The Semi-Automated Accessibility Testing Coverage Report

How a new kind of testing can stop 80%+ of accessibility issues during development

deque

Contents

The Problem

The accessibility industry needs a new strategy. We've been trying to make digital accessibility a reality for over 20 years now. While meaningful progress has been made, the state of the web is still largely inaccessible. Evident in reports like the WebAIM Million, inaccessibility is a systemic problem. Some causes of this problem include:

- The rate at which developers who aren't trained to include accessibility create new digital assets.
- The complexity of the international standard for testing accessibility: Web Content Accessibility Guidelines (WCAG).
- The time it takes to test a digital asset for accessibility
- How quickly the results of an accessibility test "rot".
    - The moment a developer or content contributor changes something on a page, they could have introduced accessibility issues.
- Not enough experts/expertise in digital accessibility to meet the world's needs/demands.

The Challenge and the Wizard

To truly make digital accessibility a reality we must make it easy for developers to write software that is accessible from the start. And we can't expect all the developers in the world to immerse themselves in the ever-changing WCAG conformance standards and understand every available assistive technology. There must be a way to help developers review their code without having them spend months, or years, becoming digital accessibility experts.

What if...what if...we had an accessibility wizard that could guide developers to quickly and easily find and fix most of the WCAG issues in their own code? This wizard would be designed to inspire developers to proactively test for accessibility, and actually help them learn how to write accessible code in the first place.

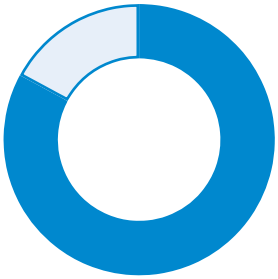
How much of digital accessibility testing can really be automated?

Let's take a step back and review fully automated testing capabilities. Automated accessibility testing is when a rules engine, such as axe-core, scans, or analyzes a web page for accessibility issues. These rules engines are built to test against accessibility standards, such as WCAG, which have predefined criteria for whether something is accessible. Automated testing tools can be browser extensions, like axe DevTools, or rules-engines built into automated test environments.

In 2021, Deque published an Accessibility Coverage Report based on expert automatic and manual testing of 13,000+ pages/page states, and nearly 300,000 issues. In this real-world data, we found that **57.38% of issues from first-time audit customers could be found with automated testing**. We are confident in the accuracy of the coverage percentage from this data set because it was derived from a large sample size and from a wide variety of first-time customers.

How can we more easily address what's left to test?

Semi-automated or wizard-based testing is a new type of accessibility testing technology developed by Deque that significantly increases testing coverage during development. This approach covers many issues that previously could only be tested manually by accessibility experts. These issues are now detectable through a series of simple guided questions and exercises. We call this technology Intelligent Guided Testing (IGT)—part of axe DevTools. In combination with automated testing, IGTs cover, on average, **80.39% of all accessibility issues.**



80.39% of Total Issues were detected using automated and semi-automated (IGT) tests

Again, we used data from a large sample size of WCAG audits done for first-time customers. Having said that, this research was more complicated than that conducted for the original report. We had to look deeper than just which WCAG success criteria failed. We had to identify and segment individual manual issue descriptions. We have an extensive manual issue description library that is vetted by our team of experts. This detailed library covers 99% of all manual issues reported in this data set. In other words, only 1% of manual issues reported are custom issues not already documented in the library.

After a thorough review, the key findings from our analysis are:

