

Mark Lubkowitz



Video-Lektionen zu

✓ Photoshop  
Elements

✓ Flash

✓ TYPO3 4.0

✓ Joomla! u.v.m.

RSS  
**HTML CSS**  
XML PHP JavaScript  
Perl **MySQL** Weblogs  
Barrierefreiheit **AJAX**

# Webseiten

programmieren und gestalten

Das umfassende Handbuch

3., aktualisierte Auflage

Galileo Computing





*Der moderne Sisyphos wälzt Formulare.*  
– Thomas Christian Dahme, dt. Publizist und Aphoristiker

## 12 Formulare

Formulare sind die beste Möglichkeit, mit einem Besucher Ihrer Webseite in Kontakt zu treten. In diesem Abschnitt werden Sie den Aufbau von Formularen lernen und erfahren, wie Sie entsprechende Elemente in ein solches Formular einfügen können.

### 12.1 Was sind Formulare?

Anmeldeformulare für Seminare, bei Behörden oder Preisausschreiben, Fragebögen von Umfragen oder auch das allseits beliebte Formular für die Steuererklärung werden Ihnen sicherlich bekannt sein. Der Einkauf bei einem Versandhandel ist meistens mit einem Bestellformular verbunden, und auch im Internet haben Sie bestimmt schon einmal Bekanntschaft mit einem Formular gemacht, z. B. bei der Anmeldung für einen Webservice wie Web.de oder bei einer Suchanfrage über Google.de. Oft gibt es viele verschiedene Felder, in denen man die unterschiedlichsten Daten und Informationen eintragen oder die verschiedensten Fragen durch das Setzen eines Kreuzes beantworten muss. HTML-Formulare verhalten sich ganz ähnlich wie Papierformulare, nur dass die Möglichkeiten weit über die eines Papierformulars hinausgehen, aber sie dienen dem gleichem Zweck: dem Sammeln von Informationen.

### 12.2 Aufbau eines Formulars

Ähnlich einer Tabellendefinition stellt HTML auch für Formulare ein Hauptelement für die Definition zur Verfügung: das `form`-Element (engl. *form*, dt. *Formular*). Zum einen wird mit dem `form`-Element das grundsätzliche Verhalten eines Formulars definiert, zum anderen dient es quasi als Container für die verschiedenen Eingabefelder, Auswahllisten und -felder sowie Schaltflächen. Diese Elemente werden im Gültigkeitsbereich des `form`-Elements notiert. Auch andere HTML-Elemente wie Tabellen, Überschriften oder Textblöcke dürfen in einem `form`-Element verwendet werden.

```
<form>
  <!-- Formularelemente und andere Elemente dürfen innerhalb des
        Start- und Ende-Tags von form stehen -->
</form>
```

Die beiden wichtigsten Attribute des `form`-Elements, nämlich `action` und `method`, werden Sie nun kennenlernen.

### 12.2.1 Das `action`-Attribut

In diesem Attribut wird das Ziel definiert, an das die eingegebenen Daten versendet werden sollen. Und schon treffen Sie auf das erste Problem bei Formularen. Sie brauchen eine Skriptsprache (z. B. Perl oder PHP), um die Daten eines Formulars auszuwerten und weiterverarbeiten zu können. Das Skript, das die Daten verarbeiten soll, wird dann im `action`-Attribut notiert, entweder als globale oder als lokale Adresse. Ein Beispiel könnte ein Perl-Skript **formular.pl** sein, das auf dem gleichen Server im **cgi-bin**-Verzeichnis liegt:

```
<form action="cgi-bin/formular.pl">
...
</form>
```

**E-Mail-Empfänger**

Eine etwas unsaubere Alternative ist die Angabe einer E-Mail-Adresse. Leider führt dies nicht immer zu einem Ergebnis, da es von dem Browser und dem Rechner des Benutzers abhängt, ob sich die Daten als E-Mail versenden lassen. Soll ein Formular z. B. an die E-Mail-Adresse **formular@irgendwo.de** gesendet werden, müssen Sie die Adresse als Parameter an `action` übergeben und davor die Zeichenfolge `mailto:` notieren.

```
<form action="mailto:formular@irgendwo.de">
...
</form>
```

### 12.2.2 Das `method`-Attribut

Neben dem Empfänger müssen Sie noch die Methode angeben, wie die Daten versendet werden sollen. Es gibt zwei verschiedene Methoden, die verwendet werden können:

#### ► **get**

Diese Methode ist die Standard-Methode für Formulare. Wird das Attribut `method` nicht gesetzt, wird diese Methode automatisch verwendet. Die Daten des Formulars werden an die in `action` definierte Zieladresse angehängt, d. h. das Ziel und die Daten werden zusammengefügt und an den Browser als Zieladresse übergeben. Dies könnte z. B. so aussehen: **http://www.ziel.de/formular.pl?vorname=Max& nachname=Mustermann&stadt=Musterstadt**. Dies kann jedoch zu Problemen führen, da eine Internetadresse maximal 255 Zeichen lang sein sollte. Würden Sie nun noch mehr Daten als nur den Vornamen, den Nachnamen und die Stadt übertragen wollen, könnte es passieren, dass Daten verloren gehen.

