Designing for Mindful Human-Computer Interaction

Prototyping for the Digital Wellbeing

Alberto Monge Roffarello

Politecnico di Torino

Academic Year 2024/2025

e-Lite

Prototypes

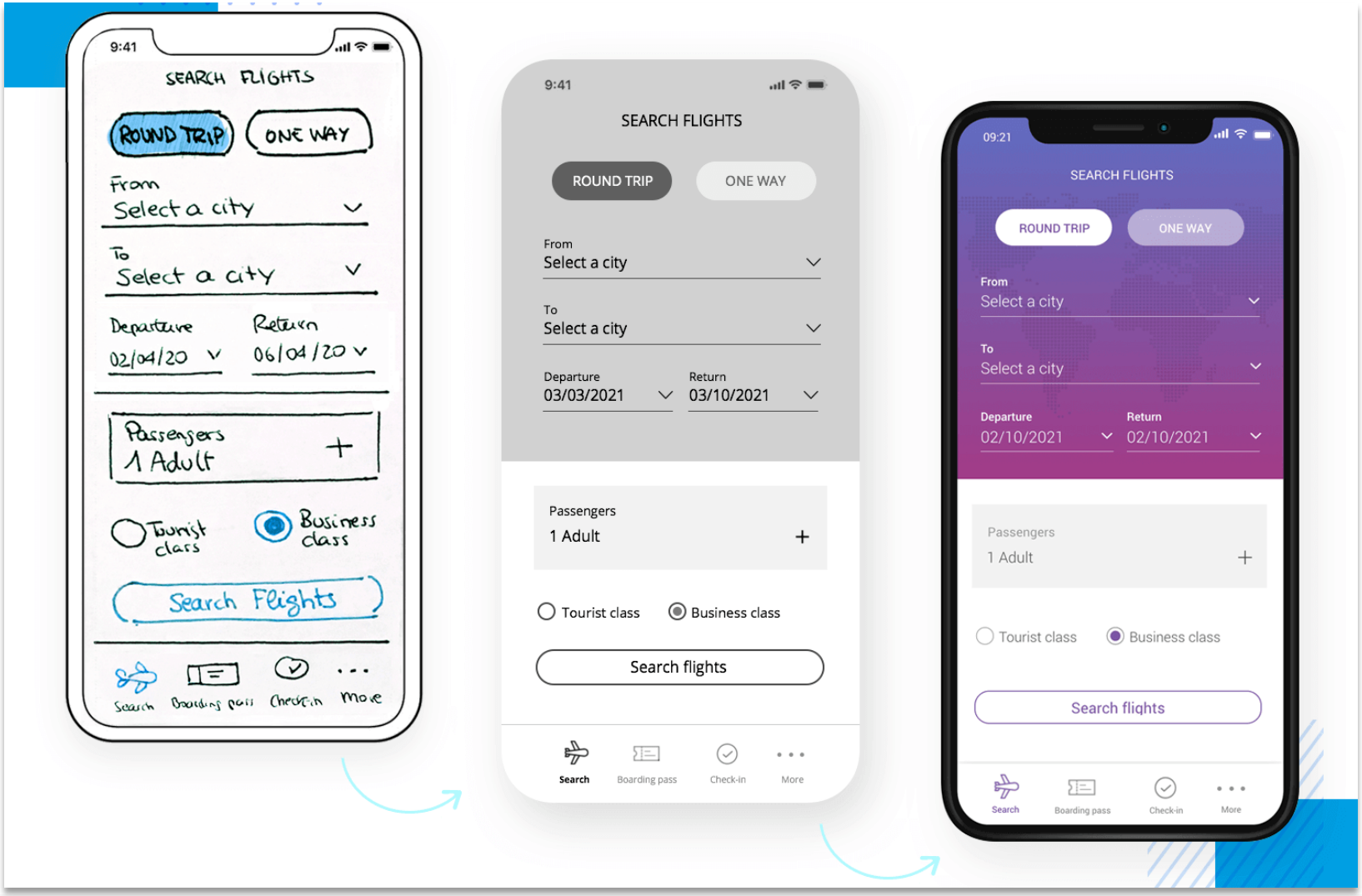
- «A prototype is a concrete but partial representation or implementation of a system design»
- «An easily modified and extensible model (representation, simulation or demonstration) of a planned software system, likely including its interface and input/output functionality»
- One of the most powerful tools for design exploration, visualization, and testing
- They let us ‘see’ and ‘feel’ interactivity (simulated or real)

Prototypes

- **Envisionment**: making ideas visible
  - Generating new ideas
  - Evaluating new ideas (within the design group)
  - Testing new ideas (with users)
- Different tools and techniques, according to
  - The stage of design (early, …, advanced, final)
  - The intended audience (designers, test users, clients, management, …)

