

HTML 5 Formulare erstellen und verarbeiten

Beispiel: www.billa.at

The screenshot shows the Billa website's contact form. At the top, there is a yellow navigation bar with the Billa logo and links for 'Vorteils-Club', 'Aktionen', 'Sortiment', 'BILLA Marken', and 'Nachhaltigkeit'. Below the navigation bar, the text 'sagt der Hausverstand.' is displayed. The main heading is 'Kontaktformular'. The form consists of several input fields and dropdown menus:

- Betreff***: A dropdown menu with the placeholder text 'Bitte auswählen'.
- Anrede***: A dropdown menu with the placeholder text 'Anrede'.
- Titel**: A dropdown menu with the placeholder text '---'.
- Vorname***: A text input field with the placeholder text 'Vorname'.
- Nachname***: A text input field with the placeholder text 'Nachname'.
- Straße**: A text input field with the placeholder text 'Straße'.
- PLZ**: A text input field with the placeholder text 'PLZ'.
- Ort**: A text input field with the placeholder text 'Ort'.
- E-Mail Adresse***: A text input field with the placeholder text 'E-Mail Adresse'.
- Vorwahl**: A dropdown menu with the placeholder text '---'.
- Rufnummer**: A text input field with the placeholder text 'Mobile Rufnummer'.
- Alter**: A text input field with the placeholder text 'Alter'.

Die Verarbeitung von Formularen erfolgt nach einem festen Prinzip. Man deklariert Datenfelder, in denen der Besucher seine Daten eintippt oder das gewünschte Objekt auswählt. Jedes Eingabefeld benötigt einen eindeutigen Namen, dem dann als Wert die Daten des Besuchers zugeordnet werden.

z.B. Namen der Datenfelder: vorname, nachname, notiz

Regeln

- wähle verständliche Namen für die Datenfelder
- Kleinschreibung verhindert Probleme bei der Verarbeitung
- Leerstellen, Umlaute und Sonderzeichen sind verboten
- Jeder Feldname darf nur einmal vorkommen

Nachdem der Besucher die Daten eingegeben hat, schickt er die Daten ab. Dazu dient meist ein Button. Für die Verarbeitung dieser Daten ist aber nicht HTML zuständig, sondern werden die Daten an den Webserver gesendet, wo im Hintergrund ein kleines Programm läuft. Das nennt man CGI (Common Gateway Interface). Das ist meist in PHP oder Perl geschrieben.

- Um CGIs nutzen zu können kann man fertige CGI-Archive aus dem Internet laden und diese in der Website integrieren. Die Auswahl ist groß und beginnt bei einfachen Gästebüchern bis hin zu ganzen Shopping-Systemen.
- Oder man verwendet ein System-CGI vom Provider. Dieser stellt oft kostenlos ein Basispaket zur Verfügung mit Kontaktprogramm, Newsletter usw.
- Daten per Mail: als Nothilfe, mit der sich Formulare auch ohne Gästebuch nutzen lassen werden die Daten eines Formulars in Form einer E-Mail verarbeitet. Dabei werden die Daten nicht verarbeitet oder ausgewertet, sondern im Klartext zugeschickt.

E-Mail Notlösung:

Gib die E-Mail Adresse im <action>-Feld ein.

```
<form action="mailto:josefus200@gmx.at"> </form>
```

Die **Verarbeitung** gilt nun wieder für beide Varianten: Sobald der Besucher die Daten des Formulars abschickt, werden diese verarbeitet. Die Möglichkeiten dafür werden mit dem Attribut <method> festgelegt, welches direkt im Form-Tag mit angegeben wird.

- **GET**
Dies ist die einfachste Methode. Dabei werden die Daten ohne Umwege direkt übergeben. Dies ist zwar besonders unkompliziert, bringt aber zwei Nachteile mit sich. Viele Browser akzeptieren dabei maximal 200 Zeichen für das gesamte Formular. Außerdem entsteht aus den Daten des Formulars ein Link, der recht kompliziert aussieht und im Browserfenster des Besuchers erscheint.
- **POST**
Damit lassen sich umfangreichere Daten absenden und verarbeiten.

Soll man die Daten mit der E-Mail Notlösung versenden, muss man die POST-Methode wählen. Nur damit übergibt der Browser die Felddaten korrekt an das E-Mail Programm.

```
<form action="mailto:josefus200@gmx.at" method="post">  
</form>
```

Einfache Eingabefelder erstellen

Eingabefelder werden immer mit dem Befehl <input> erzeugt. Das ergibt ein ganz einfaches, einzeiliges Textfeld, in das der Besucher etwas eintippen kann. Der Befehl muss nicht geöffnet und geschlossen werden, er steht für sich alleine.

Da man besser jedem Textfeld einen Namen geben sollte, stellt man diesen vor das jeweilige Feld:

```
<form...>  
  Name: <input>  
  E-Mail: <input>  
  Notiz: <input>  
</form>
```

Attribut <name>:

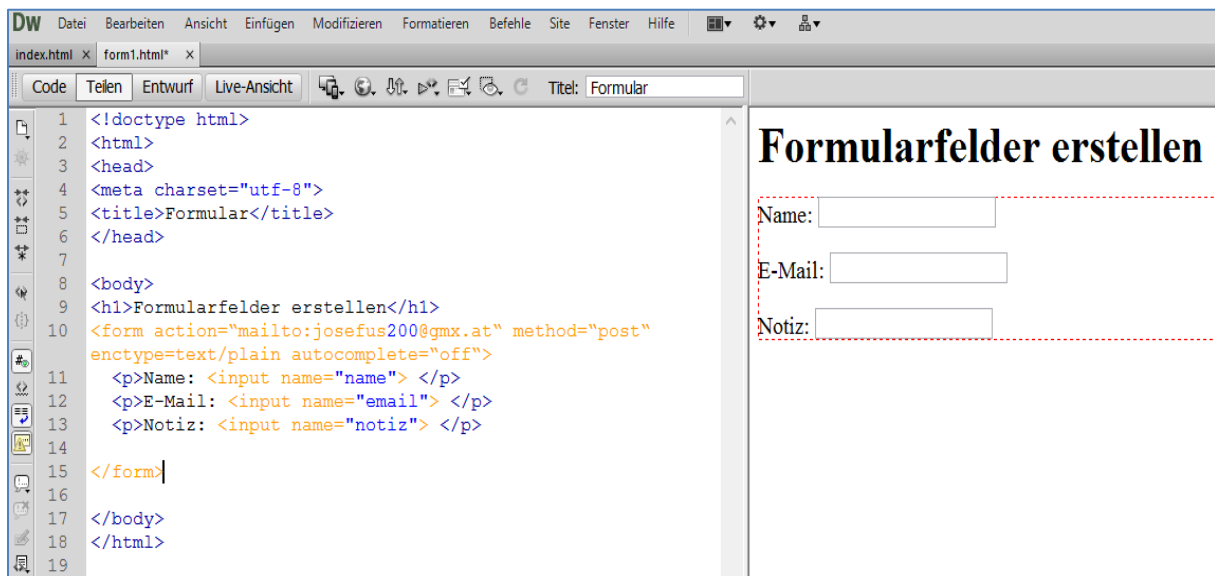
Der Name vor dem <input> stellt nur ein gestalterisches Mittel dar, da sie vom Browser angezeigt werden, aber keine technische Relevanz haben. Will man ein großes Formular erstellen, müssen die Felder eindeutig unterscheidbar sein, daher sollte man jedem <input>-Feld mit dem **Attribut <name>** einen eindeutigen Namen vergeben. Damit weiß der Webserver dann auch, unter welcher Bezeichnung er die übergebenden Daten speichern soll.

