CDS SNC Canadian Digital Service
Service numérique canadien

Canada

Accessibility Matters: Effective testing gives you back a lot of time later

Intro

Hey there! We are **Julianna & Bethany**. We work on helping the Canadian Digital Service create accessible and inclusive services for Canadians.

julianna.rowsell@tbc-sct.gc.ca

https://github.com/jrowsell

@juliannarowsell

bethany.dunfield@tbs-sct.gc.ca

https://github.com/brdunfield

@BethanyDunfield

Accessibility Matters

Inaccessible service design creates barriers for **people**.

Why designing for accessibility benefits everyone

Steps to start making products and services accessible

- Start with research
- Design with people with disabilities (co-create)
- Test with people with diverse needs
- Try automated testing tools
- Test your product or service manually for accessibility
- Conduct an accessibility audit
- Hire an accessibility subject matter expert to give you and your team advice
- Raise awareness in your organization about accessibility and inclusion

No two people with the same disability are alike

- Blind since birth
- Blind later in life
- Vision loss
- Eye fatigue from over stimulation

- Deafened since birth
- Partial hearing loss
- Hearingloss later in life
- Listening in a crowded room

Building inclusive services is not about perfection

The key is **people** need to know what to consider, be aware of and what to flag.

- Build accessibility into the beginning, it will save time and resources.
- Shorten update/release processes by clearing hurdles and lengthy approvals.
- Include persons with disabilities throughout discovery, design, development, and delivery phases.
- Build awareness on product teams in order to improve the service.
- Encourage delivery team members to engage within the community.
- Embed accessibility champions on product teams.

Why Perform Automated Testing?

- Easy to get started
- Provides excellent value for corresponding effort
- Scales well
- No excuse not to

Test your code regularly

How is Automated Testing Possible?

- Value system is well-defined (WCAG & Section 508) “Links must have features x and y”

- Properties of objects are known (DOM & accessibility tree) “My link has feature X, but not feature Y”

- Judgements can be made systematically “Feature Y is missing, therefore the link is bad”

Why Not Automate Everything?

Automated testing alone is insufficient

- Only finds about 40-50% of accessibility issues

- Risk of false-positives greater certainty of
success requires Manual Audits

Drawbacks of Manual Audits

- Require deep knowledge of accessibility
- Time-consuming
- Results are invalid by the time they can be processed
- Do not scale well
- Effort to value ratio = 1:1

Why Perform Manual Audits?

Manual audits are the only way to ensure some components of accessibility:

- Form-field labels are Accurate/Descriptive
- Text alternatives are Necessary/Accurate
- Headings are Hierarchical/Descriptive
- Content order Preserves Meaning
- Form-field types are Appropriate
- Color Conveys Information
- Skip links are Appropriate
- Frame titles are Accurate
- Images are Informative

Humans Understand Intention

Only humans can assign the following attributes:

- Appropriate
- Descriptive
- Meaningful
- Accurate
- Necessary for understanding
- Informative

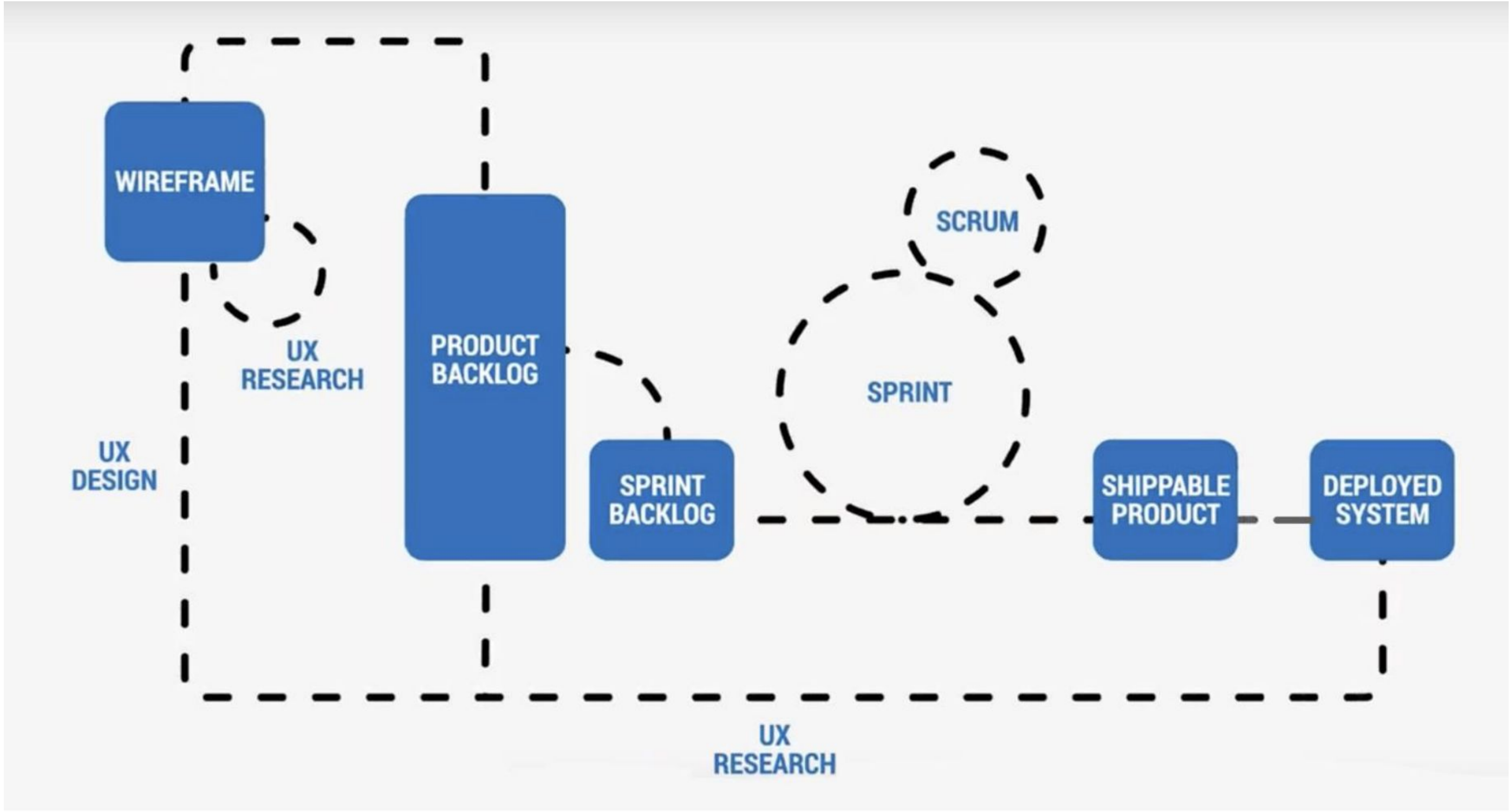
To objects, and their properties; Because only humans can infer intent when important information is missing or patterns are ambiguous.