Take from https://www.justinmind.com/prototyping/low-fidelity-vs-high-fidelity-prototypes

Fidelity: Different Information Is Conveyed



Figma

A Tool for Rapid Prototyping

Overview

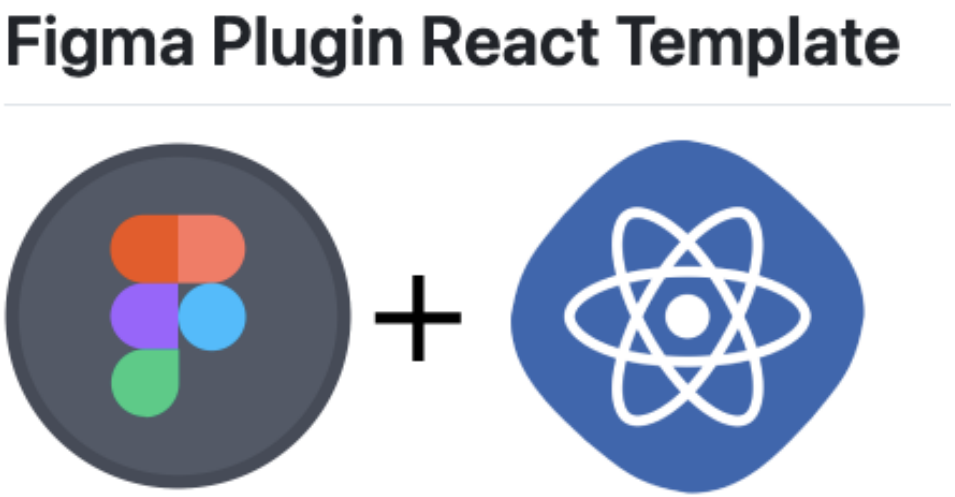
- Figma is a collaborative interface design tool: https://www.figma.com/
  - entirely browser-based;
  - free for students;
  - allows real-time collaboration on the same file.
- Some starting points:
  - https://www.figma.com/community/file/917793002372330875
  - https://youtu.be/S0J9C70PGsg?t=782 (from the HCI course)

Figma Extensions

Integrating LLMs to support designers

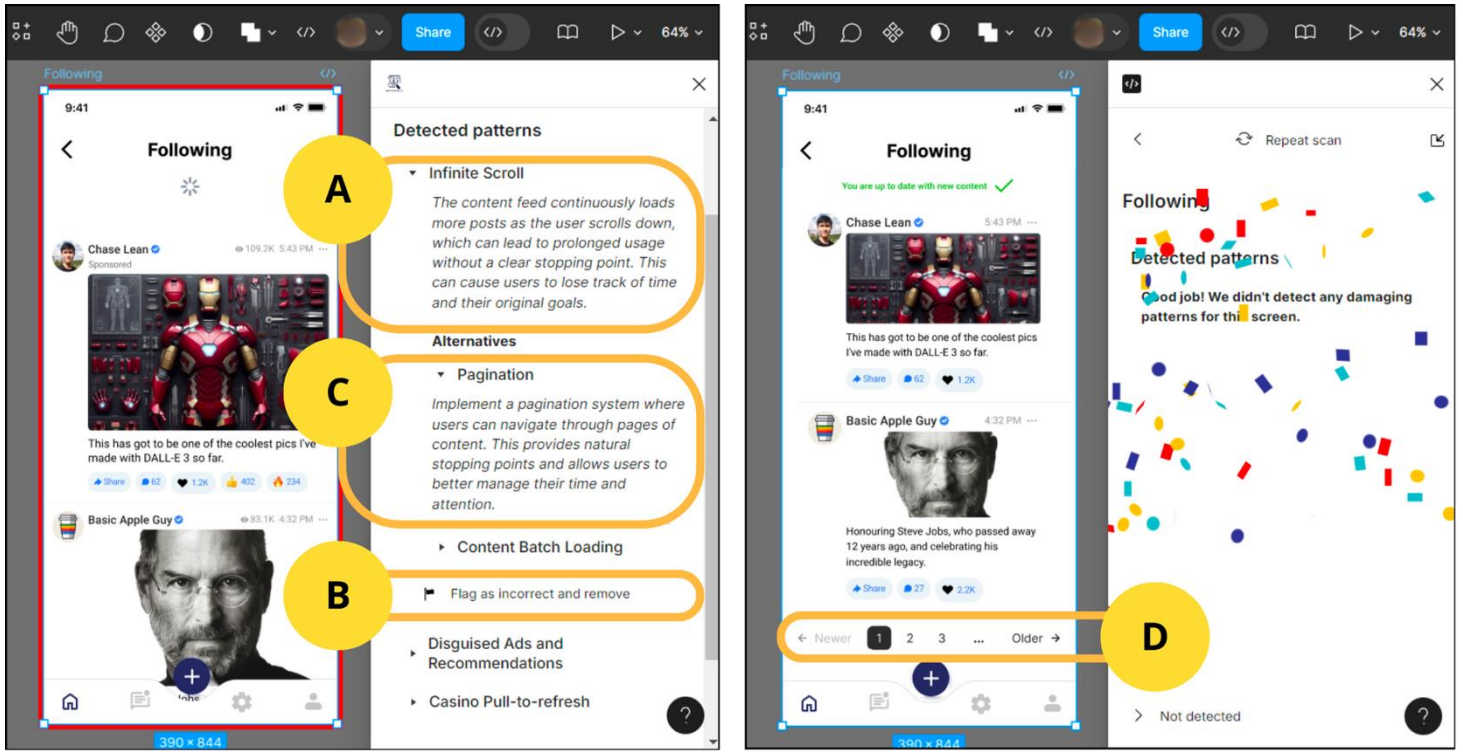
References to create and style the plugin

