



# Webprogramming Intro

---

# Agenda

Organisatorisches

Inhalt der Vorlesung

Grundlagen der Webprogrammierung

Grundsätzliche Prinzipien in der Entwicklung

Best Practises in der Entwicklung

Tech Stacks

Server / Client Prinzip

APIs

Deployment

Setup

---

# Organisatorisches



# Organisatorisches

- 2 Dozenten
  - Daniel Münch, SAP
  - Tobias Joschko, MongoDB
- 11 Termine
- Bewertung:
  - Klausur muss mindestens 50% der Note ausmachen
  - Die anderen 50% werden von Hausaufgaben bzw. Labor kommen
- GitHub repository für Code Beispiele und Übungen:  
<https://github.com/TEL22AT/webprogrammierung>
- Classroom Invite: <https://classroom.github.com/classrooms/193832587-tel22at-classroom>
- Kommunikation: Discord (Invite: <https://discord.gg/5jt4e6XYXM>)

---

# Inhalt der Vorlesung



# Inhalt der Vorlesung

- Grundlagen zu HTML + CSS
- JavaScript Einführung
- Moderne Web App Entwicklung mit Frontend Frameworks (Vue + Quasar)
- Backend (NodeJS + ExpressJS) + API Grundlagen
- Datenbanken (NoSQL vs SQL)
- Authentifizierung
- Deployment + GitHub
- (Grundlagen zu Tests)

---

# Grundlagen der Webprogrammierung



# Webentwicklung

1

**Webdesign:** Das visuelle Design einer Website, einschließlich Layout, Farbschema und Benutzererfahrung.

2

**Web-Publishing:** Das Veröffentlichen von Inhalten auf Website, oft durch Content-Management-Systeme.

3

**Web-Programmierung:** Die Entwicklung von Webanwendungen und Funktionalitäten, die auf Server- und Clientseite laufen.

4

**Datenbankverwaltung:** Das Organisieren und Verwalten von Daten, die für das Funktionieren einer Website benötigt werden, oft unter Verwendung von Datenbanksystemen wie MySQL oder MongoDB.





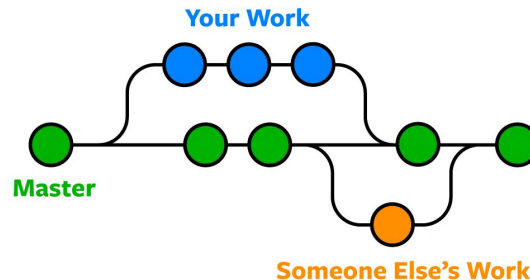
# Beispiel

E-Commerce-Website

Eine E-Commerce-Website beinhaltet  
**Webdesign** für die Benutzeroberfläche,  
**Web-Publishing** für  
Produktbeschreibungen,  
**Web-Programmierung** für  
Warenkorb-Funktionalitäten und  
**Datenbankverwaltung** für die Speicherung  
von Kundendaten.

# Grundlegende Entwicklungsprinzipien

- **Agile Entwicklung:** Iterative Entwicklung, die Flexibilität und Kundeneinbindung betont.
- **Responsive Design:** Ansatz zur Webentwicklung, der sich an verschiedene Bildschirmgrößen anpasst.
- **Code-Wiederverwendung und -Modularität:** Nutzung von Modulen oder Komponenten, um Code effizienter zu gestalten.
- **Versionierung:** Einsatz von Tools wie Git zur Verwaltung verschiedener Versionen des Codes.





# Best Practices in der Entwicklung

- **Clean Code:** Klare und verständliche Code-Schreibweise.
- **Testgetriebene Entwicklung (TDD):** Erst Tests schreiben, dann Code.
- **Sicherheit:** Implementierung von Sicherheits Praktiken, um Angriffe und Datenlecks zu verhindern.
- **Performance-Optimierung:** Techniken zur Beschleunigung der Ladegeschwindigkeit und Effizienz von Webseiten.