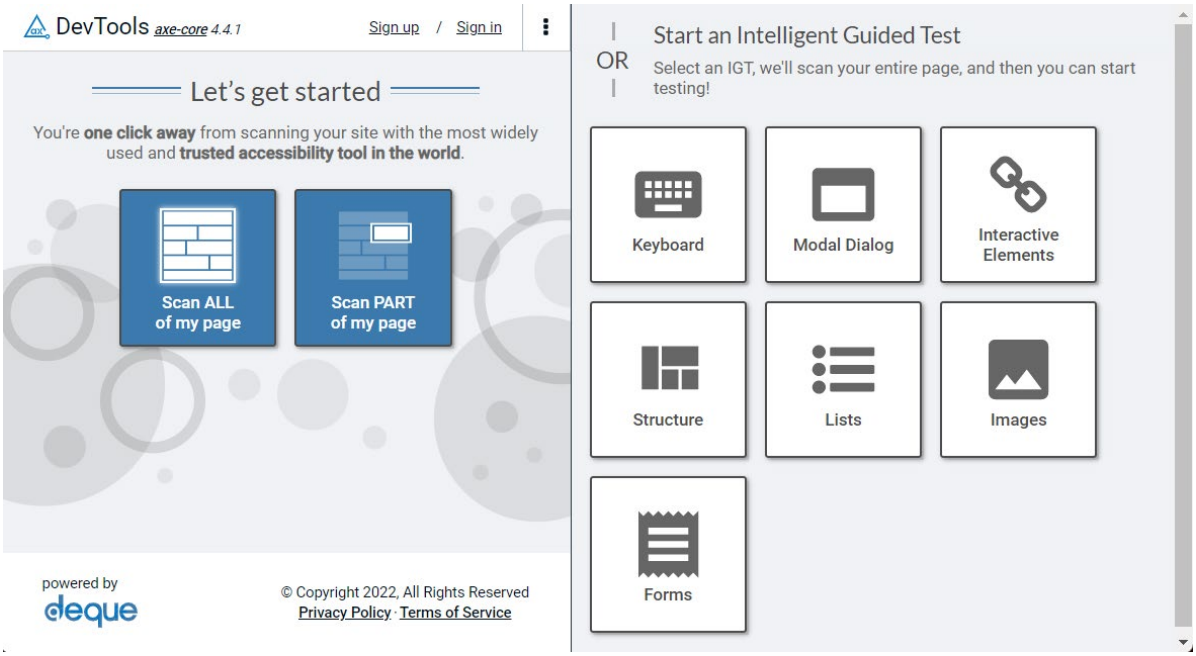
On average, across all the audits included in the sample data, we determined that if developers had used axe DevTools automated testing + Intelligent Guided Testing, they could have found over 80.39% of all the issues our experts reported.

Percent of Issues Found	Testing Method
57.38%	Axe-core (automated)
23.01%	Intelligent Guided Tests
80.39%	Total

Intelligent Guided Tests (IGT)

As previously described, IGTs are semi-automated tests that greatly increase accessibility testing coverage during development while also simplifying and streamlining the process for completing these tests. With IGTs, any developer can quickly identify and address WCAG issues for:

1. **Keyboard** (includes keyboard focusable, visual focus...)
2. **Modal Dialogs** (includes keyboard accessible, focus order...)
3. **Interactive Elements** (includes accessible name present and meaningful, appropriate role, appropriate states...)
4. **Structure** (includes semantic heading structure, appropriate page title, accurate language of page, language of parts...)
5. **Lists** (semantic list structure)
6. **Images** (appropriate alt text for informative and decorative images, images of text...)
7. **Forms** (appropriate label, field groups, visible label, informative errors...)



Manual Issues Supported by IGTs

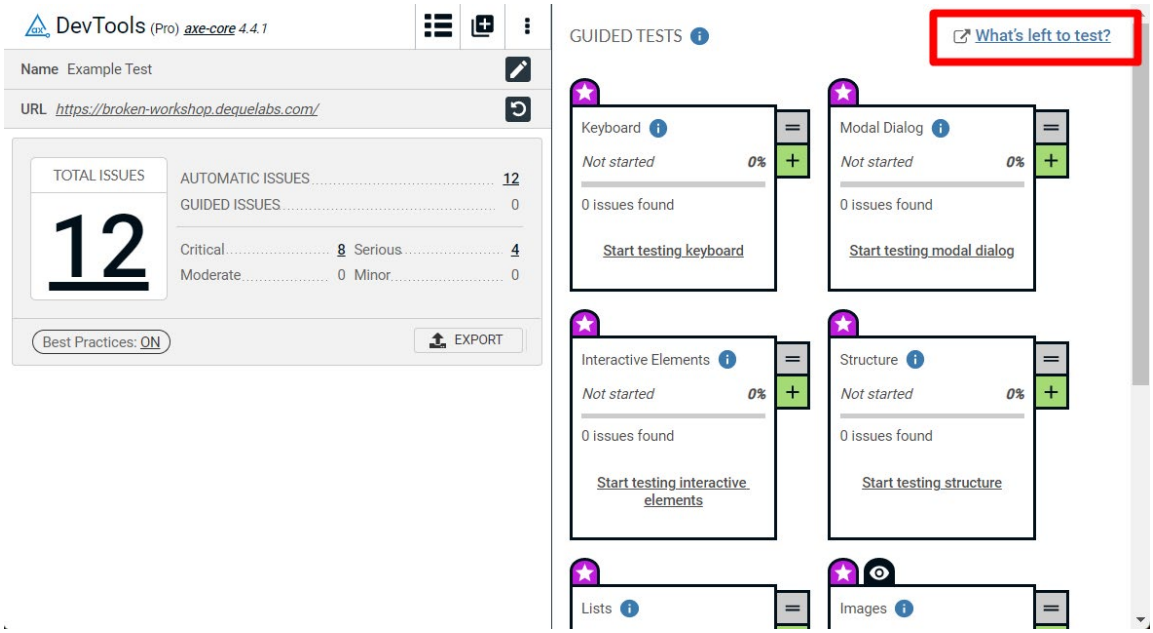
Every accessibility issue created during development raises your costs and creates access barriers for real people. We analyzed the data from our large sample of WCAG audits done for first-time customers to see how each of our guided tests make it possible for developers to find and fix issues efficiently…and how IGTs can teach them how to code it correctly the first time. The table below highlights how the seven IGTs available today empower developers to quickly and easily perform advanced semi-automated accessibility testing without having to become an accessibility expert.