► **post**

Um das Problem der `get`-Methode umgehen zu können, wird die `post`-Methode verwendet. Die Formulardaten werden dabei zu einem Datenpaket »verschnürt« und dann als Ganzes an den Server versendet. Die Menge der Daten ist dann egal, da sie nicht auf 255 Zeichen beschränkt ist.

Verwenden Sie die `get`-Methode nur dann, wenn wenig Informationen übergeben werden sollen, z. B. bei einer Suchanfrage. Daten, die durch ein Skript weiterverarbeitet werden sollen (Personendaten, Bestelldaten etc.), sollten immer mit der `post`-Methode übergeben werden – vor allem weil die mit `get` übertragenen Daten in der Adressleiste des Browsers zu sehen sind. Bei `post` werden die Daten nicht in der Adressleiste, sondern über einen eigenen Kanal übertragen. Für die Benutzeranmeldung und auch für die Übertragung von Daten, die sehr sicherheitskritisch sind, sollten Sie immer diese Methode verwenden.

[«]

## 12.3 Eingabefelder

Es gibt zwei Typen von Eingabefeldern: einzeilige und mehrzeilige. Für einzeilige Eingabefelder wird das `input`-Element (dt. *Eingabe*) und für mehrzeilige Eingabefelder das `textarea`-Element (dt. *Textbereich*) verwendet.

Einzeilige Eingabefelder werden immer dann verwendet, wenn nur kurze Informationen wie der Vorname, der Nachname oder der Wohnort eingegeben werden sollen. Wichtig ist jedoch, dass Sie im Attribut `type` den Parameter `text` angeben.

Das input-Element

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

Diese Angabe weist den Browser an, ein einzeiliges Eingabefeld darzustellen. Der Parameter `password` würde zwar auch ein einzeiliges Eingabefeld erzeugen, jedoch wird die Eingabe durch Sternchen »\*« maskiert. Dies ist gerade bei der Eingabe von Passwörtern sinnvoll.

```
<input type="password">
```

Bei der Verarbeitung von Formulardaten ist es später erforderlich, die einzelnen Daten identifizieren zu können. Mit dem Attribut `name` können Sie einem Eingabefeld einen Namen zuweisen, durch den sich die übertragenen Daten eindeutig identifizieren lassen.

```
<input type="text" name="vorname">
<input type="password" name="passw">
```

Die Attribute `size` und `maxlength` bestimmen die Größe des Feldes und die maximale Anzahl von Zeichen, die eingegeben werden dürfen. Die Angabe `size="25"` bedeutet, dass das Eingabefeld 25 Zeichen breit dargestellt werden soll, die Angabe `maxlength="50"` bedeutet, dass maximal 50 Zeichen eingegeben werden dürfen.

```
<input type="text" name="vorname" size="25" maxlength="50">
<input type="password" name="passw" size="8" maxlength="8">
```

Schlussendlich können Sie ein Eingabefeld auch noch mit einem Wert vorbelegen. Dafür müssen Sie das Attribut `value` verwenden. Die Angabe `value="Max"` würde bewirken, dass das Eingabefeld automatisch die Zeichenkette »Max« enthält.

```
<input type="text" name="name" size="25" maxlength="50"
value="Max Mustermann">
```

Beachten Sie, dass das `input`-Element über kein Ende-Tag verfügt.

**Das `textarea`-Element** Die typische Notation zum Erzeugen eines mehrzeiligen Eingabefelds bzw. Textbereichs lautet:

```
<textarea></textarea>
```

Auch hier können Sie dem Element mit dem `name`-Attribut einen Namen zuweisen.

```
<textarea name="kommentar"></textarea>
```

Anders als bei `input`-Elementen wird die Größe eines `textarea`-Elements über die Attribute `cols` (Spalten) und `rows` (Zeilen) festgelegt. Die Angabe erfolgt dabei in Zeichen und Zeilen. Ein Textbereich, der 40 Zeichen breit und 10 Zeilen hoch sein soll, wird folgendermaßen notiert:

```
<textarea name="kommentar" cols="40" rows="10"></textarea>
```

An dieser Stelle noch eine kleine Anmerkung: Oft werden Sie Formulare mit einer Kombination aus `input`- und `textarea`-Elementen nutzen. Da diese Elemente oft untereinander stehen, werden Sie diese meist gleich breit darstellen wollen. Geben Sie bei einem `input`-Element `size="30"` und einem `textarea`-Element `cols="30"` an, sollte man erwarten, dass sie gleich breit sind. Leider ist das nicht der Fall, weil für die Textbereiche eine andere Schriftart genutzt wird. Eine wirklich exakt gleiche Breite können Sie nur mithilfe von CSS erreichen.

