

KAPITEL 8 - FORMULARE

Formulare werden in HTML genutzt um Daten vom Benutzer zu sammeln. In Formularen (eng.: Forms) kann der Benutzer Eingabefelder ausfüllen, in mehrzeiligen Textfeldern Text eingeben, aus Listen Einträge auswählen usw. Wenn das Formular fertig ausgefüllt ist kann der Benutzer auf einen Button klicken, um das Formular abzusenden. Diese Operation (im englischen „Submit“ genannt) schickt das Formular dann an den Webserver der die Daten, die der Benutzer eingegeben hat, auswertet und weiterverarbeitet (z.B. durch Speichern in einer Datenbank).

1. HTML FORMULARE (WIEDERHOLUNG)

Zum Erstellen eines Formulars wird das `<form>` Tag benutzt.

```
<form>
    input elements
</form>
```

Beachten Sie dass das Formular an sich nicht sichtbar ist.

1.1. INPUT ELEMENTE

Das wichtigste Form-Element ist das `<input>` Feld. Das `<input>` Feld kann in vielen Varianten vorkommen, abhängig von seinem Attribut „type“. Ein `<input>` Element kann so z.B. vom Typ text field sein, checkbox, password, radio button, submit button, usw.

1.2. TEXT FELDER

Ein `<input type="text">` definiert ein einzeiliges Inputfeld wo der Benutzer Text eingeben kann.

```
<form>
    First name: <input type="text" name="firstname"><br>
    Last name: <input type="text" name="lastname"><br>
</form>
```

First name:

Last name:

Als Standard ist ein `<input>` Feld 20 Zeichen lang, d.h. der Benutzer kann maximal 20 Zeichen eingeben.

1.4. PASSWORT FELDER

Ein `<input type="password">` definiert ein Passwort-Feld, d.h. die Eingabe des Benutzers wird in der Anzeige durch * ersetzt.

```
<form>
    Password: <input type="password" name="pwd">
</form>
```

Password:

1.5. RADIO BUTTONS

Durch `<input type="radio">` werden Radio-Buttons definiert. Mit einem Radio-Button kann der Benutzer nur eine einzige Option von einer Auswahl an Optionen markieren.

```
<form>
    <input type="radio" name="sex" value="male">Male<br>
    <input type="radio" name="sex" value="female">Female
</form>
```

☒ Male
☐ Female

1.6. CHECKBOXES

Eine Checkbox wird durch `<input type="checkbox">` definiert. Checkboxes funktionieren ähnlich wie die Radio-Buttons mit dem Unterschied dass in Checkboxes mehrere Optionen selektiert (angeklickt) werden können.

```
<form>
    <input type="checkbox" name="vehicle" value="Bike">I have a bike<br>
    <input type="checkbox" name="vehicle" value="Car">I have a car
</form>
```

☐ I have a bike
☐ I have a car

1.7. SUBMIT BUTTON

Ein Submit Button ist komischerweise auch ein Input-Element. Er wird benutzt um die Daten an den Server zu senden und wird durch `<input type="submit">` definiert.

```
<form name="myInputForm" action="script_on_server.php" method="get">
    Username: <input type="text" name="user">
    <input type="submit" value="Send">
</form>
```

Wenn der Benutzer auf den Send-Button drückt, wird das Formular zum Server geschickt. Im Attribut „action“ gibt man an welches Script auf dem Server die Daten empfangen wird. In diesem Fall ist es ein Script mit dem Namen `script_on_server.php`. Wie solche Skripte erstellt werden, sehen wir erst im nächsten Jahr. Falls kein Server-Skript zur Verfügung steht, kann ein Platzhalter (#) verwendet werden.

```
<form name="myInputForm" action="#"></form>
```

Username:

1.8. DAS NAME-ATTRIBUT

Wie dir vielleicht bereits aufgefallen ist wurde oft das Attribut „name“ verwenden, das sowohl in den `<input>` Elementen wie auch im `<form>` Tag. Dieses Attribut ist wichtig wenn ein Server-Skript die Daten

auslesen muss. Man sollte sich also besser jetzt schon daran gewöhnen bei Formularen immer name-Attribute zu verteilen.

Selbstverständlich können wir zu dem name-Attribute auch noch ein ID-Attribut hinzuschreiben, welches wir benutzen um das HTML-Element mit JavaScript anzusteuern.

1.9. AUSWAHLLISTEN

Auswahllisten(eng.: „select lists“) wurden bereits ausführlich behandelt. Dabei handelt es sich nicht um `<input>`-Element sondern um `<select>`-Elemente, welche eine oder mehrere `<option>`-Elemente besitzen.

```
<form name="myForm" action="#">
  Land: <select name="land">
    <option value="ITA">Italien</option>
    <option value="GER">Deutschland</option>
    <option value="LUX">Luxemburg</option>
  </select>
</form>
```

2. HTML-FORMULARE UND JAVASCRIPT

JavaScript wird sehr oft dazu benutzt um Benutzereingaben zu Überprüfen bevor das Formular zum Server geschickt wird. So kann man z.B. das Formular nur abschicken wenn alle Felder ausgefüllt sind, oder eine gültige Email-Adresse angegeben wurde.