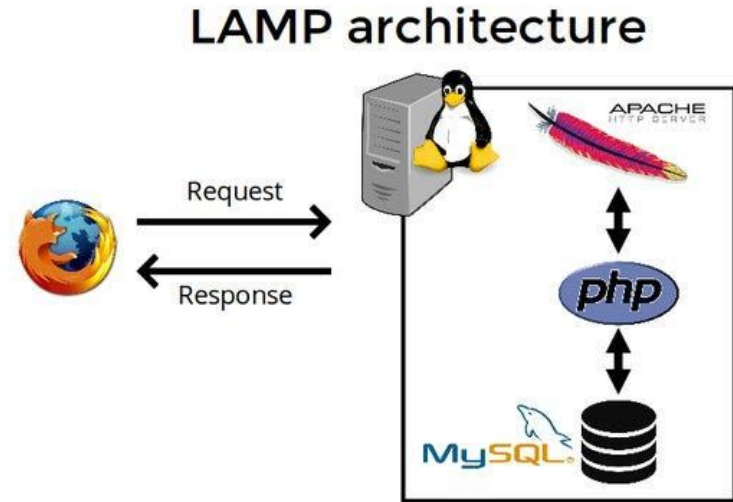
# LAMP Stack

Linux: Das Betriebssystem.

Apache: Der Webserver.

MySQL: Das Datenbanksystem.

PHP/Perl/Python: Programmiersprachen



# MEVN Stack



**MongoDB:** Die Datenbank.

**Express.js:** Das Backend-Framework.

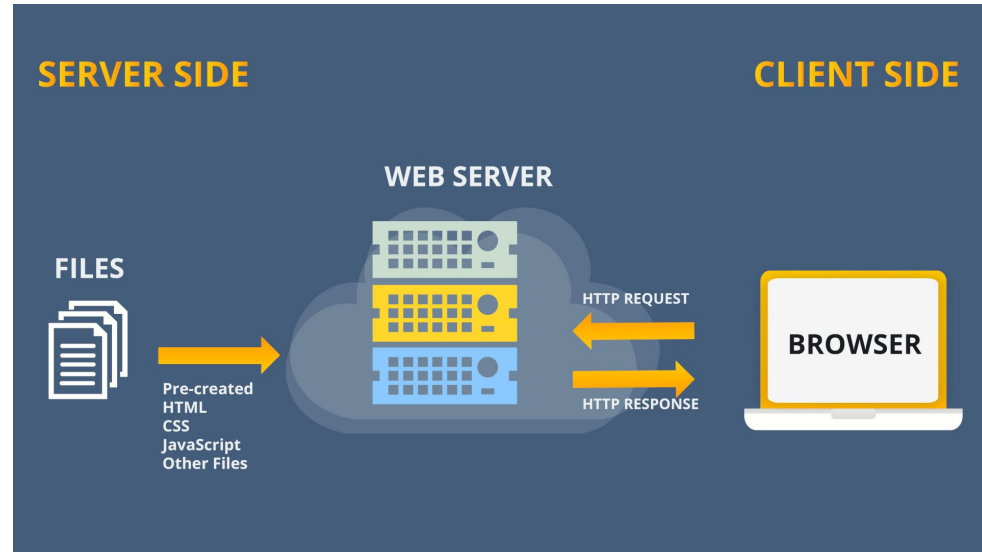
**Vue.js:** Das Frontend-Framework.

**Node.js:** Die Laufzeitumgebung.



# Server vs Client side

- Serverseitige Programmierung:
  - Erstellung von Logik, die auf dem Server läuft.
  - Benutzung unterschiedlicher Web Frameworks (z.B. Django, Express) und Verwendung verschiedener Programmiersprachen (z.B. Python, Javascript, PHP, Java, etc).
  - Datenbank-Interaktion des Servers.
- Frontend-Entwicklung:
  - Entwicklung von Benutzeroberflächen.
  - HTML/CSS/Javascript
  - Frameworks wie Vue, React oder Angular.