Empathy as a Testing Tool

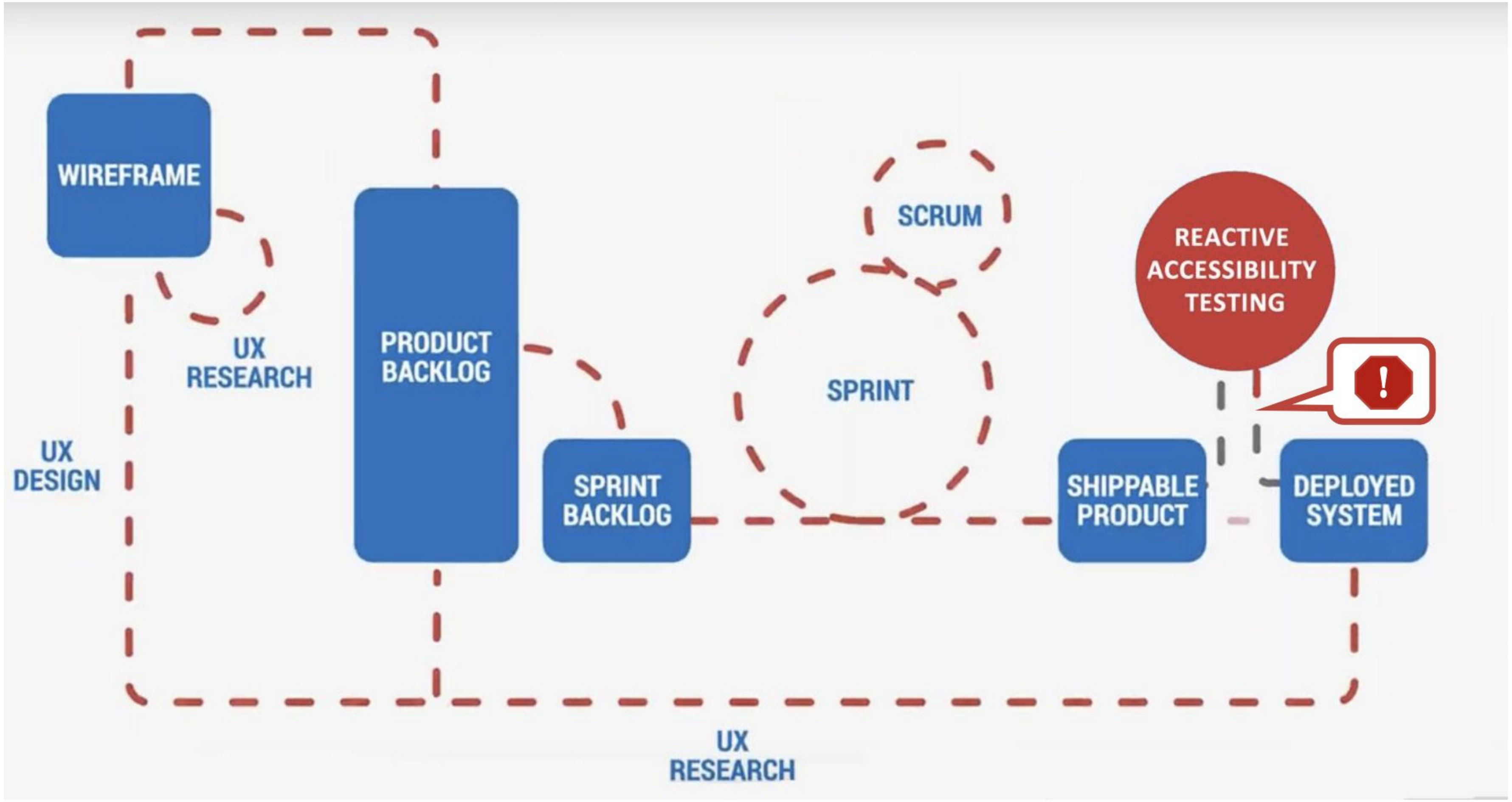
Empathy gives us insight into **intent**.

Our job as accessibility experts is to **ensure that intent survives** transit from content creators to their audiences, despite the ambiguities introduced by the medium.

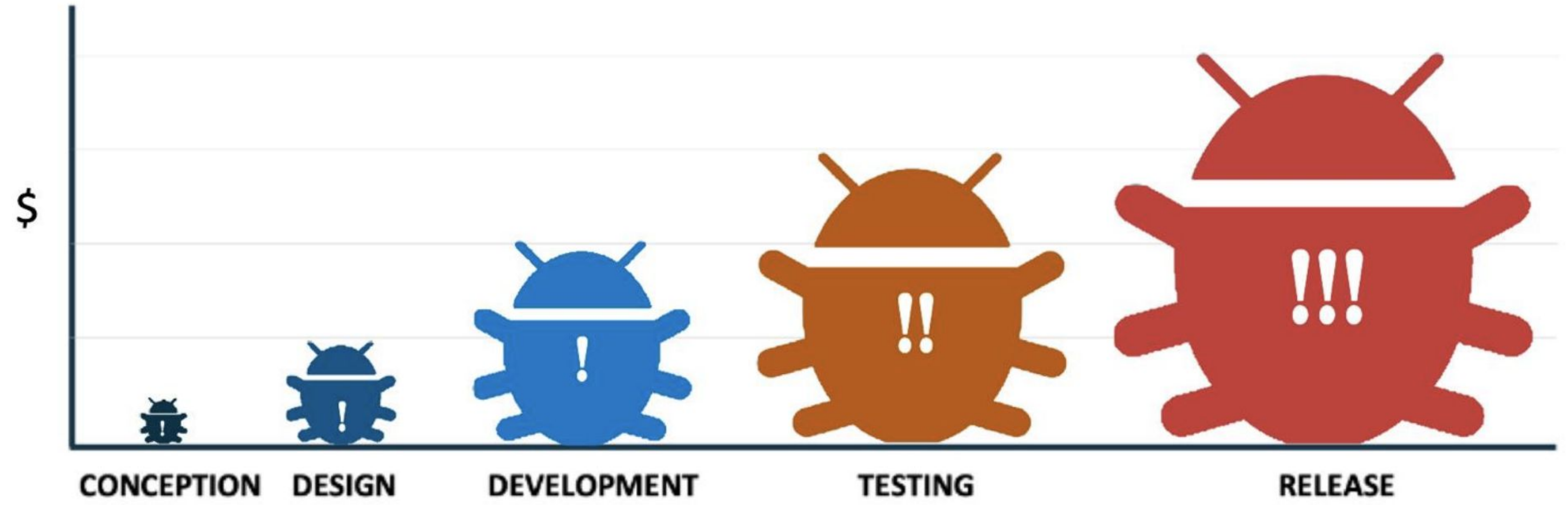
Where do you want to catch bugs?



