        </ol>
    </li><!-- Ende Action -->
```

Beispiellösung: Listen

```
    <li>Drama
        <ol>
            <li>Der Pate
                <ol type="i">
                    <li>Der Pate</li>
                    <li>Der Pate - Teil II</li>
                    <li>Der Pate - Teil III</li>
                </ol>
            </li><!-- Ende Der Pate Trilogie -->
        </ol>
    </li><!-- Ende Drama -->
</ul>
```

Tabellen

- Werden durch das Tag `<table>` eingeleitet
- Optional: Nutzen von `<thead>`, `<tbody>` und `<tfoot>` zur Angabe von Kopf, Rumpf und „Abschluss“ der Tabelle (wichtig bei größeren Tabellen)
- Einteilung der Tabelle in Zeilen und Zellen
  - Definition einer Zeile mit `<tr>`
  - Definition einer Zelle mit `<td>` und `<th>` („Heading“ für Überschriftszellen, bewirkt normalerweise fette Schrift)
- Hinzufügen eines Rahmens mit Attribut `border="1"` theoretisch möglich, entspricht aber nicht neuestem Standard, Design mit CSS vorzunehmen

Beispiel: Tabellen

```
<table>
  <thead>
    <tr>
      <th>Mannschaft</th>
      <th>Siege</th>
      <th>Niederlagen</th>
      <th>Unentschieden</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Seattle Seahawks</td>
      <td>7</td><td>2</td><td>1</td>
    </tr>
    <tr>
      <td>Arizona Cardinals</td>
      <td>4</td><td>5</td><td>1</td>
    </tr>
    <tr>
      <td>Los Angeles Rams</td>
      <td>4</td><td>6</td><td>0</td>
    </tr>
    <tr>
      <td>San Francisco 49ers</td>
      <td>1</td><td>9</td><td>0</td>
    </tr>
  </tbody>
</table>
```

Aufgabe: Tabellen

Erstellen Sie eine Tabelle, die den Tabellenstand einer Sportliga darstellt!

Zeitrahmen: 10 Minuten

Block versus Inline

Blocklevel-Element
Starten grundsätzlich auf einer neuen Zeile und nehmen die komplette Breite in Anspruch.

Beispiele:

- `<div>`
- `<h1> bis <h6>`
- `<p>`

Inline-Element
Starten **nicht** auf einer neuen Zeile und nehmen nur so viel Breite in Anspruch, wie nötig.

Beispiele:

- `<span>`
- `<a>`
- `<img>`

Warum Formulare?

Alle bisher vorgestellten Elemente sind statisch und können keine Eingaben des Nutzers verarbeiten.

Heutige Webseiten erfordern aber viel Interaktion:

- Registrierung
- Login
- Suchen auf den Seiten
- ...

Daher: Erfassen von Nutzereingaben mit Formularen

Definition von Formularen

Formulare werden mit dem Tag `<form>` definiert. Dies besitzt zwei Hauptattribute:

- `action`: Gibt die URL an, an die die Daten des Formulars gesendet werden
- `method`: Bestimmit, wie die Daten gesendet werden, zwei Möglichkeiten:
  - `get`: Daten werden an die URL angehängt (Standard)
  - `post`: Daten werden im Rumpf des Requests gesendet (meist bevorzugt)

Alles innerhalb des Formulars gehört zum Formular.

Labels mit `<label>`

Mit dem Tag `<label>` kann ein Label für ein Eingabeelement erstellt werden.

Diese haben kein besonderes Aussehen, können jedoch mit dem Attribut `for` einem Input zugeordnet werden.

Dabei bezieht man sich auf das Attribut `id`.

