Homework Assignment 4

Objectives:

1. To gain more experience programming client-side JavaScript.
2. To use event handling.
3. To use the DOM to dynamically modify XHTML/HTML documents.
4. To use cookies to store client information locally.

Note: Use Firefox. Microsoft IE does not implement the DOM correctly. You should also make sure your browser accepts or enables cookies and scripted windows for this to work.

Tag Clouds

Introduction

A cookie is a small object of information consisting of a name and a textual value stored on the client's local machine when a client visits a Web page. This information may contain user preferences, authentication information such as a username and password, session information such as the time and date a client accessed this particular site, the number of times a client visited this site, etc. Cookies have a certain lifetime which means they will be removed from your system after their lifetime expires. Your browser may have options to enable or disable cookies from being written to your local system. Your browser may also have options to delete cookies from your local system.

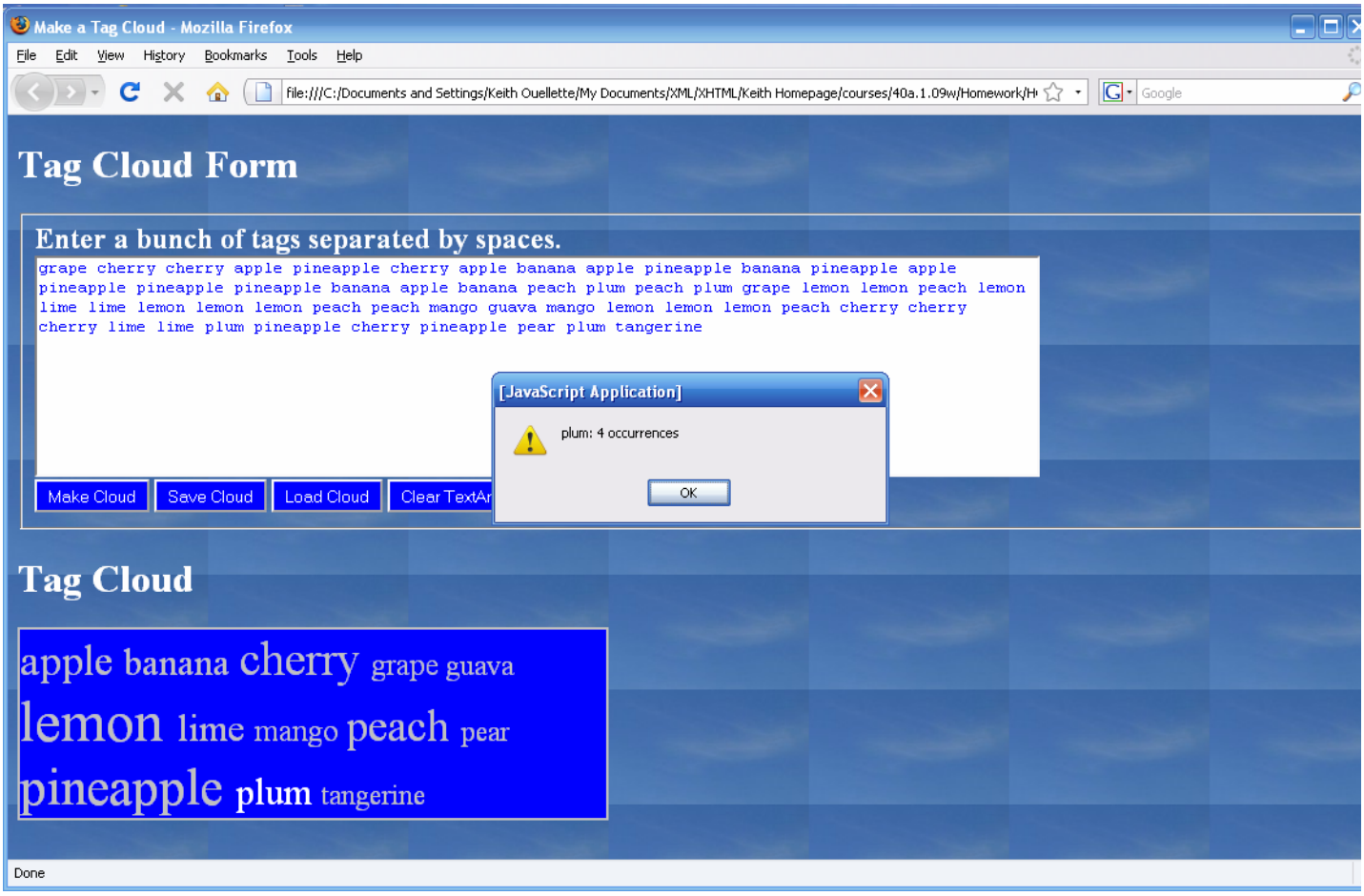
You are going to write a JavaScript script that reads a string of tags separated by spaces from the `textarea`, counts the frequency of each tag with the help of arrays, and dynamically generates a styled XHTML `div` element that will represent a tag cloud when the user clicks the "Make Cloud" button. Each tag in the tag cloud will be the content of a `span` element so that it can be styled individually. Each `span` element will be dynamically generated and added to the document via the JavaScript DOM methods. You will also register an `onclick` event handler with the `span` element that will generate an `alert` dialog displaying the tag name and the number of occurrences of the tag, for example, `"apple: 13 occurrences"`. The script will also provide event handlers for when the user clicks the "Save Cloud", "Load Cloud", and "Clear Textarea" form button controls. In particular, the "Save Cloud" button should save the contents of the `textarea` as a cookie. In particular, the "Load Cloud" button should load the contents of a previously saved cookie into the `textarea`. The "Clear TextArea" button should erase the contents of the `textarea`.

