Formulare mit jQuery Mobile

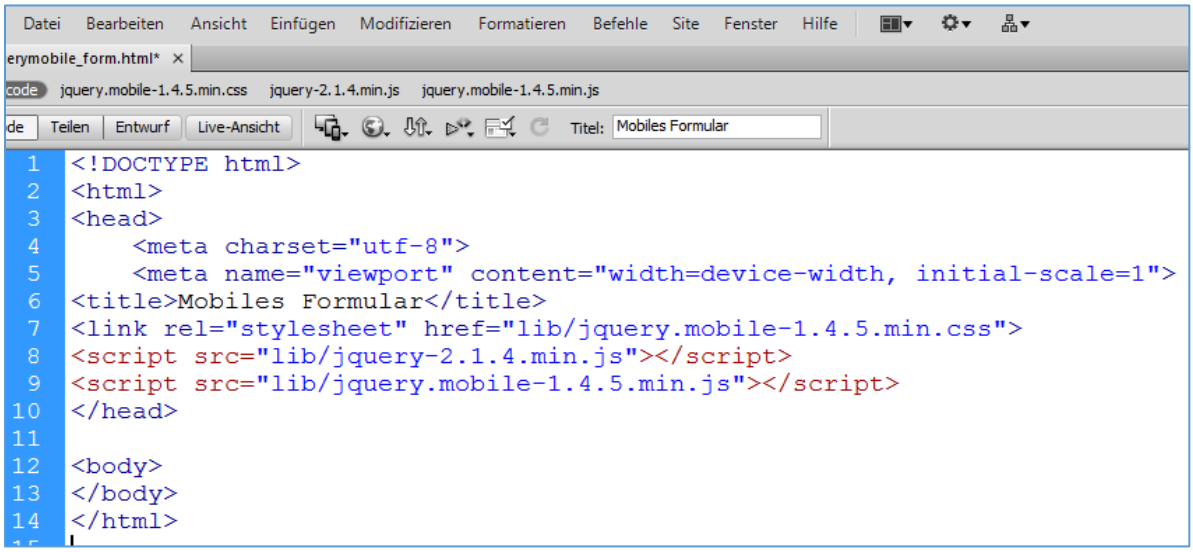
Inhalt:

a)Textfeld erstellen, Search, Clear-Button
b)Checkboxen
c)Radiobutton – vertikal und horizontal
d)Auswahl <select>
e)Zwei-Werte-Schalter FlipSwitch
f)Absenden / Zurücksetzen
g)Formularelemente in Miniformat
h)Slider
i)Übung

jQuery Mobile stellt Formularelemente „mobilgerecht“ dar. Die Standardelemente wie Textfelder und Checkboxen mögen zwar in einem Desktop-Browser gut mit der Maus bedienbar sein, aber auf einem mobilen Endgerät, und mit dem Finger als Eingabegerät, ist man jedoch schnell überfordert.

So layoutet jQuery Mobile die Formularelemente standardmäßig so, dass Checkboxen genug Fläche bieten, um mit dem Finger bequem eine Auswahl treffen zu können. Es werden auch die HTML5 spezifischen Input-Elemente wie „search, range oder number“ unterstützt.

Übung:
Erstelle eine neue Datei namens „3_jquerymobile_form.html“

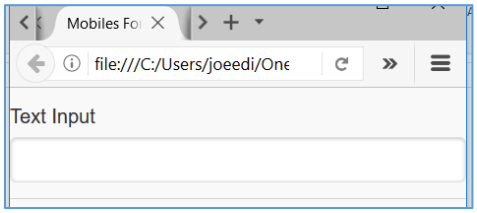


Jedes Formularelement ist inklusive seines Labels in einen Container mit dem Attribut

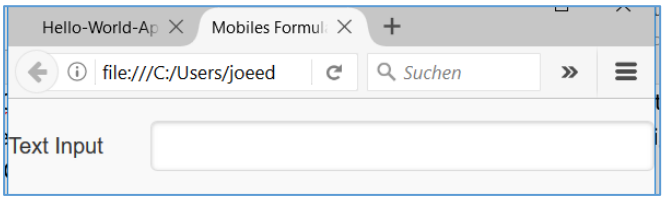
„**data-role=“fieldcontain“** verschachtelt.