Zusätzlich sollte man jedem Element einen eigenen Absatz zukommen lassen, um es für den Browser übersichtlicher zu machen und die spätere Formatierung mit CSS zu erleichtern:

```
<form action="mailto:josefus200@gmx.at" method="post">
  <p>Vorname: <input name="vorname"></p>
  <p>Nachname: <input name="nachname"></p>
  <p>E-Mail: <input name="email"></p>
  <p>Notiz: <input name="notiz"></p>
</form>
```

Übung:

Erstelle obiges Beispiel in einem neuen HTML5-Dokument:



Schaltflächen für die Verarbeitung

Standardmäßig sollte jedes Formular zwei Schaltflächen aufweisen:

- Abschieken
es werden die Eingaben im Formular abgeschickt und von CGI verarbeitet. Hat man die E-Mail-Verarbeitung gewählt, wird die E-Mail mit den Daten erstellt.

<input type="submit">

- Abbrechen
das Formular wird zurückgesetzt und die Eingaben vollständig gelöscht.

<input type="reset">

Damit der Button die Beschriftung trägt, die man selbst will, sollte man eine eigene Beschriftung angeben:

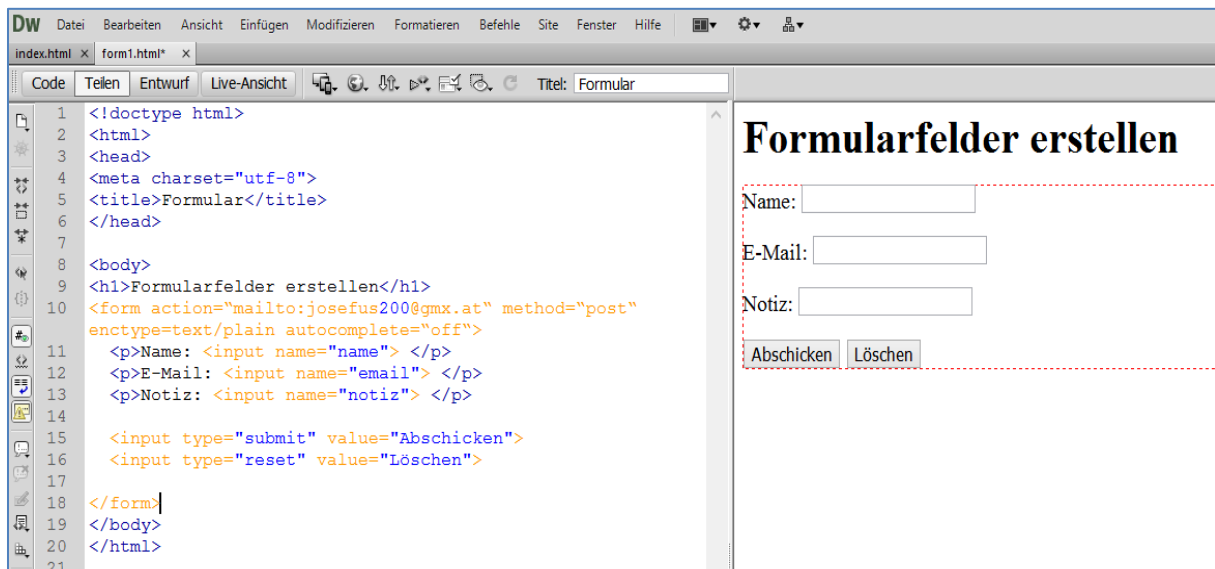
```
<input type="submit" value="Absenden">
```

bzw.

```
<input type="reset" value="Löschen">
```

Diese beiden Schaltflächen sind Pflicht bei jedem Formular und sie gehören zwingend in den Formularbereich und müssen zwischen den beiden <form>-Befehlen stehen. Die Beschriftung im Attribut <value> kann frei gewählt werden.

Übung: erstelle beide Buttons



Tipp: Man kann auch <label> verwenden

```
<form>
  <label for="name">Name:</label>
  <input name="name" id="name">
  <label for="notiz">Notiz:</label>
  <input name="notiz" id="notiz">
</form>
```

<label> markiert, dass der eingeschlossene Text die Beschriftung des Eingabefeldes ist. Dazu muss das „for“-Attribut des <label>s mit der „id“ des Eingabefeldes übereinstimmen. Das <label> kann „sein“ Eingabefeld auch umschließen, dann entfällt das „for“-Attribut, aber sonst ändert sich nichts. Man könnte <label> einfach weglassen und die Beschriftung des Eingabefeldes als Text ohne Tag schreiben, es sähe vollkommen gleich aus. Ein guter Grund für <label> ist aber die Barrierefreiheit: Lesegeräte können so klarmachen, welcher Text zu welchem Eingabefeld gehört. Aber auch im herkömmlichen Browser gibt es einen Vorteil: ein Klick auf das Label aktiviert das Eingabefeld, genauso als hätte man das Feld angeklickt. Das ist z.B. ein Vorteil auf dem Handy.

Bereiche und Gruppen definieren

Mit `<fieldset>` erstellt man einen Bereich, der sauber geöffnet und geschlossen werden muss. Alle Formularfelder, die innerhalb dieses Bereichs stehen, werden automatisch als logische Gruppe angesehen und verarbeitet. Innerhalb eines Formulars kann man beliebig viele `<fieldset>` definieren und jedes `<fieldset>` kann beliebig viele INPUT-Felder besitzen.

