Esercizi per il corso di Fondamenti di Informatica 2

Corso di Laurea in Ingegneria Gestionale – Savona
Armando Tacchella

Esercizio 1

Creare una tabella con due colonne:

- La colonna di sinistra contiene immagini (una per riga)
- La colonna di destra contiene una descrizione sintetica
- Facendo click sull'immagine o su parte della descrizione sintetica, deve aprirsi un'altra finestra con una descrizione più complessa

Esercizio 2

Creare una pagina HTML suddivisa in 2 frame:

- La frame di sinistra contiene una pagina con una serie di link, per es., pagina1, pagina2, pagina3, ecc. che si collegano ad altrettante pagine HTML
- La frame di destra contiene le pagine richiamate tramite i link sulla pagina visualizzata nel frame di sinistra

Sperimentare con diverse dimensioni delle frame e provare ad organizzare la pagina utilizzando una suddivisione orizzontale anziché verticale

Esercizio 3

Utilizzando i tag per le tabelle e per i moduli, costruire un modulo per la richiesta di dati personali:

- Nome e cognome utilizzando dei campi testo
- Età, utilizzando tre selezioni multiple: giorni, mesi, anni
- Sesso, utilizzando due bottoni "radio"
- Stato civile, utilizzando due bottoni "radio"
- Hobby preferito, dando una serie di scelte con "checkbox"

Esercizio 4

Provare gli esempi di JavaScript visti a lezione. Modificare le impostazioni di partenza e sperimentare nuove configurazioni.

Esercizio 5

Scrivere la seguente pagina web ed eseguirla utilizzando Explorer:

```
<HTML>
<BODY>
<SCRIPT>
document.write("<h2>Tabella dei fattoriali</h2>");
for(i = 1, fact = 1; i < 10; i++, fact *= i) {
    document.write(i + "! = " + fact);
    document.write("<br>");
}
</SCRIPT>
</BODY>
</HTML>
```

Modificare il codice JavaScript in modo da stampare la tavola dei fattoriali:

- Tramite un elenco puntato
- Tramite un elenco numerato
- Tramite una tabella

Esercizio 6

Scrivere la seguente pagina web ed eseguirla utilizzando Explorer:

```
<HTML>
<HEAD>
<SCRIPT>
function print_todays_date() {
        var d = new Date();                  // Data odierna
        document.write(d.toLocaleString());
}
</SCRIPT>
</HEAD>
<BODY>
<HR>Data e ora:<BR><B>
<SCRIPT>
print_todays_date();
</SCRIPT>
</B><HR>
</BODY>
</HTML>
```

Esercizio 7

Introdurre il seguente codice in una pagina web:

```
<FORM>
<INPUT TYPE="button"
       VALUE="Click here"
       onClick="alert('You clicked the button')">
</FORM>
```

Provare a cambiare il gestore di onClick con una funzione JavaScript che cambia il titolo della finestra del browser.

Esercizio 8

Il seguente codice JavaScript esegue diversi compiti:

- Dichiara una funzione report_error che viene invocata in caso di errore
- Installa la funzione report_error come gestore di errori JavaScript
- Provoca un errore in modo da causare l'invocazione di report_error

Analizzare il codice cercando di individuare le varie parti e il relativo funzionamento. Introdurre il codice in una pagina e sperimentare variazioni sullo stesso.

```
<script>
var error_count = 0;
function report_error(msg, url, line)
{
   var w = window.open("",                    // URL
                       "error"+error_count++, // nome unico
                       "resizable,status,width=625,height=400");
   var d = w.document;
   d.write('<DIV align=center>');
   d.write('<FONT SIZE=7 FACE="helvetica"><B>');
   d.write('OOPS.... A JavaScript Error Has Occurred!');
   d.write('</B></FONT><BR><HR SIZE=4 WIDTH="80%">');
   d.write('<FORM ACTION="mailto:email@fake.org" METHOD=post');
   d.write(' ENCTYPE="text/plain">');
   d.write('<FONT SIZE=3>');
   d.write('<I>Click the "Report Error" button to send a bug
report.</I><BR>');
   d.write('<INPUT TYPE="submit" VALUE="Report Error">  ');
   d.write('<INPUT TYPE="button" VALUE="Dismiss"
onClick="self.close()">');
   d.write('</DIV><DIV align=right>');
   d.write('<BR>Your name <I>(optional)</I>: ');
   d.write('<INPUT SIZE=42 NAME="name" VALUE="">');
   d.write('<BR>Error Message: ');
   d.write('<INPUT SIZE=42 NAME="message" VALUE="' + msg + '">');
   d.write('<BR>Document: <INPUT SIZE=42 NAME="url" VALUE="' + url +
'">');
   d.write('<BR>Line Number: <INPUT SIZE=42 NAME="line" VALUE="' + line
+'">');
   d.write('<BR>Browser Version: ');
```

```
   d.write('<INPUT SIZE=42 NAME="version" VALUE="'+navigator.userAgent
+ '">');
   d.write('</DIV></FONT>');
   d.write('</FORM>');
   d.close();
   return true;
}
self.onerror = report_error;
</script>

<script>
self = null;
</script>
```

