Input - Output in Javascript 2
Die ersten Formulare

Jobst-Hartmut Lüddecke

22. April 2013

Zusammenfassung

In dieser 6. Lektion verdichten sich noch weiter die bisher gelernten *Datentypen* und *Grundstrukturen*. Wer immer noch nicht *Schleifen* und *Fallunterscheidungen* auf der *Mandoline spielen kann*, hat einen sehr starken Bedarf an Nach- und Aufarbeitung. Auch die verschiedenen Arten von *Variablen* und *Arrays* müssen jetzt *intus* sein, sonst klappt es nicht mit Formularen.

Der Schwerpunkt der heutigen Lektion wird die Besprechung eines einfachen Formulars mit einigen Angaben einer Kreditkarte sein. Diese Angaben sind nicht vollständig, aber charakteristisch. Auf die Idee bin ich gekommen, weil ich bei vielen Internet-Shops nicht mit meiner *Amex*-Karte bezahlen konnte, weil der Programmierer offensichtlich nicht nachgedacht hat und unterschiedliche Arten von Kartennummern nicht bei seinem Formular berücksichtigt hat. Weiterhin steckt beim Feld für den Kontoinhaber eine Möglichkeit drin, auf zulässige Zeichen in diesem Feld zu prüfen. Nicht mit drin sind die Prüfsummen und das Ablaufdatum der Karte. Dies können wir später noch mit weiteren Typen von Eingabefeldern ausbauen.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Formulare

Formulare sind ein Mittel der Eingabe, bei dem *HTML* und *Javascript* Hand in Hand gehen.

Zunächst ist mit dem *HTML-Befehl* `<form>` ein *Formular* zu definieren. Mit weiteren *Optionen* im *form-tag* ist der *Name* des Formulars, die *Methode der Übertragung* und die aufzurufende *Javascript-Funktion* festzulegen.

```
<form name="karte" method="post" onSubmit="return kartentest()">
...
</form>
```

Abbildung 1: Beispiel einer Formular-Definition

In unserem Beispiel hat das *Formular* jetzt den Namen *karte*, es wird *nach* dem Ausfüllen *(post)* und der Betätigung des *Absendeknopfes (submit)* an die Funktion *kartentest* übergeben.

In *Javascript* kann nun mit `document.karte...` auf die *HTML*-Seite (document), dem Formular mit Namen *karte* und weiteren Teilen darunter zugegriffen werden.

Heute wollen wir die Eingabefelder vom Typ *Text* und vom Typ *Radio-Button* näher betrachten. Das Eingabefeld vom Typ *Checkbox* wird angerissen, da es starke Gemeinsamkeiten mit den *Radio-Button* hat.

Weiterhin kommen *Buttons* vom Typ *submit* zum Abschicken des Formulars und vom Typ *reset* zum Löschen der Formularinhalte im Beispiel vor. Diese sind allerdings so einfach, dass sie keinen eigenen Unterpunkt zur Erklärung benötigen.

1.1 Textfeld

Das einfachste Element eines Formulars ist ein *Textfeld* von einer Zeile.

```
<input type="text" size="40" maxlength="40" name="nummer">
```

Abbildung 2: Beispiel einer Textfeld-Definition

Die Optionen haben folgende Bedeutung:

type : Datentyp dieses Feldes. In diesem Fall *text*.

size : **Sichtbare Länge des Eingabefeldes**. In diesem Fall ist es 40 Zeichen lang. Dies ist nicht die mögliche Länge der Eingabe. Ist diese größer, wird die Eingabe im Eingabefeld horizontal *gescrollt* (verschoben).

maxlength : Diese Option bestimmt die **maximale Anzahl der eingebbaren Zeichen**. Will man also, wie im vorliegendem Beispiel, nicht mehr als 40 Zeichen zulassen ist *maxlength* zu setzen. Die o.a. Option *size* bestimmt nur die *sichtbaren Zeichen*. Natürlich sollten beide Optionen sinnvoll korrespondieren.

name : Ist der **Bezeichner dieses Feldes** innerhalb des Formulars. Ohne Namen können wir nicht auf dieses Eingabefeld zugreifen.