```
<label for="my_input">Ihre Eingabe:</label>
<input id="my_input">
```

Klickt man nun mit der Maus auf das Label, wird das referenzierte Eingabeelement fokussiert oder ausgewählt (je nach Typ).

Eingabeelement

Mögliche Elemente:

- `<input>`: Einzeiliges (Text-) Eingabeelement
- `<textarea>`: Mehrzeiliges Texteingabeelement
- `<select>`: Dropdown-Liste
- `<button>`: Klickbarer Knopf

Der Zugriff auf die Werte der einzelnen Element erfolgt über das `name`-Attribut.

Eingabe mit `<input>`

- „Universelles“ einzeiliges Eingabeelement
- Normalerweise Texteingabe
- konfigurierbar über Attribut `type`

Wichtige Attribute:

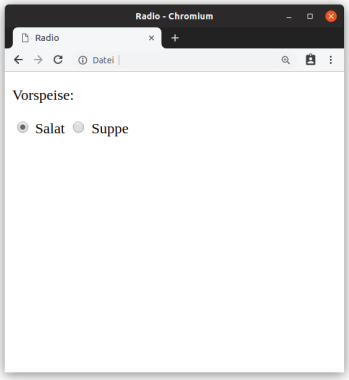
- `name`: Definiert den Namen, mit dem serverseitig nach Absenden auf die Daten zugegriffen werden kann
- `value`: Ändert Wert, der an den Server gesendet wird (Radiobutton, Checkbox) oder Anzeige
- `required`: Falls angegeben, prüft der Browser vor dem Absenden eines Formulars, ob ein entsprechender Wert eingegeben wurde und verhindert eventuell das Absenden Formulars

Inputs: Werte für das Attribut `type`

Attributwert	Funktion
`text`	Standard, keine weiteren Funktionen
`email`	Überprüfung, ob Eingabe eine Emailadresse sein kann (nicht perfekt!)
`password`	Anzeige von Sternen oder Kreisen anstatt eigentlicher Eingabe
`number`	(Ganz-)Zahlenfeld
`submit`	Knopf, der das Formular absendet
`reset`	Knopf, der das Formular zurücksetzt
`radio`	Einfache Auswahl
`checkbox`	Mehrfachauswahl
Viele weitere in HTML5 (`color`, `date`, `search`, `range`, ...)	

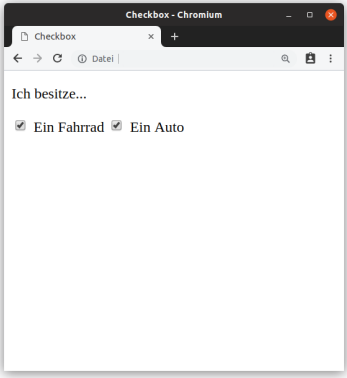
Einfachauswahl mit **radio**

```
<!-- Einfachauswahl -->
<p>Vorspeise:</p>
<label for="salat">
    <input type="radio" name="starter"
           id="salat"> Salat
</label>
<label for="suppe">
    <input type="radio" name="starter"
           id="suppe"> Suppe
</label>
</body>
```



Mehrfachauswahl mit **checkbox**

```
<!-- Mehrfachauswahl -->
<p>Ich besitze...</p>
<label for="ovbi">
    <input type="checkbox" id="ovbi"
           name="owned_vehicle_bike">
    Ein Fahrrad
</label>
<label for="ovc">
    <input type="checkbox" id="ovc"
           name="owned_vehicle_car">
    Ein Auto
</label>
```



Dropdowns mit `<select>`

Dropdowns werden mit dem Tag `<select>` angelegt.

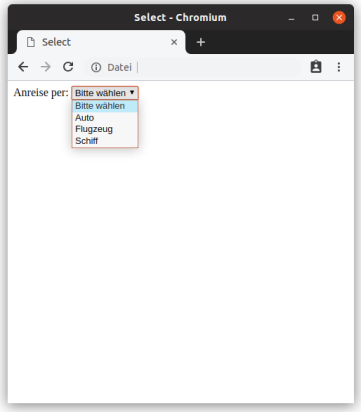
Die wählbaren Optionen werden in `<option>`-Tags angegeben.

- Das Attribut `value` sendet einen einfacheren Wert an den Server (zum Beispiel eine Zahl)
- Das Attribut `selected` trifft eine Vorauswahl

Falls die Auswahl einer Option mit dem Attribut `required` erzwungen werden soll, muss das erste Element einen leeren Wert haben.

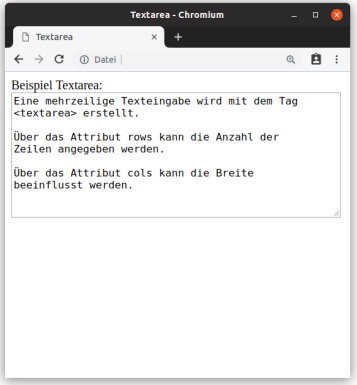
Beispiel: <select>