Da das Handy entweder in vertikaler oder horizontaler Ausrichtung gehalten werden kann, unterstützt jQM die dynamische Darstellung. Dafür muss man eine semantische Information zu den Element-Gruppen verfügen, damit der Platz besser ausgenutzt werden kann. Deshalb müssen sich zusammengehörige Elemente in einen „Field-Container“ befinden:

vertikal: untereinander

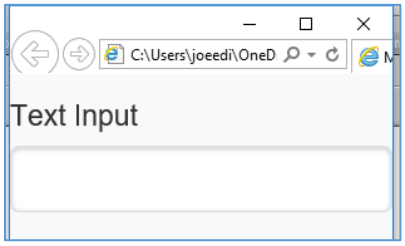


horizontal: nebeneinander



a)Textfeld erstellen

```
12  <body>
13  <form action="#" method="get">
14  <!--Textinput-->
15  <div data-role="fieldcontain">
16      <label for="name">Text Input</label>
17      <input type="text" name="name" id="name " value="">
18  </div>
19  </form>
20  </body>
21  </html>
```



Hinweis:

Stimmen die Attribute „for“ beim Label nicht mit der „id“ beim Form-Element überein, kann jQM die Spalten-Formatierung nicht mehr durchführen.

Das Label-Element hat den Vorteil, dass jQM die Textzeile auf gleicher Höhe darstellt wie den Text des Eingabe-Elements. Wird stattdessen kein Label, sondern reiner Text verwendet, wird er Kopf-bündig dargestellt, was nicht besonders schön aussieht.

Search:

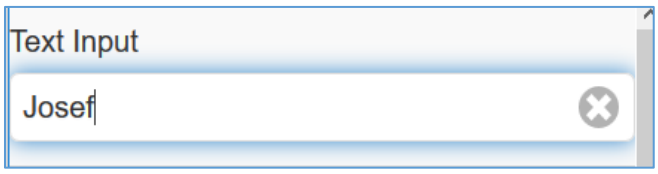
Interessant ist das Input-Element vom Typ „search“. Die Ecken werden in diesem Fall mehr abgerundet und zusätzlich wird das Lupen-Icon eingeblendet.

Clear-Button:

Neben den üblichen HTML5-Attributen wie „placeholder“ etc. stellt jQM das Attribut „data-clear.btn=“true“ zur Verfügung. Es wird erst eingeblendet, wenn etwas eingegeben wurde.

Dabei wird automatisch ein „Clear-Button“ für das Löschen des Textes zur Verfügung gestellt. Der Tooltip-Text ist änderbar mit dem Attribut „data-clear-btn-text=“löschen“.

```
<!--Textinput-->
<div data-role="fieldcontain">
    <label for="name">Text Input</label>
    <input type="text" name="name" id="name " value="" data-clear-btn="true">
</div>
<!--Ende Textinput-->
```

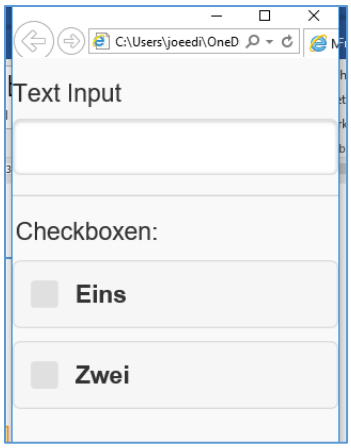


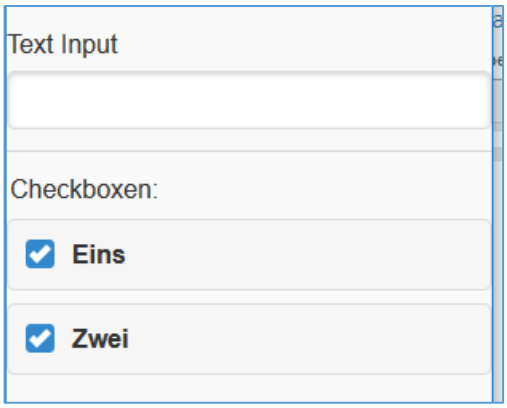
Leider funktioniert der Clear-Button nicht bei Textarea-Elementen.