Directions

1. Download the files tagcloud.html, cloud.jpg, and tcstyle.css.
2. Define an external JavaScript file `tagcloud.js` which contains the following functions:
   - `makeCloud()` This function is the event handler for button "Make Cloud". It should
     1. grab the contents of the textarea, a string of tags separated by spaces.
     2. parse that string into an array of tag strings, sorted by lexicographic ordering. (Remember this array may have repeat tags.)
     3. Create two arrays: One array contains unique tags and second corresponding array will contain the frequencies of each of those tags.
     4. The unique tag array and the frequency tag array pair is an example of parallel arrays. If the unique array contains elements apple, banana, cherry in indeces 0, 1 and 2 respectively, then the frequency array will contain elements 5,3,4 (for example) in indices 0,1 and 2. Where 5,3 and 4 are frequencies for apple, banana and cherry respectively.
     5. keep track of the maximum frequency for the set of tags. (Write a helper function to compute this.)
     6. create a `div` element containing all the tags as `span` elements as described above. (You should write a helper fuction to achieve this.)
     7. use the properties of the DOM address for the `div` element to set a `.1em solid silver` border for the `div`. Give the `div` a blue background. a silver foreground color, and an extra large serif font.
     8. loop through all the `span` child elements (the tags) in the `div` (tag cloud) and assign a suitable font size to them. The font size should at least be 15pt. The font size will be determined by taking the frequency at which the tag occurs divided by the max number of occurrences, then the resulting fraction is multiplied by 20. This will give you a decimal number between 0 and 20. Round the number to nearest integer and add 15. Now append the string "pt" and you have a string that is something like "22pt". This string should be used in setting the font-size. (I suggest defining a helper function to do this step.)
     9. Add onclick function to each span element. When you click on the span element it should alert how many times the tag appears. See image below for an example.
     10. remove the `div` element already belonging to `tagcloud.html` and replace it with the new `div` you just created. The new `div` should be in the same position in the DOM tree as the old `div` was.
   - `saveCloud()` This function is the event handler for button "Save Cloud".
   - `loadCloud()` This function is the event handler for button "Load Cloud".
   - `clearArea()` This function is the event handler for button "Clear TextArea".

A reminder to always start from a point where your script works and then modify a little at a time. Use alert dialogs for tracing the flow of your script. Comment out parts that don't work and then gradually uncomment them out a little at a time.

Your application should look like this aftre the user clicks "Make Cloud" and then clicks the plum `span` element:



As always, be sure to validate all your XHTML.

Make sure all of the following files are uploaded to your `public_html` and he relevant files uploaded to `CCLE`:

- `tagcloud.html`
- `tagcloud.js`
- `tcstyle.css` You do not need to upload this to CCLE unless you created your own style
- `cloud.jpg` (This file should be only on your public directory. Do not need to upload to CCLE.)

Grade Breakdown:

Function	Criteria	Points
`tagcloud.js makeCloud()`	steps 1-8	10 points each
`tagcloud.js saveCloud()`		7 points
`tagcloud.js loadCloud()`		7 points
`tagcloud.js clearArea()`		6 points
	Total	100 points