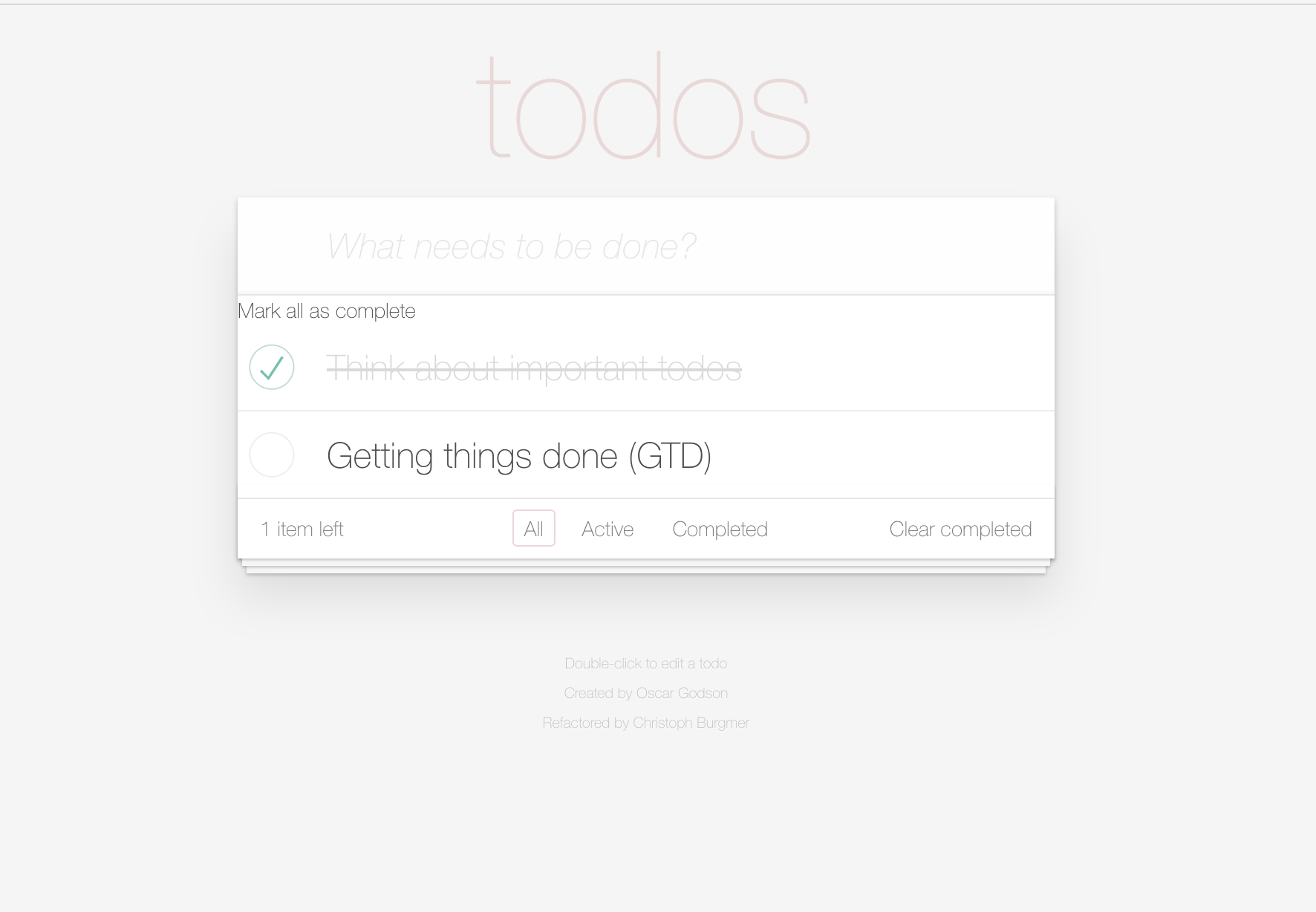
The to-do-list-app



Probably the world’s best online to-do-app.

