Canada

Design System Project Team

Sprint 5 Summary

December 13 - January 14, 2022

Sprint goals

1. Establish workflows:
    - a11y testing workflows (automatic and manual)
    - Figma integration workflow
2. Break down a component:
    - Button component:
        i. Stencil – button variations
        ii. How we're classifying things
        iii. Text/labels
        iv. States – hover, active, focus

Button requirements

Applying best practices and design logic to improve the overall button experience and usability

Documented:

- **Style properties** - configurable or not, and why
- **States and state behaviour** - hover, focus, disabled, active
- **Proposals** - what to keep, change, remove, add

Data model: categories

Components have 3 categories of properties:

- **universal properties**: properties that all components have - these follow a consistent naming convention
- **component-specific properties**: properties that are only applicable to the component's unique model
- **custom overrides**: properties that override default settings with custom values

Universal properties

- **type**: the name of the functional behaviour that distinguishes a component from other components
- **task**: the main styles that facilitate a component's different tasks
- **variant**: the versions of a component that look different from the standard task styles or consume different data for a specific purpose
- **state**: options that dictate the component's default state
- **lang**: the language a component needs to load

Draft documentation model

- Definition
- How a button works
- Use buttons for
- Don’t use buttons for
- Styles
  - solid
  - outline
  - text only

- States
  - hover
  - active
  - focus
  - disabled
- Make buttons usable
  - labels
- Make buttons accessible
  - auditory
  - cognitive
  - motor
  - visual

Button page content

Buttons

A button is an interactive object that highlights an important or common action for the user.

How a button works

A short label or icon tells the user what the button does. The user communicates a button's command by launching a click action, like a mouse key, keydown action, a tap, or clicking a mouse.

By activating a button, the user initiates a response or a change of location or state.

Use buttons for [icon]

Use buttons to make common or important actions more obvious, such as:

- saving, deleting, downloading.
- clearing entered data.
- submitting a form or request.
- giving consent or agreement.
- making a binary choice.
- entering or exiting a web product, like a secure area login, a form, or tutorial.

In applications, buttons are used in place of text links for things like:

- controlling movement, like going from one screen to another in a flow.
- external links.

Don't use buttons for [icon]

Consider links instead when:

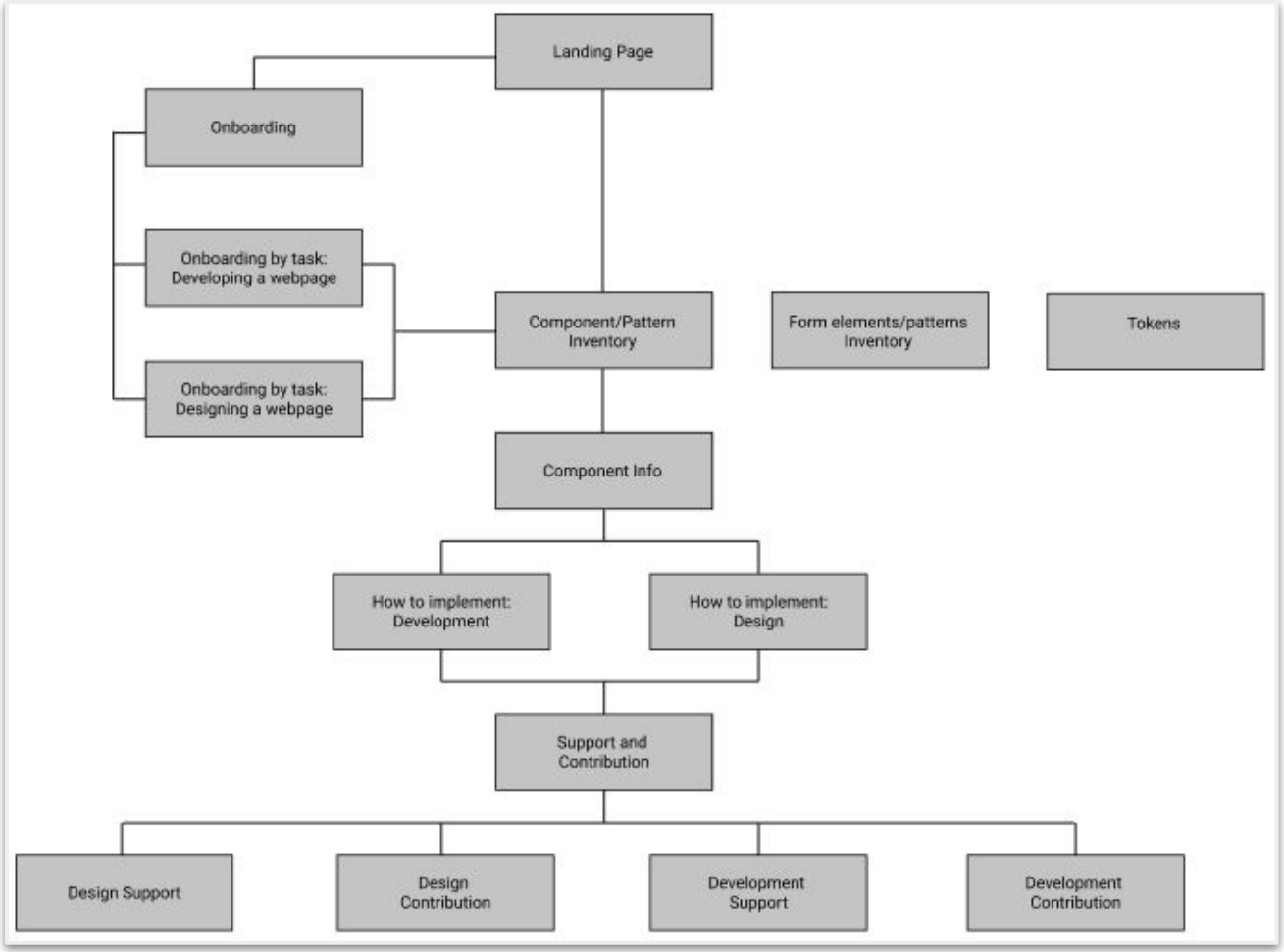
- taking a user off a web page.
- an action is less important.

Consider menus or selections, like dropdowns or radio buttons, instead when:

- the user must choose between more than 2 options.

Site map

Building out conceptually how sections inter-relate



Accessibility testing

Research questions

- Does having interactive elements, like a button, in the shadow DOM create an accessibility issue?
- If so, how can these be addressed?

Testing

- Ran test with two screen readers (VoiceOver and NVDA)
- Used an HTML button, the poc button (gc-ds-button), and another web component button (ionic framework)

Each button was recognized as a button to the screen reader

Accessibility tools

CDS tools

- A11y-tracker app - GitHub
- A11y-checker GitHub action - GitHub
- A11y-multiple-page-checker GitHub action - GitHub

Manual testing

Tools a developer can run in their browser:

- Accessibilityinsights.io
- Axe
- WAVE Toolbar
- WebAIM: Contrast Checker
- Google Chrome - Lighthouse

Blog post: Guide to accessible UI components

Automated testing - Web components/Stencil

For a11y testing:

- Stencil generates unit and E2E(end-to-end) tests for each component
- Use E2E tests for interactions
- Run tests on components repo when a pull request is opened/modified

CSS stage 0 polyfill

Successful experiment in WET-BOEW

- Leverages pure CSS instead of SASS/Less
- Allows authors to write pure CSS at stage 0+ and test it live in a browser

https://wet-boew.github.io/wet-boew-experimental/src/sites/wb.dev-css/example.html

GC Design System Product Team

GC Design Slack • dto.btn@tbs-sct.gc.ca