# Grundlagen von APIs

- Definition: APIs (Application Programming Interfaces - Programmierschnittstellen) als Mittler zwischen verschiedenen Softwareanwendungen.
- REST (API): Ein populärer API-Standard, der das HTTP Protokoll nutzt (z.B. GET, POST, DELETE).

```
{
  "original_title": "Interstellar",
  "budget": 165000000,
  "genres": [
    { "id": 12, "name": "Adventure" },
    { "id": 18, "name": "Drama" },
    { "id": 878, "name": "Science Fiction" }
  ],
  "homepage": "http://www.interstellarmovie.net/",
  "id": 157336,
  "overview": "The adventures of a group of
explorers who ...",
  "poster_path": "/gEU2QniE6E77NI6lCU6MxlNBvIx.jpg"
}
```



# Nutzung von APIs

**Als Datengrundlage für eigene Websites:**

Beispiel Webshop:

- Produktinformationen liegen in einer Datenbank
- Backend REST API Service läuft, um Daten abzurufen oder zu ändern
- Website nutzt die REST API, um Produkte abzurufen

**Als Integration Dritter:**

Einbeziehung von Diensten wie Google Maps ,Twitter oder Instagram über APIs.





# Deployment

- Deployment Prozess:
  - Continuous Integration/Continuous Deployment (CI/CD): Automatisierung des Deployments.
  - Staging vs. Production: Unterschiede zwischen Test- und Produktionsumgebungen.
- Hosting-Optionen:
  - Shared Hosting (STRATO, 1&1 Ionos, etc.)
  - Virtual Private Server (simuliert “dedicated” Server mit Partitionierung)
  - Cloud Hosting (AWS, Azure, GCP, etc...)
  - Dedicated Hosting

---

# Setup



# Setup

IDE	<a href="#"><u>Visual Studio Code</u></a>
Laufzeitumgebung	<a href="#"><u>NodeJS</u></a>
Paketverwaltung	<a href="#"><u>npm</u></a> (included in node) + <a href="#"><u>yarn</u></a> (also included in NodeJS via corepack)
JavaScript Framework	<a href="#"><u>Vue</u></a>
Frontend Framework	<a href="#"><u>Quasar</u></a>
Versionierung	<a href="#"><u>GIT</u></a>



# HTML

---

# Agenda

Geschichte und Entstehung von HTML

HTML Grundlagen

HTML Elemente + Übungen

---

# Geschichte und Entstehung von HTML



# SGML

- Standard Generalized Markup Language
- Ende der 80er erfunden
- Standard für Auszeichnungssprachen
- Trennung von Inhalt + Layout
- Inhalt wird durch Elemente strukturiert
- DTDs (Document Type Definitions) beschreiben Regeln und Inhalte von Dokumenten

```
<!DOCTYPE NEWSPAPER [  
  
  <!ELEMENT NEWSPAPER (ARTICLE+)>  
  <!ELEMENT ARTICLE (HEADLINE,BYLINE,LEAD,BODY,NOTES)>  
  <!ELEMENT HEADLINE (#PCDATA)>  
  <!ELEMENT BYLINE (#PCDATA)>  
  <!ELEMENT LEAD (#PCDATA)>  
  <!ELEMENT BODY (#PCDATA)>  
  <!ELEMENT NOTES (#PCDATA)>  
  
  <!ATTLIST ARTICLE AUTHOR CDATA #REQUIRED>  
  <!ATTLIST ARTICLE EDITOR CDATA #IMPLIED>  
  <!ATTLIST ARTICLE DATE CDATA #IMPLIED>  
  <!ATTLIST ARTICLE EDITION CDATA #IMPLIED>  
  
  <!ENTITY NEWSPAPER "Vervet Logic Times">  
  <!ENTITY PUBLISHER "Vervet Logic Press">  
  <!ENTITY COPYRIGHT "Copyright 1998 Vervet Logic Press">  
  
]>
```



# HTML

- HyperText Markup Language
- Ursprünglich Anwendung von SGML (HTML5 NICHT!!!) → Auszeichnungssprache
- Wird vom W3C und WHATWG weiterentwickelt
- Aktuelle Version seit 2017 HTML 5.2 (wird von vielen Browsern unterstützt)
- Dient zur semantischen Strukturierung von Inhalten, NICHT zur Formatierung
- Grund für Entwicklung: Dokumente auf elektronischem Weg zwischen mehreren Personen austauschen und miteinander verknüpfen
- <https://html.spec.whatwg.org/>