```
<label for="arrival_method">
    Anreise per:
</label>
<select name="arrival_method"
        id="arrival_method" required>
    <option value="">Bitte w&auml;hlen</option>
    <option value="car">Auto</option>
    <option value="plane">Flugzeug</option>
    <option value="ship">Schiff</option>
</select>
```



Texteingaben mit `textarea`

```
<label for="my_ta">Beispiel Textarea:</label>
<textarea rows="10" cols="50" id="my_ta">
Eine mehrzeilige Texteingabe wird mit dem Tag
&lt;textarea&gt; erstellt.

Über das Attribut rows kann die Anzahl der
Zeilen angegeben werden.

Über das Attribut cols kann die Breite
beeinflusst werden.
</textarea>
```



Beispiel: Formular (1)

```
<form action="/newUser" method="post">
    <div>
        <label for="name">Name:</label>
        <input name="name" id="name" required>
    </div>
    <div>
        <label for="email">Email:</label>
        <input type="email" name="email" id="email" required>
    </div>
    <div>
        <label for="passwort">Passwort:</label>
        <input type="password" name="passwort" id="passwort" required>
    </div>
```

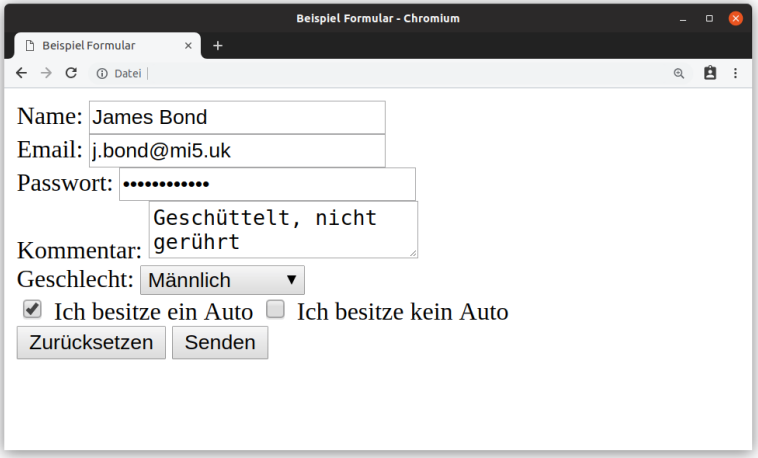
Beispiel: Formular (2)

```
<div>
    <label for="comment">Kommentar:</label>
    <textarea name="comment" id="comment" placeholder="Ihr Kommentar...">
    </textarea>
</div>
<div>
    <label for="geschl">Geschlecht:</label>
    <select name="geschl" id="geschl" required>
        <option value="" selected>Bitte w&auml;hlen...</option>
        <option value="m">M&auml;nnlich</option>
        <option value="f">Weiblich</option>
        <option value="d">Divers</option>
    </select>
</div>
```

Beispiel: Formular (3)