Nun kann man auf verschiedene Arten und und auf verschiedene Informationen dieses Eingabefeldes zugreifen:

document.karte.nummer : ist die Variable *nummer* im Formular *karte* auf der aktiven HTML-Seite, also die Speicherplatz-Adresse.

document.karte.nummer.value : ist der *value* (Wert), also der Inhalt, bzw. der Text im Feld *nummer* im Formular *karte* auf der *HTML*-Seite.

document.karte.nummer.value.length : ist die Länge des Textes, also die Anzahl der eingegebenen Zeichen im Feld *nummer* im Formular *karte* auf der *HTML*-Seite.

document.karte.nummer.value[i] : da dies eine **Textvariable** ist, kann über den **Index** auf jedes **einzelne Zeichen** zwischen 0 und der Obergrenze zugegriffen werden.

1.2 Radio-Button

Radio-Button sind *Knöpfchen* von denen nur **genau eins** gedrückt werden kann. Hiervon kann es im Formular mehrere mit gleichem Namen (*name*), aber unterschiedlichen Werten (*value*) geben, aber nur das *gedrückte Knöpfchen* ist *wahr*.

Die Menge der definierten *Knöpfchen* ist ein mit dem Index 0 beginnendes *Array*. Die *Obergrenze* kann man wieder mit *length*-Funktion ermitteln und hat damit das Ende seiner *Schleife*. Die Funktion *checked* liefert den Wert *true*, wenn dieser Button gedrückt ist und *false* bei nicht gedrückten Knopf. Zum Schluss liefert noch die Funktion *value* den Inhalt des Button. Dieser Inhalt interessiert uns aber nur für den *gedrückten Knopf*.

Nun gilt es mit Schleifen und If-Abfragen den Wert des gedrückten Knopfes zu ermitteln. Klingt alles recht schwer, aber hinten im Beispiel ist es deutlich beschrieben.

1.3 Checkbox

Eine Abwandlung vom *Radio-Button* ist die *Checkbox*. Das *Handling* entspricht dem *Radio-Button*. Der wesentliche Unterschied liegt darin, dass beim *Radio-Button* nur genau **ein Kopf** oder **kein Knopf** gedrückt werden kann. Bei der *Checkbox* können **beliebig viele** Knöpfe gedrückt, oder mit *Häkchen versehen* (checked) sein. Entsprechend muss die Auswertung modifiziert werden.

2 Beispiel Kreditkarten

```
<html>
<head>
<title>Eingabeformular fuer Kreditkarten</title>

<meta http-equiv="content-type" content="text/html;CHARSET=iso8859-2">
<meta http-equiv="expires" content="0">

<meta name="AUTHOR" content="Jobst-Hartmut Lueddecke">
<meta name="description" lang="de"
    content="Beispiel fuer ein Eingabeformular fuer Kreditkarten
            und der Ueberpruefung mit Javascript">
<meta name="keywords"
    content=" Jobst Hartmut Lueddecke Eingabemaske Kreditkarten ">
<meta name="ROBOTS" content="noindex nofollow">
<meta name="LANGUAGE" content="de">
<meta name="COPYRIGHT" content="Jobst-Hartmut Lueddecke">
<meta name="PAGE-TOPIC" content="Beispiel">
<meta name="AUDIENCE" content="all">
<meta name="REVISIT-AFTER" content="10 days">

<meta name="DC.Title"
    content="Beispiel Eingabemaske fuer Kreditkarten">
<meta name="DC.Creator" content="Jobst-Hartmut Lueddecke">
<meta name="DC.Subject" content="Lehrbeispiel">
<meta name="DC.Description"
    content="Beispiel für ein Eingabeformular für Kreditkarten
                und der Überprüfung mit Javascript">
<meta name="DC.Publisher" content="Jobst-Hartmut Lueddecke">
<meta name="DC.Contributor" content="">
<meta name="DC.Date" content="2005-11-14">
<meta name="DC.Type" content="text">
<meta name="DC.Identifier"
      content="javascript/karte.html">
<meta name="DC.Source" content="">
<meta name="DC.Language" content="de">
<meta name="DC.Coverage" content="Lehrbeispiel">
<meta name="DC.Rights" content="Alle Rechte liegen beim Autor">

<link rel=stylesheet type="text/css"
      href="style/lueddecke.css">
```