# XML

XML (Extensible Markup Language)

- Ende der 90er erfunden
- Anwendung von SGML
- Metasprache → ermöglicht die Entwicklung weiterer Sprachen durch Erweiterung
- Plattformunabhängiger Austausch von Daten zwischen Computersystemen
- Mit XML können neue Sprachen entwickelt werden, indem DTDs oder XML Schemas erstellt werden
- Durch XML Schemas können auch Inhalte eingeschränkt werden
- Ohne Schema oder DTD ist die Struktur frei wählbar (nur Syntax muss stimmen)



# XML Schema Beispiel

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="shiporder">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="orderperson" type="xs:string"/>
        <xs:element name="shipto">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="address" type="xs:string"/>
              <xs:element name="city" type="xs:string"/>
              <xs:element name="country" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="item" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="title" type="xs:string"/>
              <xs:element name="note" type="xs:string" minOccurs="0"/>
              <xs:element name="quantity" type="xs:positiveInteger"/>
              <xs:element name="price" type="xs:decimal"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="orderid" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>

</xs:schema>
```



# XHTML

- Extensible Hypertext Markup Language
- XML Sprache
- Idee: XML Syntax fuer HTML nutzen und weitere XML Sprachen einbinden

Probleme:

- Zu wenig XML Kenntnisse von Autoren → kaum XHTML Websites
- Späte Unterstützung in Browsern
- → Wurde 2009 zugunsten HTML5 eingestellt

---

# HTML Grundlagen



# HTML Struktur

```
<!DOCTYPE html>
<!-- This is a comment -->
<html>
  <head>
    <title>HTML5 Example Page</title>
  </head>
  <body>
    <p>This is some content</p>
  </body>
</html>
```



# HTML head Elemente

title	Title der Website
meta	Metadaten der Website (z.B. Autor, Keywords, Beschreibung)
base	Basis URL der Website
link	Verlinkung der Website zu anderen Ressourcen (z.B. Stylesheets)
script	Bindet code einer Skriptsprache ein
style	Formatierungs- und Stil Informationen



# HTML Entities

- Reservierte Zeichen müssen in HTML mit Entities ersetzt werden
- z.B. < oder >
- Beginnt mit & und endet mit ;
- Beispiele:
  - < = &lt;
  - > = &gt;



## Block Elemente vs. Inline Elemente

Block Elemente	Inline Elemente
Erzeugen neuen Absatz	Erzeugen KEINEN neuen Absatz
Erzeugen einen Abstand vor und nach dem Element	Erzeugen KEINEN Abstand vor und nach dem Element
Können Block und Inline Elemente beinhalten	Können nur Inline Elemente beinhalten
Nutzt volle verfügbare Breite links und rechts	Nur so breit wie nötig





# HTML Attribute

- Jedes Element kann Attribute haben
- Attribute konfigurieren das Element
- Wichtige Universalattribute für jedes Element:
  - id: eindeutiger Name zur Identifizierung des Elements
  - class: CSS Klassen, denen das Element angehört
  - style: Formatierung des Elements durch CSS Befehle
  - onclick: Event für das Anklicken eines Elements

---

# HTML Elemente



# HTML Elemente (Beispiele)

## Block

<h1>-<h6>	<p>	<div>	<ol>	<ul>	<dl>
<nav>	<table>	<section>	<article>	<aside>	

## Inline

<a>	 	<img>	<input>	<script>	<select>
<span>	<textarea>				



# Überschriften

`<h1>Das ist ein h1 Element.</h1>`

`<h2>Das ist ein h2 Element.</h2>`

`<h3>Das ist ein h3 Element.</h3>`

`<h4>Das ist ein h4 Element.</h4>`

`<h5>Das ist ein h5 Element.</h5>`

`<h6>Das ist ein h6 Element.</h6>`

**Das ist ein h1 Element.**

**Das ist ein h2 Element.**

**Das ist ein h3 Element.**

**Das ist ein h4 Element.**

**Das ist ein h5 Element.**

**Das ist ein h6 Element.**



# Absatz

`<p>Dies ist der erste Absatz.</p>`

`<p>Dies ist der zweite Absatz.</p>`

Dies ist der erste Absatz.

Dies ist der zweite Absatz.



# div

- Allgemeines Blockelement
- Kann mehrere andere Elemente beinhalten
- Dient als Container zur Formatierung
- Beliebig tief schachtelbar
- Keine sichtbare Auswirkung ausser:
- Ist ein Block Element → neue Zeile

**Dies ist ein h2 Element in einem div.**

Dies ist ein Absatz im gleichen div.