Intelligent Guided Test	% of Total IGT Issues	Just a few examples of the types of issues easily found with axe DevTools IGT	Applicable WCAG Success Criteria
Lists	3.67%	• Visual list is not marked up as a list. • Content is not a list but is marked as such.	1.3.1
Modal Dialog	6.06%	• The element appears and functions like a modal dialog but is missing the required ARIA role(s) and/or attribute(s). • Keyboard focus is not returned to the triggering element • Keyboard focus is not maintained within the modal. It is possible to tab out of the modal.	1.3.2, 2.4.3
Forms	14.07%	• Group of form controls not associated with group label • Label not descriptive	1.3.1, 2.4.6, 3.3.1, 3.3.2, 3.3.3, 3.3.4
Images	14.59%	• Text alternative does not serve the same purpose as image • Decorative image has a non-empty alt attribute • Image of text is used instead of real text	1.1.1, 1.4.5
Keyboard	15.66%	• Visual focus indicator is missing. • There is no way to perform the function using only the keyboard.	2.1.1, 2.4.7
Structure	18.19%	• Visual heading text is not marked as heading • Text should not be marked as a heading • Page <title> does not identify purpose of page • Language of page not accurate • Lang-change-not-marked	1.3.1, 2.4.2, 3.1.1, 3.1.2
Interactive Elements	27.75%	• Missing or incorrect role for interactive elements including button, link, checkbox, select, radio, slider… • Missing or incorrect state for interactive elements including button, checkbox, select, radio, slider…	2.4.4, 4.1.2

There are clearly more WCAG criteria that must be manually tested. We are working on many of those tests today and will continue to add IGTs to axe DevTools. We can do this because we built axe DevTools to support regular web development practices. We are committed to helping developers write code that is born accessible.

Advanced Manual Tests: The 19.61% of "What's left to test"

After you've run axe DevTools automated testing and the semi-automated IGTs, the next logical question is, "What's left to test"? Based on the actual number of completed tests, we document the remaining tests in the "What's left to test?" link on the axe DevTools dashboard.



Below are the types of tests in that 19.61% of What's Left to Test/Advanced Manual Testing instructions (axe DevTools Pro login required):

- Site-wide Tests
  - Consistency
  - Session Timeout
- Page-Level Tests
  - Interruption on Page Load
  - Multiple ways
  - Orientation
- Page State Tests
  - Automatic Behavior
  - Page Meaning
  - Page Structure
  - Interactive Elements
  - Static Content
  - Sensory Characteristics
  - Forms
  - Focus Management
  - Test Resize, Reflow, and Spacing
  - Status Messages
  - Alternatives for Timed Media
  - Tables
  - Time Limits
  - Motions and Gestures

Five of the most common issues that are not yet covered by Intelligent Guided Tests (and would have to be tested for manually) are:

1. State of active component lacks 3 to 1 contrast ratio (1.4.11)
2. Link or button text lacks 4.5:1 contrast ratio on hover or focus (1.4.3)
3. Status message is not automatically announced by the screen reader (4.1.3)
4. Parts of an icon (with no text) do not have a contrast ratio of 3 to 1 against adjacent color(s). These icon parts are required for understanding. (1.4.11)
5. Content is lost, clipped, or obscured when the page is zoomed to 200%. (1.4.4)

Completing this "last mile" of testing currently requires in-depth accessibility knowledge. We are actively working to create more automated rules and more IGTs to cover more and more WCAG requirements. Our mission is to help your developers be even more efficient and effective in writing accessible code from the start.

The Impact of Testing Accuracy

Not All Accessibility Tools Are Created Equal

The accuracy of accessibility tools depends on close collaboration between developers and the accessibility experts who create these tools.

When Deque reports issues using automated and semi-automated testing, we work very hard to prevent false positives. This means that any issues we cannot state are–in fact–issues, with 100% certainty are rarely reported. False positives can waste time, erode trust, and derail progress. When the rare false positive is reported, we treat it as a very high priority bug and research fix it quickly.