b)Checkboxen

Im Falle von Checkboxen solle man noch einen Block mit dem HTML-Element <fieldset> hinzufügen sowie ein <legend>-Element für die Beschriftung der zusammengehörigen Checkboxen. Dies sollte man auch bei Radiobuttons machen.
Info: **Unterschied zu Radio-Buttons**: Checkboxen, 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.

```
<!--Ende Textinput-->
<!--Checkboxen-->
<div data-role="fieldcontain">
    <fieldset>
        <legend>Checkboxen:</legend>
        <label for="checkbox-1">Eins</label>
        <input type="checkbox" name="checkbox-1"
        id="checkbox-1">
        <label for="checkbox-2">Zwei</label>
        <input type="checkbox" name="checkbox-2"
        id="checkbox-2">
    </fieldset>
</div>
</form>
</body>
</html>
```



angeklickt:


Möchte man die Checkbox nicht wie standardmäßig angeboten links haben, so kann man die Position mit dem Attribut „data-iconpos“ platzieren. Möglich sind left, right, top und bottom.

Da es sich bei Checkboxen ja um eine Mehrfachauswahl handelt, können alle Punkte in der Checkliste „abgehakt“ werden.

c)Radiobutton – vertikal und horizontal

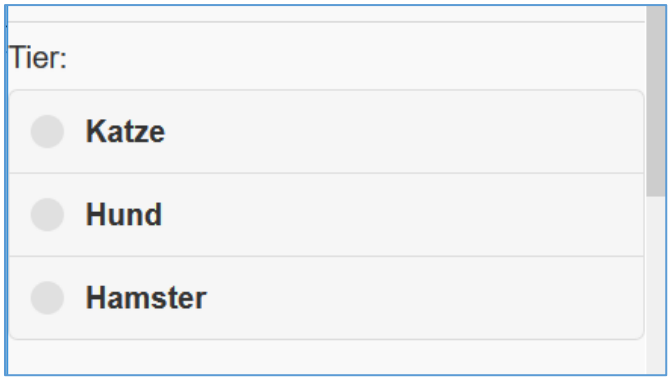
Der Radiobutton wird genau wie eine Checkbox erstellt, nur der type wird anstelle von „checkbox“ auf „radio“ geändert.

Radiobuttons werden bei der Auswahl **einer Option** aus mehreren verwendet. Sie werden automatisch „de-selektiert“, wenn in anderer Radiobutton in der gleichen Gruppe selektiert wird.

Analog zu den Checkboxen lassen sich die Radiobuttons in horizontale und vertikale Gruppen organisieren. Beide lassen sich auch über die Angabe von „data-theme“ im input-Tag über die Farbschemata von jQuery Mobile einfärben, z.B. Farbschema b.

Beispiel:

Verwendung von „fieldset“ mit der Rolle „controlgroup“.



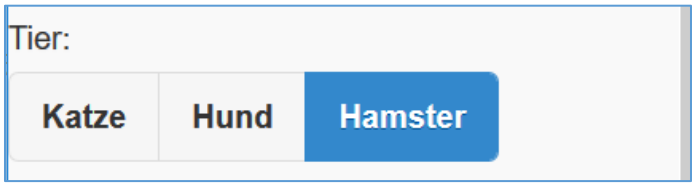

```
<!--Start Radio Buttons -->
<fieldset data-role="controlgroup">
        <legend>Tier:</legend>
        <input type="radio" name="tierwahl" id="tierwahl-1" value="wahl-1">
```

```
        <label for="tierwahl-1">Katze</label>
         <input type="radio" name="tierwahl" id="tierwahl-2" value="wahl-2">
         <label for="tierwahl-2">Hund</label>
          <input type="radio" name="tierwahl" id="tierwahl-3" value="wahl-3">
        <label for="tierwahl-3">Hamster</label>
</fieldset>
<!--Ende Radio Buttons-->
```