```
<div>  
  <h2>Dies ist ein h2 Element in einem div.</h2>  
  <p>Dies ist ein Absatz im gleichen div.</p>  
</div>
```



## span

- Inline Element
- Kann weitere **Inline** Elemente beinhalten
- Dient als Container zur Formatierung
- Keine sichtbare Auswirkung

Dies ist ein Absatz mit mehreren Spans

```
<p>  
    <span>Dies</span>  
    <span>ist</span>  
    <span>ein</span>  
    <span>Absatz</span>  
    <span>mit</span>  
    <span>mehreren</span>  
    <span>Spans</span>  
</p>
```



# Umbrüche

`<p>Dies ist ein Absatz und hier kommt ein Umbruch: <br>Und hier geht es im gleichen Absatz weiter. </p>`

`<p>Neuer Absatz. Hier kommt eine Trennlinie: </p>`

`<hr>`

`<p>Weiterer Absatz. </p>`

Dies ist ein Absatz und hier kommt ein Umbruch:  
Und hier geht es im gleichen Absatz weiter.

Neuer Absatz. Hier kommt eine Trennlinie:

---

Weiterer Absatz.





# HTML Textformatierung

<p>

<b>Dies ist ein fetter Text.</b> <br>

<i>Dies ist ein kursiver Text.</i> <br>

<u>Dies ist ein unterstrichener Text.</u> <br>

<s>Dies ist ein durchgestrichener Text.</s> <br>

Normal

<sub> Dies ist ein tiefgestellter Text</sub>

<sup> Dies ist ein hochgestellter Text</sup> <br>

<small>Dies ist ein kleiner Text.</small> <br>

<big>Dies ist ein großer Text.</big> <br>

<tt>Dies ist ein Schreibmaschinen-Text.</tt> <br>

<!-- nicht fuer HTML 5 supported! -->

<font size="3" face="Arial, Verdana">

Dies ist ein Text mit Größe 3 und Schriftart Arial, Verdana.

</font> <br>

**Dies ist ein fetter Text.**

*Dies ist ein kursiver Text.*

Dies ist ein unterstrichener Text.

~~Dies ist ein durchgestrichener Text.~~

Normal Dies ist ein tiefgestellter Text. Dies ist ein hochgestellter Text.

Dies ist ein kleiner Text.

**Dies ist ein großer Text.**

Dies ist ein Schreibmaschinen-Text.

Dies ist ein Text mit Größe 3 und Schriftart Arial, Verdana.

<p>

<code>Dies ist ein Code-Text. </code> <br>

<kbd>Dies ist ein Tastatur-Text. </kbd> <br>

<samp>Dies ist ein Beispiel-Text. </samp> <br>

<var>Dies ist ein Variablen-Text. </var> <br>

<cite>Dies ist ein Zitat-Text. </cite> <br>

<dfn>Dies ist ein Definition-Text. </dfn> <br>

<abbr title="HyperText Markup Language" >HTML</abbr> <br>

<acronym title="World Wide Web Consortium" >W3C</acronym> <br>

<address>Dies ist eine Adresse. </address> <br>

<blockquote>Dies ist ein Blockzitat. </blockquote> <br>

<q>Dies ist ein Zitat. </q> <br>

<pre>Dies ist ein vorgeformatierter Text. </pre>

<em>Dies ist ein betonter Text. </em> <br>

<strong>Dies ist ein starker Text. </strong> <br>

<mark>Dies ist ein markierter Text. </mark> <br>

</p>

Dies ist ein Code-Text.

Dies ist ein Tastatur-Text.

Dies ist ein Beispiel-Text.

*Dies ist ein Variablen-Text.*

*Dies ist ein Zitat-Text.*

*Dies ist ein Definition-Text.*

HTML

W3C

*Dies ist eine Adresse.*

Dies ist ein Blockzitat.

“Dies ist ein Zitat.”

Dies ist ein vorgeformatierter Text.

*Dies ist ein betonter Text.*

**Dies ist ein starker Text.**

**Dies ist ein markierter Text.**

# Grafik