Indeed, our products can also find potential issues that require manual verification as well as best practice issues. But for the integrity and accuracy of this report, if a flagged item needed manual verification, we did not include it unless it had been manually tested and confirmed as a real accessibility issue by one of our experts.

Exclusion of potential issues that were not actually issues and the exclusion of best practice issues ensured that we did not inflate the coverage percentage. This also helped us stay true to the needs of development and QA managers that need to budget and plan for resources ahead of time. They require an accurate understanding of accessibility testing coverage to forecast resources that will be needed to meet the product deliverables, timelines, and budget.

Comprehensive Testing of Complex Dynamic Pages

Comprehensive testing of complex web pages requires understanding how the page works—including all the possible states of that dynamic page. For example, if a page has ten different states and testing is only performed on three of those states, the testing results will be incomplete.

So, the accuracy of testing depends on the person doing the testing to understand and complete the following:

- Know all of the possible states of the page
  a. Run automated testing on all the possible states of the page
  b. Complete the semi-automated intelligent guided tests on all the possible states of the page
  c. Have an accessibility expert complete the “what’s left to test” * manual testing on all the possible states of the page

While axe DevTools greatly simplifies the testing process, we acknowledge that it can still be time-consuming to test every possible state of a complex dynamic page. Nevertheless, this is necessary to return comprehensive and accurate results. This is one of the drivers behind our commitment to make accessibility testing work for those who must deliver on the task—from developers through accessibility experts.

Repeat issues in common components only counted once

Modern web pages very often include templates (like header, footer, navigation, etc.) repeated across multiple pages. Any accessibility issues present on these templates can most likely be fixed once and update all the pages on which they are included. Therefore, we account for issues on these common templates only once for this analysis.

For example, if a header had eight issues that were repeated across 10 pages, instead of counting these as 80 issues our analysis includes only eight issues. Counting all 80 issues would lead to an increase in the overall percentage of issues discovered using automated and semi-automated testing. While this may not be a completely accurate representation of user experience on these 10 pages, it aligns more closely with the effort required to fix the issues in the header.

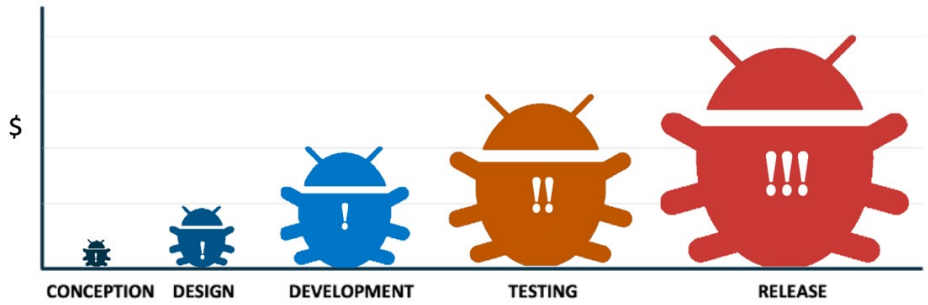
In Summary

Accessibility coverage should not just be defined by the number of WCAG Success Criteria that are addressed, but also by the volume of issues that can be covered in real-life examples. Our large sample size covers a wide range of first-time audits and provides an accurate estimate of how much issue coverage to expect from automated and semi-automated accessibility technology.

The coverage percentage of 57.38% for automated testing gives development teams and accessibility experts a more accurate understanding of the value they receive from using automated tools and reduces the barrier of entry for non-experts who’d like to use free automated tools.

Adding a semi-automated testing approach, like the Intelligent Guided Tests offered in axe DevTools, can increase this coverage to 80.39% on average, and in many cases even higher.

Cost of Accessibility Bugs



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

As we all continue to make the web a better, more inclusive place, it is critical to employ automation to help us move the needle. The web is changing faster than any manual actions can keep pace with. Accurately communicating the coverage automation offers enables us all to reconsider our approaches, our processes, and our goals for accessibility. The ultimate impact can be tremendous, helping to remove any hesitancy from newcomers and putting us all on a better, more sustainable path toward universal digital accessibility.

*Axe and Intelligent Guided Testing (IGTs) are trademarks of Deque Systems, Inc.

