Engineer

axe

🕷 рушій доступності, яким перевіряють цей сайт

Що це таке

axe, що виходить як бібліотека JavaScript із відкритим кодом під назвою axe-core, — це набір автоматичних правил доступності, який працює всередині вебсторінки, перевіряє те, що відмалював браузер, і повідомляє про кожен елемент, що порушує одне з цих правил. Підтримує його компанія Deque Systems, яка займається доступністю, за ліцензією Mozilla Public License 2.0. На ньому будують інші інструменти: категорія доступності в Lighthouse від Chrome зважує свої аудити за оцінками впливу з axe, а власний посібник Playwright із тестування доступності запускає axe через невеликий супутній пакет.

Рушій — це один скрипт. Тестовий раннер або розширення браузера завантажує його в сторінку, яку перевіряють, і викликає `axe.run`, а рушій обходить документ, зокрема вкладені фрейми й shadow DOM, і застосовує кожне правило, умови якого виконано. Оскільки він читає сторінку вже після того, як браузер розташував її елементи, він може перевірити те, чого не видно з самого коду: чи текст видимий, на якому кольорі він стоїть, чи прихований елемент. З тієї ж причини деяким правилам потрібен справжній браузер. Проєкт прямо пише, що його правило контрасту кольорів не працює в JSDOM, імітованому документі, яким користується багато налаштувань для модульних тестів.

Кожне правило має теги. Один із них каже, до якої версії й рівня WCAG належить правило, наприклад `wcag2a`, `wcag2aa`, `wcag21aa` чи `wcag22aa`, або позначає його як `best-practice`, коли жоден критерій успіху його не вимагає. Інші теги пов'язують правила з EN 301 549, Section 508 або правилами W3C для тестування відповідності. Саме тегами команда визначає, що перевіряє прогін.

Повертається не одне «пройдено» чи «не пройдено», а чотири списки:

- `violations` : елементи, що порушили правило, кожен з оцінкою впливу (minor, moderate, serious або critical), коротким описом виправлення й посиланням на докладніше пояснення.
- `passes` : елементи, що пройшли перевірку; саме з них команда бачить, що правило справді спрацювало.
- `incomplete` , які документація також називає «needs review»: елементи, які рушій не може з певністю ні зарахувати, ні відхилити, тож на них має подивитися людина.
- `inapplicable` : правила, яким не було чого перевіряти, як-от правила для відео на сторінці без відео.

Застосуйте поняття до рішення

Застосуйте «Що таке axe?» до конкретного рішення. Ми можемо пов'язати поняття із системою, яку ви створюєте або купуєте. Інтерфейси, що швидко завантажуються, працюють на телефоні й поводяться однаково в усіх браузерах ваших клієнтів.

Переглянути наші послуги Зв'язатися з нами →

Перевірте припущення за терміном

Використайте пояснення та пов'язані роботи, щоб сформулювати запитання для проєкту. Ми можемо перевірити припущення разом до вибору інструмента. Керувала full-stack-розробкою GIS-карт — full-stack-картографія — від бази даних до браузера

Керувала full-stack-розробкою GIS-карт Зв'язатися з нами →

Чому це важливо

Рішення, якому служить axe, стосується того, які частини доступності команда передає машині і чи має збій зупиняти випуск. Дві властивості роблять рушій придатним для цього. Він дешевий у запуску, тож може перевіряти кожну збірку, а не один аудит на рік. І заявлена мета його авторів — нуль хибних спрацювань, якщо не рахувати помилок у самому рушії: усе, в чому рушій не впевнений, іде до списку на перегляд, а не до списку порушень. Перевірку, яка даремно кричить «вовк», вимикають; ту, що повідомляє лише доведене, можна зробити підставою для падіння збірки, і ніхто на неї не ображатиметься.

Обмеження полягає в охопленні. Deque пише, що axe-core у середньому автоматично знаходить 57% проблем WCAG, а непевні результати передає на ручний розгляд. Це число описує виявлені проблеми, а не частку критеріїв успіху, які машина може оцінити.

Поза досяжністю лишається передусім судження. axe може підтвердити, що зображення має альтернативний текст, кнопка має ім’я, а поле форми — мітку. Він не може сказати, чи текст описує зображення, чи ім’я каже, що робить кнопка, і чи сторінка має сенс, коли її читають уголос по порядку. Для цих питань потрібна людина з екранним читачем і клавіатурою, а час, який машина заощаджує на механічних перевірках, ця людина може витратити на них.

Вибір тегів — окремий компроміс. Прогін лише правил із тегами WCAG дає результат, який чітко лягає на заяву про відповідність, і ні на що інше. Додавши правила best-practice, команда ловить проблеми, яких WCAG не називає, але з якими користувачі все одно стикаються, як-от вміст поза всіма орієнтирами чи сторінка без заголовка верхнього рівня. Ціна — кілька збоїв, на які аудит відповідності ніколи б не послався.

Як це виглядає на практиці

Дві сторінки цього сайту показують, що саме довіряють рушієві.

У тестовому наборі однієї платформи він стоїть поряд із чотирма рівнями тестів, і гейт на axe-core вимагає нуля порушень WCAG 2.0 і 2.1 рівнів A та AA, тож доступність стає падінням збірки, а не знахідкою аудиту через кілька місяців. Суть у часі: дефект, через який падає збірка, виправляють одразу, поки щойно написана зміна ще відкрита.

