Interfaces de Usuario

Grado en Ingeniería Informática

Course 2017 / 2018

Programming exercises

(Assignments)

TECHNOLOGIES FOR DEVELOPING WEB USER INTERFACES

Client script languages (JavaScript & jQuery)

Table of contents

1. Introduction

This document presents the **second set of programming exercises**. These exercises aim to provide you with a hands-on introduction to a script language designed to execute code in web navigators. In particular, we will provide a short introduction to the JavaScript language and describe three programming exercises. This section describes how the exercises will be carried out (estimated effort, date of submission), assessed (assessment criteria), and submitted.

Estimated effort

The estimated number of hours each student will need to devote to completing the exercises is 12, which will be distributed along **four** working weeks and sessions. The sessions will take place in small lecture rooms, wherein the lecturers will (i) introduce JavaScript to the students, (ii) go through a number of examples and (iii) solve general doubts or concerns.

Submission

The programming exercises will be submitted on the **8th week of the course** (see the timetable of lab sessions in Aula Global). The submission will consist of one exercise. The exercise to be submitted will be specified at the start of the submission session.

Assessment

In this set of exercises, we will evaluate: **functionality** (i.e. what you are asked to do), **code** (e.g. can a person who is not you understand the code? Have you written comments in the code?) and **user interface design** (e.g. think about a human user interacting with your page – colors, size of text…)

This set of exercises corresponds to 33% of the final mark of the labs of the course, i.e. 10% of the final mark. Students **will not pass** the course if they either copy the exercises from web pages or from another group, or allow them to copy their exercises.

Organization of the document

This document is divided into three sections. Section 2 gives an overview of Script languages and JavaScript. Section 3 describes the exercises and the submission procedure, along with the assessment criteria.

2. Script languages executed on the client side

Script languages executed on the client allows web developers to run functions and programs in web navigators, and this provides the end-user with a richer interaction with web applications, since these are more dynamic. Script languages are crucial in programming dynamic web applications, whose behavior changes depending on the needs of the end-user, execution conditions or the context of execution (e.g. web navigators). Examples of script languages are VBScript (Visual Basic) and JavaScript. In this document, we will focus on JavaScript.

JavaScript

JavaScript is an interpreted language. JavaScript is also an object-oriented programming language, which is executed in web navigators. The web navigator provides JavaScript with an execution context, which has pre-defined objects representing different elements of the web navigator and the web page. With respect to user interfaces, JavaScript allows us to:

- Modify the text of an HTML document, as it is possible to insert text into a document (e.g. the value of a variable).
- React to events, as it allows web pages to execute JavaScript code depending on user- and navigator-based actions, such as loading a web page or clicking on a button. Event-oriented programming is key to code the dynamic behavior of user interfaces.
- Read and modify HTML labels, since JavaScript enables web developers to add, modify or delete any HTML element. This presents us with an opportunity to modify the structure, content and presentation of a web page. We do this, in JavaScript code, through the DOM (Document Object Model) interface, which enables us to easily manipulate the tree of any HTML document.
- Validate data provided by the end-user. With JavaScript, we can check whether the data provided by the end-user is valid before sending or processing it (the typical example is an online form).

Web navigators might implement different versions of the DOM interface, and this fact often leads to compatibility issues of web pages running JavaScript code. Ensuring the compatibility of web pages in as many web navigators as possible is very important – not all the users go online with the same web navigator. Hence, web developers need to be aware of the (un)supported JavaScript elements in different web navigators

The JavaScript tutorial of the w3c schools provides a number of interactive examples and an extensive reference to DOM objects [1]. For more advanced aspects, [2] provides tutorials, and the books [3] and [4], which are available at Safari Books Online, can be used as reference manuals.

