Technical Documentation

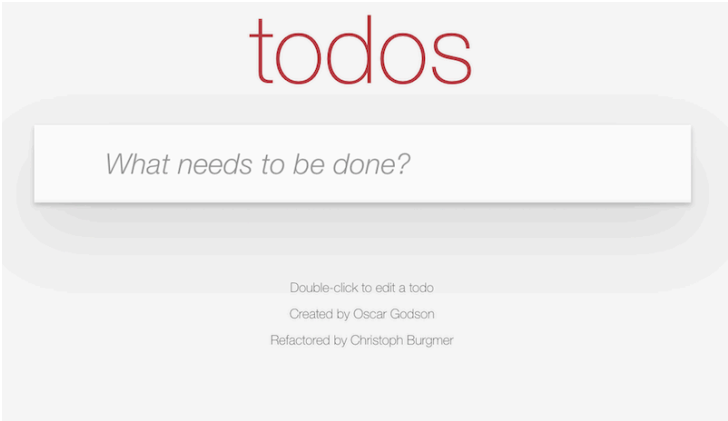
Prepared for: OpenClassrooms, Front-End Project 8
Prepared by: Thomas Di Giovine, Front-End Developer
6 January 2020

About the app

The “**to-do app**" lets a user create, read, update and delete to-do tasks.
A task has a title and can be completed or active. The user can add as many task as needed.
Tasks Can be mark as completed all at once. Completed tasks can be deleted all at once.

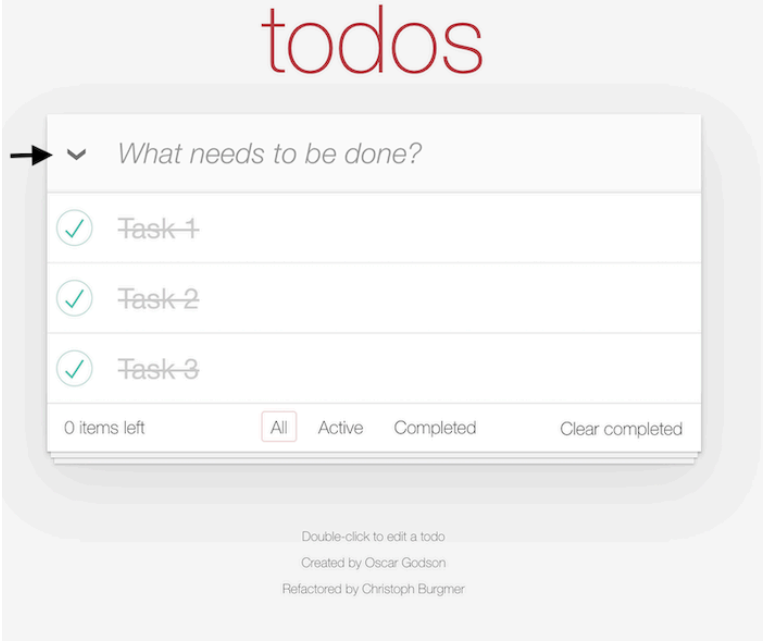
How it works

1. Screen has a toolbar where is possible add new to-do tasks with a title/description.



2. Once one or more tasks are added, it will appear a filtering menu to allow to switch between “All”, "Active" and “Completed” tasks. Tasks can be marked completed/ active.
   On hover, listed task will shows an “X” to delete the task.
   Once a task is marked as completed will appear a “Clear completed” option on the bottom right corner.
   Completed task can be deleted all at once by pressing the “Clear completed”.

3. Pressing the arrow on the top left corner of the app, all task will be marked as completed.



4. The user can edit an existing task by double-clicking on it.

Audit Performance Document

Audit Performance Document from the competitor site: “http://todolistme.net/"

1. Website Performance:

Website performance works well with an overall score of 92%.
First Contentful Paint (time at which the first text or image is painted), is fast and appear in 0.9s.
While the Time to Interactive (the amount of time it takes for the page to become fully interactive), is within the average and is complete in 4.3s.

- **Performance improvement:**

I. Eliminate render-blocking resources:

This are external links included in the html document.

- Move critical code from the render-blocking URL to an inline script or style block tag in the HTML page to speed up page loading.
- If there’s not-critical code in a render-blocking URL, keep it in the URL, and mark the URL with async or defer attributes.
- Code that isn't being used at all should be removed.

II. Ensure text remains visible during webfont load.

Some browsers hide text until the font loads, causing a flash of invisible text (FOIT).

- Include font-display: swap in @font-face style, can avoid FOIT.
- Add the &display=swap parameter to the end of your Google Fonts URL.

III. Serve static assets with an efficient cache policy

HTTP caching can speed up your page load time on repeat visits.

- When possible, cache immutable static assets for a long time, such as a year or longer.
- Use no-cache if the resource changes and freshness matters, but you still want to get some of the speed benefits of caching.

IV. Minimize main-thread work

The main-thread of the renderer process handles most code, including parsing and executing html, css, and JavaScript. It also processes user events.

- Only send code that your users need by implementing code splitting.
- Minify and compress code.
- Remove unused code.

- **Positive performance audits:**

- Properly size image.
- Serve image in next-gen formats.
- Efficiently encode images.
- Minify CSS and JavaScript.
- Removed unused CSS.
- Enable text compression.
- Avoid multiple page redirects.

2. Website Accessibility

Website Accessibility has a low overall score of 38%.

Accessibility improvement:

- Choose background and foreground colors with stronger contrast ratio: low-contrast text is difficult or impossible for many users to read.
- All [id] attributes should be unique.
- <frame> or <iframe> elements should have a title.
- Image elements should have [alt] attributes.
- Form elements should have associated labels.
- <html> element should have a [lang] attribute.

3. Website Best Practices

Website Best Practice has an average overall score of 64%.

Best practices improvement:

- Add HTTPS protocol. All sites should be protected with HTTPS.
- Do not use `document.write()`. For users on slow connections, external scripts dynamically injected via `document.write()` can delay page load by tens of seconds.
- Replace third-party scripts that may contain security vulnerabilities easily identified and exploited by attackers, with more secure ones.
- Resolve browser console errors. They can come from network request failures and other browser concerns.

4. Website SEO

Website SEO has an average overall score of 78%.

SEO improvement:

- Add a `<meta name="viewport">` tag to optimize your app for mobile screens.
- Add alt attribute to all image elements.

For the full audit, please refer to the document “todolistme.net-performance.html" located in the same folder as this pdf file.