

COMP1073 - CLIENT SIDE SCRIPTING

INSTRUCTOR - DEREK BUTTINEAU

TODO List Web App

Author name : Armagan Tekdoner

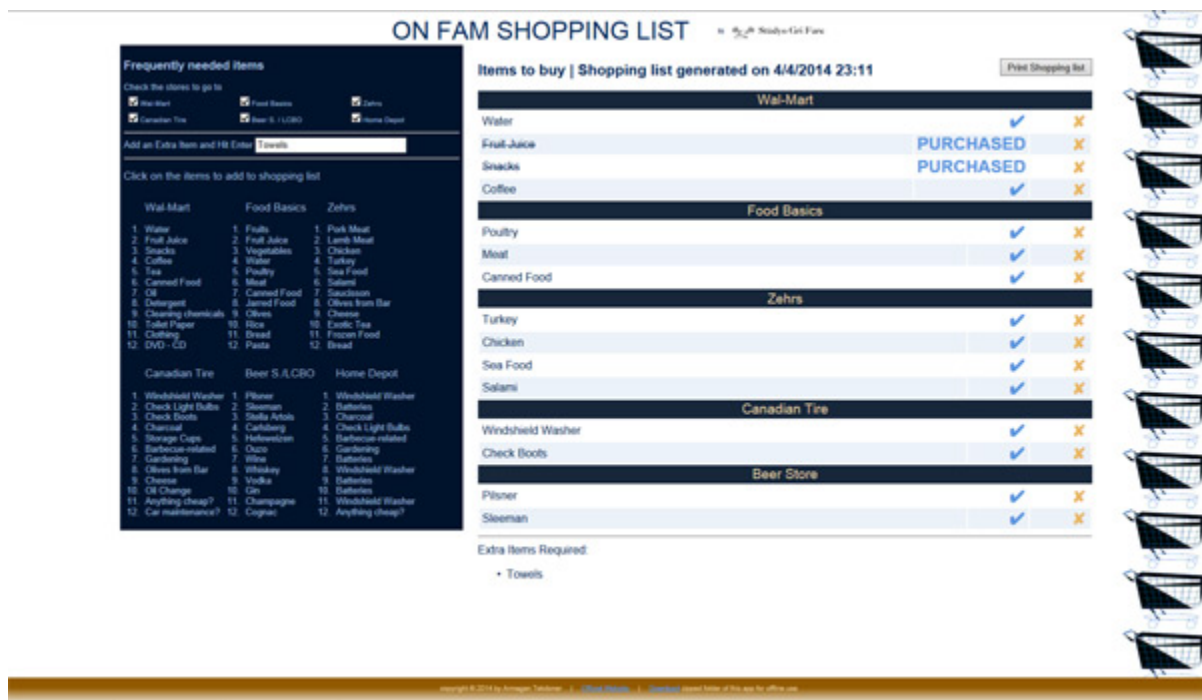
Date : April 2014

URL : <http://webdesign4.georgianc.on.ca/~200253439/comp1073/to-do/to-do.html>

Description

Design and build an interactive TODO list web application that will allow a user to maintain a list of tasks (or TODOs) to be completed for the current day. The user should have the following abilities:

- Add TODOs
- Remove TODOs
- Edit TODOs
- Mark TODOs as complete



Objectives

1. Design an interface for the application using HTML 5

The application uses HTML5 document type.

2. CSS should utilize DOM selectors as much as possible. Avoid using the id attribute unless absolutely necessary.

DOM selectors is the only selection method used in the CSS file. There is not one single ID attribute for any element, throughout the application. There is only one class attribute in total, dynamically assigned from within the js file.

3. Ensure that your design is standards compliant

Please check the validator. The document fully validates.

<http://validator.w3.org/check?uri=http%3A%2F%2Fwebdesign4.georgianc.on.ca%2F~200253439%2Fcomp1073%2Fto-do%2Fto-do.html>

4. The user interface should be intuitive, meaning that without explaining the functionality of the interface a user should be able to grasp the usage.

Even Fred Flintstone should be able to use this app.

5. The application should be designed to be self contained. This means that all interactions should happen within dialogs and not direct the user away from the application or cause a page reload.

Everything related to the application takes place inside one page. Page loads happen, if only links in the footer to additional resources are clicked.

6. Implement functionality to add TODOs

Adding takes place as follows:

From "Frequently needed items" list

User chooses a store name by clicking checkboxes on the left-hand side

Relevant items to choose are then automatically listed along with their shop's names underneath the checkboxes

User selects an item

That item is displayed on the right-hand side

By typing in the text field below "Add an Extra Item ":

User types anything up to 50 characters

That word is posted under the list on the right-hand side when enter key is hit

If user clicks in the field and hits the enter key, an alert box pops up and nothing is posted

Each and any of these items can be deleted by clicking them

7. Implement functionality to remove TODOs

Any item added can be removed by clicking a cross sign that has been created on the far right-hand side, along with that item's insertion.

8. Implement functionality to edit TODOs

Any item added, can be edited by clicking that item.

User clicks the item

A text field appears

The user types in text

The new text replaces the old item once the user leaves the text field

If the user clicks in the text field and leaves without typing, a confirm dialogue appears which either deletes the newly-inserted text or waits for user input

9. Implement functionality to mark TODOs as completed

Any item added can be marked as "PURCHASED" and be stricken-through, by clicking a check sign that has been created on the far right-hand side, along with that item's insertion.

10. Implement validation for user input with intuitive feedback (for example, do not let the user enter a blank TODO)

Blank entries trigger warnings and no empty field can be posted.

11. Provide a help system to guide a hapless end user through using your application. This can take the form of tooltips that appear when a user hovers over a button or option, or you could go as far as preparing how to video documentation.

The application is very easy to use and all functionalities are properly titled/labeled.

There is a help box written in plain language. Although the "how-to video" of this application is like a tutorial for calculating the area of a rectangle, even that has been provided for the sake of exercise.

12. Give your application a name (and a suitable logo) and prepare your application for the web. Your application should be prepared for external visitors and should look the part.

The application is named ON FAM SHOPPING LIST, standing for "Ontario Family Shopping List". It is in the header, followed by company logo. Although it can work offline, it is live online and a link in the footer enables downloading a zipped folder that includes the jQuery file for offline use. A favicon icon has been installed as well. The footer comprises documentation and links about the job ownership.

Extras:

- Responsive design: The application has separate styles for different screen sizes as listed below.
 - Greater than or equal to 1366px
 - Between 721px – 1365px
 - Between 481px – 720pxIf requested by devices with smaller than 481px screen, a reminder is issued telling that the device is not compatible
- Separate printer-friendly stylesheet:
 - Only the enlisted items are sent to printer and nothing else is printed. No backgrounds, fields, buttons, images or large fonts.
- A functionality under the title of "Extra Items Required" adds any text written to the list.
- Styling and documentation are intended to well exceed the expectations.
- Files are made available on GitHub.
- Extensive testing was done, the app works perfectly, with following notes:
 - Chrome (latest) ok
 - Firefox (latest) ok
 - Yandex (latest) ok
 - Internet Explorer (9+) ok (mp4 Help video is not played by the browser)
 - Safari ok (mp4 Help video is not played by the browser)
 - Opera mp4 Help video is not played by the browser
 - webkit-calc does not work, there is a rendering error in devices Between 721px – 1365px