Luigi De Russis, Teodoro Montanaro

LAB 7 – JQUERY, AJAX, REST: A FULL EXERCISE

EXERCISE 1 – IMPLEMENT A COMPLETE TODO LIST MANAGER: INSERT

Extend the ToDo List REST client implemented in classroom (you can clone the branch "ajax_client" of the following repository: https://github.com/AmI-2017/python-lab6[1]), to provide basic support for **inserting** a task by JavaScript.

Specifically, by using the jQuery library, you should define what will happen when the user will click on the "**Enter**" button present at the bottom of the page. To do this, you should specify which function will be called when the "*onsubmit*" event associated to the "add-task" form will be handled:

1. Define a function (ex. "*insertNewTask()*") inside your script ("*tasks.js*")
2. Assign the defined function ("*insertNewTask()*") to the handler of the "*onsubmit*" event associated to the button (ex. *<form id="add-task" … onsubmit="insertNewTask(event);">*)

   WARNING: to avoid form submission (the default action of the event) you should
   a) Pass "event" as argument of the function (*onsubmit="insertNewTask(event);"*)
   b) Accept the event as function argument (*function insertNewTask(event)*)
   c) Add this line of code at the end of the *insertNewTask* function:
   event.preventDefault();

3. Then modify the "*insertNewTask()*" function to perform the following actions:
   a) Get the string contained in the text box (which id is "*description*")
   b) Get the value of the *Urgent* checkbox (after assigning an id to it)
   c) Perform an asynchronous HTTP POST request to send the acquired data to the server.

   ```
   var json = {description: text, urgent: urgent};
   $.ajax("/api/v1.0/tasks", {
     method: 'POST',
     contentType: 'application/json',
     data: JSON.stringify(json),
     success: function (data, status) {
       // update the list of printed tasks: called when the INSERT is complete
     }
   }
   ```

   d) As soon as the server responds with a success response, update the list of printed tasks.

Suggestion: if, when you run your server, it cannot connect to the database, do not forget to set the *working directory* in the Configuration settings (Run -> Edit configurations).

[1] To clone a branch of a repository you can use the following command:

```
git clone <repository-url> -b <branch> --single-branch
```

EXERCISE 2 – IMPLEMENT A COMPLETE TODO LIST MANAGER: DELETE

Extend the application developed in the previous exercise to provide basic support for **deleting** a task.

Consequently, you should

1. extend the front-end jQuery code to add a “Delete” button near each task (in the Action column).
2. store the ID of each task in each “Delete” button
3. implement a “deleteTask” function that actually performs the deletion and, then, updates the list of printed tasks.

EXERCISE 3 – IMPLEMENT A COMPLETE TODO LIST MANAGER: UPDATE

Extend the application developed in the previous exercise to provide basic support for **updating** a task.

Consequently, you should

1. add an “Update” button near the “Delete” one for each printed task
2. implement a “setTextboxForUpdate” function that performs the following operations whenever the “Update” button is pressed:
   a. Modify the label “Insert a new task” into “Update the task”
   b. Load the information related to the selected task in the “description” textbox and in the “urgent” checkbox
   c. Change the action and method of the “add-task” form in the ones needed to perform the update
3. implement an “updateTask” function that performs the following operations:
   a. Send the data to the server
   b. Update the shown list of tasks
   c. Modify the label “Update the task” into “Insert a new task”
   d. Empty the “description” textbox and the “urgent” checkbox
   e. Reset the action and method of the “add-task” form in the ones needed to perform the insert