Reactive accessibility testing creates risk and impacts resources

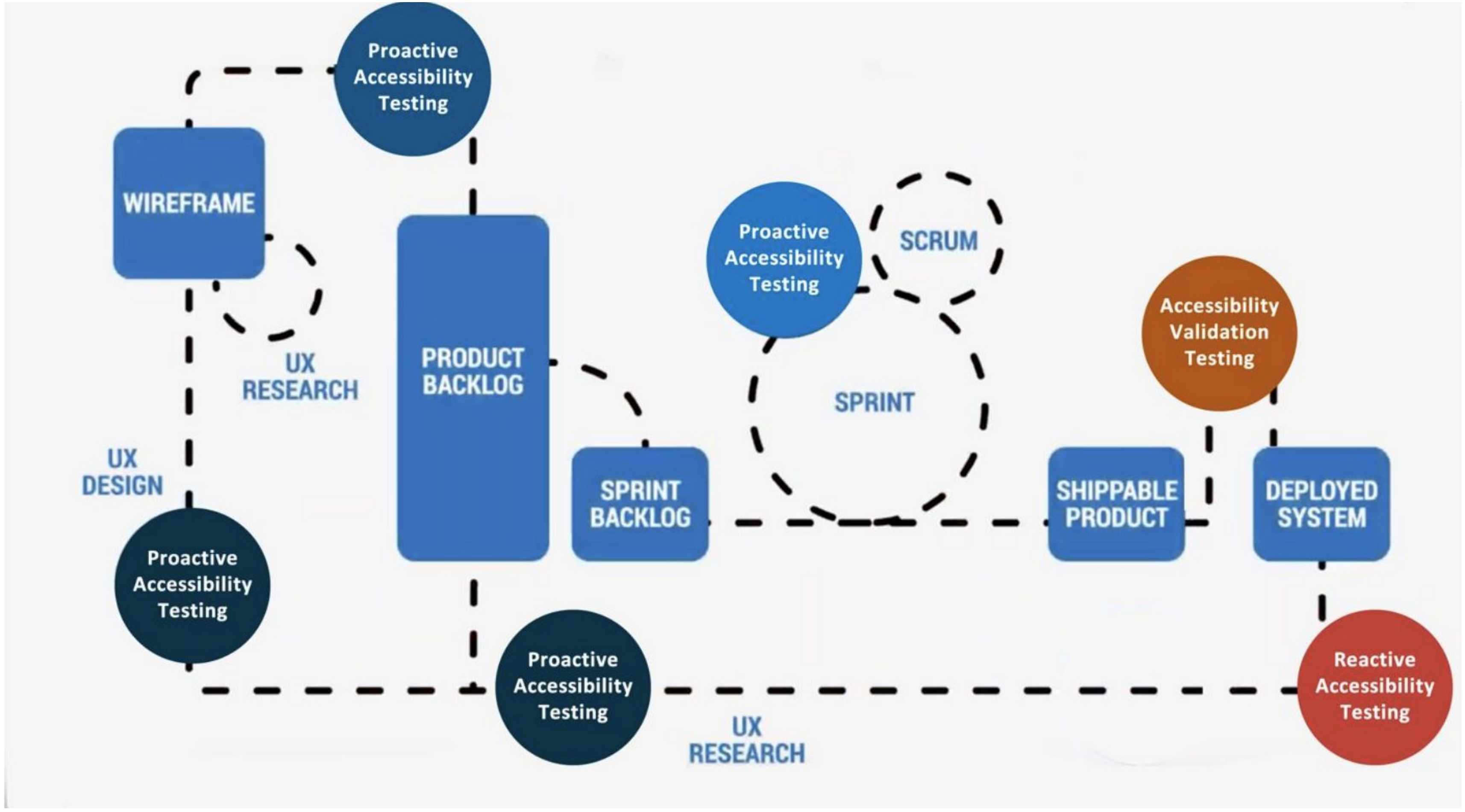


Cost of accessibility bugs is exponential



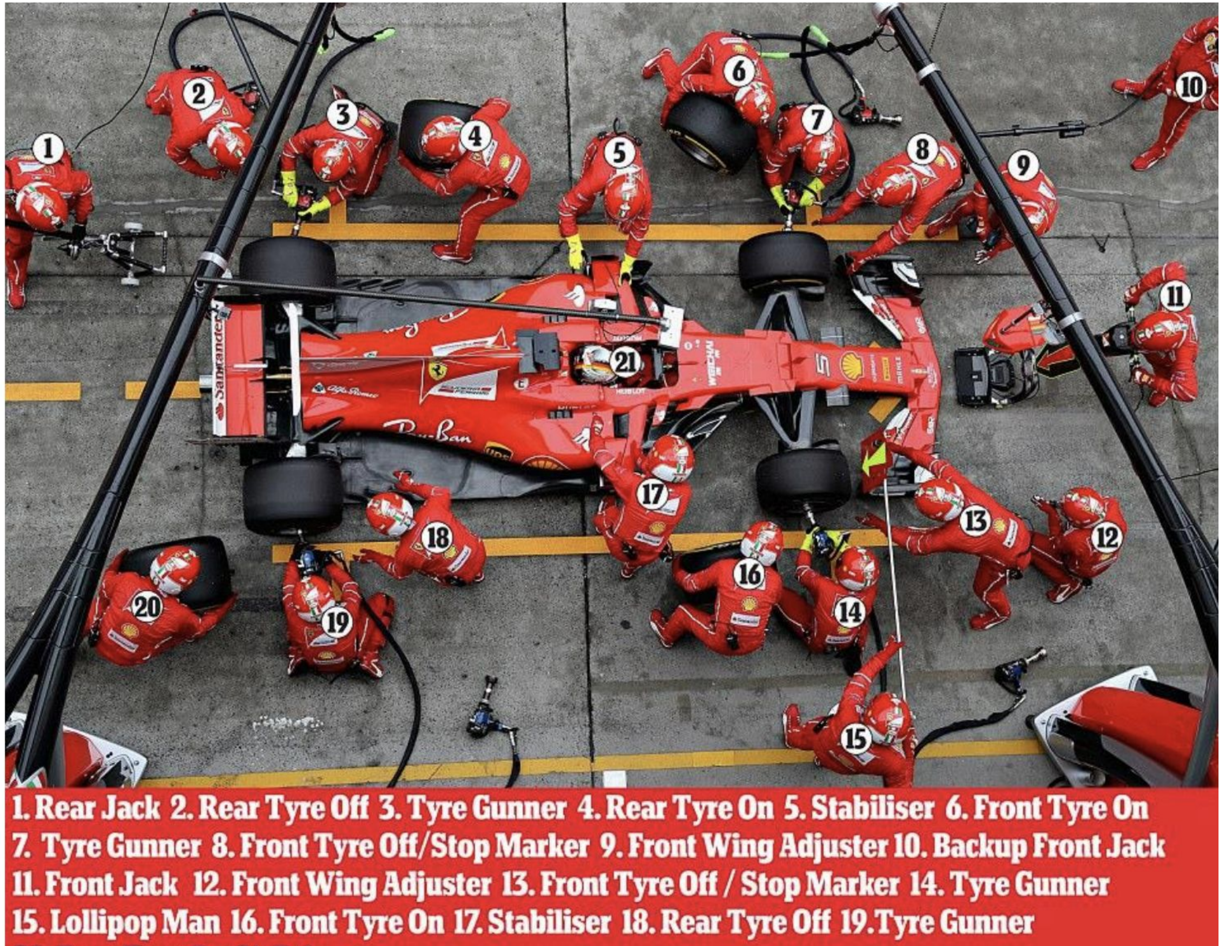
The longer it takes to discover an accessibility bug, the more it will cost your organization to fix it.