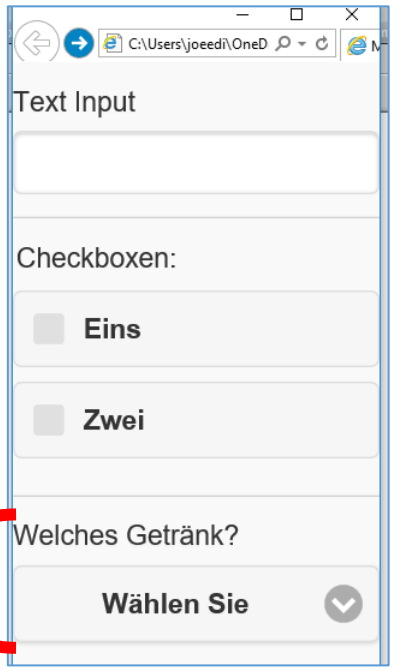
Soll die Darstellungsform auf „**horizontal**“ gestellt werden, muss zusätzlich „data-type=“horizontal“ angegeben werden:

```
<fieldset data-role="controlgroup" data-type="horizontal">
    <legend>Tier:</legend>
    <input type="radio" name="tierwahl" id="tierwahl-1" valu
    <label for="tierwahl-1">Katze</label>
```

Ergebnis:



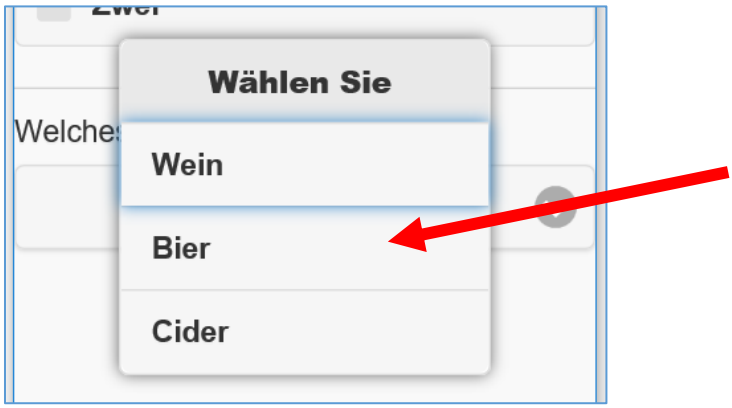
d)Auswahl <select>



Eine Besonderheit weist das <select>-Element auf. Während standardmäßig das native Menü aufklappt, wenn man auf eine Selectbox tippt, kann man mit dem Attribut

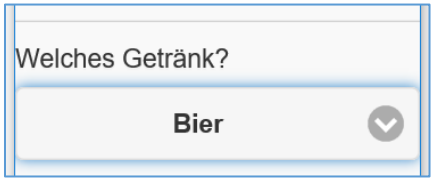
„data-native-menu=“false“

das Menü von jQuery Mobile aktivieren.



```
</div>
<!--Ende Checkboxen-->
<!--start select-->
<div data-role="fieldcontain">
    <label for="select-a" class="select">Welches Getränk?</label>
    <select name="select-a" id="select-a" data-native-menu="false">
        <option>Wählen Sie</option>
        <option value="1">Wein</option>
        <option value="2">Bier</option>
        <option value="3">Cider</option>
    </select>
 </div>
<!--ende select-->
```

Ergebnis nach einer getroffenen Wahl:



Tipp: Man kann ein <select>-Menü auch mit einer multiplen Auswahl ausstatten. Wenn man das Attribut „multiple=“multiple“ angibt, kann der Benutzer mehrere Optionen auswählen.

e)Zwei-Werte-Schalter FlipSwitch

Der FlipSwitch kennt zwei Werte, die über einen Select mit zwei Optionen repräsentiert werden.

Das Attribut lautet

data-role=“flipswitch“.

Dies kann sowohl für select- als auch für Checkbox-Tags gewählt werden. Um den Labels bei der Checkbox-Variante andere Werte als „On“ und „Off“ zu geben, kann man über die data-Attribute „data-on-text“ und „data-off-text“ individuelle Werte vergeben.