Es sollte aber auf jeden Fall eine erneute Überprüfung der Daten auf dem Server stattfinden, da der Benutzer JavaScript einfach deaktivieren kann um so die Überprüfung zu übergehen. Wenn man die Eingaben sowieso auf dem Server überprüfen muss, wieso macht man sich dann überhaupt die Mühe eine (redundante) Überprüfung mit JavaScript durchzuführen. Die Antwort ist einfach um den Server zu entlasten. So wird eine erste Überprüfung der Eingaben bereits auf der Client-Seite durchgeführt, der Server wird im Moment damit noch nicht belastet.

In JavaScript erfolgt der Zugriff auf Formular-Elemente normalerweise über das form-Objekt. In diesem Kurs wird aber das bereits bestbekannte `document.getElementById()` benutzt. Jedes Input-Element muss also seine eigene ID bekommen.

2.1. JAVASCRIPT UND TEXTFELDER

Textfelder werden mit dem JavaScript-Codewort „value“ ausgelesen.

```
<form name="f" action="#">
  <input type="text" id="myInput">
</form>
<script>
  var inputField = document.getElementById(„myInput“);
  alert(inputField.value);
</script>
```

2.2. JAVASCRIPT UND RADIO-BUTTONS/CHECKBOXES

Radio-Buttons und Checkboxes können in JavaScript wie folgt ausgelesen werden.

```
<form name="f" action="#">
```

```
<input type="radio" name="gender" id="gender_Male" value="Male">
<input type="radio" name="gender" id="gender_Female" value="Female">
</form>
<script>
if (document.getElementById('gender_Male').checked) {
    //Male radio button is checked
} else if (document.getElementById('gender_Female').checked) {
    //Female radio button is checked
}
</script>
```

2.3. FORMULARVALIDIERUNG MIT JAVASCRIPT

JavaScript kann dazu benutzt werden eine Formularvalidierung durchzuführen, d.h. das Formular wird nur dann abgeschickt wenn exakt definierte Bedingungen erfüllt sind. Deshalb muss eine eigene JavaScript-Funktion programmiert werden welche entweder „true“ oder „false“ als Rückgabewert hat. Diese Funktion wird in einem speziellen onSubmit-Event aufgerufen (also kurz vor dem Versenden des Formulars).

```
<form name="myForm" action="#" onSubmit="return validateMyForm();"></form>
<script>
    function validateMyForm()
    {
        if (...)
        {
            alert(„There was an error in your form.“);
            return false;
        }
        return true;
    }
</script>
```

Wenn die Funktion „false“ zurückgibt, wird das Abschicken des Formulars zum Server abgebrochen. Falls die Funktion „true“ zurückgibt, wird das Formular ganz normal gesendet. Die Funktion sollte aber in jedem Fall immer einen der beiden Werte zurückgeben.

3. AUFGABEN

Versuche nun folgende Aufgaben zu lösen. Diese Aufgaben bauen aufeinander auf und fügen jedes Mal neue Bedingungen hinzu welche beim Absenden des Formulars erfüllt werden müssen.

AUFGABE 1

Erstelle folgendes Formular auf einer HTML-Seite.

Vorname (*)	<input type="text"/>
Nachname (*)	<input type="text"/>
Email-Adresse (*)	<input type="text"/>
Passwort (*)	<input type="password"/>
Passort (Verifikation) (*)	<input type="password"/>
	<input type="button" value="Abschicken"/> <input type="button" value="Zuruecksetzen"/>

AUFGABE 2

Ändere nun deine HTML-Seite mit JavaScript so um dass das Formular nur dann abgeschickt wird wenn alle Felder ausgefüllt sind. Bei unvollständigem Formular soll eine Meldung ausgegeben werden.

AUFGABE 3

Ändere deine Seite so um dass das Formular nur dann abgeschickt wird wenn beide Passwörter übereinander stimmen. Bei fehlerhaftem Formular soll eine Meldung ausgegeben werden.

AUFGABE 4

Ändere deine Seite so um dass das Formular nur dann abgeschickt wird wenn das Passwort mindestens 8 Zeichen lang ist. Bei fehlerhaftem Formular soll eine Meldung ausgegeben werden.

AUFGABE 5

Füge jetzt eine Auswahlliste für das Land hinzu. Entsprechenden vorgefertigten Code mit allen möglichen Ländern findest du auf dem Internet.