Damit man die `<fieldset>` (=Feldgruppen) besser unterscheiden kann, gibt es zusätzlich den Befehl `<legend>`. Damit gibt man den Gruppen Namen, die auch der Browser anzeigt.

```
8 <body>
9 <h1>Formularfelder erstellen</h1>
10 <form action="mailto:josefus200@gmx.at" method="post" enctype=text/plain
    autocomplete="off">
11 <fieldset>
12 <legend>Kontaktdaten</legend>
13
14 <p>Name: <input name="name"> </p>
15 <p>E-Mail: <input name="email"> </p>
16 <p>Notiz: <input name="notiz"> </p>
17
18 <input type="submit" value="Abschicken">
19 <input type="reset" value="Löschen">
20
21 </fieldset>
22 </form>
23 </body>
```

Ergebnis:

Formularfelder erstellen

Kontaktdaten

Name:

E-Mail:

Notiz:

Übung: Erstelle folgendes Formular – vergrößere das bestehende um die „Hobbys“

Formularfelder erstellen

Kontaktdaten

Name:

E-Mail:

Notiz:

Hobbys

im Winter:

im Sommer:

Lösung:

```
9 <h1>Formularfelder erstellen</h1>
10 <form action="mailto:josefus200@gmx.at" method="post" enctype=text/plain
    autocomplete="off">
11 <fieldset>
12 <legend>Kontaktdaten</legend>
13
14 <p>Name: <input name="name"> </p>
15 <p>E-Mail: <input name="email"> </p>
16 <p>Notiz: <input name="notiz"> </p>
17
18 </fieldset>
19
20 <br>
21
22 <fieldset>
23 <legend>Hobbys</legend>
24
25 <p>im Winter: <input name="winterhobby"> </p>
26 <p>im Sommer: <input name="sommerhobby"> </p>
27
28 <input type="submit" value="Abschicken">
29 <input type="reset" value="Löschen">
30
31 </fieldset>
32
33 </form>
34 </body>
```

Eingabefelder für Text erstellen

Auf Basis des `<input>`-Befehls lassen sich mit unterschiedlichen Attributen viele neue Felder definieren. Es gibt folgende Attribute: `accesskey`, `checked`, `disabled`, `maxlength`, `name`, `size`, `tabindex`, `type` und `value`.

Textfeld

Dabei handelt es sich um eine Zeile, in die der Besucher direkt etwas eintippen kann. Im Prinzip haben wir hier bereits diese Textfelder erzeugt, da mit dem Befehl `<input>` genau sowas erzeugt wird. Aber es ist sinnvoll, ab nun das Attribut hinzu zu schreiben:

```
<type="text">
```

Feldgröße:

Beim Definieren von Textfeldern hat man zunächst keinen Einfluss darauf, wie groß bzw. lang dieses Feld ist. Das kann jeder Browser etwas anders machen. Mit dem Attribut `<size>` gibt man die Größe des Eingabefeldes in Form von Stellen bzw. Buchstaben an.

Mit dem Attribut `<maxlength>` gibt man die maximale Anzahl von Zeichen an. Mehr kann dann nicht eingegeben werden.

```
<p>Name: <input name="name" type="text" size="30" maxlength="50"> </p>
```

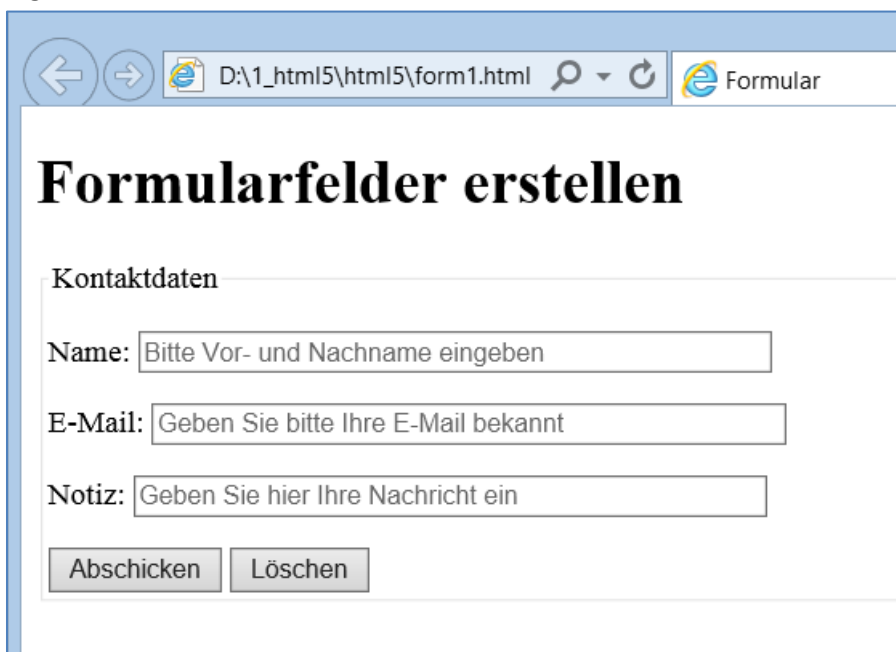
Beispiele oder Hilfstexte angeben

Diese Hilfetexte sollen dem Besucher Hilfe bieten, damit er das Formular besser ausfüllen kann.

Mit dem Attribut **<placeholder>** legt man den Wert fest, die der Browser im Eingabefeld einblenden soll. Allerdings wird dieser ausgeblendet, sobald der Besucher mit der Maus in das jeweilige Feld klickt. Die meisten Browser stellen diesen Platzhalter auch heller und in einer anderen Farbe dar.

```
<form action="mailto:josefus200@gmx.at" method="post"
  enctype=text/plain autocomplete="off">
  <p>Name: <input name="name" type="text" size="50"
    maxlength="50" placeholder="Bitte Vor- und Nachname
    eingeben"> </p>
  <p>E-Mail: <input name="email" type="text" size="50"
    maxlength="50" placeholder="Geben Sie bitte Ihre E-Mail
    bekannt"> </p>
  <p>Notiz: <input name="notiz" type="text" size="50"
    maxlength="200" placeholder="Geben Sie hier Ihre
    Nachricht ein"> </p>
</form>
```

Ergebnis:



The screenshot shows a web browser window with the address bar displaying 'D:\1_html5\html5\form1.html' and the page title 'Formular'. The main content area has a heading 'Formularfelder erstellen' and a section titled 'Kontaktdaten'. Below this section are three text input fields: 'Name: Bitte Vor- und Nachname eingeben', 'E-Mail: Geben Sie bitte Ihre E-Mail bekannt', and 'Notiz: Geben Sie hier Ihre Nachricht ein'. At the bottom of the form are two buttons: 'Abschicken' and 'Löschen'.

