

# Les corrections apportées à l'application TODOS

todos

*What needs to be done?*

Double-click to edit a todo

Created by Oscar Godson

Refactored by Christoph Burgmer

# Sommaire

1. Correction du bug " faute de frappe"
2. Correction liée au conflits avec les ID
3. Amélioration du code
4. Tests

## 1. Correction du bug "faut de frappe"

La faute de frappe est dans le fichier **controller.js**, il y avait un "d" en trop.  
Controller.prototype.addlItem et non Controller.prototype.addItem

```
95      //fix addItem (wrong syntax)
96
97  [-] Controller.prototype.addItem = function (title) {
98      let self = this;
99
100  [-]  if (title.trim() === '') {
101      |    return;
102      |  }
103
104  [-]  self.model.create(title, function () {
105      |    self.view.render('clearNewTodo');
106      |    self._filter(true);
107      |  });
108  };|
```

## 2. Correction liée au conflits avec les ID

La faute se situe dans le fichier **store.js** avec un souci d'identifiant unique pour chaque nouvel ID.

Le conflit d'intérêt lié aux IDs vient avec la création d'ID.

Pour fixer ce problème, la meilleure solution consiste à utiliser la méthode `Date.now()`. Ainsi la fonction va retourner un nombre unique et donner un nombre de millisecondes écoulées depuis le 1er Janvier 1970 00:00:00 UTC.

```
77 Store.prototype.save = function (updateData, callback, id) {
78   let data = JSON.parse(localStorage[this._dbName]);
79   let todos = data.todos;
80
81   callback = callback || function () {};
82
83   // // Generate an ID
84   // let newId = "";
85   // let charset = "0123456789";
86   // If an ID was actually given, find the item and update each property
87   if (id) {
88     for (let i = 0; i < todos.length; i++) {
89       if (todos[i].id === id) {
90         for (let key in updateData) {
91           todos[i][key] = updateData[key];
92         }
93         break;
94       }
95     }
96
97     localStorage[this._dbName] = JSON.stringify(data);
98     callback.call(this, todos);
99   } else {
100
101     // Generate an unique ID
102     // Use the Method Date.now() which looking for a number returned since 1970/01/01
103     // Assign an ID
104     updateData.id = Date.now();
105     //parseInt(newId);
106     todos.push(updateData);
107     localStorage[this._dbName] = JSON.stringify(data);
108     callback.call(this, [updateData]);
109   }
110 };
```