```
</img>
```





# Übung 1: Grundaufbau

- Erstellen Sie eine Webseite mit den Grundelementen (html, head, title, body)
- Fügen Sie folgende Elemente ein:
  - Überschrift
  - Paragraph
  - Ein Image
  - Paragraph mit anderer Schriftart erstellen



# Links

- Können verweisen auf:
  - Andere Dateien im Webserver durch relative Pfadangaben
  - Andere Dateien im Webserver durch absolute Pfadangaben
  - URI
  - URL

```
<a href="https://google.com">Google</a>
```

```
<a href="../paragraph.html">Paragraph</a>
```

```
<a href="/index.html">Home</a>
```

```
<a href="mailto:tom.ate@company.com">Email us</a>
```

```
<a href="https://google.com" target="_blank">Google in new Tab</a>
```

[Google](#) [Paragraph](#) [Home](#) [Email us](#) [Google in new Tab](#)



# Listen

- List Element 1
- List Element 2
- List Element 3

- II. Ordered List Element 1
- V. Ordered List Element 2
- VI. Ordered List Element 3

- Definition Term 1
  - Definition Description 1
- Definition Term 2
  - Definition Description 2
- Definition Term 3
  - Definition Description 3

```
<!-- type can be circle|disk|square -->
<ul type="disc">
  <li>List Element 1</li>
  <li>List Element 2</li>
  <li>List Element 3</li>
</ul>

<!-- type can be 1|A|a|I|i -->
<ol type="I" start="2">
  <li>Ordered List Element 1</li>
  <li value="5">Ordered List Element 2</li>
  <li>Ordered List Element 3</li>
</ol>

<dl>
  <dt>Definition Term 1</dt>
  <dd>Definition Description 1</dd>
  <dt>Definition Term 2</dt>
  <dd>Definition Description 2</dd>
  <dt>Definition Term 3</dt>
  <dd>Definition Description 3</dd>
</dl>
```

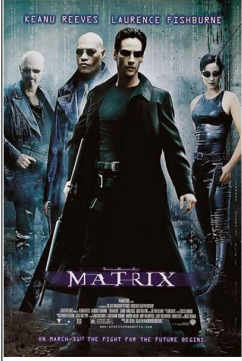
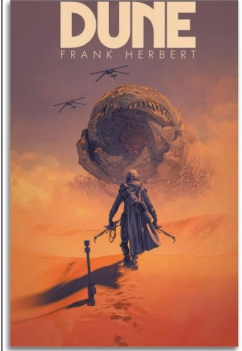
# Tabellen

Name	Quantity	Price
Name 1	3	3\$
Total	3	9\$

```
<table border="1">
  <caption>Table Caption</caption>
  <thead>
    <tr> <!-- table row -->
      <th>Name</th> <!-- table header cell -->
      <th>Quantity</th>
      <th>Price</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Name 1</td> <!-- table data cell -->
      <td>3</td>
      <td>3$</td>
    </tr>
  </tbody>
  <tfoot>
    <tr>
      <td>Total</td>
      <td>3</td>
      <td>9$</td>
    </tr>
  </tfoot>
</table>
```

## Übung 2

- Erstellen einer Top 3 Filmeliste
  - Rang, Titel, Bild und Streamingplattformen
  - Übersichtlich gestalten → Tabelle verwenden
  - Überschriften sollen sich vom Rest abheben → <th>
  - Streaming Platform Liste sollte eine Unordered List sein
  - Verlinkung des Filmnamens mit der dazugehörigen Wikipedia Seite

Rang	Bild	Titel	Streaming-Plattform
1		<a href="#">The Matrix</a>	<ul style="list-style-type: none"><li>• Netflix</li><li>• Amazon Prime</li><li>• Disney Plus</li></ul>
1		<a href="#">Dune</a>	<ul style="list-style-type: none"><li>• Netflix</li><li>• Amazon Prime</li><li>• Disney Plus</li></ul>





# Input (Textfeld)