Um nun einen Textbereich mit einem Wert vorzubelegen, notieren Sie diesen im Gültigkeitsbereich des `textarea`-Elements, also zwischen `<textarea>` und `</textarea>`.

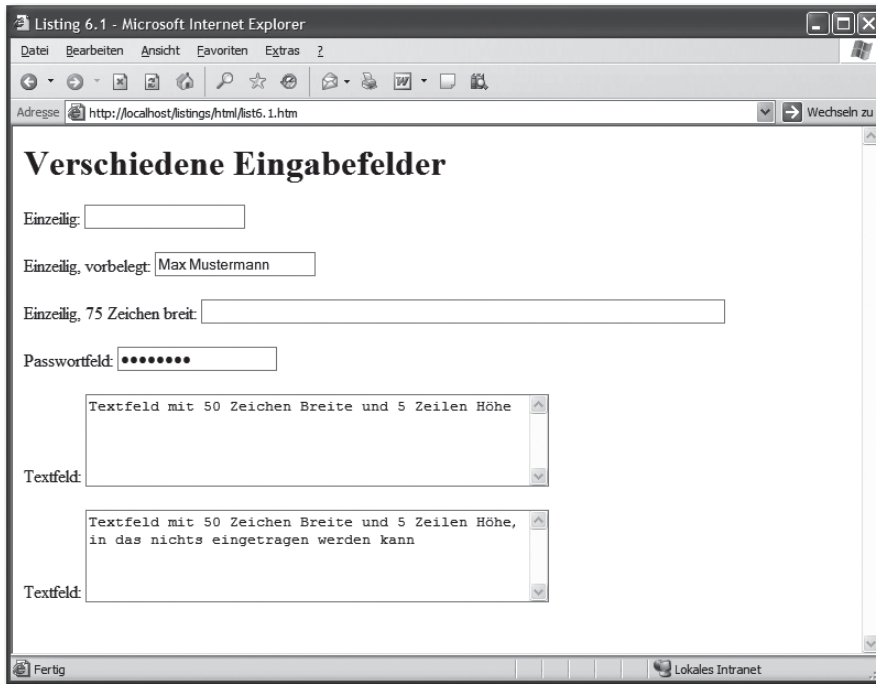
```
<textarea name="kommentar" cols="40" rows="10">Tolle Sache, diese
Textbereiche</textarea>
```

Auf die maximale Zeichenanzahl in einem Textbereich haben Sie keinen Einfluss. Sollte ein Text über die darstellbare Anzahl Zeichen hinausgehen, wird das `textarea`-Element einfach mit einer Bildlaufleiste versehen.

**Das `readonly`-Attribut** Sollen die einzeiligen und mehrzeiligen Eingabefelder nur zur Wiedergabe von Werten verwendet werden, können Sie mit Hilfe des `readonly`-Attributs die Felder auf »Nur lesen« setzen. Notieren Sie es einfach wie üblich im Start-Tag des Elements, aber ohne Angabe eines Parameters.

```
<input type="text" value="Nur lesen" readonly>
```

Im Folgenden sehen Sie ein Listing mit einigen Beispielen für Eingabefelder in Zusammenhang mit einem Formular und anschließend eine Abbildung des Listings im Internet Explorer 6.0.



**Abbildung 12.1** Darstellung des Listings 12.1 im Internet Explorer 6.0

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<!-- Formulare und Eingabefelder -->
<html>
  <head>
    <title>Listing 6.1</title>
  </head>
  <body>
    <form action="formular.pl" method="post">
      <h1>Verschiedene Eingabefelder</h1>
      <p>Einzeilig: <input type="text" name="f1"></p>
      <p>Einzeilig, vorbelegt: <input type="text"
        name="f2" value="Max Mustermann"></p>
      <p>Einzeilig, 75 Zeichen breit: <input type="text"
        name="f3" size="75"></p>
      <p>Passwortfeld: <input type="password" name="f4"
        value="passwort"></p>
      <p>Textfeld: <textarea name="t1" cols="50"
        rows="5">Textfeld mit 50 Zeichen Breite und 5
        Zeilen H&ouml;he</textarea></p>
      <p>Textfeld: <textarea name="t1" cols="50" rows="5"
```

```

        readonly>Textfeld mit 50 Zeichen Breite und 5
        Zeilen H&ouml;l;he, in das nichts eingetragen wer-
        den kann.</textarea></p>
    </form>
</body>
</html>

```

**Listing 12.1** Beispiele für einzeilige und mehrzeilige Eingabefelder

**[x]** Sollten Sie nun nach einer Möglichkeit suchen, Formularelemente durch ein Attribut zu beschriften, muss ich Sie leider enttäuschen. Sie müssen dabei auf konventionellere Maßnahmen zurückgreifen, z. B. auf Textabsätze oder Tabellen. Letztere eignen sich optimal dazu, Formulare sauber zu formatieren und den einzelnen Feldern eine Beschriftung zuzuweisen. Ein kurzes Beispiel:

```

<form action="formular.pl" method="post">
  <table>
    <tr>
      <td>Vorname:</td>
      <td><input type="text" name="vorname"></td>
    </tr>
    <tr>
      <td>Nachname:</td>
      <td><input type="text" name="nachname"></td>
    </tr>
    <tr>
      <td>E-Mail-Adresse:</td>
      <td><input type="text" name="email"></td>
    </tr>
  </table>
</form>

```

Dieses Beispiel zeigt pro Zeile einen Text und ein Eingabefeld an. Sowohl der Text als auch das Eingabefeld werden in jeweils unterschiedlichen Zellen dargestellt, sodass die Text- und Eingabefelder zueinander linksbündig ausgerichtet sind.

## 12.4 Schaltflächen

Es gibt zwei Varianten, um Schaltflächen für ein Formular zu erzeugen: eine herkömmliche Variante, die mit allen aktuellen Browsern funktioniert (Internet Explorer ab Version 3.x und Netscape ab Version 2.x), und eine neue Variante, die zwar nicht unter allen Browsern funktioniert (Internet Explorer ab Version 4.x und Netscape ab Version 6.x), aber sehr viel logischer ist.

Schaltflächen  
mit dem input-  
Element

Schaltflächen in Formularen sollen prinzipiell nur zwei Aufgaben bewältigen: das Versenden des Formulars oder das Löschen des Formulars. Zum Versenden steht der Typ `submit` (dt. *senden*) und zum Löschen der Typ `reset` (dt. *zurücksetzen*) zur Verfü-

gung. Sie werden mit dem `input`-Element in ein Formular eingebunden und bekommen als Parameter für das Attribut `type` entweder `submit` oder `reset` zugewiesen.

Parameter	Übersetzung	Erklärung
<code>submit</code>	übertragen	Erzeugt eine Schaltfläche zum Übertragen der eingegebenen Daten mit der im <code>form</code> -Start-Tag angegebenen Methode.
<code>reset</code>	zurücksetzen	Erzeugt eine Schaltfläche, mit der Sie die Felder des Formulars auf den Startwert zurücksetzen, der mit <code>value</code> definiert wurde. Andernfalls wird der Inhalt der Felder gelöscht.

**Tabelle 12.1** Mögliche Parameter für das Attribut `type` des `input`-Elements, um Schaltflächen zu erzeugen

```
<input type="submit">
<input type="reset">
```

Mit dem Attribut `name` können Sie auch für die Schaltflächen einen Namen vergeben, und mit dem Attribut `value` können Sie den Schaltflächen eine Beschriftung zuweisen.

```
<input type="submit" value="Eingaben abschicken!">
<input type="reset" value="Eingaben l&ouml;schen">
```

Den einzelnen Feldern einen Namen zuzuweisen, ist nicht unbedingt erforderlich. Nur dann, wenn Sie zwei verschiedene Schaltflächen zum Übertragen der Daten definieren, sollten Sie mit dem `name`-Attribut einen Namen vergeben, um unterscheiden zu können, mit welcher Schaltfläche die Daten abgeschickt worden sind. Das Attribut `value` sollten Sie auf jeden Fall immer mit angeben. Andernfalls wird eine vom Browser abhängige Standardbeschriftung verwendet, die in der Regel wenig mit der Aufgabe des Formulars zu tun hat.

**[x]**

Schaltflächen mit dem `button`-Element zu definieren, ist die neuere und auch logischere Variante, denn hier werden Schaltflächen über das entsprechend benannte `button`-Element erzeugt. Außerdem bietet es für die Gestaltung sehr viel mehr Möglichkeiten als das `input`-Element.

Schaltflächen  
mit dem `but-  
ton`-Element

Auch hier können Sie den Typ des Buttons wieder mit dem Attribut `type` bestimmen. Der Parameter `submit` erzeugt dann eine Schaltfläche zum Versenden und `reset` eine Schaltfläche zum Löschen der Formulardaten. Zusätzlich können Sie im Gültigkeitsbereich weitere Elemente notieren, die dann auf der Schaltfläche dargestellt werden.

```
<button type="submit" name="Versenden">
  <b>Absenden</b>
</button>
<button type="reset" name="Loeschen">
  <b>L&ouml;schen</b>
</button>
```

Das Beispiel aus Listing 12.2 erzeugt Schaltflächen, die als Beschriftung eine Grafik verwenden.