Appendix: By the Numbers

Automated Accessibility Data

Access the Automated Testing Coverage Report at: https://www.deque.com/automated-accessibility-testing-coverage/

Semi-Automated Intelligent Guided Testing Data

Cost of Accessibility Bugs

#	SC#	Success Criteria	Axe DevTools Coverage	axeDev Tools Coverage Percent	What Percent is NOT covered
1	1.1.1	Non-Text Content	Complete	100%	0%
2	1.3.1	Info and Relationships	Partial	84%	16%
3	1.3.2	Meaningful Sequence	Partial	6%	94%
4	1.4.1	Use of Color	Partial	12%	88%
5	1.4.3	Contrast (Minimum)	Partial	83%	17%
6	1.4.5	Images of Text	Complete	100%	0%
7	2.1.1	Keyboard	Partial	47%	53%
8	2.2.1	Timing Adjustable	Partial	2%	98%
9	2.4.1	Bypass Blocks	Partial	97%	3%
10	2.4.2	Page Titled	Complete	100%	0%

#	SC#	Success Criteria	Axe DevTools Coverage	axeDev Tools Coverage Percent	What Percent is NOT covered
11	2.4.3	Focus Order	Partial	47%	53%
12	2.4.4	Link Purpose (In Context)	Partial	55%	45%
13	2.4.6	Headings and Labels	Complete	100%	0%
14	2.4.7	Focus Visible	Complete	100%	0%
15	3.1.1	Language of Page	Complete	100%	0%
16	3.1.2	Language of Parts	Complete	100%	0%
17	3.3.1	Error Identification	Partial	95%	5%
18	3.3.2	Labels or Instructions	Complete	100%	0%
19	3.3.3	Error Suggestion	Complete	100%	0%
20	3.3.4	Error Prevention	Complete	100%	0%
21	4.1.1	Parsing	Complete	100%	0%
22	4.1.2	Name, Role, Value	Partial	96%	4%

Complete Coverage: All the issues in the Total Issues column could have been discovered with IGT.
Partial coverage: The rules in IGT do not cover all possible scenarios for these success criteria. A percentage of issues (depending on the page content) in the Total Issues column could have been discovered with IGT.

axe DevTools Coverage of the top 15 SC that account for 94.54% of all WCAG Issues

Maintaining velocity is critical to both development teams and the businesses they support. Leveraging automated and semi-automated testing during development saves time, effort, and rework, which directly reduces costs while still supporting velocity. As outlined in the table below and in our original Automated Accessibility Coverage Report, looking at issue coverage by volume enables teams to focus on and proactively fix the majority of issues.

#	SC #	Success Criteria Name	axe DevTools Coverage	% of Issues from Auto	% of Issues from IGT	% of Issues from What's Left to Test	% of ALL Issues by SC	Cumulative
1	1.4.3	Contrast (Minimum)	83.11%	83.11%	0.00%	16.89%	30.08%	30.08%
2	4.1.2	Name, Role, Value	96.00%	54.42%	41.58%	4.00%	16.37%	46.45%
3	1.3.1	Info and Relationships	84.00%	45.17%	38.83%	16.00%	12.33%	58.78%
4	4.1.1	Parsing	100.00%	100.00%	0.00%	0.00%	11.69%	70.47%

#	SC #	Success Criteria Name	axe DevTools Coverage	% of Issues from Auto	% of Issues from IGT	% of Issues from What's Left to Test	% of ALL Issues by SC	Cumulative
5	1.1.1	Non-text Content	100.00%	67.57%	32.43%	0.00%	8.04%	78.51%
6	2.4.3	Focus Order	47.00%	0.00%	47.00%	53.00%	3.24%	81.75%
7	2.1.1	Keyboard	47.00%	2.49%	44.51%	53.00%	3.19%	84.94%
8	2.4.7	Focus Visible	100.00%	0.00%	100%	0.00%	2.48%	87.42%
9	1.4.11	Non-text Contrast	0.00%	0.00%	0.00%	100.00%	1.54%	88.96%
10	1.4.1	Use of Color	12.17%	12.17%	0.00%	87.83%	1.26%	90.22%
11	1.3.2	Meaningful Sequence	6.00%	0.00%	6.00%	94.00%	1.12%	91.34%
12	3.3.2	Labels or Instructions	100.00%	20.42%	79.58%	0.00%	0.86%	92.20%
13	2.4.1	Bypass Blocks	97.00%	79.00%	18.00%	3.00%	0.86%	93.06%
14	2.4.2	Page Titled	100.00%	11.26%	88.74%	0.00%	0.75%	93.81%
15	3.1.1	Language of Page	100.00%	91.81%	8.19%	0.00%	0.74%	94.54%
-	-	Rest of WCAG 2.1 A/AA					5.46%	100.00%
		TOTAL (overall % coverage by issue volume)	80.39	57.38%	23.01%	19.61%		