JavaScript libraries

Both an increasing number of functionalities provided by web applications and the need to provide rich interactions in web pages have increased the complexity of developing web interfaces. To solve this problem, a number of JavaScript libraries have been created. These libraries provide us with pre-developed components, which can be used while developing web applications. Examples are Dojo Toolkit, Google Web Toolkit, UI Library and *jQuery*. We will focus on *jQuery*, which is used by companies like Google, Microsoft, IBM and Netflix.

jQuery

jQuery is an open source library designed to help web developers create web interfaces. The aim of *jQuery* can be summarized as: "write less, do more", i.e. to provide web developers with pre-developed components, which allow us to create complex functionalities in one line of code. Moreover, *JQuery* is compatible with nearly all the web navigators we are currently using to access to web pages.

jQuery allows us to change a webpage without re-loading it, thanks to manipulating the DOM object and AJAX events, effects and requests. We use $() o jQuery(). Thus, the syntax consists of a selector to select an HTML element followed by an action: **$(*selector*).*action*()**

Key characteristics of *jQuery* are:

- Interaction with HTML documents: selection and manipulation of DOM components and proprieties defined in CSS.
- Management of HTML events: the controllers of events are methods executed when there is a concrete type of interaction with an HTML document. We often talk about an action being triggered or fired by an event. For example, the instruction **$(document).ready(*function*)** will call the *function* when the document (HTML) is *ready*, i.e. the web page has been loaded.
- Animations: special actions which can be associated to HTML elements. Some examples are
    - **hide() / show()**
    - **slideDown() / slideUp() / slideToggle()**
    - **animate()**

For further information about *jQuery*, we recommend the tutorial provided by w3c schools, which provides a number of interactive examples and an extensive reference to actions, effects and animations [1]. The official page of *jQuery* provides lots of documentation, and can be used as a reference guide [2]. Another reference is a book available at Safari Books online [3].

3. Exercises

This section consists of 3 programming exercises, all of which are considered mandatory. For each exercise, we describe its main objective, suggest examples and provide supporting material.

Description of the exercises

Exercise 1

The aim of this exercise is to get familiar with the main elements of JavaScript for creating a web form for signing up to a website for sharing images.

The webpage will be the same as the first exercise of the first set (HTML5 & CSS3), except for the following changes:

- A section with information about the account will be added before the other two already in the page (personal information and billing information). This new section will contain username and password.
- All fields in the form will be editable except for the email
- At the end of the form, a check for "I have read and agree to terms of use, privacy policy and use of cookies" and two buttons "save" and "cancel" will be added.
- The form will be divided into the following sections and include the following fields:
    - Account information
        - Username (mandatory)
        - Password with a maximum of 8 chars as letters [a-z] and digits [0-9] (mandatory)
    - Personal information:
        - Name and surname (mandatory)
        - Email with the format username@domain.ext (mandatory)
        - Birthday (mm/dd/yyyy) (mandatory)
        - Language (optional) – to select one among the listed languages
        - Profile image (optional)
    - Billing information:
        - Address (mandatory)
        - Payment method (mandatory) – Depending on the selected payment method, the following fields will change as follow
            - Credit card – it will show the card number, the valid thru and the three-digit code
            - PayPal – it will show a link to the PayPal page

        - Bank transfer – it will show the account holder name, account number, and bank name
    - I have read and agree to terms of use, privacy policy and use of cookies (mandatory)
    - Buttons “save” and “cancel”
- When the page is loaded, a modal dialog box will be opened asking for email and password. If a cookie with the same email already exists, the form will be loaded with the information stored in the cookie. If it does not exist, the form will be opened with empty fields, except for email and password.
- Clicking on the "save" button, a cookie will be stored with the email and all the remaining information contained in the form. If a cookie with the same email already exists, it will check if any information contained in the fields has been modified. If so, a modal dialog box will be shown informing about the changes.
- Clicking on the "cancel" button, the form will be reset to its default values.
- The validation process can be implemented using HTML5, JavaScript and jQuery.

We encourage you to look at the examples available at http://www.w3schools.com/js/default.asp to complete this exercise, paying special attention to the JS String and JS Validation entries, and http://www.w3schools.com/html/html5_form_attributes.asp, to know more about HTML5 Form Attributes. This information can be expanded with the DOM object available at: http://www.w3schools.com/js/js_ex_dom.asp

Exercise 2

The aim of this exercise is to get familiar with the main elements of JavaScript to design a web gallery to share images.

The webpage will be the same of the second exercise of the first set (HTML5 & CSS3), except for the following changes:

- When the page is loaded, a modal dialog box will be opened asking for email and password. If a cookie with the same email already exists, the username and the profile image will be shown in the right part of the header. If not, the previous exercise will be opened.
- Clicking on the like and the sharing button, the counter will be increased.
- The other videos in the gallery can be drag and drop to the central part of the page. When dropped, it will be swapped with the video already there.
- Clicking on the “+” icon below each video, a short description will be shown.