```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<!-- Schaltflächen -->
<html>
  <head>
    <title>Listing 6.2</title>
  </head>
  <body>
    <form action="formular.pl" method="post">
      <h1>Wer sind Sie?</h1>
      <p>Vorname: <input type="text" name="vorname"
        value="Max"></p>
      <p>Nachname: <input type="text" name="nachname"
        value="Mustermann"></p>
      <p>Alter: <input type="text" name="alter"
        value="150"></p>
      <p>
        <b>Schaltflächen mit <kbd>input</kbd>: </b>
        <input type="submit">
        <input type="reset">
      </p>
      <p>
        <b>Schaltflächen mit <kbd>button</kbd>: </b>
        <button type="submit">
          
        </button>
        <button type="reset">
          
        </button>
      </p>
    </form>
  </body>
</html>

```

### Listing 12.2 Beispiel für Schaltflächen in HTML

In Listing 12.2 wurden vier Schaltflächen eingefügt: zwei über das `input`- und zwei über das `button`-Element. Die beiden mit `input` erzeugten Schaltflächen können als Beschriftung lediglich einen Text darstellen. Die Schaltflächen, die mit dem `button`-Element definiert wurden, verwenden anstelle eines einfachen Textes eine kleine Grafik. Auch die Angabe von Text und Grafik ist möglich, wie das folgende Beispiel zeigt.

```

<button type="submit">
  <br>Daten abschicken
</button>

```

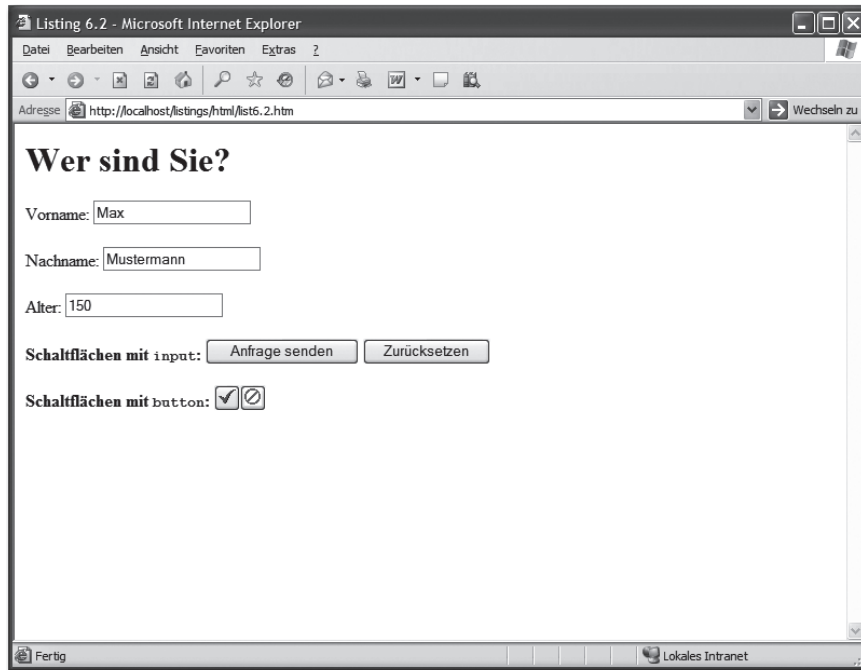


Abbildung 12.2 Darstellung des Listings 12.2 im Internet Explorer 6.0

## 12.5 Wahlprogramm

Wenn der Benutzer einmal nichts eingeben, sondern nur aus vorgegebenen Möglichkeiten die eine oder andere auswählen soll, stehen Ihnen drei Auswahlarten zur Verfügung: Radiobuttons, Checkboxes und Auswahllisten.

Radiobuttons sollten Sie dann verwenden, wenn verschiedene Auswahlmöglichkeiten gegeben sind, und der Benutzer nur eine davon auswählen soll. Solche Buttons werden mit dem `input`-Element erstellt und erhalten als Parameter für das `type`-Attribut `radio`.

**Radiobuttons**

```
<p>
  <input type="radio" name="geschlecht" value="m">
    männlich<br>
  <input type="radio" name="geschlecht" value="w">
    weiblich<br>
  <input type="radio" name="geschlecht" value="k">
    keine Ahnung
</p>
```

Bei Radiobuttons müssen Sie mit dem `value`-Attribut einen Wert definieren. Dieser Wert wird dann beim Absenden des Formulars an den Empfänger gesendet und kann über den in `name` angegebenen Namen identifiziert werden. Geben Sie keine Werte

an oder unterscheiden die Werte sich nicht voneinander, kann es passieren, dass die Radiobuttons im Browser nicht korrekt funktionieren.

**[»]** Achten Sie darauf, dass alle Radiobuttons, die zu einer Gruppe gehören sollen, als Parameter im Attribut `name` den gleichen Wert zugewiesen bekommen. Nur dadurch ist die Gruppierung der Radiobuttons möglich. Unterscheiden sich die Werte für das `name`-Attribut, werden alle Radiobuttons gleichzeitig auswählbar.

**Checkboxen** Checkboxen sollten Sie immer dann verwenden, wenn der Benutzer aus verschiedenen Möglichkeiten mehrere auswählen können soll. Auch hier wird wieder das `input`-Element verwendet, diesmal als Typ jedoch `checkbox` angegeben.