Table of Contents

[Introduction 2](#__RefHeading___Toc471_2771898783)

[Detailed View of the controller 3](#__RefHeading___Toc473_2771898783)

[Storage 5](#__RefHeading___Toc475_2771898783)

[Future concepts and comparison with the comparison site 6](#__RefHeading___Toc477_2771898783)

[Folder Concept 6](#__RefHeading___Toc479_2771898783)

[Information about the items 6](#__RefHeading___Toc481_2771898783)

[Sorting 7](#__RefHeading___Toc483_2771898783)

## Introduction

The to-do-list app is designed using the Model-View-Controller(MVC) design pattern. The only layer of this multitier architecture, that is called from the presentation layer (aka the HTML or Web-UI), is the controller. Every other manipulation that is done for example with the model, is triggered by the controller.

The controller is also the entry point for an automated test using the Jasmine Framework.

For more information on design pattern itself, see Gang of Four(GoF) and Design Pattern, please.

Overview of the layers:

- Controller: All the available actions are triggered from here:

* + newTodo(title)
  + itemEdit(item)
  + itemEditDone(item)
  + itemEditCancel(item)
  + itemRemove(item)
  + itemToggle(item)
  + removeCompleted()
  + toggleAll(status)
  + showAll
  + showActive
  + showCompleted
  + updateCount
  + filter
  + updateFilterState
* Views: Renders and filters the different views, like all, 'active', 'completed’.
* Model: Manages, i. e. create, read, update, remove, removeAll and counts, the data in the   
   storage.
* Storage: In this layer, the data is stored, using the session storage.
* Helpers: A bunch of helper methods for querying the selector(s) and encapsulating the   
   DOM.

## Detailed View of the controller

Robert C. Martin (a.k.a 'Uncle Bob') wrote a book about clean code. My understanding of his theory is, that the when you use clear and self-explaining function and method names, there is no need for an extra documentation.

In this section, all the methods of the controller are listed with the call hierarchy.

* + newTodo(title) => addItem(title)
    - trim
    - model.create
    - view.render
  + itemEdit(item) => editItem(item.id)
  + itemEditDone(item) => editItemSave(item.id, item.title)
    - trim
    - model.update
    - view.render
    - if item == null remove
  + itemEditCancel(item) => editItemCancel(item.id)
  + itemRemove(item) => removeItem(item.id)
    - model.read
    - iterate
      * Show remove message
    - model.remove
    - view.render
    - filter
  + itemToggle(item) => toggleComplete(item.id, item.completed)
    - model.update
    - view.render
    - filter
  + removeCompleted() => removeCompletedItems()
    - model.read
    - iterate
      * removeItem
  + toggleAll(status) => toggleAll(status.completed)
    - model.read
    - iterate
      * toggleComplete
    - filter
  + showAll
    - read
    - render('showEntries', data)
  + showActive
    - read
    - render('showEntries', data)
  + showCompleted
    - read
    - render('showEntries', data)
  + updateCount
    - model.getCount
    - view.render(updateElementCount)
    - view.render(clearCompleteButton)
    - view.render(toggleAll)
    - view.render('contentBlockVisibility’)
  + filter
  + updateFilterState

## Storage

There are three different types of client storage technologies. Local storage, session storage and cookies. Local Storage and session storage provide 5 MB of local memory on the hard disk, i.e. per domain 5 MB are reserved. Depending on the used browser, it might be more than 5 MB.

