Übung 2 – Mensch-Machine-Interaktion 2

Interaction Tracking With JavaScript

URLs of further resources and source code for this exercise are available from
http://murx.medien.ifi.lmu.de/~richard/mmi2/form

1 JavaScript (Ecmascript)

(Execises session)

Acquaint yourself with the basic facts about the following features of JavaScript:

- Syntax and semantics: Functions, global/local variables, automatic type conversion etc.
- Embedding JavaScript code in HTML documents (using `<script>` or attribute values)
- Traversing the HTML DOM (document object model) tree
- Changing values in the document tree
- User-generated events: `onfocus`, `onmouseover` etc.
- Useful debugging techniques: Changing the status bar text, creating popup messages
- Usage of the Mozilla/Firefox DOM inspector and JavaScript console (very useful for writing/debugging JavaScript code)

2 Tracking Interactions in HTML Forms

(Per-group homework, 2 weeks)

To evaluate the usability of a complex web form, it can be of interest to collect information about how people fill in the forms. In particular, it is interesting

- in what order form fields are filled in
- how much time is spent on a particular input field
- whether the contents of an already filled-in field are changed later
- whether copy & paste is used

Task: Without changing its appearance, extend a form and add the capability to track user interaction. It should be tracked when a form element is changed, when the mouse is placed on a particular element and when text is selected in a text field. The information should be stored and transmitted along with the "normal", user-entered data when the form is submitted.

This technique can be used to perform a usability test without additional infrastructure at the client side. However, it is important to inform the the users that they are tracked. A similar approach may be used to monitor regular usage and provide help depending on the users' actions.

For your work, choose any already existing form with 10 or more input elements that is available from a website. Samples of forms:

- https://ovr.ss.ca.gov/votereg/OnlineVoterReg
- http://www.google.com/advanced_search?hl=en
- http://www.google.de/search?q=conference+submission

Before experimenting with a form, change the `action` attribute of the `<form>` element to the following URL: http://murx.medien.ifi.lmu.de/~richard/mmi2/http-get-post
This script will output a list of all values submitted by the form.

Submission:

- Solutions must include the modified HTML code for the form you chose, the URL of the original form, and a short description of how the code works
- Your JavaScript code must work with Firefox (and should work with Internet Explorer).
- All important aspects of the JavaScript/HTML must be commented.
- Submission is by email to mmi2.2005@hcilab.org
  Please use an HTML attachment named `uebung2-gruppeN.html` (N is the number of your group) and send the description in the main body of the mail.
- Each group must hand in one solution. Please state if anyone has left your group.
- Also state in the mail the date of the execises session where you want to present your work (Friday, May 13th or Wednesday, May 18th). At least three group members must be present, all group members must be able to explain the group's solution.
- Deadline for email submission: **Thursday, May 12th 2005, 8 a.m.**

Example:
(Complete document available as http://murx.medien.ifi.lmu.de/~richard/mmi2/form)

```
...
  <script language="JavaScript" type="text/javascript">
  <!-- // Comment prevents very old browsers from displaying script

  // Return string of the form "2004-12-31,23:59:59"
  function datestamp() {
    var now = new Date(); // "var" => is local variable
    return now.getFullYear() + "-" + (now.getMonth()+1) + "-"
      + now.getDate() + "," + now.getHours() + ":" + now.getMinutes()
      + ":" + now.getSeconds();
  }

  logVal = ""; // Initialised when page loads. Contains entire log of actions

  // Called from the form elements below to log things with a timestamp
  // @param domObject Reference to a node in the DOM tree
  // @param text String to write to log
  function writeLog(domObject, text) {
    var objectId = domObject.getAttribute("id");
    var logLine = datestamp() + " " + objectId + " " + text;
    window.status = logLine; // Display logLine in status bar
    logVal = logVal + logLine + "\n"; // Add logLine to interaction log
  }
  // -->
  </script>

...

<form name="mmiform" method="post"
    action="http://murx.medien.ifi.lmu.de/~richard/mmi2/http-get-post"
    onsubmit="document.mmiform.log.value = logVal">
  <!-- Example form element. For the JavaScript code in the "onfocus"
       etc attribute values, the value of "this" corresponds to the
       DOM node for <input> -->
  <input id="i1" name="i1" type="text" value=""
    onfocus="writeLog(this, 'focus')"
    onchange="writeLog(this, 'change')"
    onselect="writeLog(this, 'select')"
    onmouseover="writeLog(this, 'mouseover')">
  <!-- Hidden form element which we use to transmit our log back to
       the server -->
  <input type="hidden" name="log" value="">
  <input type="submit" value="Send">
</form>
```