- https://github.com/nirsky/figma-plugin-react-template
- https://github.com/figma/plugin-samples



Source code of Digital Wellbeing Lens (with LLM integration)

- https://git.elite.polito.it/public-projects/digital-wellbeing-lens



Chrome Extensions

Prototyping DSCTs for the Web

Overview

- Extensions are written with the same web technologies used to create web applications:
  - **HTML** is used as a content markup language;
  - **CSS** is used for styling;
  - **JavaScript** is used for scripting and logic.
- A Chrome extension can access two different set of APIs:
  - "traditional" JavaScript APIs (https://developer.mozilla.org/en-US/docs/Web/API);
  - Chrome APIs (https://developer.chrome.com/docs/extensions/reference/).

Ingredients

- The **Manifest**:
  - required file;
  - must have a specific file name: manifest.json;
  - it must be located in the extension's root directory;
  - it records important metadata, defines resources, declares permissions, and identifies which files to run in the background and on the page.
- The **Service Worker:**
  - handles and listens for browser events (e.g., navigating to a new page, closing a tab, …);
  - it can use the Chrome APIs, but it cannot directly interact with the content of web pages.

Ingredients

- **Content Scripts**:
  - execute Javascript in the context of a web
  - read and modify the DOM of the pages they're injected into.
  - can only use a subset of the Chrome APIs:
    - may interact with a service worker to access additional APIs
- The **Popup** and **other pages:**
  - an extension can include various HTML files, such as a popup, an options page, and other HTML pages;
  - all these pages have access to Chrome APIs.

Hello, World!

1. Create a directory for your extension
2. Create a file named *manifest.json* and add the following code:

```
{
      "manifest_version": 3,
      "name": "Hello Extensions",
      "description": "Base Level Extension",
      "version": "1.0",
      "action": {
      "default_popup": "hello.html",
      "default_icon": "hello_extensions.png” }
}
```

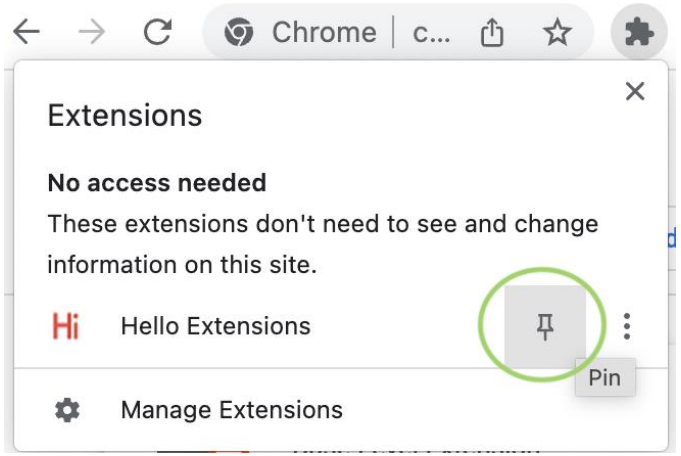
Hello, World!

