Ensuring Your Critical User Flows Remain Accessible

Wilco Fiers

Deque Systems
Axe-core & Attest HTML product owner

AX Remediation Using E2E Testing

- Build in vs bolted on AX
- E2E Testing and AX
- Remediation Action Plan
- Walking or crawling apps
- Basic plumbing & water management strategies

Web, but more…

- Web based solutions
  - Cucumber
  - CypressJS
  - AGet
- Mobile
- Desktop

AX: Build In Or Bolted ON

Consider AX Early

- Designs account for AX
- AX Included in tests
- Chose accessible suppliers
- Budget for training and audit costs

The Remediation Problem

- Costly bugs (30x increase*)
- Conflicts with design guides
- Tied to unaccessible software
- Locked into contract with unsupportive suppliers

* ftp://ftp.software.ibm.com/software/rational/info/do-more/RAW14109USEN.pdf

Most of the web is inaccessible

and rebuilding it is not an option

Challenges Of Retrofitting AX

- Lack of training
- No AX testing
- Overwhelming scope
- Limited budget

Mopping With An Open Tap

When you resolve a problem, make sure it stays solved

Ensuring Critical User Flows

Make this text bigger

Effective AX Retrofitting

- Prioritised to user’s needs
- Fits with development workflow
- Prevent regressions
- Educate as you fix

Remediation Action Plan

1. Identify a user flow
2. Code and test the user flow
3. Resolve any found issues
4. Enable AX user flow testing on CI

Required E2E Testing

End To End (E2E) Testing

What Is End To End (E2E) Testing

- Complete system
  - Full UI, including real data
- Feature driven
- Simulates real user interaction

E2E Tools

- Cucumber
- Protractor
- Cypress
- Nightwatch
- WorldSpace Attest - AGet
- …

Cypress

```
describe('deque.com', () => {
 it('should be accessible', () => {
   const win = await cy.visit('http://deque.com/');
   const { violations } = await axe.run(win.document);

   expect(violations).to.be.empty;
 });
});
```

Cucumber

```
Feature: Example using default Capybara setup
 Default driver is :rack_test, uses :selenium by default
 for any test tagged with @javascript.

 Background:
   Given I visit "http://deque.com/"

 Scenario: Audit whole page
   Then the page should be audited for accessibility
```

AGet CLI

```
"name": "Deque contact example",
"pageList": [{
  "url": "http://deque.com/contact/",
  "actions": [
    "analyze the page",
    "click element '#submit'",
    "analyze only element '#contactForm'"
  ]
```

Choosing An E2E Framework

E2E testing in code

(Cypress, Protractor)

- Easier to use for devs
- Reuse existing test code
- Greater versatility

E2E testing in plain language

(AGet, Cucumber)

- Actions can be record
- Works (almost) everywhere
- Readable scripts for QA & SME

Remediation Action Plan

Remediation without regression

Step 1. Identify a user flow

- Business-critical pages

  login page, shopping cart, etc.
- Upcoming features
- Analytics for existing features
- User feedback & AX feedback

Step 2. Code And Test The User Flow

1. Define the user flow in plain language
2. Encode the user flow
    - Record the user flow (AGet)
    - Encode the user flow (other)
3. Add E2E tests, (disabled on CI)
4. SME test the user flow
5. Create tickets for any issues found

Step 3. Resolve Any Found Issues

- Tips for creating tickets:
- Group tickets by user flow (epics, projects, milestones)
- Explain for who this is a problem
- Remediation suggestion
- AX background info

Such as Deque University help pages & courses

Step 4. Enable AX user flow testing on CI

Enable CI testing with every fix:

- Analyze only certain pages
- … only elements A, B, C
- … using rule X, Y, Z
- … (or) using ruleset WCAG2A

Benefits

Iterative & Agile

- Fits in familiar workflow
- Integrates with existing tools
- Learn as you go
- Rapid incremental improvements

Coffee-Break Feedback Cycle

Test Method	Duration	Dev Response
Automated testing	1 - 10 min	Immediate fix
Dev team SME test	1 - 3 hours	Multi-tasking
Crawler testing	1 - 5 days	Backlog, same sprint
SME test	1 - 3 weeks	Backlog, future sprint

Walk Your Site, Don't Crawl

Scripted actions get you,

- Interaction with forms, widgets, SPAs, etc
- Control which elements are tested
- Control per element which test runs

Limitations of E2E Testing

- Finds no more than 50% of AX issues
- Only tests encoded scenarios
- Selectors can be fragile

Unsuited for monitoring or compliance

In Conclusion

Close The Tap

- Avoid regression with E2E
- Identify key users flows
- Solve problems iteratively
- Walk, don’t crawl your site

Connect with us

- @dequesystems
- dequelabs
- deque-system-inc
- dequesystemsinc