```
<p>
  <input type="checkbox" name="hobbies" value="fussball"> Fußball<br>
  <input type="checkbox" name="hobbies" value="basketball"> Basketball<br>
  <input type="checkbox" name="hobbies" value="handball"> Handball
</p>
```

Die in `value` angegebenen Werte werden dann an den Empfänger übertragen und lassen sich über den in `name` angegebenen Wert identifizieren.

**Vorauswahl** Mit dem Attribut `checked` ohne Angabe eines Parameters können Sie bei Radiobuttons einen und bei Checkboxen mehrere Einträge vorselektieren.

```
<input type="radio" value="Radiobutton" checked>
<input type="checkbox" value="Checkbox" checked>
```

Checkboxen sollen dem Benutzer die Wahl zwischen keiner, einer oder mehreren Möglichkeiten bieten. Radiobuttons hingegen sollen dem Benutzer nur eine Wahl lassen, die er auch zu treffen hat. Daher sollten Sie bei den Radiobuttons immer eine der Möglichkeiten mit `checked` vorselektieren, da es andernfalls vorkommen kann, dass keine der Auswahlmöglichkeiten selektiert wurde.

**Auswahllisten** Auswahllisten sind eine Kombination aus Radiobuttons und Checkboxen. Bei diesen hat der Benutzer die Möglichkeit, mehrere, mindestens aber eine der Möglichkeiten auszuwählen. Die Definition einer Auswahlliste gestaltet sich allerdings vollkommen anders. Hier finden zwei Elemente Anwendung: `select` und `option`. Mit `select` (dt. *auswählen*) wird eine Auswahlliste definiert, die als Container für die `option`-Elemente gilt. Mit `option` werden dann die einzelnen Auswahlmöglichkeiten festgelegt.

```
<select name="Verein" size="1">
  <option>Hertha BSC</option>
  <option>Bayer Leverkusen</option>
  <option>Borussia Dortmund</option>
  <option>FC Schalke 04</option>
</select>
```

Mit dem Attribut `size` können Sie die Höhe der Auswahlliste bestimmen. Entsprechend viele Einträge werden gleichzeitig angezeigt. Um dem Benutzer nun eine mehr-



fache Wahl zu ermöglichen, müssen Sie im `select`-Start-Tag das Attribut `multiple` (dt. *mehrfach*) notieren.

```
<select name="Verein" size="4" multiple>
```

Einen oder mehrere Einträge, je nach Typ der Auswahlliste, können Sie mit dem `selected`-Attribut markieren. Auch bei diesem Attribut wird kein Parameter notiert.

```
<option selected>Hertha BSC</option>
```

Wenn Sie im Start-Tag der `option`-Elemente das Attribut `value` und einen Parameter notieren, können Sie so den Wert beeinflussen, den der Empfänger bzw. das Empfänger-Skript der Daten erhält. Ansonsten werden die im Gültigkeitsbereich der `option`-Elemente notierten »Beschriftungen« als Wert versendet/verwendet.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<!-- Radiobuttons, Checkboxes und Auswahllisten -->
<html>
  <head><title>Listing 6.3</title></head>
  <body>
    <form action="auswahl.pl" method="post">
      <h1>Umfrage</h1>
      <p>
        <h3>Welchen Geschlechts sind Sie?</h3>
        <input type="radio" name="geschlecht" value="m"
          checked> männlich
        <input type="radio" name="geschlecht" value="w">
          weiblich
      </p><p>
        <h3>Welche Medien nutzen Sie, um sich über
          Fußball zu informieren?</h3>
        <input type="checkbox" name="medien" value="tv"
          checked> Fernsehen
        <input type="checkbox" name="medien"
          value="radio"> Radio
        <input type="checkbox" name="medien" value="www"
          checked> Internet
        <input type="checkbox" name="medien" value="zei-
          tung"> Zeitung
      </p><p>
        <h3>Welcher ist Ihr Lieblingsverein?</h3>
        <select name="verein" size="1">
          <option selected>Hertha BSC</option>
          <option>Bayer Leverkusen</option>
          <option>Borussia Dortmund</option>
          <option>FC Schalke 04</option>
        </select>
      </p><p>
        <input type="submit"><input type="reset">
      </p>
```

```

    </form>
  </body>
</html>