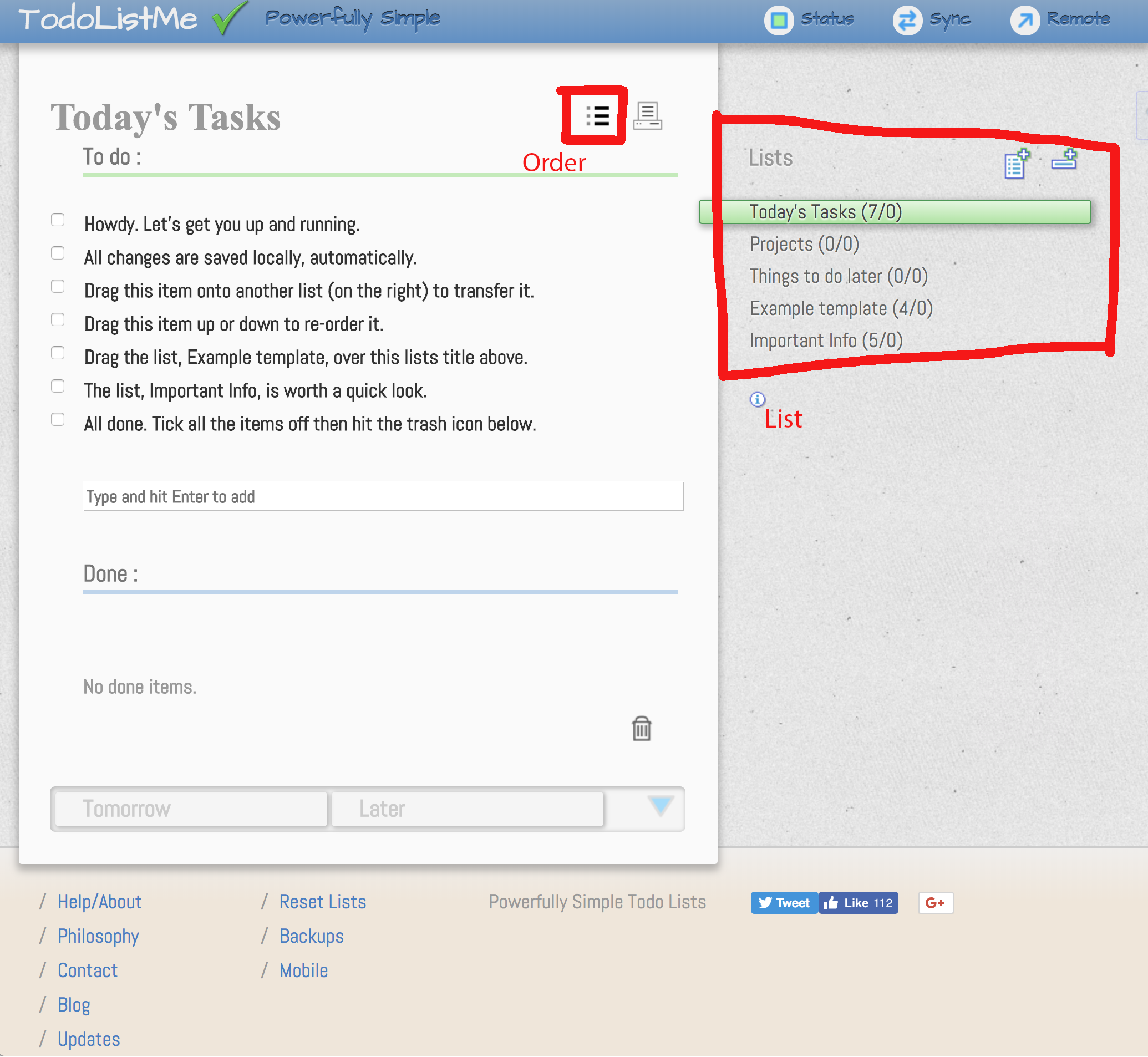
5 MB are equal to 5 \* 1024 \* 1024 = 5,242,880 characters, enough to store a book with over 300 pages. Even though not every character can be used to save the todos. Some bytes are lost to the JSON format.

Cookies, on the other hand, allow only the usage of 180 KB, that is equal to 180 \* 1024 = 184,320 characters. Complexe to-do lists might exceed this length.

We value the privacy of our customers. Therefore no information is transferred over the internet to our servers. All the data is stored on the client device in the session storage. After quitting the app, all the information is not availible any more.

## Future concepts and comparison with the competitor site

### Folder Concept



The competitor app offers storing the items in different lists, to keep them better organized. The technical base in our application is already prepared to do so, only a dialog with a selection box should allow the customer to create, change or delete lists. Lists in our case labels do not change the behaviour of the application or the performance.

### Information about the items

There is no information provided by our app, the competitor offers the information a tooltip when the item was inserted, changed or done.

### Sorting

As of today, it is not possible to sort the to-dos either by task, date, etc. A great, in both tools not yet present feature, would be to have and sort by priority.