Werte vorgeben, sperren oder erzwingen

1)Vorgeben <value> (eher selten verwendet)

Der Unterschied zum <placeholder> besteht darin, dass die vorgegebenen Werte erhalten bleiben und als gültige Formularwerte akzeptiert werden. Sie verschwinden nicht aus dem Formular, wenn der Besucher das Feld mit der Maus anklickt. Der Besucher kann die Werte ganz einfach löschen, überschreiben, verändern oder ergänzen.

Man muss dafür das Attribut **<value>** verwenden. Lehrzeichen sind erlaubt.

```
<p>Name: <input name="name" type="text" size="50"
      maxlength="50" value="Vorname Nachname"> </p>
```

2)sperren <readonly>

Manchmal sollen Felder nicht verändert werden, wenn z.B. eine automatische Kundennummer angezeigt wird. Dafür können Felder den Zusatz **<readonly>** erhalten. Dann können die vordefinierten Werte nicht verändert werden. Allerdings zeigen die Browser diesen Nur-Lese-Status nicht gesondert an und die Felder sehen aus wie sonst auch. Das kann irritierend sein.

```
<p>Kunden-Nummer: <input name="kdnr" type="text" size="9"
      maxlength="9" value="33015" readonly> </p>
```

3)erzwingen <required>

Vorname*	Nachname*
<input type="text" value="Vorname"/>	<input type="text" value="Nachname"/>

Hier **muss der Besucher** ein bestimmtes Feld **ausfüllen**. Das ist z.B. sinnvoll bei Namens- oder Adressfeldern, damit man mit dem Besucher in Kontakt treten kann.

Meist trägt der Bereich ein Sternchen *. Das gib gleich nach dem Text einfach dazu:

```
<p>Vorname:*<input name="vorname" type="text" size="30"
      maxlength="40" value="Vorname" required> </p>
```

Allerdings gibt es auch Browser, die diese Anweisung missachten. Außerdem bedeutet dies ja nicht, dass der Besucher eine sinnvolle Angabe macht, er kann auch nur drei xxx einfügen.

Textbereich für längere Eingaben <textarea>

Um einen längeren Textbereich zu erzeugen verwendet man den Befehl <textarea>.

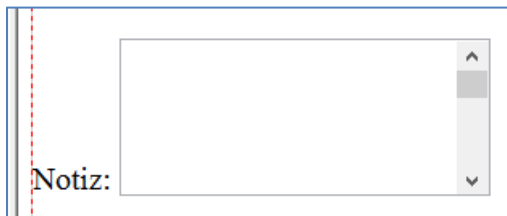
Die Größe lässt sich nicht über <size> erzeugen, sondern über die Attribute ROWS und COLS. Damit gibt man die gewünschten Zeilen und Spalten an.

Die meisten Browser zeigen den Textbereich dann mit Bildlaufleiste bzw. Scrollbalken an.

<p>Notiz: <textarea name="notiz" cols="30" rows="5" wrap="wrap"></textarea></p>

- Das Attribut rows bestimmt die Anzahl der Zeilen,
- cols die Anzahl der Spalten.
- wrap legt fest, dass die Zeilen umbrochen werden, wenn das Ende des Textfeldes erreicht ist.

Lösung:

A screenshot of a web browser showing a text area. The text area is labeled "Notiz:" and contains a scroll bar on the right side. The text area is empty.

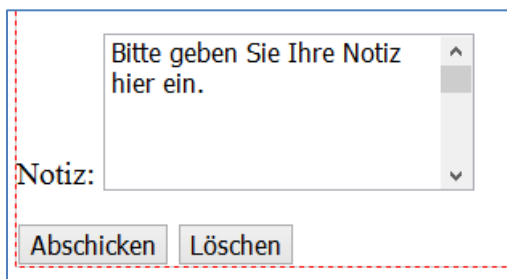
Im Textarea-Bereich kann man alle oben erwähnten Attribute verwenden, wie z.B. <maxheight>, <required> und <readonly>. Auch <placeholder> funktioniert in <textarea> !

Inhalt vorausfüllen:

Statt dem <input> erfolgt hier das Attribut <textarea>, das auch wieder geschlossen werden muss.

<p> Notiz: <textarea name="notiz" cols="30" rows="5" placeholder="Bitte geben Sie hier Ihre Notiz ein."> </textarea></p>

Ergebnis:

A screenshot of a web browser showing a text area. The text area is labeled "Notiz:" and contains the placeholder text "Bitte geben Sie Ihre Notiz hier ein." Below the text area are two buttons: "Abschicken" and "Löschen".

Spezielle Input-Typen für das Eingabefeld

Die Eingabe von Text bei mobilen Geräten kann sehr mühsam sein. Deshalb bieten Browser unterschiedliche Tastaturen an, je nach Eingabefeld. Muss zum Beispiel eine Telefonnummer eingegeben werden, so ergibt es keinen Sinn, dem Benutzer Buchstaben anzubieten.

Die Unterstützung solcher verschiedenen Tastaturen in Web-Apps ist Bestandteil von HTML5 und wird dort Web Forms 2.0 genannt. Die Input-Elemente wurden um das Attribut „type“ erweitert, z.B. für die Eingabe eines Datums:

```
<input type="date" name="date">
```

Weitere Typen findet man unter <http://wufoo.com/html5>

Richtige Tastatur

Dank den Formular-Input-Feldern in HTML5 wird bei Smartphones automatisch die richtige Tastatur ausgewählt. Es wäre sehr unangenehm, wenn man bei der Eingabe einer E-Mail Adresse erst umständlich nach dem @-Zeichen suchen müsste. Besser ist es, wenn die richtige Tastatur vorgeschlagen wird.

Beispiel:

- E-Mail:
<input type="email" name="mailadresse" id="mailadresse" />



- URL:
<input type="url" name="webadresse" id="webadresse" />



- Telefon:
<input type="tel" name="telefonnummer" id="telnummer" />



- o Datum:

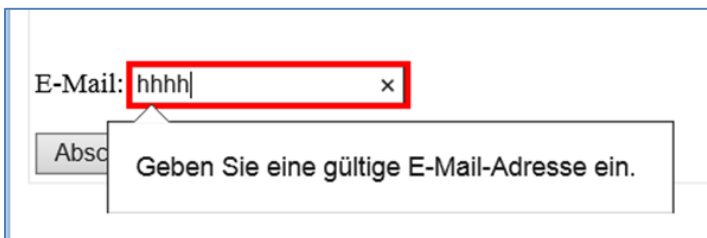


1) E-Mails überprüfen

In HTML5 gibt es den <input>-Typen namens **<email>**, der die eingegebene E-Mail Adresse (automatisch) auf Richtigkeit überprüft. Dabei übernimmt der Browser die ganz einfache Prüfung. Nämlich das Vorhandensein des @-Zeichens und zulässiger Zeichen und Ziffern davor und danach eine mögliche Domain. Das bedeutet nicht, dass die Adresse auch richtig ist und funktioniert.

```
<form...>
<p> E-Mail-Adresse: <input name="mail" type="email"></p>
</form>
```

Wenn keine korrekte E-Mail eingegeben wird, erscheint automatisch eine Fehlermeldung:



Das gleiche gilt für die Typen:

<type="tel"> hier wird Smartphones wird eine passende Tastatur eingeblendet

<type="url"> hier wird Smartphones wird eine passende Tastatur eingeblendet, die das @-Zeichen direkt anbietet.