Effectively shift accessibility to the left

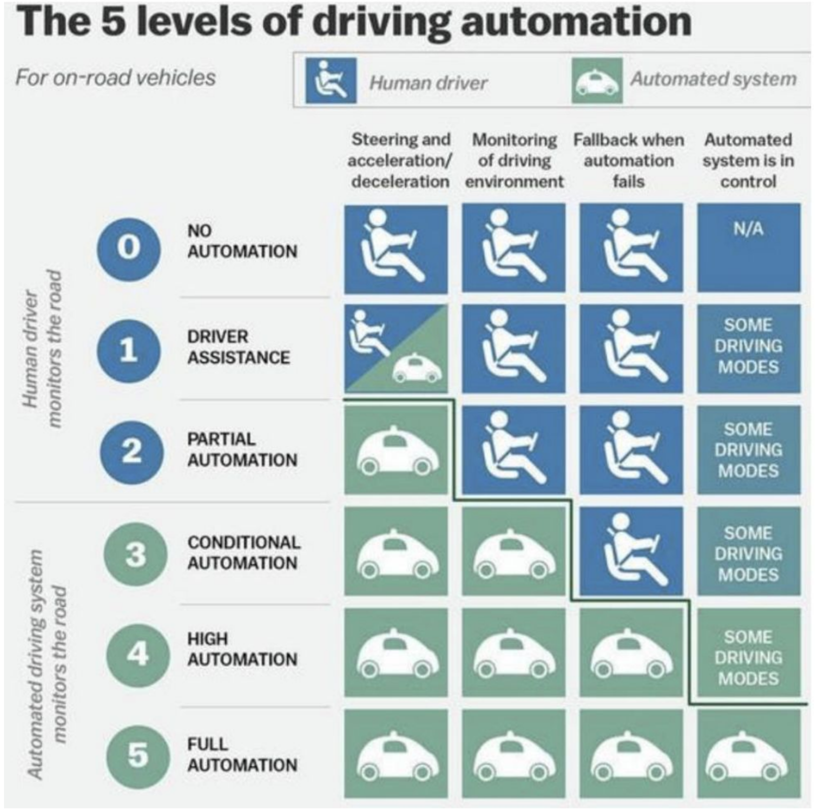

Proactive Accessibility Testing
WIREFRAME
UX RESEARCH
UX DESIGN
Proactive Accessibility Testing
PRODUCT BACKLOG
Proactive Accessibility Testing
SPRINT BACKLOG
Proactive Accessibility Testing
SCRUM
SPRINT
Accessibility Validation Testing
SHIPPABLE PRODUCT
DEPLOYED SYSTEM
Reactive Accessibility Testing
UX RESEARCH

Efficient accessibility testing for developers

- Automated
- Smart Manual Testing


1. Rear Jack 2. Rear Tyre Off 3. Tyre Gunner 4. Rear Tyre On 5. Stabiliser 6. Front Tyre On
7. Tyre Gunner 8. Front Tyre Off/Stop Marker 9. Front Wing Adjuster 10. Backup Front Jack
11. Front Jack 12. Front Wing Adjuster 13. Front Tyre Off / Stop Marker 14. Tyre Gunner
15. Lollipop Man 16. Front Tyre On 17. Stabiliser 18. Rear Tyre Off 19. Tyre Gunner

Levels of self driving cars and accessibility for web developers



The 5 levels of driving automation

For on-road vehicles

🟦 Human driver 🟩 Automated system

		Steering and acceleration/ deceleration	Monitoring of driving environment	Fallback when automation fails	Automated system is in control
Human driver monitors the road	0 NO AUTOMATION	🟦	🟦	🟦	N/A
	1 DRIVER ASSISTANCE	🟦🟩	🟦	🟦	SOME DRIVING MODES
	2 PARTIAL AUTOMATION	🟩	🟦	🟦	SOME DRIVING MODES
Automated driving system monitors the road	3 CONDITIONAL AUTOMATION	🟩	🟩	🟦	SOME DRIVING MODES
	4 HIGH AUTOMATION	🟩	🟩	🟩	SOME DRIVING MODES
	5 FULL AUTOMATION	🟩	🟩	🟩	🟩

- 0: No A11Y Automation
- 1: Dev Testing
- 2: Partial Automation
  - • Auto A11Y
  - • Integrate into every unit test
- Default part of Integration Testing
- Dev Testing
  - • Run auto testing tool in browser
  - • Perform targeted manual tests
- 3: Dev Testing in CI/CD

Combining Our Toolsets

Focusing brain power where it counts

Anatomy of an Automated Test

1. **Find objects** that match a given set of criteria

2. **Extract** properties from each object

3. **Apply** rules to determine if properties satisfy values

Auditing Tools replace Step 3 with: 3. **Present** objects and properties to human tester for analysis

Accessibility Testing Tools

- Microsoft Accessibility.insights.io
- Deque’s axecore
- Siteimprove’s accessibility extensionChris Pederick’s Web Developer Toolbar
- Paul Adam’s Bookmarklets for Accessibility
- Testing WebAIM’s WAVE Evaluation Tool
- Social Security Administration’s ANDI
- Google Web Developer Toolbar

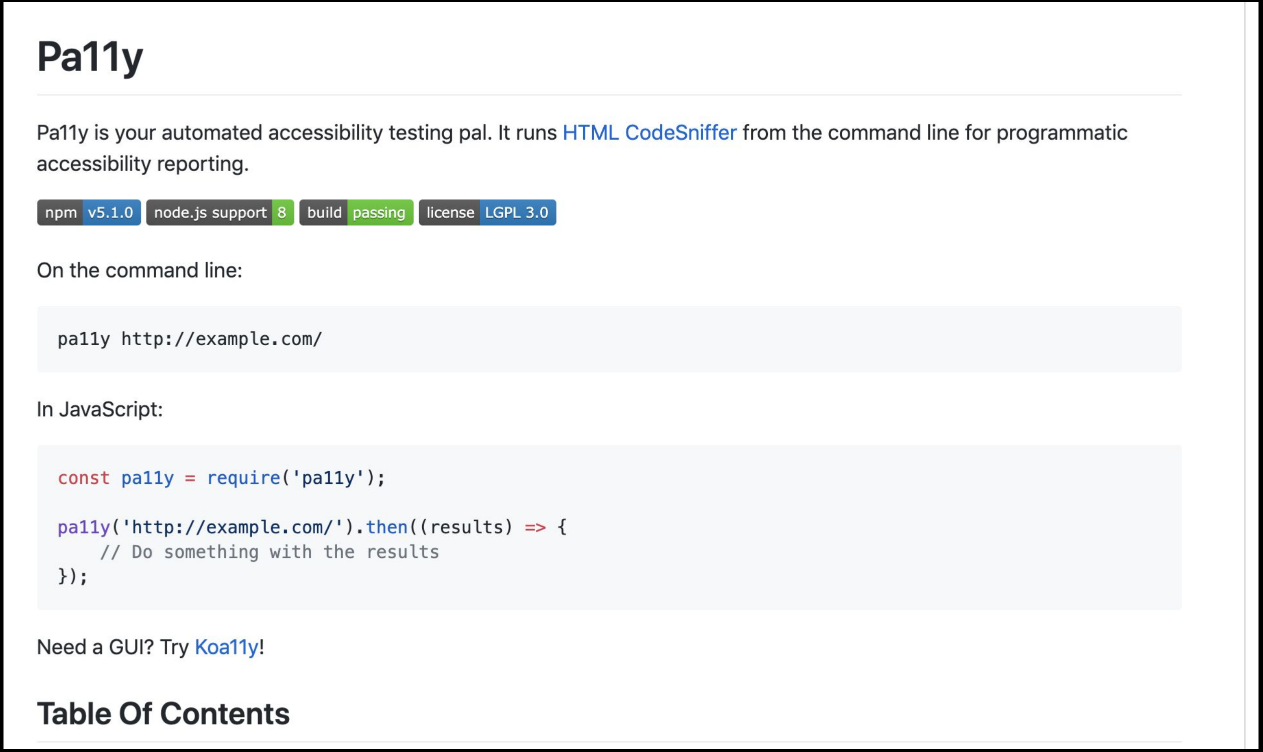
There are lots of resources online

The A11Y Project
Accessibility in government
Easy Checks - A First Review of Web Accessibility
Basic screen reader commands

Build accessibility into your CI/CD process

pA11y

https://www.ssa.gov/accessibility/andi/help/install.html



Pa11y

Pa11y is your automated accessibility testing pal. It runs HTML CodeSniffer from the command line for programmatic accessibility reporting.