3. Download the icon for the extension and save it in the directory with the name "*hello-extension.png*"
4. Create a file named *hello.html* and add the following code:

```
<html>
    <body>
        <h1>Hello, World!</h1>
    </body>
</html>
```

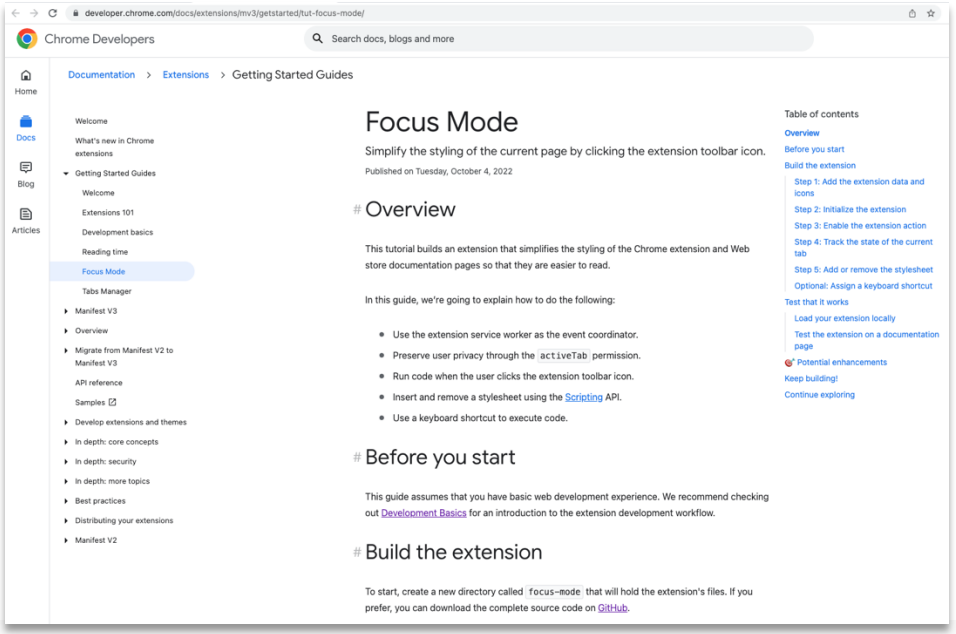
Hello, World!

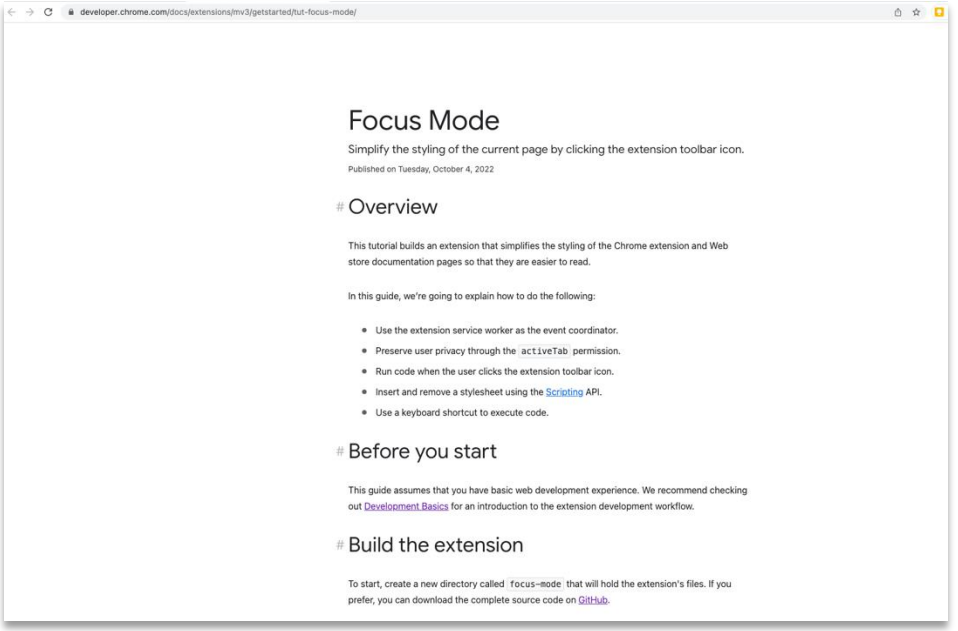
5. Load the extension as an an unpacked extension:
   - go to *chrome://extensions* in a new tab;
   - enable Developer Mode;
   - click the *Load unpacked* button and select the extension directory.
6. Pin your extension to the toolbar to quickly access your extension during development.



EXAMPLE: Focus Mode

- Simplifying the styling of a web page by removing menus, search bars, …
  - SOURCE: https://developer.chrome.com/docs/extensions/mv3/getstarted/tut-focus-mode/





License

- These slides are distributed under a Creative Commons license "**Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0)**"

- **You are free to:**
  - **Share** — copy and redistribute the material in any medium or format
  - **Adapt** — remix, transform, and build upon the material
  - The licensor cannot revoke these freedoms as long as you follow the license terms.

- **Under the following terms:**
  - **Attribution** — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - **NonCommercial** — You may not use the material for commercial purposes.
  - **ShareAlike** — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - **No additional restrictions** — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

- https://creativecommons.org/licenses/by-nc-sa/4.0/