```
<input name="firstname" type="text" placeholder="First name" required>  
<input name="lastname" type="text" placeholder="Last name" required>  
<input name="password" type="password" placeholder="Password" required>
```

# Input (Checkbox und Radiobutton)

```
<input type="radio" name="gender" value="male">Male  
<br>  
<input type="radio" name="gender" value="female">Female
```

☐ Male  
☐ Female

```
<input type="checkbox" name="packinglist" value="shampoo">Shampoo  
<br>  
<input type="checkbox" name="packinglist" value="toothbrush">Toothbrush  
<br>  
<input type="checkbox" name="packinglist" value="towel" checked>Towel  
<br>  
<input type="checkbox" name="packinglist" value="underwear">Underwear
```

☐ Shampoo  
☐ Toothbrush  
☒ Towel  
☐ Underwear



# Input (Verstecktes Feld)

```
<input type="hidden" name="hiddenfield" value="hiddenvalue">
```



## Input (Fileupload)

```
<input type="file" name="fileupload" accept="image/*">
```

Choose File No file chosen



## Input (Button) / Button

```
<input type="button" name="button" value="Button">  
<br>  
<button name="button" value="Button">ButtonText</button>
```



Button



ButtonText

# Select

```
<select name="color" size="1">
  <option>Blue</option>
  <option>Red</option>
  <option>Green</option>
</select>
<br><br><br><br><br>
<select name="color" size="3" multiple>
  <option>Blue</option>
  <option>Red</option>
  <option>Green</option>
</select>
```



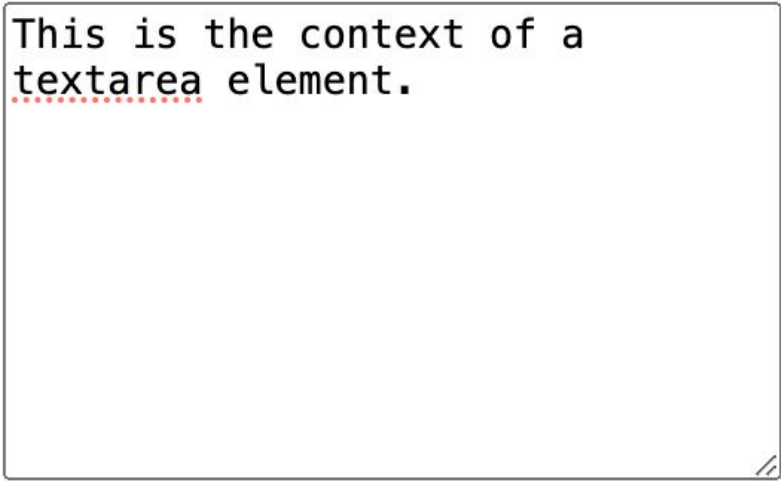


# Textarea

```
<textarea name="textarea" cols="30" rows="10">
```

```
This is the context of a textarea element.
```

```
</textarea>
```



This is the context of a  
textarea element.

# Formulare

```
<form action="mail.php" method="post">
  <label for="subject">Subject</label>
  <input type="text" placeholder="Your subject" id="subject" name="subject">
  <br>
  <label for="message">Message</label>
  <textarea placeholder="Your message" id="message"
name="message"></textarea>
  <br>
  <label for="confirmation">Confirmation</label>
  <input type="checkbox" id="confirmation" name="confirmation">
  <br>
  <input type="submit" value="Submit">
  <input type="reset" value="Reset">
</form>
```

Subject

Message

Confirmation ☐





## Übung 3

- Kontaktformular erstellen
  - Namensfeld → Required
  - Email Adressfeld → Required
  - Textarea für eine Nachricht
  - Submit und Reset Button