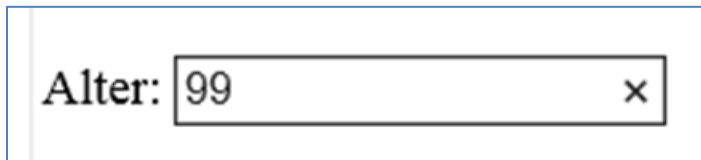
2) Felder für Nummern und Zahlen erstellen

Der Wert **<number>** macht aus einem `<input>`-Feld ein Zahlenfeld. Der Browser erzeugt automatisch Schaltflächen oder Scrollbalken, mit denen sich der gewünschte Wert einstellen lässt.

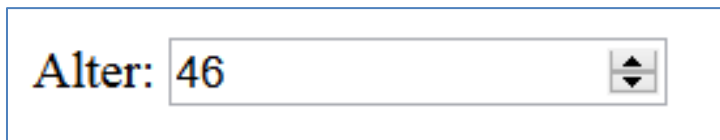
```
<form...>
<p> Alter: <input name="alter" type="number"></p>
</form>
```

Browser-Unterschiede:

IE erzeugt keinen Scrollbalken, sondern nur die Möglichkeit zum Löschen

A screenshot of an Internet Explorer browser window. It shows a text input field with the label "Alter:" and the value "99". To the right of the input field is a small square button with an "X" inside, used for clearing the field.

Firefox erzeugt die Schaltflächen zum weiterzählen:

A screenshot of a Firefox browser window. It shows a text input field with the label "Alter:" and the value "46". To the right of the input field is a spin button with up and down arrows, used for incrementing or decrementing the value.

3) Wertebereiche – Maximum, Minimum und Schritte

Damit wählt der Besucher stufenlos aus einer vorgegebenen Spanne aus. Mit dem Befehl `<range>` im `<input>`-Feld erzeugt der Browser einen Schieberegler.

```
<input name="zahl" type="range">
```

Dies erstellt einen inhaltslosen Schieberegler. Nun muss man noch angeben, welchen Bereich dieser Regler abdecken soll. Man muss einen minimalen und einen maximalen Wert angeben. Negative Werte soll man mit einem Minuszeichen angeben, positive Werte dürfen aber kein Pluszeichen besitzen.

```
<input name="zahl" type="range" min="-10" max="10">
```

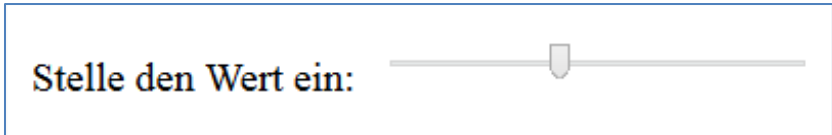
Die Größe der Schritte wird durch das Attribut `<step>` angegeben. Man kann hier ganze Zahlen oder auch Kommastellen mit einem Punkt angeben.

```
<input name="zahl" type="range" min="-10" max="10" step="1">
```

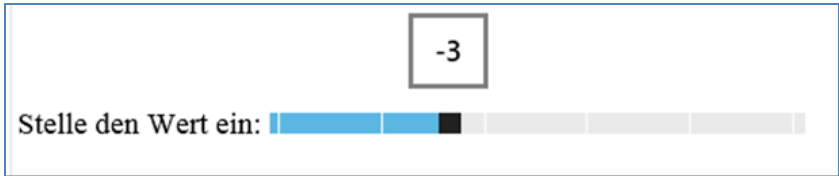
Zum Schluss muss man noch angeben, auf welchen Wert der Schieberegler in der Ausgangsposition stehen soll. Das tut man mit dem bekannten Wert `<value>`.

```
<input name="zahl" type="range" min="-10" max="10" step="1" value="0">
```

Ergebnis im Firefox



Der IE zeigt diesen Typ besser an, da er auch die Zahl anzeigt, wenn man den Regler schiebt

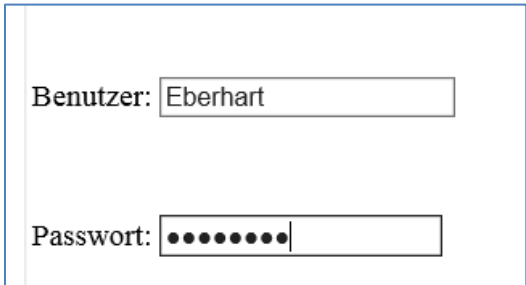


Leider wird nur ein leerer Regler von den Browser angezeigt, ohne Wertanzeige. Daher eignet er sich maximal um zwischen den Wörtern „gut“ und „schlecht“ eine sinnvolle Anwendung zu finden.

4)Passwörter ausblenden

Hier heißt der Typ <password>. Damit kann der Nachbar die Eingabe nicht sehen.

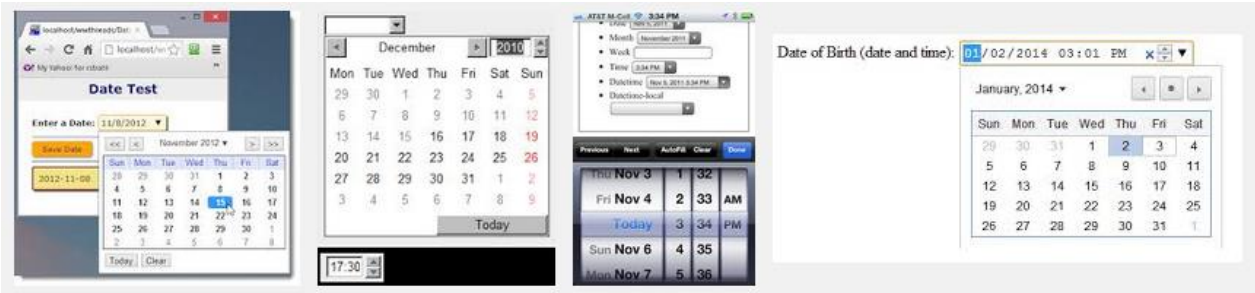
```
<p>Benutzername: <input name="user" type="text"></p>
<p>Passwort: <input name="pass" type="password"></p>
```



5)date, time (z.B. type="date")

In Datumsfeldern erzeugen HTML5-fähige Geräte entsprechende Auswahlelemente, die Zeiteingaben auch in Wochen oder Monaten zulassen und entsprechende Überprüfung möglich machen.