### 3. Amélioration du code

L'amélioration apportées est dans le fichier **controller.js**

Elle concerne une boucle forEach et un console.log

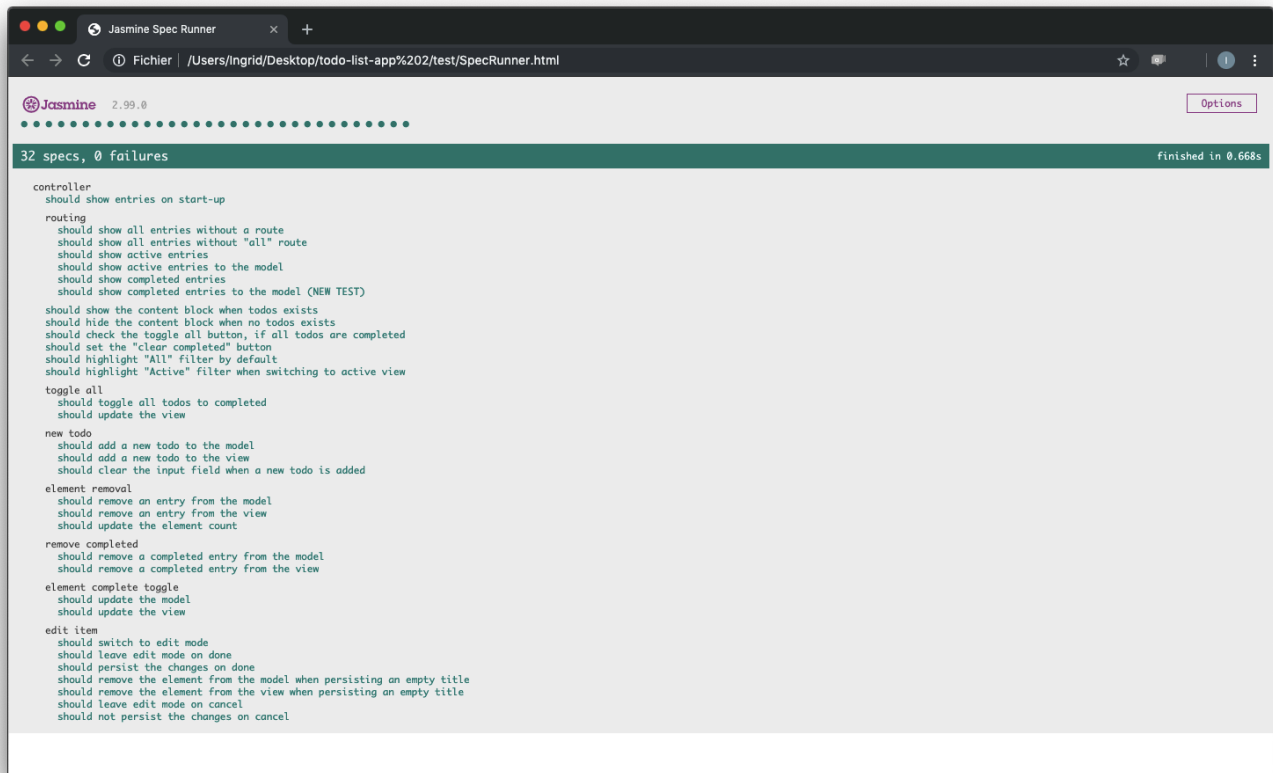
La boucle forEach est pour ce cas inutile et le console.log apporte une mauvaise information, il est mieux de le placer après le render.

```
160 Controller.prototype.removeItem = function (id) {
161     let self = this;
162     let items;
163     self.model.read(function(data) {
164         items = data;
165     });
166
167     // Console.log don't give the good information Put it after the render
168     // forEach loop is useless
169
170     self.model.remove(id, function () {
171         self.view.render('removeItem', id);
172         console.log("Element with ID: " + id + " has been removed.");
173     });
174     self._filter();
175 }
```

## 4. Tests

Les tests ont été faits dans le fichier **ControllerSpec.js**.

J'ai ajouté 2 tests car tous les tests testent le Model et la View et il manquait 2 tests sur le Model.



The screenshot shows the Jasmine Spec Runner interface in a web browser. The browser's address bar displays the file path: `/Users/Ingrid/Desktop/todo-list-app%202/test/SpecRunner.html`. The Jasmine logo and version `2.99.0` are visible in the top left, and an `Options` button is in the top right. A green progress bar indicates `32 specs, 0 failures` and `finished in 0.668s`. The main area lists the following test specifications:

```
controller
  should show entries on start-up
routing
  should show all entries without a route
  should show all entries without "all" route
  should show active entries
  should show active entries to the model
  should show completed entries
  should show completed entries to the model (NEW TEST)
  should show the content block when todos exists
  should hide the content block when no todos exists
  should check the toggle all button, if all todos are completed
  should set the "clear completed" button
  should highlight "All" filter by default
  should highlight "Active" filter when switching to active view
toggle all
  should toggle all todos to completed
  should update the view
new todo
  should add a new todo to the model
  should add a new todo to the view
  should clear the input field when a new todo is added
element removal
  should remove an entry from the model
  should remove an entry from the view
  should update the element count
remove completed
  should remove a completed entry from the model
  should remove a completed entry from the view
element complete toggle
  should update the model
  should update the view
edit item
  should switch to edit mode
  should leave edit mode on done
  should persist the changes on done
  should remove the element from the model when persisting an empty title
  should remove the element from the view when persisting an empty title
  should leave edit mode on cancel
  should not persist the changes on cancel
```