На цьому сайті сторінка подяк описує, як рушій перевіряє добірку репрезентативних сторінок, по одній на кожен шаблон кожною мовою, на відповідність WCAG 2.2 рівнів A та AA в обох колірних темах, і робить це в межах обов’язкових перевірок, а не окремим звітом осторонь. Прогін в обох темах означає, що збій контрасту, який з’являється лише в темній темі, однаково може зупинити збірку. Вибірка по одній сторінці на шаблон спирається на те, що сторінки з одного шаблону мають однакову структуру, і саме це в жовтні 2026 року дало змогу 38 сторінкам представляти сайт на 2 268 сторінок. Автоматичні правила не охоплюють усіх оцінок, яких потребує WCAG, тому сайт веде свій звіт окремо за кожним критерієм успіху, а не як єдину позначку «пройдено».

Де виникають проблеми

Більшість збоїв з axe — це збої довіри: прогін, що перевірив менше, ніж здавалося, або результат, який прочитали як більше, ніж він каже.

Список тегів пропускає правила. Теги не накопичуються. `wcag2aa` містить лише правила для критеріїв рівня AA з WCAG 2.0, тож прогін, обмежений ним, пропускає всі правила рівня A, і документація API прямо каже, що потрібні обидва теги, `wcag2a` і `wcag2aa`. Так само `wcag21aa` містить лише правила для критеріїв, які додала версія 2.1, а прогін, обмежений тегами WCAG, ніколи не виконує правил `best-practice`. В описах правил правила WCAG 2.2 позначено як вимкнені за замовчуванням, тож прогін без тегів їх пропускає; якщо назвати їхній тег, вони вмикаються. Називайте кожен тег, який має покрити прогін, і тримайте список поруч із твердженням, яке він підтримує:

```
const results = await axe.run(document, {
    runOnly: {
        type: "tag",
        values: ["wcag2a", "wcag2aa", "wcag21a", "wcag21aa", "wcag22aa"],
    },
});
```

Правило, якого немає в жодному з чотирьох списків, не запускалося взагалі, тож перечитуйте списки щоразу, коли змінюється конфігурація.

Чистий прогін сприймають як доступну сторінку. Нуль порушень означає, що правила, які запускалися, не довели жодного збою. Це нічого не каже про те, чи правильний альтернативний текст, чи фокус переміщується в розумному порядку і чи людина з екранним читачем зможе виконати завдання. Сприймайте прогін як мінімум і плануйте перевірки, яких він зробити не може.

Список на перегляд викидають. Гейт, що падає лише на порушеннях, пропускає результати `incomplete`, і їх ніхто не бачить. Це елементи, які рушій розглянув і залишив без рішення, а отже саме вони найбільше потребують людини. Передавайте їх людині, яка їх прочитає, або нехай збірка падає й на них.

Сторінку перевіряють в одному стані. Рушій перевіряє те, що відмальовано в момент запуску. Закрите меню, діалог, який ще ніхто не відкривав, повідомлення про помилку, що з'являється лише після невдалого надсилання, та інша колірна тема — усе це лишається поза прогоном сторінки в тому вигляді, в якому вона щойно завантажилася. Доводьте сторінку до кожного стану, що має значення, і запускайте рушій знову в кожному з них.

Рушій працює там, де нічого не малюється. Імітований документ не має ні геометрії елементів, ні обчислених кольорів, і проєкт називає контраст правилом, яке там не працює. Модульні тести з axe однаково ловлять структурні проблеми; контраст і все інше візуальне належить справжньому браузеру.

Правило вимикають, щоб випустити реліз. Вимкнення правила для всього прогону через один компонент приховує його збої всюди, зокрема на сторінках, які напишуть пізніше. Натомість виключайте з перевірки один елемент і залишайте примітку, чому й коли цей виняток слід прибрати.

Що почитати далі

- axe-core — вихідний код рушія, його заявлені принципи й список підтримуваних браузерів.
- Rule Descriptions — кожне правило з його впливом і тегами, згруповані за версією WCAG, best practice, AAA, експериментальними й застарілими правилами.
- axe-core API — теги, параметр `runOnly`, як включити чи виключити частини сторінки та будова чотирьох списків результатів.
- Accessibility testing — посібник Playwright про запуск axe у браузерних тестах, зокрема про перевірку сторінки після взаємодії з нею.

Словник

Обирайте за фактичними умовами

Опишіть робочий процес, наявні інтерфейси та користувачів, яким потрібна зміна. Ми можемо визначити перевірюваний результат і спосіб передачі, який дасть вашій команді змогу підтримувати його.

Дивитися, скільки це коштує Зв'язатися з нами →

Зробіть термінологію практичною

Запитайте, як «Що таке axe?» стосується вашої організації. Конкретного прикладу з поточної системи достатньо для початку розмови.

Переглянути наші послуги

welcome@engineer.company +45 50 10 61 84 Копенгаген, Данія

Візьміть цю сторінку із собою

Візитівка

Дбайливо зроблено з любов'ю — D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Авторське право © 2025 – дотепер · Engineer ApS · Усі права захищено.

Надруковано з https://engineer.company/uk/glossary/axe/