Unterschiedliche Browser liefern unterschiedliche Ergebnisse:



In Google Chrome:

Geben Sie Ihre Formularein:

Kontaktdaten:

Vorname:

Nachname:

E-Mail:

Notiz:

Alter:

Benutzername:

Passwort:

Geburtsdatum:

Abschicken Löschen

Übung: Erstelle folgendes <fieldset>:

Ziel:

Bitte füllen Sie folgendes Formular aus

Benutzerdaten:

Vorname:*

Nachname:*

E-Mail:*

Nachricht:

max. 200 Zeichen

Terminwunsch:

Uhrzeit:

```
23 <form method="post" action="mailto:josefus200@gmx.at">
24 <fieldset>
25 <legend>Benutzerdaten:</legend>
26 <p>Vorname:* <input name="vorname" size="50" required></p>
27 <p>Nachname:* <input name="nachname" size="50" required></p>
28 <p>E-Mail:* <input name="email" size="50" type="email" required></p>
29 <p>Nachricht: </p>
30 <textarea name="notiz" cols="50" rows="10" maxlength="200" placeholder="max. 200 Zeichen"></textarea>
31 <p>Terminwunsch: <input name="datum" type="date"></p>
32 <p>Uhrzeit: <input name="zeit" type="time"></p>
33 </fieldset>
34 </form>
```

<p>Nachricht: </p>

```
<textarea name="notiz" cols="50" rows="10" maxlength="200" placeholder="max. 200 Zeichen"></textarea>
```

<p>Terminwunsch: <input name="datum" type="date"></p>

<p>Uhrzeit: <input name="zeit" type="time"></p>

Übung: Formular „nachbauen“

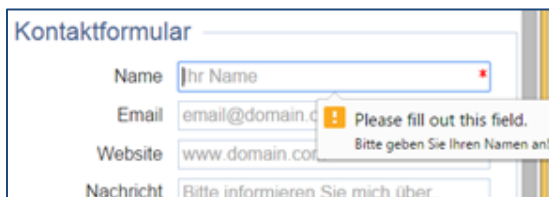
autofocus:

Damit setzt man den Cursor auf ein Eingabefeld eigener Wahl, sodass beim Seitenaufruf sofort mit der Eingabe begonnen werden kann.

required:

Damit markiert man das Feld als Pflichtfeld. Manche Browser tun sich damit aber noch schwer (Safari) und die Unterstützung von Android und Opera Mini hinkt ebenfalls nach. Die Browser, bei denen es einwandfrei funktioniert Firefox, Chrome, Opera und IE10 geben jedoch eine optisch und textlich unterschiedliche Rückmeldung für die Verletzung dieses „required-Attributes“.

z.B. Chrome:



The screenshot shows a contact form titled 'Kontaktformular' with fields for Name, Email, Website, and Nachricht. The Name field contains 'Ihr Name' and the Email field contains 'email@domain.c'. A yellow tooltip with a warning icon is displayed over the Name field, containing the text 'Please fill out this field.' and 'Bitte geben Sie Ihren Namen an!'.

Firefox:



The screenshot shows the same contact form in Firefox. The Name and Email fields are highlighted with a red border. The Name field contains 'Ihr Name' and the Email field contains 'Bitte füllen Sie dieses Feld aus.'.

Durch die Zuweisung der unterschiedlichen Typen für die <input>-Felder werden auf mobilen Geräten die Tastaturen entsprechend angepasst.

- type="email" sorgt dafür, dass das @-Zeichen ohne einen weiteren Klick zugänglich ist
- type="url" liefert eine zusätzliche Taste „.com“ für die schnelle Eingabe

Mit dem Container „div.formset“ gruppiert man die einzelnen Label-Eingabefeld-Paare und vereinfacht damit deren Ausrichtung.

```

<form action="post">
  <fieldset>
    <legend>Kontaktformular</legend>
    <div class="formset type-text">
      <label for="name">Name</label>
      <input type="text" name="name" id="name" placeholder="Ihr Name"
        required autofocus title="Bitte geben Sie Ihren Namen an!">
    </div>
    <div class="formset type-mail">
      <label for="email">Email</label>
      <input type="email" name="email" id="email" placeholder="email@domain.com"
        required title="Bitte geben Sie Ihre E-Mail-Adresse an!">
    </div>
    <div class="formset type-url">
      <label for="website">Website</label>
      <input type="url" name="website" id="website" placeholder="www.domain.com">
    </div>
    <div class="formset type-text">
      <label for="message">Nachricht</label>
      <textarea name="message" id="message" placeholder="Bitte informieren
        Sie mich über.."></textarea>
    </div>
    <input class="submit-btn" type="submit" value="Senden" />
  </fieldset>
</form>

```

Ergebnis:

Kontaktformular

Name *

Email *

Website

Nachricht

Lesetipp: HTML-Formulare

www.wufoo.com/html5

Buttons, Checkboxes erstellen

1)Radio-Buttons – Auswahlknöpfe erstellen

Dies dient für die Auswahl zwischen verschiedenen Optionen. Die Antwort-Möglichkeiten sind dabei vorgegeben, z.B. Schulnoten von 1-5 oder ja, Nein oder Weiß nicht.

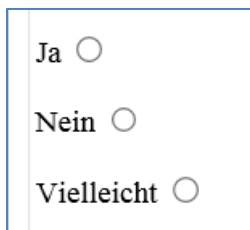
Alle Attribute dieser Auswahlliste müssen **denselben Namen** haben. Nur so werden sie als zusammengehörige Auswahlliste erkannt. Weicht der Name bei einem Element ab funktioniert die Auswahl nicht.

Dem <input>-Befehl muss der Typ <radio> zugewiesen werden.