```

**Listing 12.3** Beispiel für Radiobuttons, Checkboxes und Auswahllisten

The screenshot shows a web browser window titled 'Listing 6.3 - Microsoft Internet Explorer'. The address bar shows 'http://localhost/listings/html/list6.3.htm'. The page content is a survey form titled 'Umfrage'. It has three main sections: 1. 'Welchen Geschlechts sind Sie?' with two radio buttons labeled 'männlich' and 'weiblich'. 2. 'Welche Medien nutzen Sie, um sie über Fussball zu informieren?' with four checkboxes labeled 'Fernsehen', 'Radio', 'Internet', and 'Zeitung'. 3. 'Welcher ist Ihr Lieblingsverein?' with a dropdown menu currently showing 'Hertha BSC'. Below these sections are two buttons: 'Anfrage senden' and 'Zurücksetzen'. At the bottom, there is a collapsed section titled 'Geschlecht' which, when expanded, shows the same gender question and radio buttons as the first section. The browser's status bar at the bottom shows 'Fertig' and 'Lokales Intranet'.

**Abbildung 12.3** Darstellung des Listings 12.3 im Internet Explorer 6.0

Das Formular aus Listing 12.3 bietet eine Kombination der vorgestellten Auswahlmöglichkeiten. Es umfasst sowohl Radiobuttons als auch Checkboxes und eine Auswahlliste.

## 12.6 Elemente gruppieren

Um ein Formular übersichtlicher gestalten zu können, lassen sich die Elemente gruppieren und mit Überschriften versehen. Mit der Element-Kombination `fieldset` (dt. *Feldgruppe*) und `legend` (dt. *Legende*) können Sie einen Rahmen um mehrere Elemente ziehen und eine Überschrift zuweisen.

```

<fieldset>
  <legend>Geschlecht</legend>
  Welchen Geschlechts sind Sie?<br>
  <input type="radio" name="geschlecht" value="m"
    checked> männlich

```

```

☐
    weiblich
</fieldset>

```

Im Gültigkeitsbereich des `legend`-Elements können Sie auch weitere Elemente zur logischen oder physischen Textauszeichnung verwenden. So könnte durch die Verwendung des `b`-Elements die Überschrift fett dargestellt werden.

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<!-- Elemente gruppieren -->
<html>
  <head>
    <title>Listing 6.4</title>
  </head>
  <body>
    <form action="gruppieren.pl" method="post">
      <fieldset>
        <legend><b>Persönliche Daten</b></legend>
        Vorname: <input type="text" name="vorname"><br>
        Nachname: <input type="text" name="nachname"><br>
        Wohnort: <input type="text" name="wohnort"><br>
        E-Mail: <input type="text" name="email"><br>
      </fieldset><br>
      <input type="submit" value="Abschicken">
    </form>
  </body>
</html>


```

**Listing 12.4** Gruppierung von Elementen mit `fieldset` und `legend`

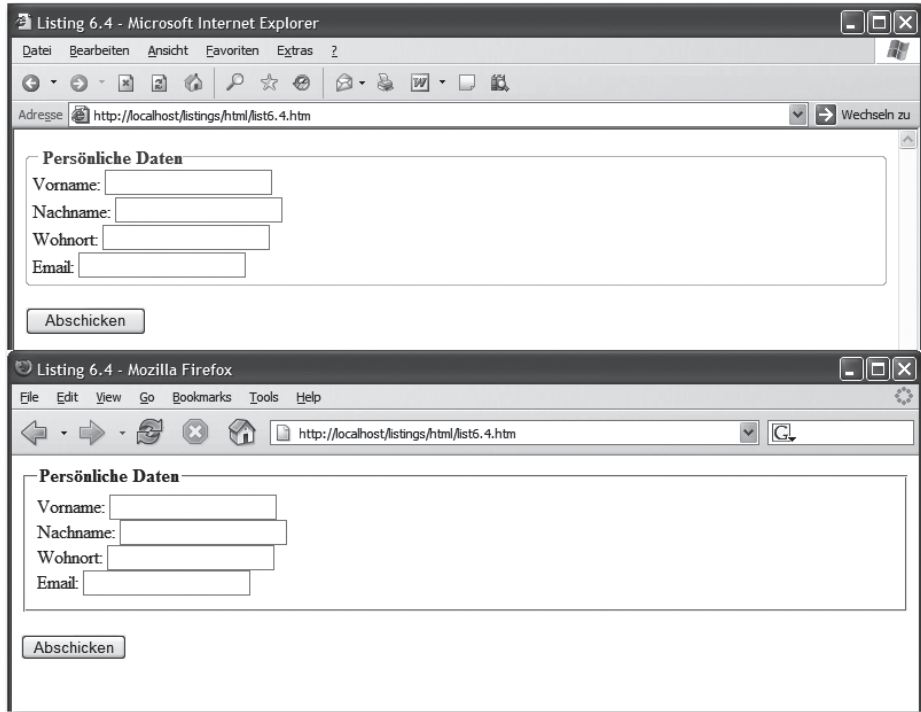
Wie Sie anhand des Beispiels aus Listing 12.4 und der entsprechenden Abbildung 12.4 erkennen können, ist die Darstellung des Rahmens nicht nur vom Browser, sondern auch vom eingesetzten Betriebssystem abhängig. Es gibt sogar Unterschiede zwischen den einzelnen Versionen der Betriebssysteme.

[x]

## 12.7 Reihenfolge

Sollte ein Formular über mehrere Eingabefelder, Radiobuttons, Checkboxes, Textbereiche, Auswahllisten und Schaltflächen verfügen, ist es meistens sehr angenehm, mit der Tabulator-Taste  zwischen diesen Elementen hin und her springen zu können. Normalerweise wird die Sprungreihenfolge durch die Reihenfolge der Notation der Elemente im Dokument bestimmt. Mit dem Attribut `tabindex` und einem Zahl-Parameter (eine Zahl zwischen 0 und 32.767) können Sie die Reihenfolge aber Ihren eigenen Wünschen anpassen, vorausgesetzt, der Benutzer verwendet den Internet Explorer ab Version 4 oder Netscape ab Version 6. Das Formularelement mit dem `tabindex="0"` wird dabei als Erstes angesprungen, das Formularelement mit dem höchsten Wert als Letztes.





**Abbildung 12.4** Darstellung des Listings 12.4 in Internet Explorer und Firefox

```
<form action="tabindex.pl">
  Feld 1: <input type="text" name="f1" tabindex="3">
  Feld 2: <input type="text" name="f2" tabindex="1">
  Feld 3: <input type="text" name="f3" tabindex="2">
  Feld 4: <input type="text" name="f4" tabindex="0">
</form>
```

**Schnelltaste** Im Internet Explorer ab Version 5 und Netscape ab Version 6 können Sie mit dem Attribut `accesskey` (dt. *Zugriffstaste*) eine Taste von [A] bis [Z] festlegen, über die in Kombination mit der Taste [Alt] zu einem bestimmten Formularelement gesprungen werden kann.

```
<form action="accesskey.pl">
  Feld1 (Alt+B): <input type="text" name="f1"
  accesskey="b">
  Feld2 (Alt+G): <input type="text" name="f2"
  accesskey="g">
  Feld3 (Alt+E): <input type="text" name="f3"
  accesskey="e">
  Feld4 (Alt+L): <input type="text" name="f4"
  accesskey="l">
</form>
```

Das `accesskey`-Attribut darf in den Elementen `input`, `textarea`, `select`, `legend` und `button` verwendet werden.

Dass diese Art des Zugriffs möglich ist, sollten Sie dem Benutzer auf geeignete Weise mitteilen, da die Browser dies nicht von allein tun. So könnten Sie z. B. einen Buchstaben in der Beschriftung des Felds, der mit dem »Accesskey« übereinstimmt, unterstreichen, oder, wie im vorherigen Beispiel, indem Sie die Tastenkombination in Klammern dahinter schreiben.

[x]

## 12.8 Zusammenfassung

- ▶ Das Element `form` enthält alle Formularelemente. Das Attribut `action` bestimmt das Empfänger-Skript, und über das Attribut `method` wird die Versandart festgelegt.
- ▶ Das `input`-Element ist sehr vielseitig; der Parameter `text` im Attribut `type` erzeugt ein einzeliges Eingabefeld, `radio` einen Radiobutton, `checkbox` eine Checkbox und `submit` eine Schaltfläche zum Versenden der Formulardaten.
- ▶ Für mehrzeilige Eingabefelder wird das `textarea`-Element verwendet.
- ▶ Auswahllisten werden durch die Kombination der Elemente `select` und `option` erzeugt.

## 12.9 Fragen und Übungen

1. Welchen Parameter müssen Sie im Attribut `type` des `input`-Elements setzen, um ein einzeliges Eingabefeld zu erzeugen, dessen Eingaben durch »\*« maskiert werden?
2. Erstellen Sie ein Formular mit sechs einzeligen Eingabefeldern für Vorname, Nachname, Straße, Hausnummer, PLZ und Stadt. Vergeben Sie sinnvolle Bezeichner (`name`-Attribut). Geben Sie als Empfänger-Skript **ziel.php** und eine beliebige Methode Ihrer Wahl an.
3. Fügen Sie eine **Absenden**- und eine **Zurücksetzen**-Schaltfläche hinzu.
4. Fügen Sie dem Formular außerdem ein Textfeld für die Eingabe eines längeren Textes (z. B. eines Kommentars) hinzu, das 50 Zeichen breit und 10 Zeilen hoch sein soll.
5. Gruppieren Sie die Elemente logisch mit den Elementen `fieldset` und `legend`. Verwenden Sie als Überschriften für die Gruppen »Name«, »Anschrift« und »Ihre Meinung«.
6. Welche Versandmethode (`method`-Attribut) ist für dieses Formular die sinnvollere Wahl?