Abbildung 3: Einfaches Eingabeformular für Kreditkarten. Wie immer der Vorspann.

```
<script language="JavaScript">

function kartentest(karte)
{
    var i;

    var kartenart;
    var kartentyp;
    var num1 = document.karte.nummer1.value;
    var num2 = document.karte.nummer2.value;
    var num3 = document.karte.nummer3.value;
    var num4 = document.karte.nummer4.value;
    var nummer_ok = false;
    var karteninhaber = document.karte.inhaber.value;
    var msg;
    var msg_inhaber;

// Herausfinden welcher Knopf gedrückt (checked) ist und den möglichen
// Wert zuweisen. Es sind im Formular 3 Möglichkeiten für den Namen
// "art" vorgegeben und man kann "art" damit als array mit den Indices
// von [0] bis [2] betrachten. Der Wert steckt für alle im art[i].value
// und die Gültigkeit wird durch das art[i].checked bestimmt.
// Die Obergrenze des arrays steht in document.karte.art.length.

    for(i=0; i<document.karte.art.length; i++)
        {
            if(document.karte.art[i].checked)
                {
                    kartenart = document.karte.art[i].value;
                }
        }

// Gültige Werte sind "a", "m" und "v". Wurde kein Knopf gedrückt,
// gilt "default".

    switch(kartenart)
        {
            case "a":   kartentyp="American Express"; break;
            case "m":   kartentyp="Mastercard"; break;
            case "v":   kartentyp="Visa"; break;
            default:    kartentyp="leer"; return false; break;
        }
```

Abbildung 4: Definition der Auswertefunktion. Auswertung welcher Radio-Button für den Kartentyp gedrückt worden ist.

```
// Kartennummer überpüfen: Erste Variante
// ======================================
// Die Kartennummer wird in 4 nummerische Felder zerlegt.
// Dies ist einfacher für die Typ- Und Format-Prüfung.
// Der Typ muss jeweils ein Zahlenfeld sein.
// Das Format bei einer Amex-Karte sind 3 Felder mit 4, 6 und
// 5 Ziffern, das 4. Feld ist leer.
// Das Format bei der Visa-Karte und der Mastercard sind 4 Felder
// mit jeweils genau 4 Ziffern.
//
// 1. Fehleingabe: Eines der ersten 3 Zahlenfelder ist leer
// das 4. Feld muss bei der Amex-Karte leer sein.

    if(num1 ==""||num2 ==""||num3 =="")
        {
            alert("Ohne Nummer geht es nicht!");
            return false;
        }

// 2. Fehleingabe: Keine Zahlen! Dazu bietet sich die Funktion
// "isNaN()" für "is not a Number" an. Mit der zu prüfenden
// Variable als Parameter, liefert die Funktion den Wert "true",
// wenn der Inhalt der Variablen "keine Nummer" ist, also "false",
// wenn wir eine Zahl haben.

    if(isNaN(num1)||isNaN(num2)||isNaN(num3))
        {
            alert("Die Kartennummer besteht nur aus Ziffern!");
            return false;
        }
    else
        {
            // Bis hierher ist alles ok, jetzt kommen die
            // Besonderheiten der einzelnen Kartenarten dran

            // Amex-Karte mit 3 Feldern der Länge 4,6,5
            if(kartenart=="a"
                &&num1.length==4
                &&num2.length==6
                &&num3.length==5
                &&num4.length==0)
                {
                    alert("korrektes Format American Express Karte");
                    nummer_ok = true;
                }
```

Abbildung 5: Auswertung des Nummernschemas für Amex-Karten.

```
            // Visa und Master mit 4 Feldern der Länge 4,4,4,4
            // ausserdem muss das 4. Feld eine Zahl sein.
            if((kartenart=="v" || kartenart=="m")
                &&num1.length==4
                &&num2.length==4
                &&num3.length==4
                &&num4.length==4
                && !(isNaN(num4))
                )
                {
                    alert("korrektes Format der Visa/Master Karte");
                    nummer_ok = true;
                }
        }

// Karteninhaber überprüfen:
// =========================
// 1. Fehlermöglichkeit:
// "Keine Eingabe" bzw. "leere Eingabe" rausfischen

    if(karteninhaber =="")
        {
            msg_inhaber = "Es geht um Ihre Kreditkarte und nicht ";
            msg_inhaber += "um ein Schweizer Nummernkonto!\n\n";
            msg_inhaber += "Also bitte mit dem Namen des Inhabers ";
            msg_inhaber += "der Karte!\n";
            alert(msg_inhaber);
            return false;
        }
```