Damit der Webserver und CGIs die Angaben richtig verarbeiten können, muss man noch das Attribut <value> mit dem entsprechenden Wert anhängen.

```
<p> Ja <input type="radio" name="antwort" value="ja"></p>
<p> Nein <input type="radio" name="antwort" value="nein"></p>
<p> Vielleicht <input type="radio" name="antwort" value="vielleicht"></p>
```

Ergebnis:



Ja

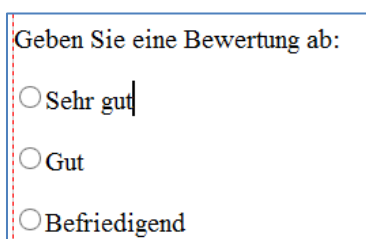
Nein

Vielleicht

Andere Variante – Radio-Button vorne

```
45 <p>Geben Sie eine Bewertung ab:</p>
46 <p><input type="radio" name="note" value="1">Sehr gut</p>
47 <p><input type="radio" name="note" value="2">Gut</p>
48 <p><input type="radio" name="note" value="3">Befriedigend</p>
49
```

Ergebnis:



Geben Sie eine Bewertung ab:

Sehr gut

Gut

Befriedigend

Soll ein Feld bereits vorausgewählt werden, muss man das Attribut `<checked>` in das Input-Feld des gewünschten Punktes setzen.

```
<p> Ja <input type="radio" name="antwort" value="ja" checked></p>
```

Weiteres Beispiel:

```
29 <fieldset>
30 <h1>Welche Farbe soll das Auto haben?</h1>
31 <p>rot <input name="farbe" type="radio" value="rot"></p>
32 <p>grün <input name="farbe" type="radio" value="grün"></p>
33 <p>schwarz <input name="farbe" type="radio" value="schwarz" checked></p>
34 </fieldset>
```

Welche Farbe soll das Auto haben?

rot

grün

schwarz

Bei Radio-Buttons ist immer nur die Auswahl von einem Punkt möglich. Die Auswahl eines Punktes deaktiviert jeden anderen.

Soll eine mehrfache Auswahl möglich gemacht werden, verwende eine Checkbox.

2)Checkbox

Dafür gibt man im Input-Feld den Type **<checkbox>** an. Wie beim Radio-Button ist es hier wichtig, dass alle Chekboxen im Attribut **<name>** denselben Namen besitzen. So werden sie als eine zusammengehörige Gruppe erkannt und verarbeitet. Der Besucher kann so viele Boxen anklicken wie er möchte.

Unterschied zu Radio-Buttons: Checkboxes, auch wenn sie den selben Gruppennamen haben, werden NICHT zu Gruppen zusammengefasst. Klickt man eine an, ändert das nichts am Zustand der andern, selbst wenn sie denselben Namen haben. Stattdessen schaltet man sie mit einem Klick an, mit einem weiteren Klick wieder aus. In HTML ist der einzige Unterschied, dass der Typ nicht **<radio>** sondern **<checkbox>** lautet.

```
50
51 <p>Checkbox: Wähle deine Hobbys: </p>
52 <p><input type="checkbox" name="hobbys" value="lesen">Lesen</p>
53 <p><input type="checkbox" name="hobbys" value="tennis">Tennis</p>
54 <p><input type="checkbox" name="hobbys" value="fuß">Fußball</p>
55 <p><input type="checkbox" name="hobbys" value="schwim">Schwimmen</p>
56
```

Ergebnis:

Checkbox: Wähle deine Hobbys:

Lesen

Tennis

Fußball

Schwimmen

Auch hier lässt sich eine Vorauswahl mit Hilfe des Attributs **<checked>** treffen.

3)Ausklappbare Auswahlliste

Sobald der Besucher mit der Maus das Auswahlfeld anklickt, öffnet sich eine Liste und bietet die Optionen an. Nach der Auswahl schließt sich die Liste wieder. Dies ist sinnvoll z.B. für Fragebögen, Bewertungen, Produktgruppen usw.

- **<select>** erstellt die Auswahlliste
- **<option>** legt die Auswahlpunkte fest

```

57
58 <h2>Auswahlliste</h2>
59 <p><select name="klassen"></p>
60   <option>1AK</option>
61   <option>2AK</option>
62   <option>3AK</option>
63   <option>4AK</option>
64   <option>5AK</option>
65 </select>
66

```

Ergebnis:



Normalerweise klappen die Auswahllisten immer nach unten auf. Dabei sind sie automatisch so lange, wie sie Optionen besitzen. Manchmal ist dies unschön. Um dies zu ändern verwendet man das Attribut **<size>**. Damit legt man die Länge fest, aber die eigentliche Listenfunktion geht etwas verloren, da die List zu einer Art scrollbarem Fenster wird.

In der ersten Zeile mit dem `<select>` füge das `<size>` Attribut ein: z.B. 2

```

57
58 <h2>Auswahlliste</h2>
59 <p><select name="klassen" size="2"></p>
60   <option>1AK</option>
61   <option>2AK</option>

```

Ergebnis:



- **Multiple Auswahl – mehrere Punkte auswählen**

Dadurch, dass die Liste in einem scrollbaren Fenster angezeigt wird, ergibt sich aber eine ganz neue Möglichkeit. Weil jetzt mehrere Punkte der Liste auf einmal sichtbar sind, ist es auch möglich mehrere Optionen auf einmal auszuwählen. Möchte man dem Besucher das erlauben, muss man das Attribut **<multiple>** in das Eröffnungsfeld setzen.

Man sollte den Besucher auch darauf hinweisen, dass durch Festhalten der STRG-Taste mehrere Punkte ausgewählt werden können.

```

57
58 <h2>Auswahlliste</h2>
59 <p><select name="klassen" size="2" multiple|></p>
60 <option>1AK</option>
61 <option>2AK</option>

```

Ergebnis:



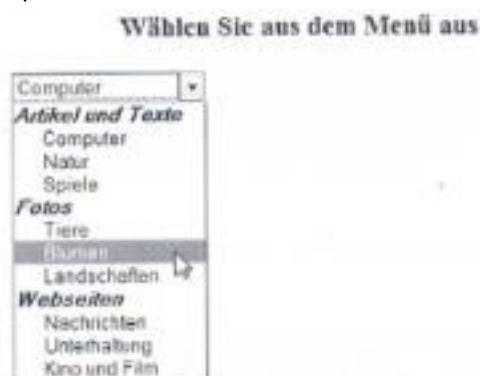
- **Vorauswahl durch das Attribut <selected>**

Damit wird eine Option bereits vorausgewählt, z.B. Länderangabe in einem Onlineshop.

- **Lange Auswahllisten mit <optgroup> strukturieren**

Damit erstellt man beliebige Gruppen innerhalb einer Auswahlliste (einer <select>-Liste). Dadurch sind die Auswahlpunkte in Gruppen sortiert.

Beispiel:



4)Diverse Elemente

a)Füllbalken erzeugen <meter>

Damit kann man ein paar Daten grafisch darstellen. Man erzeugt individuelle Füllbalken, die einen beliebigen Wert anzeigen. Auch wenn sich der Balken nicht selbständig passend zu einem Ereignis verändert, kann man doch diesen händisch auf verschiedenen Seiten mit Prozent erhöhen. Somit sieht es dynamisch aus.