npm v5.1.0 | node.js support 8 | build passing | license LGPL 3.0

On the command line:

```
pa11y http://example.com/
```

In JavaScript:

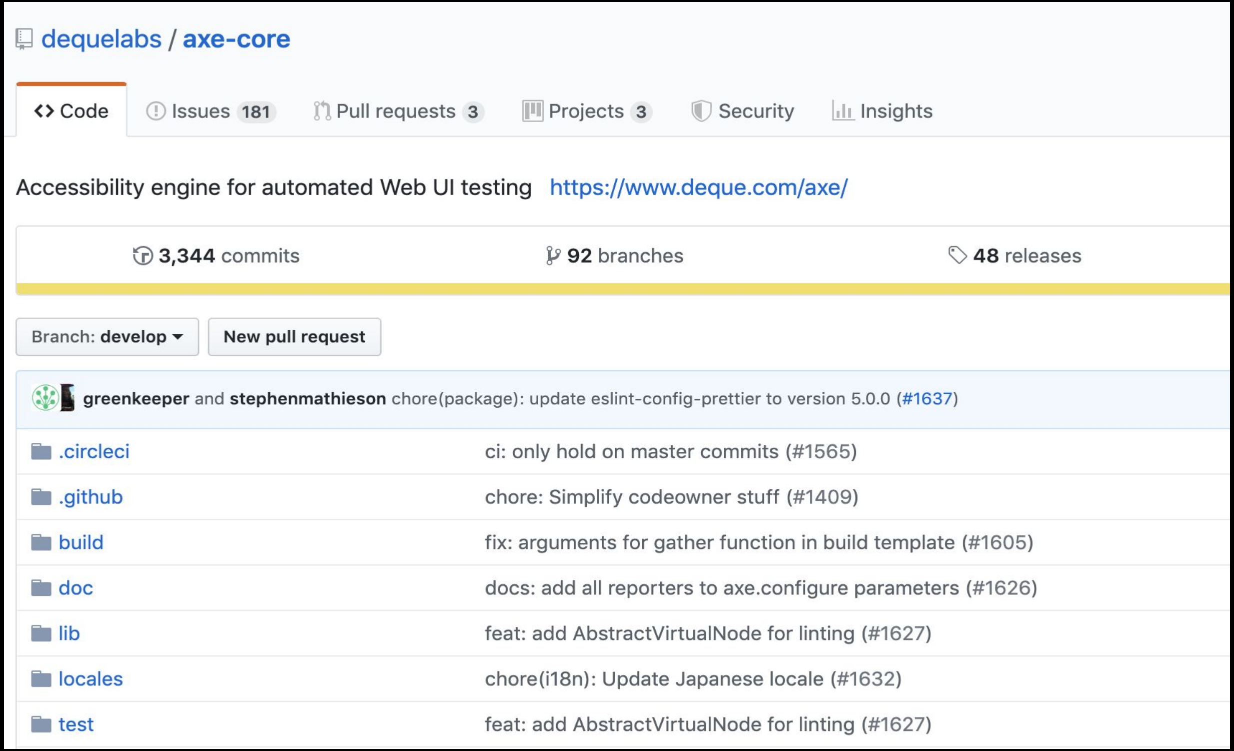
```
const pa11y = require('pa11y');

pa11y('http://example.com/').then((results) => {
    // Do something with the results
});
```

Need a GUI? Try Koa11y!

Table Of Contents

axecore

https://github.com/dequelabs/axe-core



dequelabs / axe-core

<> Code | Issues 181 | Pull requests 3 | Projects 3 | Security | Insights

Accessibility engine for automated Web UI testing https://www.deque.com/axe/

3,344 commits | 92 branches | 48 releases

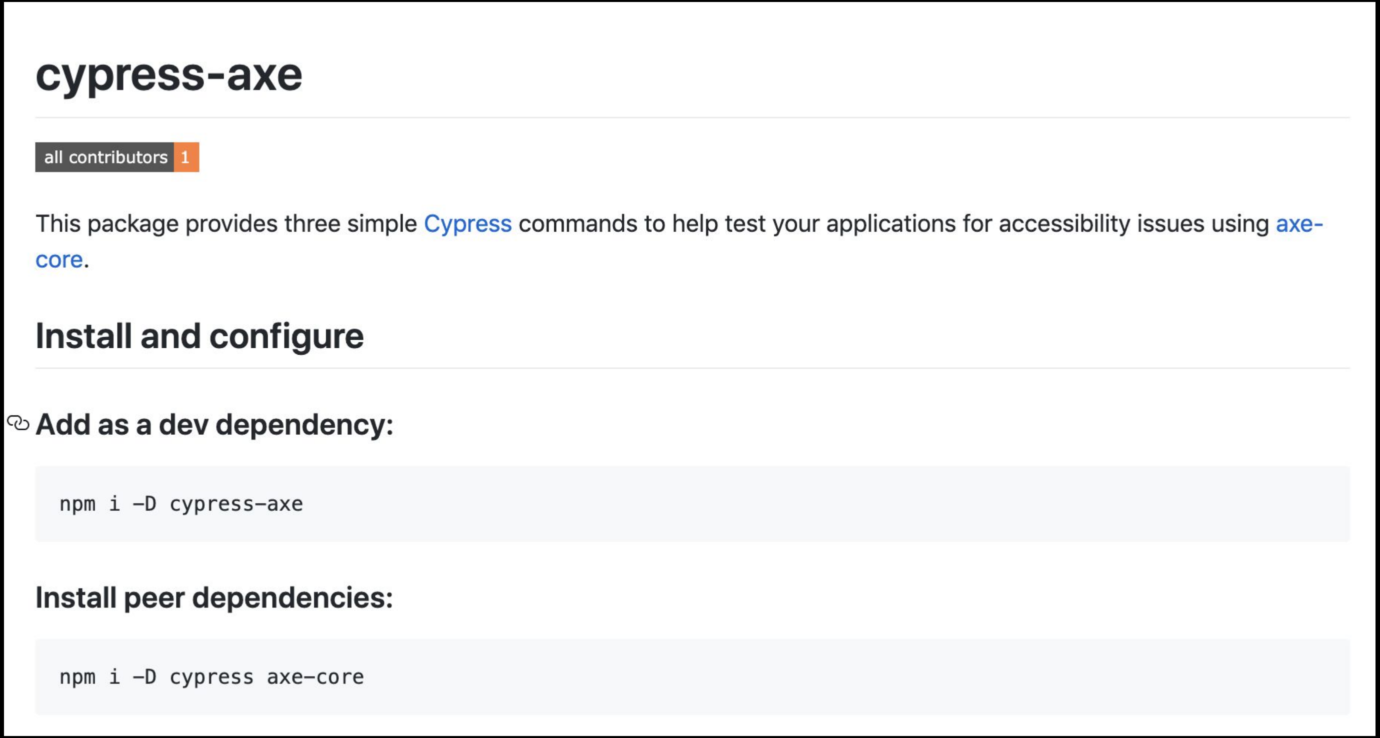
Branch: develop | New pull request

greenkeeper and stephenmathieson chore(package): update eslint-config-prettier to version 5.0.0 (#1637)

.circleci	ci: only hold on master commits (#1565)
.github	chore: Simplify codeowner stuff (#1409)
build	fix: arguments for gather function in build template (#1605)
doc	docs: add all reporters to axe.configure parameters (#1626)
lib	feat: add AbstractVirtualNode for linting (#1627)
locales	chore(i18n): Update Japanese locale (#1632)
test	feat: add AbstractVirtualNode for linting (#1627)

Cypress.io

https://github.com/avanslaars/cypress-axe



cypress-axe

all contributors 1

This package provides three simple Cypress commands to help test your applications for accessibility issues using axe-core.

