MDN's new design is in Beta! A sneak peek: https://blog.mozilla.org/opendesign/mdns-new-design-beta/

Learn web development

Cross browser testing

This module focuses on testing web projects across different browsers. We look at identifying your target audience (e.g. what users, browsers, and devices do you most need to worry about?), how to go about doing testing, the main issues that you'll face with different types of code and how to mitigate them, what tools are most useful in helping you test and fix problems, and how to use automation to speed up testing.

Level-up your cross-browser compat skills with the Mozilla and Sauce Labs Cross-browser Testing Tutorial.

Test on Sauce Labs   ❯

Prerequisites

You should really learn the basics of the core HTML, CSS, and JavaScript languages first before attempting to use the tools detailed here.

Guides

Introduction to cross browser testing

This article starts the module off by providing an overview of the topic of cross browser testing, answering questions such as "what is cross browser testing?", "what are the most common types of

problems you'll encounter?", and "what are the main approaches for testing, identifying, and fixing problems?"

Strategies for carrying out testing

Next, we drill down into carrying out testing, looking at identifying a target audience (e.g. what browsers, devices, and other segments should you make sure are tested), low fi testing strategies (get yourself a range of devices and some virtual machines and do adhoc tests when needed), higher tech strategies (automation, using dedicated testing apps), and testing with user groups.

Handling common HTML and CSS problems

With the scene set, we'll now look specifically at the common cross browser problems you will come across in HTML and CSS code, and what tools can be used to prevent problems from happening, or fix problems that occur. This includes linting code, handing CSS prefixes, using browser dev tools to track down problems, using polyfills to add support into browsers, tackling responsive design problems, and more.

Handling common JavaScript problems

Now we'll look at common cross browser JavaScript problems and how to fix them. This includes information on using browser dev tools to track down and fix problems, using polyfills and libraries to work around problems, getting modern JavaScript features working in older browsers, and more.

Handling common accessibility problems

Next we turn our attention to accessibility, providing information on common problems, how to do simple testing, and how to make use of auditing/automation tools for finding accessibility issues.

Implementing feature detection

Feature detection involves working out whether a browser supports a certain block of code, and running different code dependent on whether it does (or doesn't), so that the browser can always provide a working experience rather crashing/erroring in some browsers. This article details how to write your own simple feature detection, how to use a library to speed up implementation, and native features for feature detection such as `@supports`.

Introduction to automated testing

Manually running tests on several browsers and devices, several times per day, can get tedious and time consuming. To handle this efficiently, you should become familiar with automation tools. In this article we look at what is available, how to use task runners, and the basics of how to use commercial browser test automation apps such as Sauce Labs and Browser Stack.

Setting up your own test automation environment

In this article, we will teach you how to install your own automation environment and run your own tests using Selenium/WebDriver and a testing library such as selenium-webdriver for Node. We will also look at how to integrate your local testing environment with commercial apps like the ones discussed in the previous article.

Was this article helpful?

Learn the best of web development

✖

Get the latest and greatest from MDN delivered straight to your inbox.

you@example.com

SIGN UP NOW