

Overview

The requirement of this practical was to produce a single page web application with local data storage. The function of the web application was to provide a task manager, in which tasks could be added, removed, edited and marked as complete.

Design and Implementation

The project was developed using three modules following the Model-View-Controller pattern. The model contains the application state and logic, the view creates the visual appearance using HTML, and the controller manages events and makes method calls to both modules.

We were provided with a HTML page which ran scripts from each of the modules, provided a new task form, and an empty class for the task list. To display the tasks I appended HTML to the task list for each task, and then referred to it when necessary using calls to the DOM using JavaScript, and JQuery when possible.

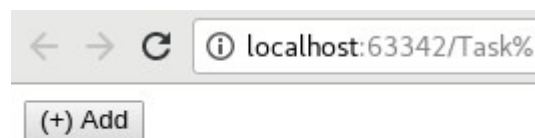
In the View module I used JQuery calls to CSS functions to make the background colours of the text box and check box variable depending on their value. I also included some CSS in the taskman.css file to style the form and task list. I also used the module to hide empty tasks before they were assigned values (otherwise they flashed up for a short time). I finally edited the provided new task form to allow it to be reused for editing of tasks.

In the Model module I wrote functions to take data from the new task form and create tasks, sort them by dates, and then write them to localStorage. To do this I created empty tasks and appended them to the taskList class in the DOM, then used the task data provided in the form into the inner HTML of the relevant elements. I used the Moment.js library to compare dates to correctly order the tasks, and also to compare dates with the current date to highlight overdue tasks. I also wrote a method to delete tasks from both the localStorage and the DOM, and a method to edit the task, by deleting the current task and adding the updated one after editing in the new task form.

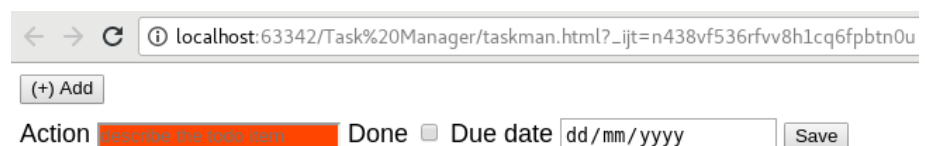
The Controller module was used to call methods at the correct time after the DOM tree had finished parsing, and in the correct order. It also contained JQuery functions called when elements on the page were clicked or changed, which called the appropriate methods in the other two classes.

Testing

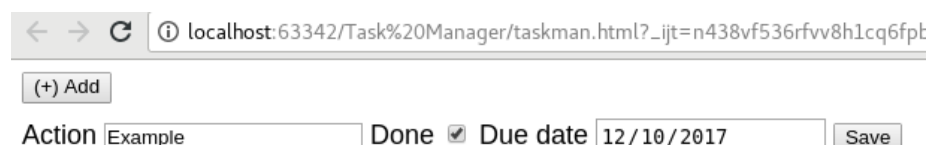
Initial view:



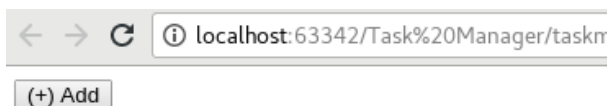
Initial task form:



Adding a task to the task manager:




The task is overdue but done, so is highlighted in red and green.



Example 2017-10-12 Edit Delete Done ☒

Sorting multiple tasks by date:

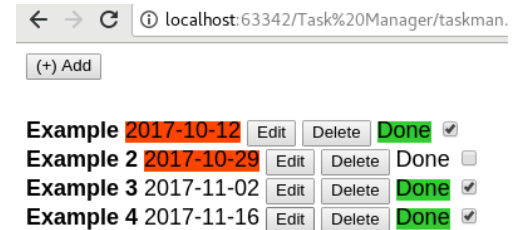
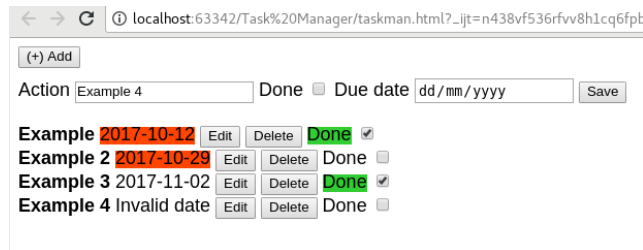
Example 4 was created with no date, so the date is considered invalid and put the bottom of the list.



Example 2017-10-12 Edit Delete Done ☒
Example 2 2017-10-29 Edit Delete Done ☐
Example 3 2017-11-02 Edit Delete Done ☒
Example 4 Invalid date Edit Delete Done ☐

Editing a task:

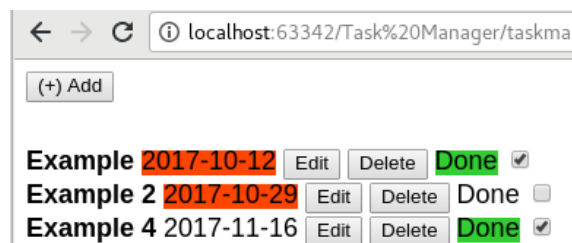
The edit form is pre-filled. Example 4 is given a date and marked done.



Example 2017-10-12 Edit Delete Done ☒
Example 2 2017-10-29 Edit Delete Done ☐
Example 3 2017-11-02 Edit Delete Done ☒
Example 4 2017-11-16 Edit Delete Done ☒

Deleting a task:

Example 3 is deleted, and removed from LocalStorage and the list of tasks.



Example 2017-10-12 Edit Delete Done ☒
Example 2 2017-10-29 Edit Delete Done ☐
Example 4 2017-11-16 Edit Delete Done ☒

Key	Value
0	["Example",true,"2017-10-12"]
1	["Example 2",false,"2017-10-29"]
3	["Example 4",true,"2017-11-16"]

Extensions

I added a green background to the tick box to make it clear when a task was completed. I also included the functionality to store tasks with undefined dates, or invalid dates such as 31/04/2017, displaying them at the bottom of the list of tasks.

Evaluation

Overall I felt that I understood how to meet the requirements of the practical without too much difficulty, and that the vast majority of the time spent was trying to work out how to achieve this in JavaScript. I had not coded a project in JavaScript before, and therefore didn't understand how to properly set up the modules required for the the Model-View-Controller development pattern. This wasn't made clear in lectures or the specification, which led to a significant amount of unnecessary work later on trying to rewrite code once I learnt how to do this.

Throughout the practical I did extend my knowledge of JQuery, JavaScript manipulation of the DOM tree, and LocalStorage. If I was to attempt the practical again, I would better plan out the HTML structure of the tasks to make some of the methods which called to task elements easier and more efficient.

Conclusion

This practical took much more time than any other so far this year, but I did learn a lot about using JavaScript, libraries, LocalStorage and the DOM tree when creating a web application. The finished task manager is fully functional, meeting all of the basic requirements and implementing some extensions.