Install and configure

Add as a dev dependency:

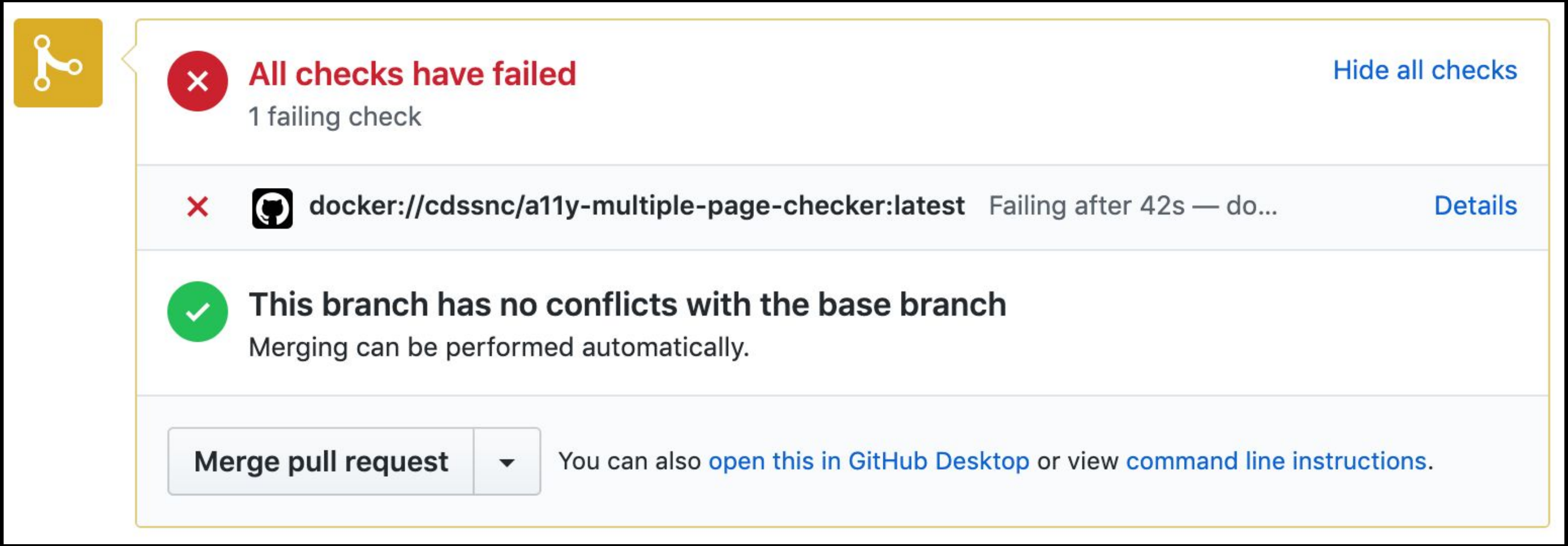
```
npm i -D cypress-axe
```

Install peer dependencies:

```
npm i -D cypress axe-core
```

Integrating automated a11y testing into the Github workflow

Github action on Pull Request deployments



All checks have failed

1 failing check

Hide all checks

✗ docker://cdssnc/a11y-multiple-page-checker:latest Failing after 42s — do... Details

✓ This branch has no conflicts with the base branch

Merging can be performed automatically.

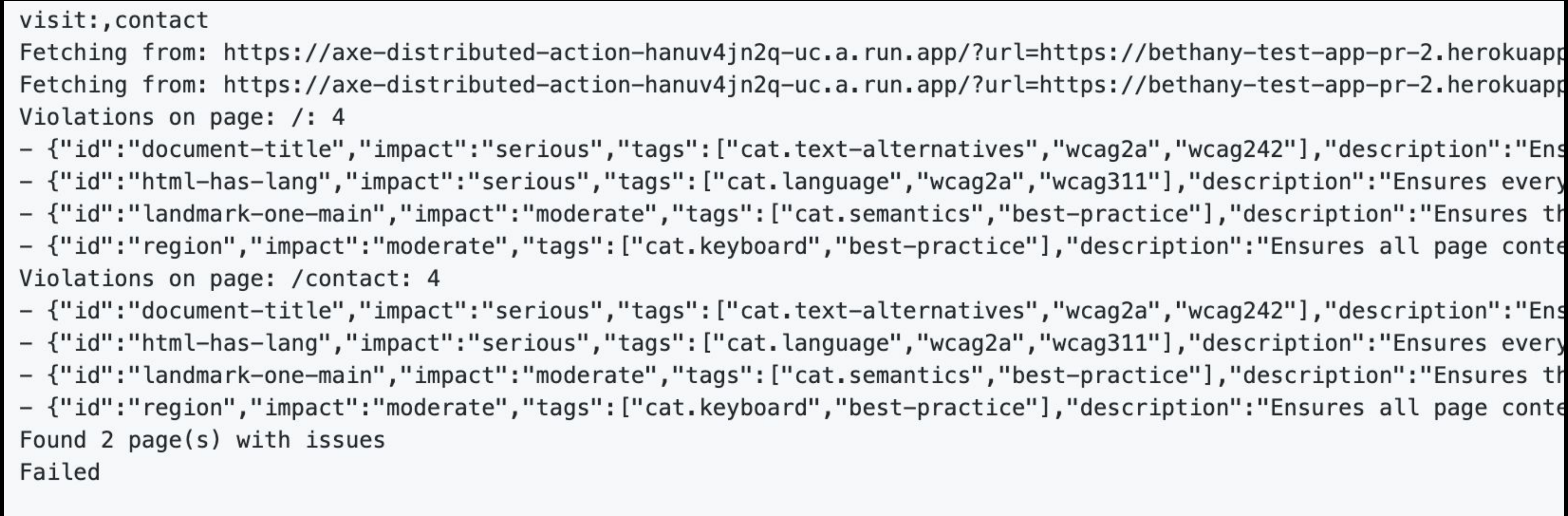
Merge pull request

You can also open this in GitHub Desktop or view command line instructions.

Integrating automated a11y testing into the Github workflow

Github action sends scan requests to a Google Cloud Run container (using axe-puppeteer)

If one or more of the requests fail, the github action reports a failure and logs the violations



```
visit:,contact
Fetching from: https://axe-distributed-action-hanuv4jn2q-uc.a.run.app/?url=https://bethany-test-app-pr-2.herokuapp
Fetching from: https://axe-distributed-action-hanuv4jn2q-uc.a.run.app/?url=https://bethany-test-app-pr-2.herokuapp
Violations on page: /: 4
- {"id":"document-title","impact":"serious","tags":["cat.text-alternatives","wcag2a","wcag242"],"description":"Ens
- {"id":"html-has-lang","impact":"serious","tags":["cat.language","wcag2a","wcag311"],"description":"Ensures every
- {"id":"landmark-one-main","impact":"moderate","tags":["cat.semantics","best-practice"],"description":"Ensures th
- {"id":"region","impact":"moderate","tags":["cat.keyboard","best-practice"],"description":"Ensures all page conte
Violations on page: /contact: 4
- {"id":"document-title","impact":"serious","tags":["cat.text-alternatives","wcag2a","wcag242"],"description":"Ens
- {"id":"html-has-lang","impact":"serious","tags":["cat.language","wcag2a","wcag311"],"description":"Ensures every
- {"id":"landmark-one-main","impact":"moderate","tags":["cat.semantics","best-practice"],"description":"Ensures th
- {"id":"region","impact":"moderate","tags":["cat.keyboard","best-practice"],"description":"Ensures all page conte
Found 2 page(s) with issues
Failed
```