Übung: Füge unter dem <select> die „flipsitch“ ein:

```
 </div>
<!--ende select-->
<!--start flipswitch-->
    <select data-role="flipswitch">
        <option value="off">Aus</option>
        <option value="on">Ein</option>
     </select>
<!--ende flipswitch-->
```

Es ist die erste Option als Default-Wert selektiert. Wenn man das ändern möchte, vergibt man dem zweiten Wert das Attribut „selected“.

Hinweis:
Durch die feste Größe des FlipSwitch ist es nur für die Auswahl zwischen zwei Optionen mit kurzem Namen geeignet, da längere Textlabels schlicht und einfach nicht mehr lesbar sind.

z.B. ein/aus, wahr/falsch, Mann/Frau

f)Absenden / Zurücksetzen

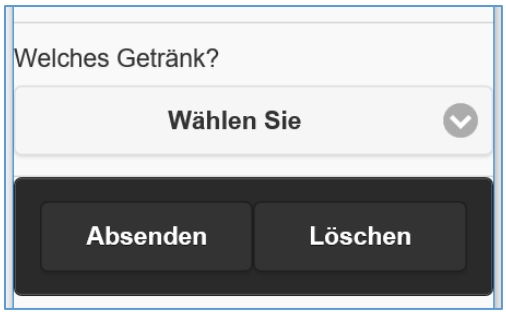
Hier sollen zwei Buttons nebeneinanderliegen.

Hier kann man statt dem Attribut „data-role=“fieldcontain“ auch mal anders designen:

<div class=“ui-body ui-body-b ui-corner-all“>

Zusätzlich soll mit Hilfe von einem Layout-Raster ein zweispaltiges Design verwendet werden, damit beide Buttons nebeneinanderliegen: Normalerweise nehmen Inhalte im Content-Bereich die volle Breite des Screens ein. Möchte man Spalten darstellen, kann man auf Layout Raster verwenden, die jQuery Mobile anbietet. Dabei verwendete man „ui-grid“ und „ui-block“.

```
43  <!--ende select-->
44  <!--absenden-->
45  <div class="ui-body ui-body-b ui-corner-all">
46      <fieldset class="ui-grid-a">
47          <div class="ui-block-a">
48              <button type="submit">Absenden</button>
49          </div>
50          <div class="ui-block-b">
51              <button type="reset">Löschen</button>
52          </div>
53      </fieldset>
54  </div>
55  <!--ende absenden-->
56  </form>
```



f)Datumseingabe

Der HTML5 Input-Type „date“ und ähnliche funktionieren unter dem iPhone sehr gut, aber leider auf dem Desktop und unter Android nicht. Deshalb muss eine separate Bibliothek verwendet werden. Eine gute Wahl ist jQuery Mobile Datebox von JTSage.

http://dev.jtsage.com/jQM-DateBox2

Es gibt zwei Versionen, wobei die jQuery Mobile mit dem Namen „jQuery Mobile Datebox“ die bessere ist.

Es gibt mehrere Modi für die Datumseingabe wie z.B. DateBox, FlipBox. Für die Zeitangabe gibt es die TimeBox und die Time-FlipBox. Eine direkte Datumseingabe ist nicht mehr möglich, es wird sofort die entsprechende Dialogbox geöffnet. Für die Internationalisierung gibt es Übersetzungen in vielen Sprachen, auch in Deutsch. Diese muss man halt zusätzlich einbinden.

http://dev.jtsage.com/jQM-DateBox/demos/api/i18n.html

Anwendung:
Beim input-Element muss in guter jQM-Manier die Rolle „databox“ angegeben werden und als Option der Modus. Der Typ muss „text“ sein, da es sonst mit der nativen Implementierung Konflikte gibt.

```
<script src="lib/datebox/jqm-datebox.core.js"></script>
<script src="lib/datebox/jqm-datebox.mode.calbox.js"></script>
<script src="lib/datebox/jquery.mobile.datebox.i18n.de.utf8.js"></script>

    <div data-role="fieldcontain">
    <label for="date">Datum:</label>
    <input name="date" id="date" type="text" data-role="datebox"
            data-options='{"mode":"calbox", "useNewStyle":true}'>
    </div>
```