We encourage you to look at the following entries available at http://www.w3schools.com/js/default.asp to complete this exercise:

- JS Functions (http://www.w3schools.com/js/js_functions.asp)
- JS Events (http://www.w3schools.com/js/js_events.asp)
- JS HTML DOM (http://www.w3schools.com/js/js_htmldom.asp)
- DOM CSS (http://www.w3schools.com/js/js_htmldom.asp)

Exercise 3

The aim of this exercise is to get familiar with jQuery library and its benefits to develop a web memory game to pair equal images.

The game will include the following steps.

1. First of all, the page will show a form to ask to the user the number N of images to include in the game (limited between 3 and 10) and the time in seconds that the game has to last (limited between 10 and 120).
2. Once the user has specified the number of images, the page will show a board with N*2 boxes, a counter for the number of found pairs, a timer and a "restart" button. Each box can have a background color (the same for all the boxes) or an image. There will be a total of N pairs of boxes with the same image. Initially, all the boxes will show the background color, hiding the images.
3. Clicking on a box, the box will turn around showing the hidden image.
4. After clicking on two boxes, the page will check if the related images are the same. If yes, the counter will be increased and the boxes will be left showing the images. If not, the images will be hidden again showing the background color.
5. If the user is able to pair all the images, the counter will show the N number (total number of pairs), the timer will stop, and a popup window will communicate that the user has won and how long the game has lasted.
6. The timer will decrease each second from the number chosen by the user in step 1. If it arrives to 0 before ending the game, a popup window will communicate that the user has lost the game. **Implementing the timer mechanism is optional.**
7. Clicking on the "restart" button, the game will restart from step 2. In each iteration, the boxes will be reorganized randomly.

We encourage you to look at the examples available at the official page of jQuery (http://jqueryui.com/) and other articles about JavaScript, HTML5 and CSS3 animations.

Material

Whereas no specific editor or tool is required to complete these exercises, we recommend that you use free tools such a Notepad++ and HTML-Kit. We also encourage you to use a JavaScript editor to debug your code, such as Firebug for Firefox. The lecturers will not help the students to use these tools.

4. Norms

The realization and submission of the programming exercises is guided by the following set of rules. If you do not comply with them, your mark **won't be more than 3** in the exercises.

Conducting the exercises

The exercises will be carried out in groups of two people.

The members of each group will belong to the same lab group.

The members of the group cannot be altered throughout the course.

The exercises will be carried out by using HTML5 and CSS3.

The exercises will be tested with either Mozilla Firefox version 16 (or above) or Chrome 26 (or above).

The exercises will be coded to be visualized in screens with resolution = 1024x768

IMPORTANT: The lecturers will not solve problems via e-mail.

Submitting the exercises

The exercises will be submitted **at the beginning of the session** indicated in the introduction of this document. Exercises submitted afterwards will not be considered.

The submission norms are:

- All the files will be submitted through Aula Global.
- All the files will be either zip or rar files, with the following filename:

Ep02_grXX.zip

- XX is the ID of your group. For example, group 5 will submit ep01 as:

Ep02_gr05.zip

The zip or rar files will have the following structure:

- ExN. Root folder. HTML files.
- ExN/style. CSS styles.
- ExN/images. Images and material.

N = number of exercise (1 – 3).

IMPORTANT: Exercises must be submitted as it has been stated before. Other forms of submission will not be considered

.

5. References

[1] "*HTML Tutorial*", HTML Tutorial, W3 Schools: http://www.w3schools.com/html/
[2] "*HTML 4.01 Specification*": http://www.w3.org/TR/html4
[3] "HTML5", http://www.w3.org/TR/html5/
[4] "*CSS Tutorial*", http://www.w3schools.com/css/
[5] "*CSSPlay*", http://www.cssplay.co.uk/index.html
[6] "*Cascading Style Sheets, level 1*", http://www.w3.org/TR/REC-CSS1
[7] "*Cascading Style Sheets, level 2*", http://www.w3.org/TR/REC-CSS2
[8] "HTML5 and CSS3: Visual QuickStart Guide, Seventh Edition", Elizabeth Castro; Bruce Hyslop. Ed. PeachPit Press, 2011