Esecizio 9

Il seguente codice HTML e JavaScript consente di generare una form e gestirne gli eventi. Analizzare il codice per comprenderne il funzionamento e sperimentare variazioni sullo stesso.

```
<FORM NAME="everything">  <!-- A one-of-everything HTML form... -->
 <TABLE BORDER CELLPADDING=5>   <!-- ...in a big HTML table. -->
   <TR>
     <TD>Username:<BR>[1]<INPUT TYPE=text NAME="username" SIZE=15></TD>
     <TD>Password:<BR>[2]<INPUT TYPE=password NAME="password"
SIZE=15></TD>
     <TD ROWSPAN=4>Input Events[3]<BR>
       <TEXTAREA NAME="textarea" ROWS=20 COLS=28></TEXTAREA></TD>
     <TD ROWSPAN=4 ALIGN=center VALIGN=center>
       [9]<INPUT TYPE=button VALUE="Clear" NAME="clearbutton"><BR>
       [10]<INPUT TYPE=submit NAME="submitbutton" VALUE="Submit"><BR>
       [11]<INPUT TYPE=reset NAME="resetbutton"
VALUE="Reset"></TD></TR>
   <TR>
     <TD COLSPAN=2>Filename: [4]<INPUT TYPE=file NAME="file"
SIZE=15></TD></TR>
   <TR>
     <TD>My Computer Peripherals:<BR>
       [5]<INPUT TYPE=checkbox NAME="peripherals" VALUE="modem">28.8K
Modem<BR>
       [5]<INPUT TYPE=checkbox NAME="peripherals"
VALUE="printer">Printer<BR>
       [5]<INPUT TYPE=checkbox NAME="peripherals" VALUE="tape">Tape
Backup</TD>
     <TD>My Web Browser:<BR>
       [6]<INPUT TYPE=radio NAME="browser" VALUE="nn">Netscape
Navigator<BR>
       [6]<INPUT TYPE=radio NAME="browser" VALUE="ie">Internet
Explorer<BR>
       [6]<INPUT TYPE=radio NAME="browser"
VALUE="other">Other</TD></TR>
   <TR>
     <TD>My Hobbies:[7]<BR>
       <SELECT multiple NAME="hobbies" SIZE=4>
         <OPTION VALUE="programming">Hacking JavaScript
```

```
          <OPTION VALUE="surfing">Surfing the Web
          <OPTION VALUE="caffeine">Drinking Coffee
          <OPTION VALUE="annoying">Annoying my Friends
        </SELECT></TD>
      <TD align=center valign=center>My Favorite Color:<BR>[8]
        <SELECT NAME="color">
          <OPTION VALUE="red">Red        <OPTION VALUE="green">Green
          <OPTION VALUE="blue">Blue      <OPTION VALUE="white">White
          <OPTION VALUE="violet">Violet  <OPTION VALUE="peach">Peach
        </SELECT></TD></TR>
 </TABLE>
</FORM>
<DIV ALIGN=center>        <!-- Another table--the key to the one above.
-->
  <TABLE BORDER=4 BGCOLOR=pink CELLSPACING=1 CELLPADDING=4>
    <TR>
      <TD ALIGN=center><B>Form Elements</B></TD>
      <TD>[1] Text</TD>  <TD>[2] Password</TD>  <TD>[3] Textarea</TD>
      <TD>[4] FileUpload</TD> <TD>[5] Checkbox</TD></TR>
    <TR>
      <TD>[6] Radio</TD>  <TD>[7] Select (list)</TD>
      <TD>[8] Select (menu)</TD>  <TD>[9] Button</TD>
      <TD>[10] Submit</TD>  <TD>[11] Reset</TD></TR>
  </TABLE>
</DIV>
<SCRIPT LANGUAGE="JavaScript1.1">
// This generic function appends details of an event to the big
Textarea
// element in the form above. It will be called from various event
handlers.
function report(element, event)
{
    var t = element.form.textarea;
    var name = element.name;
    if ((element.type == "select-one") || (element.type == "select-
multiple")){
        value = " ";
        for(var i = 0; i < element.options.length; i++)
            if (element.options[i].selected)
                value += element.options[i].value + " ";
    }
    else if (element.type == "textarea") value = "...";
    else value = element.value;
    var msg = event + ": " + name + ' (' + value + ')\n';
    t.value = t.value + msg;
}
// This function adds a bunch of event handlers to every element in a
form.
// It doesn't bother checking to see if the element supports the event
handler,
// it just adds them all. Note that the event handlers call report()
above.
function addhandlers(f)
{
    for(var i = 0; i < f.elements.length; i++) {
        var e = f.elements[i];
        e.onclick = new Function("report(this, 'Click')");
```

```
        e.onchange = new Function("report(this, 'Change')");
        e.onfocus = new Function("report(this, 'Focus')");
        e.onblur = new Function("report(this, 'Blur')");
        e.onselect = new Function("report(this, 'Select')");
    }
    // Special case handlers for the buttons:
    f.clearbutton.onclick =
        new Function("this.form.textarea.value=''; report(this,
'Click');");
    f.submitbutton.onclick =
        new Function("report(this, 'Click'); return false");
    f.resetbutton.onclick =
        new Function("this.form.reset(); report(this, 'Click'); return
false");
}
// Activate our form by adding all possible event handlers!
addhandlers(document.everything);
</SCRIPT>
```