- <meter> muss immer sauber geöffnet und geschlossen werden
- Mit dem Attribut <value> gibt man den Wert an
- Damit der Balken richtig gefüllt werden kann, muss man über die Attribute <min> und <max> den Bereich festlegen

- Verwendet man die Angabe in Prozent im <value>-Attribut, dann sind die Angaben <min> und <max> nicht nötig
- Zwischen den Tags kann man einen alternativen Text angeben. Dieser wird nur angezeigt, wenn der Browser den Füllbalken nicht anzeigen kann

```
<p>Fortschritt im Fragebogen:</p>
<meter value="2" min="0" max="10">2 out of 10</meter><br>
<meter value="0.6">60%</meter>
```

Ergebnis:

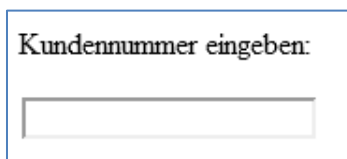


b) Zahlenfolge überprüfen <pattern>

Der <pattern>-Befehl erlaubt es, Zeichenfolgen logisch miteinander zu vergleichen oder sie anhand eines vorgegebenen Musters zu überprüfen. So kann der Browser in einem Schnellcheck sicherstellen, dass im Namensfeld nur Buchstaben eingegeben werden oder bei einer Kundennummer nur Zahlen. Oder wie hier im Beispiel eine Kombination beider:

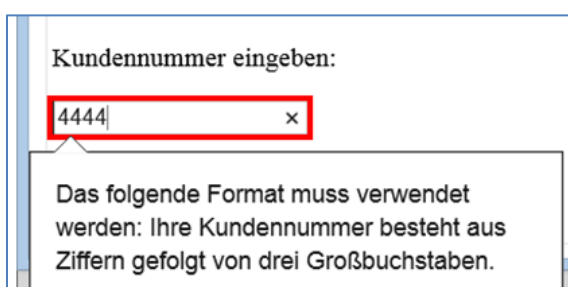
```
69 <br>
70 <p>Kundennummer eingeben:</p>
71 <input pattern="[0-9][A-Z]{3}" name="kdn" title="Ihre Kundennummer besteht aus
Ziffern gefolgt von drei Großbuchstaben.">
72
```

Ergebnis:



Über das Attribut <title> kann eine Meldung angegeben werden, die erscheint, sobald der Kunde das Feld anklickt oder das Formular mit unpassenden Daten abgesendet werden soll.

Fehlermeldung für den Kunden:

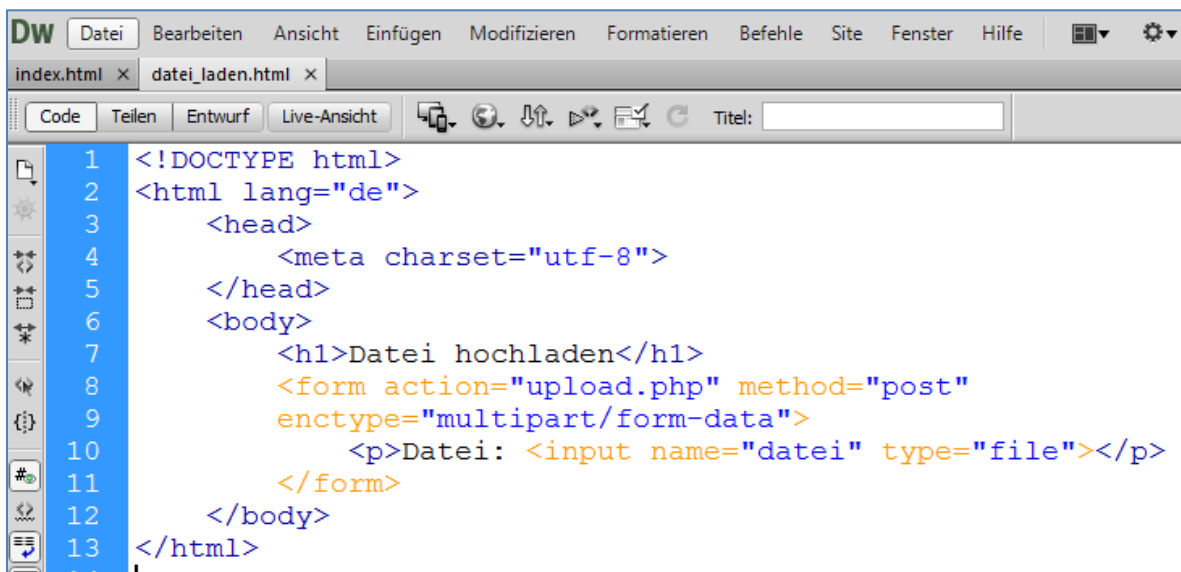


5)Dateien hochladen

Ein `<input type="file" name="datei">` genügt.

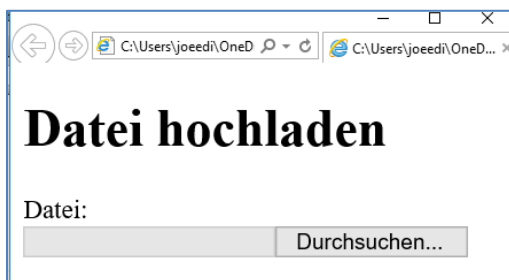
Die einzigen Einschränkungen sind folgende:

- Das Formular muss mit „method="post“ geschickt werden, ein „get“-Request kann keine Dateien transportieren.
- Am Formular muss das Attribut „enctype="multipart/form-data“ gesetzt werden. Dieses Attribut steuert, wie Formulardaten im „post-Request“ verpackt werden.



```
1 <!DOCTYPE html>
2 <html lang="de">
3   <head>
4     <meta charset="utf-8">
5   </head>
6   <body>
7     <h1>Datei hochladen</h1>
8     <form action="upload.php" method="post"
9       enctype="multipart/form-data">
10      <p>Datei: <input name="datei" type="file"></p>
11    </form>
12  </body>
13 </html>
```

Ergebnis:



Nur zur INFO:

Pattern

Mit pattern kann man die Eingabe mit regulären Ausdrücken überprüfen.

z.B.:

Die Postleitzahl-Eingabe benötigt eine vierstellige Zahl in Österreich

```
<input type="text" name="plz" id="plz" pattern="[0-9]{4}" title="Bitte geben Sie eine vierstellige Postleitzahl ein.">
```

Auf der Website <http://html5pattern.com> findet man Beispiele thematisch geordnet.