AUFGABE 6

Füge jetzt Input-Felder für die Hausnummer, die Straße, die Postleitzahl und das Bundesland hinzu.

AUFGABE 7

Setze jetzt eine Bedingung hinzu dass das Input-Feld für das Bundesland nur dann sichtbar ist, wenn „Deutschland“ ausgewählt ist.

AUFGABE 8

Füge eine Checkbox an das Ende deines Formular hinzu mit folgender Beschreibung: „Ich bin mit den allgemeinen Benutzerbedingungen einverstanden“. Das Formular soll auch nur dann abgeschickt werden wenn diese Checkbox aktiviert ist. Ist dies nicht der Fall muss eine entsprechende Meldung angezeigt werden.

AUFGABE 9

Folgende Funktion kann dazu benutzt werden um eine Email-Adresse auf ihre Gültigkeit zu überprüfen:

```
function validateEmail(email)
{
    var atpos = email.indexOf("@");
    dotpos = email.lastIndexOf(".");
    if (atpos < 1 || (dotpos - atpos < 2))
    {
        return false;
    }
    return true;
}
```

Versuche nun diese Funktion in deine Formular-Validierungs-Funktion einzubauen, so dass das Formular nur dann abgeschickt wird wenn die eingegebene Email-Adresse gültig ist.

Bemerkung: Die Funktion validateEmail() testet nur ob wenigstens ein „@“-Zeichen und ein Punkt (.) in der Email-Adresse vorhanden sind. Zusätzlich darf das „@“ nicht am Anfang der übergebenen Zeichenkette (=Email-Adresse) stehen und der letzte Punkt (es können ja mehrere Punkte in einer Email-Adresse vorkommen; z.B. in „max.musterman@ltam.lu“) muss wenigstens durch 1 Zeichen vom „@“ getrennt sein. Es handelt sich hierbei also um eine sehr banale Methode um eine Email-Adresse zu überprüfen.

AUFGABE 10

Schreibe nun deine HTML/JavaScript-Seite so um dass anstatt Fehlermeldungen herausgegeben werden, die entsprechenden Felder im Formular mit einem roten Stern (*) markiert werden. Hat der Benutzer z.B. eine ungültige Email-Adresse eingegeben wird dieses Feld mit einem roten * gekennzeichnet.

AUFGABE 11

Schreibe eine neue HTML/JavaScript-Seite welche folgende Frage in einem Formular anzeigt: „Welches ist die Hauptstadt von Spanien?“. In einer Radio-Button-Gruppe sollen folgende 3 Antworten angezeigt werden: „a. Malaga“, „b. Barcelona“, „c. Madrid“. Füge nun ein „Senden“-Button hinzu („Submit-Button“), welcher das Formular nur dann sendet wenn die richtige Antwort ausgewählt wurde. Mit einer Fehlermeldung soll der Benutzer ansonsten auf seine fehlerhafte Antwort aufmerksam gemacht werden.

4. CAPTCHA-VALIDIERUNG

Das Captcha wurde eingeführt, um den Spam in Gästebüchern, Foren, Kontaktformularen, etc. zu bekämpfen. CAPTCHA steht für „Completely Automated Public Turing-Test to Tell Computers and Humans Apart“ – wörtlich „Vollautomatischer öffentlicher Turing-Test, um Computer und Menschen zu unterscheiden“.

Dazu muss man wissen, dass der Spam nicht von Hand eingetragen wird, sondern von sogenannten Robots, welche die Seite nach Formularen abgrasen und dann ihren Spam einfüllen und das Formular absenden. Mit Hilfe des Captcha, welches meist für die Robots nicht lesbar ist, wird der Spam abgefangen und gelangt so nicht in die Datenbank.

Das Captcha-Zahlen-Bild hat einen fixen Hintergrund und meist willkürliche Zahlenkombinationen, welche vom User in ein dafür vorgesehenes Feld eingegeben werden muss. Nach Absenden des Formulars wird diese Zahl durch das Script geprüft, entspricht es der tatsächlichen Zahl, so kann der Eintrag gespeichert werden.

Es gibt mittlerweile aber auch weitere Methoden, zum Beispiel erkennen von Tieren oder Rechnen von Rechnungen.



AUFGABE 12

Füge zu deinem HTML/JavaScript Formular ein Rechen-Captcha hinzu. Dieses Captcha besteht aus einer einfachen Rechenaufgabe, z.B. 3+5, oder 2+7; also einer Addition von zwei einstelligen Ziffern. Diese Ziffern sollen dabei per Zufall bestimmt werden. In einem Eingabefeld muss der Benutzer dann das richtige Resultat eingeben um das Formular abschicken zu können. Ist dies nicht der Fall darf das Formular nicht abgeschickt werden können und eine entsprechende Meldung muss ausgegeben werden.