Integrating automated a11y testing into the Github workflow

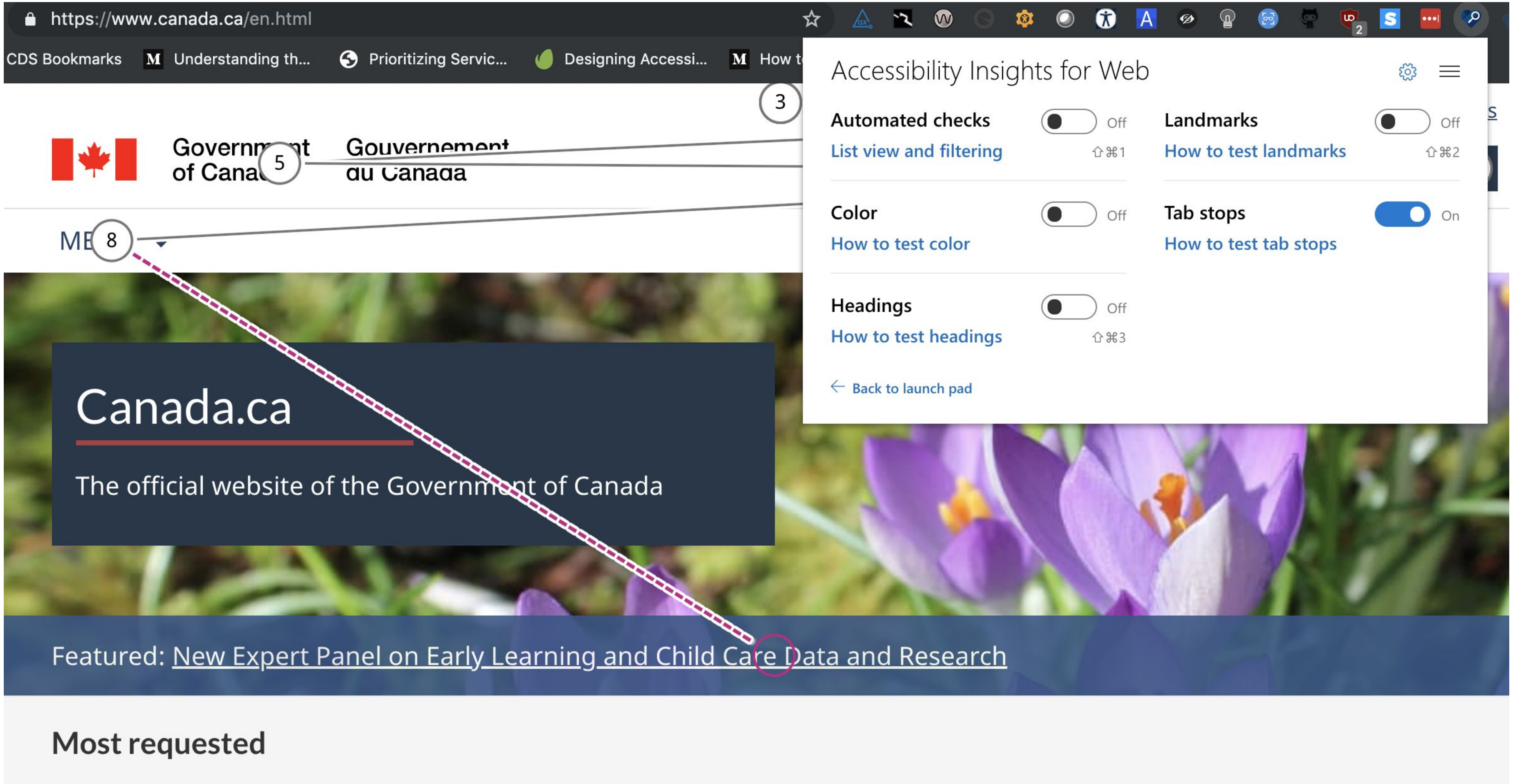
- Benefits:
  a. Can be set up from the beginning of the development process
     - Incorporates accessibility awareness and thinking into developers early on
  b. Catches low-hanging fruit, and makes sure new code isn’t being merged into master which fails accessibility rules.

Things you can do

Tab through your web content

[DEMO GOV.UK]

Is the tab order logical?



https://www.canada.ca/en.html

CDS Bookmarks | Understanding th... | Prioritizing Servic... | Designing Accessi... | How t

Accessibility Insights for Web

Automated checks	Off	Landmarks	Off
List view and filtering	⇧⌘1	How to test landmarks	⇧⌘2
Color	Off	Tab stops	On
How to test color		How to test tab stops	
Headings	Off		
How to test headings	⇧⌘3		