Esercizio 10

Il seguente esempio riguarda la convalida dei dati di una form. Analizzare il codice cercando di comprendere le funzionalità dei vari componenti. Riportare il codice su una pagina web e sperimentare variazioni dello stesso.

```
<SCRIPT LANGUAGE="JavaScript1.1">
// A utility function that returns true if a string contains only
// whitespace characters.
function isblank(s)
{
    for(var i = 0; i < s.length; i++) {
        var c = s.charAt(i);
        if ((c != ' ') && (c != '\n') && (c != '\t')) return false;
    }
    return true;
}
// This is the function that performs form verification. It will be
invoked
// from the onSubmit() event handler. The handler should return
whatever
// value this function returns.
function verify(f)
{
    var msg;
    var empty_fields = "";
    var errors = "";


    // Loop through the elements of the form, looking for all
    // text and textarea elements that don't have an "optional"
property
    // defined. Then, check for fields that are empty and make a list
of them.
    // Also, if any of these elements have a "min" or a "max" property
defined,
```

```
    // then verify that they are numbers and that they are in the right
range.
    // Put together error messages for fields that are wrong.
    for(var i = 0; i < f.length; i++) {
        var e = f.elements[i];
        if (((e.type == "text") || (e.type == "textarea")) &&
!e.optional) {
            // first check if the field is empty
            if ((e.value == null) || (e.value == "") ||
isblank(e.value)) {
                empty_fields += "\n          " + e.name;
                continue;
            }
            // Now check for fields that are supposed to be numeric.
            if (e.numeric || (e.min != null) || (e.max != null)) {
                var v = parseFloat(e.value);
                if (isNaN(v) ||
                    ((e.min != null) && (v < e.min)) ||
                    ((e.max != null) && (v > e.max))) {
                    errors += "- The field " + e.name + " must be a
number";
                    if (e.min != null)
                        errors += " that is greater than " + e.min;
                    if (e.max != null && e.min != null)
                        errors += " and less than " + e.max;
                    else if (e.max != null)
                        errors += " that is less than " + e.max;
                    errors += ".\n";
                }
            }
        }
    }
    // Now, if there were any errors, then display the messages, and
    // return true to prevent the form from being submitted. Otherwise
    // return false.
    if (!empty_fields && !errors) return true;
    msg  = "_____________________________________________________\n\n"
    msg += "The form was not submitted because of the following
error(s).\n";
    msg += "Please correct these error(s) and re-submit.\n";
    msg += "_____________________________________________________\n\n"
    if (empty_fields) {
        msg += "- The following required field(s) are empty:"
                + empty_fields + "\n";
        if (errors) msg += "\n";
    }
    msg += errors;
    alert(msg);
    return false;
}
</SCRIPT>
<!-----------------------------------------------------------------
---
     Here's a sample form to test our verification with. Note that we
     call verify() from the onSubmit() event handler, and return
whatever
     value it returns. Also note that we use the onSubmit() handler as
```

```
     an opportunity to set properties on the form objects that verify()
     will use in the verification process.
-------------------------------------------------------------------------
--->
<FORM onSubmit="
    this.firstname.optional = true;
    this.phonenumber.optional = true;
    this.zip.min = 0;
    this.zip.max = 99999;
    return verify(this);
">
First name: <INPUT TYPE=text NAME="firstname">
Last name: <INPUT TYPE=text NAME="lastname"><BR>
Address:<BR><TEXTAREA NAME="address" ROWS=4 COLS=40></TEXTAREA><BR>
Zip Code: <INPUT TYPE=text NAME="zip"><BR>
Phone Number: <INPUT TYPE=text NAME="phonenumber"><BR>
<INPUT TYPE=submit>
</FORM>
```