```
    <div>
        <label for="owns_car">
            <input type="checkbox" name="owns_car" id="owns_car">
            Ich besitze ein Auto
        </label>
        <label for="not_owns_car">
            <input type="checkbox" name="owns_car" id="not_owns_car">
            Ich besitze kein Auto
        </label>
    </div>
    <div>
        <input type="reset">
        <input type="submit">
    </div>
</form>
```

Beispiel: Formular (Browser, ausgefüllt)



Aufgabe: Formulare

Bearbeiten Sie auf `it4all` die Aufgabe 4 im Bereich Web.

Zeitrahmen: 15 Minuten

HTML5

Im Jahr 2014 wurde mit HTML5 eine neue Version von HTML veröffentlicht und HTML 4.01 abgelöst.

Dieser Standard besteht aus

- Vereinfachungen (zum Beispiel Angabe des Dokumenttyps)
- und Erweiterungen:
  - Neue Attribute für Eingabeelemente
  - Semantische Elemente
  - Grafikelemente
  - Multimediaelemente

Semantische Elemente

Bis HTML 4 wurde vor allem `<div>`s benutzt, um eine Seite einzuteilen, dies war jedoch schwer maschinenlesbar

HTML 5 führt neue semantische Elemente ein, um eine Seite (auch für Maschinen) besser strukturieren zu können

- `header`, `footer`: Kopf- und Fussabschnitt eines Dokuments
- `article`: Definiert einen Artikel im Dokument
- `section`: Abschnitt in einem Dokument
- `nav`: Navigationsbereich der Seite
- ...

Diese bieten keine neue Funktionsweise (eigentlich nur andere Namen für), sind aber besser maschinenlesbar (Crawling, Site Reader).

Grafik in HTML5

HTML 5 definiert zwei Möglichkeiten, Grafiken auf einer Webseite zu zeichnen:

- `<canvas>`:
  - Container, um mit Javascript Grafiken zu zeichnen
  - Abhängig von der Auflösung
  - Nutzung für grafikintensive Darstellungen (Spiele)
- `<svg>`:
  - „Scalable Vector Graphics“
  - XML-basierte Sprache, kann direkt in HTML eingebunden werden
  - Auflösungsunabhängig

Multimedia-Elemente

Um Audio- oder Videodateien in HTML einzubinden, war früher die Nutzung eines externen Tools (z. B. Adobe Flash Player) notwendig.

Die Entwicklung mancher externen Tools war teilweise chaotisch, die Folge waren viele Fehler und Sicherheitslücken („5 Bugs per Week“ im Adobe Flash Player, siehe Artikel von 2011).

Daher wurden in HTML5 spezielle Elemente für Audio und Video eingeführt. Diese werden um Beispiel bereits auf Youtube genutzt.

Audio und Video

Definition jeweils über `<video>` beziehungsweise `<audio>`

- Attribute width und height für Video wie bei `<img>`
- Attribut controls aktiviert (browsereigene) Kontrollschaltflächen für Play/Pause, Lautstärke, Fortschritt, Vollbild, ...
- Angabe der Quelldatei in Kindtag `<source>`
- Mehrfache Angabe von `<source>` für verschiedene Dateitypen möglich, Browser wählt passenden Dateitypen automatisch aus

Beispiel: Audio und Video

```
<audio controls>
    <!-- Angabe verschiedener Versionen der Audiodatei, Browser wählt passende
    <source src="horse.ogg" type="audio/ogg">
    <source src="horse.mp3" type="audio/mpeg">
    <!-- Text, falls Browser kein Audio unterstützt -->
    Your browser does not support the audio tag.
</audio>
<video controls width="320" height="240">
    <!-- Angabe verschiedener Versionen des Videos, Browser wählt passendes aus
    <source src="movie.mp4" type="video/mp4">
    <source src="movie.ogg" type="video/ogg">
    <!-- Text, falls Browser kein Video unterstützt -->
    Your browser does not support the video tag.
</video>
```

Aufgabe: Audio und Video

Bearbeiten Sie auf `it4all` die Aufgabe 5 im Bereich Web.

Zeitrahmen: 15 Minuten