← Back to launch pad

Canada.ca

The official website of the Government of Canada

Featured: New Expert Panel on Early Learning and Child Care Data and Research

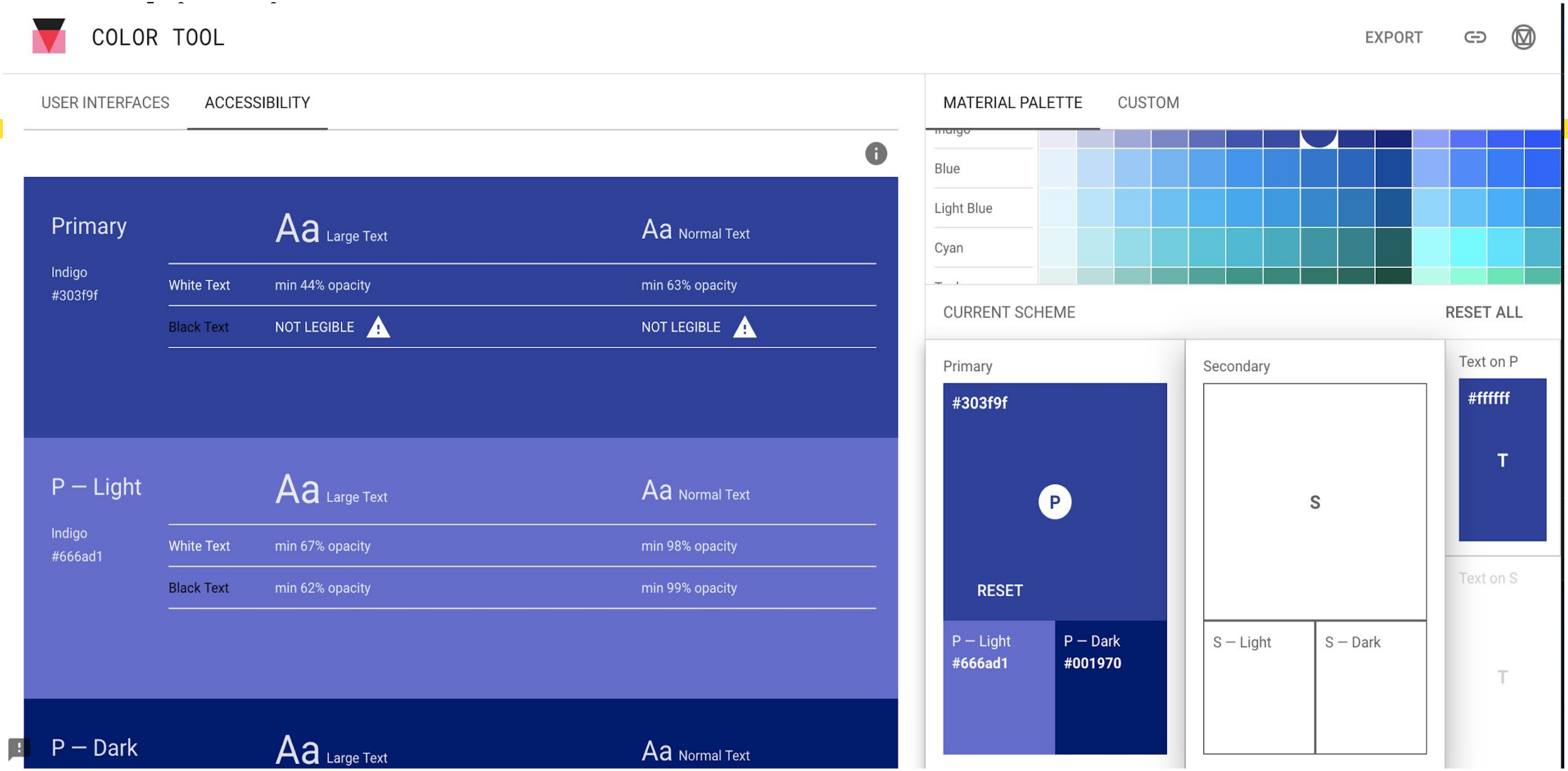
Most requested

Try using a screen reader

DEMO: Github

Make sure everything works with the keyboard only

Allows you to measure the accessibility level of any color



COLOR TOOL

EXPORT

USER INTERFACES | ACCESSIBILITY

Primary
Indigo
#303f9f

	Aa Large Text	Aa Normal Text
White Text	min 44% opacity	min 63% opacity
Black Text	NOT LEGIBLE ⚠	NOT LEGIBLE ⚠

P – Light
Indigo
#666ad1

	Aa Large Text	Aa Normal Text
White Text	min 67% opacity	min 98% opacity
Black Text	min 62% opacity	min 99% opacity

P – Dark

	Aa Large Text	Aa Normal Text

MATERIAL PALETTE | CUSTOM

Blue
Light Blue
Cyan

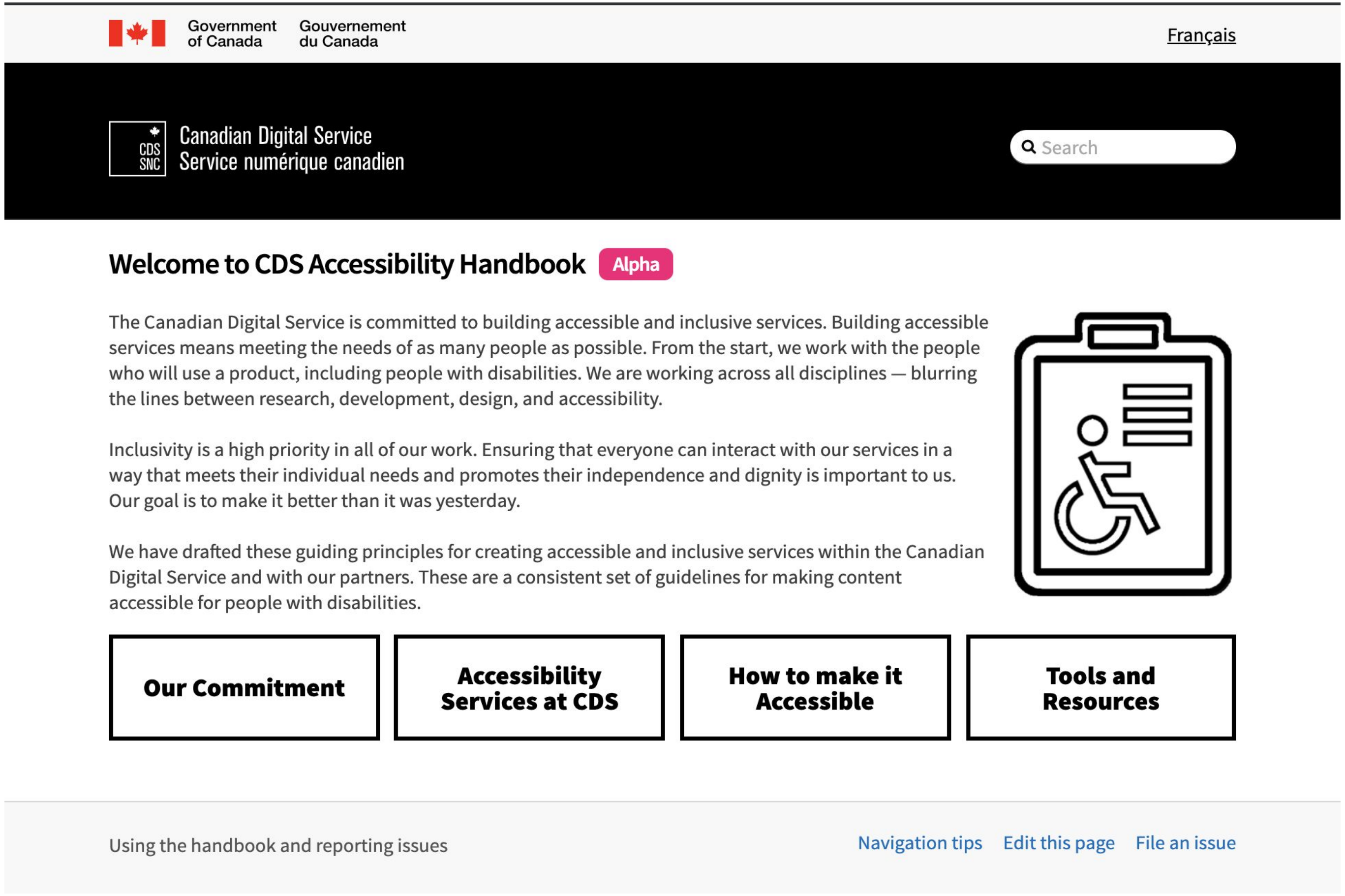
CURRENT SCHEME — RESET ALL

Primary: #303f9f, P, RESET
P – Light #666ad1
P – Dark #001970

Secondary: S
S – Light
S – Dark

Text on P: #ffffff, T
Text on S: T

Check out the Accessibility Handbook



Government of Canada Gouvernement du Canada

Français

Canadian Digital Service
Service numérique canadien

Search

Welcome to CDS Accessibility Handbook Alpha

The Canadian Digital Service is committed to building accessible and inclusive services. Building accessible services means meeting the needs of as many people as possible. From the start, we work with the people who will use a product, including people with disabilities. We are working across all disciplines — blurring the lines between research, development, design, and accessibility.

Inclusivity is a high priority in all of our work. Ensuring that everyone can interact with our services in a way that meets their individual needs and promotes their independence and dignity is important to us. Our goal is to make it better than it was yesterday.

We have drafted these guiding principles for creating accessible and inclusive services within the Canadian Digital Service and with our partners. These are a consistent set of guidelines for making content accessible for people with disabilities.

Our Commitment

Accessibility Services at CDS

How to make it Accessible

Tools and Resources

Using the handbook and reporting issues

Navigation tips Edit this page File an issue

Together, we can build more inclusive and accessible services.

Questions?

CDS
SNC

Thank you!

Merci!

digital.canada.ca · @CDS_GC | numerique.canada.ca · @SNC_GC