Abbildung 6: Auswertung des Nummernschemas für Visa-Karten und Auswertung des Kontoinhabers auf leere Eingabe.

```
// 2. Zulässige Zeichen sind die Buchstaben "A" bis "Z" oder die
// Zeichen ".", "-", " " (Blank). Alle anderen Zeichen sind ein
// Fehler!
//
// Dazu ist die ganze Eingabe in "document.karte.inhaber.value"
// Zeichen für Zeichen abzuklappern. Die Länge der Eingabe und
// damit die Obergrenze der Schleife steht in
//          "document.karte.nummer.value.length".
//
// Der eingegebene Text steht in "document.karte.inhaber.value"
// und wie bei jeder Textvariablen kann auf die einzelnen Zeichen
// über einen Index zugegriffen werden. Hier
//          "document.karte.inhaber.value[i]"

    for(i=0;i<document.karte.inhaber.value.length;++i)
        {
            // Test-Ausgabe
            // alert(document.karte.inhaber.value[i]);
            test = document.karte.inhaber.value[i];
            if((test >= "A" && test <= "Z")
                || test == "."
                || test == "-"
                || test == " "
                )
                {
                    // Korrektes Namensformat weitertesten
                    // Test-Ausgabe
                    // alert("Korrektes Namensformat");
                }
            else{
                    // Falsches Namensformat und Abpfiff
                    msg = "Es können nur große Buchstaben\n";
                    msg += "Bindestriche und Leerzeichen\n";
                    msg += "im Feld Kontoinhaber vorkommen\n";
                    alert(msg);
                    return false;
                }
        }
```

Abbildung 7: Auswertung des Kontoinhabers auf zulässige Zeichen.

```
// Zusammenbau und Ausgabe der Kontrollmeldung
// ==========================================
// Achtung: "nummer_ok" ist eine boolsche Variable
// und die Bedingung ist erfüllt, wenn "nummer_ok"
// den Wert "true" hat.

if(nummer_ok)
    {
        msg = "Sie Zahlen mit Ihrer ";
        msg += kartentyp;
        msg += " Karte \n";
        msg += "mit der Nummer: ";
        msg += num1 + " ";
        msg += num2 + " ";
        msg += num3 + " ";
        msg += num4 + " ";
        msg += " \n";
        msg += "Kontoinhaber: ";
        msg += karteninhaber;
        msg +=" \n\n";
        alert(msg);
    }
else{
        alert("Dies ist keine korrekte Nummer für diesen Kartentyp!");
        return false;
    }
}
</script>
</head>
```

Abbildung 8: Zusammenbau und Ausgabe der Kontroll-Meldung.

```
<body>
<h2>Bitte geben sie hier Ihre Kartendaten ein:</h2>
<form name="karte" method="post" onSubmit="return kartentest()">
<input type="radio" name="art" VALUE="a">American Express<br>
<input type="radio" name="art" VALUE="m">Mastercard<br>
<input type="radio" name="art" VALUE="v">Visa<br>
<br>
<table border="0">
<tr>
<td>Kartennummer:</td>
    <td>
        <input type="text" size="4" name="nummer1">
        <input type="text" size="6" name="nummer2">
        <input type="text" size="5" name="nummer3">
        <input type="text" size="4" name="nummer4">
    </td>
</tr>
<tr>
<td>Kontoinhaber:</td>
<td><input type="text" size="40" name="inhaber"></td>
</tr>
<tr>
<td>Formular:</td>
<td><input type="submit" value="abschicken">
    <input type="reset" value="l&ouml;schen"></td>
</tr>
</table>
</form>
</body>
</html>
```

Abbildung 9: Der HTML-Body mit dem eigentlichen Formular.

3 Lösungen

3.1 Vokale zählen

Man nehme das Segment aus der Prüfung der e-Mail-Adresse, wo auf ein @-Zeichen geprüft wird und erweitere dies zu einem switch.

```
// Eingabe nach @ durchsuchen
    email=false;
    for(i=0; i<mailadr.length; i++)
        {
            test=mailadr[i];
            if(test=="@")
                {
                    email=true;
                }
        }
```

Abbildung 10: Segment Prüfung auf @

```
// Eingabe prüfen und a, e, i, o, u zählen. Dabei sollen die großen
// Vokale gleichberechtigt mit den kleinen Vokalen gezählt werden.
// Da wir auch i zählen, nehmen wir als Index die Variable zeiger.

    var zeiger;
    var a = 0;
    var e = 0;
    var i = 0;
    var o = 0;
    var u = 0;
    for(zeiger=0; zeiger<eingabe.length; zeiger++)
        {
            test=eingabe[zeiger];
            switch(test){
                case "A" :  a++; break;
                case "a" :  a++; break;
                case "E" :  e++; break;
                case "e" :  e++; break;
                case "I" :  i++; break;
                case "i" :  i++; break;
                case "O" :  o++; break;
                case "o" :  o++; break;
                case "U" :  u++; break;
                case "u" :  u++; break;
                default  :  break;
            }
        }
```

Abbildung 11: Segment Zählen von a, e, i, o, u