Eine neuere Version ist der Datepicker:

```
<script src="lib/datepicker/datepicker.js"></script>
<script id="mobile-datepicker" src="lib/datepicker/jquery.mobile.datepicker.js"></script>
```

```
<div class="ui-field-contain">
      <label for="date">Degustiert:</label>
      <input data-role="date" required type="text" name="date" id="date" data-mini="true">
</div>
```

g)Formularelemente in Miniformat

Alle Formularelemente können über das zusätzlich zu setzende Attribut

data-mini="true"

geringfügig verkleinert dargestellt werden. Dies hat rein optische Bedeutung und keinen Einfluss auf die Funktion des Formulars.

Beispiel:

```
<fieldset data-role="fieldcontain" data-mini="true">
```

h)Slider

Der Slider ist ein input-Element vom Typ „range" und hat einen Start- und Endwert:

```
<div data-role="fieldcontain">
    <label for="slider"> Slider:</label>
    <input type="range" name="slider" id="slider" value="50" min="0" max="100"
    data-highlight="true" data-mini="true">
 </div>
```

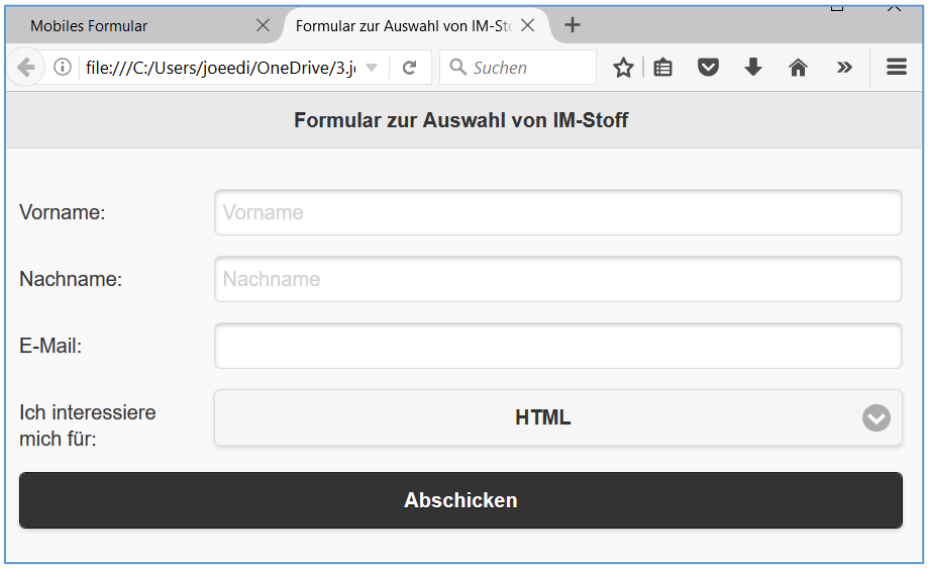
Der blaue Teilbalken wird mit data-highluight="blue" erzeugt. Ansonsten sind beide Teile des Balken grau.

i)Übung Formular

Erstelle eine Datei „4.jqm_formular.html“.

- Verwende data-role=“fieldcontain“
- Nutze einen <header> und ein <conent> aber keinen <footer>
- Verwende für die ganze <page> das Theme „a“.
- Fixiere den <header>
- Überschrift 1. Grades: „Formular zur Auswahl von IM-Stoff“
- <form action=““>; Methode soll „post“ sein
- Benutze bei den Formularfeldern <label for=““>
- Nutze Placeholder für alle Textfelder
- Für die E-Mal verwende den type=“email“
- Verwende ein <select>-Element mit 5 <option>, nämlich HTML, Joomla, JavaScript, Video und Audio.
- Darunter das „Absenden“-Element, d.h. <submit>. Dafür verwende das Theme „b“.

Ziel:



Weitere Übung:

erstelle eine jQuery-Mobile Datei mit folgendem Ergebnis: