

Chrome DevTools for Testers

The parts of DevTools a tester uses every day, and what each one helps you find.

Shortcuts

KEYS	DOES
F12	Open or close DevTools
Ctrl+Shift+I	Same (Mac: Cmd+Option+I)
Ctrl+Shift+C	Pick an element on the page
Ctrl+Shift+J	Open straight to the Console
Ctrl+Shift+M	Phone and tablet view
Ctrl+Shift+P	Command Menu: type any action
Ctrl+F	Search inside the open panel
Ctrl+Shift+R	Hard reload (DevTools open)

On a Mac use Cmd instead of Ctrl, and Cmd+Option instead of Ctrl+Shift for I, J and C.

Console errors you cannot see on the page

Red lines are JavaScript errors. Copy the full text and the file name into your bug report.

Tick **Preserve log** here too, so errors are not lost on reload.

Filter by level: Errors, Warnings, Info.

Test a locator: `$$("css")` or `$x("//xpath")`

Elements the page's HTML and CSS

Right-click anything → **Inspect** to jump to its HTML.

Double-click text to change it. Try a very long name or price to see if the layout breaks.

Styles pane → **:hov** to force hover, focus or active states.

Ctrl+F in Elements finds by text, CSS selector or XPath.

Network the most useful tab for testers

Status column: spot red 4xx and 5xx rows. Filter with **Fetch/XHR** to see only API calls.

Click a request: Headers, Payload (what was sent), Response (what came back), Timing.

Preserve log: keeps requests when the page reloads or redirects, e.g. after login.

Disable cache: loads everything fresh, like a first-time visitor.

Throttling: pick Slow 4G, 3G or Offline to test a weak network.

Right-click a request → Copy → Copy as cURL. Paste it in the bug report or import it into Postman.

Export HAR (sanitized): saves every request to a file developers can open. The sanitized one leaves out cookies and tokens.

Ctrl+F in Network searches inside every request and response.

Read the Timing tab why is this call slow?

Waiting for server response is long: the server or database is slow. Report the endpoint.

Content Download is long: the response is too big. Look at its size.

Queueing or **Stalled** is long: the page fires too many requests at once.

A 5-minute DevTools check for any new page

- 1 Console open, page reloaded. Any red errors?
- 2 Network, filter **Fetch/XHR**. Any 4xx or 5xx calls?
- 3 Disable cache, set **Slow 4G**, hard reload. Does it still work?
- 4 Device mode at **360 px**. Anything cut off or scrolling sideways?
- 5 Delete the session cookie and refresh. Sent to login cleanly?
- 6 Lighthouse, Accessibility only. Note the issues it lists.

Chrome DevTools for Testers (continued)

Device mode Ctrl+Shift+M

Pick a phone or type any width, e.g. 360, 768, 1280.

Rotate, zoom and switch on touch to test tap targets.

Good for layout bugs. It is still a desktop browser, so check real phones before sign-off.

Application storage and login state

Cookies, Local storage and **Session storage**: see what the site saves.

Delete the session cookie and refresh: you should be sent to login, with no crash.

Storage → **Clear site data** gives you a clean first visit.

Other browsers and real phones

Edge has the same DevTools as Chrome. **Firefox** opens its own with F12.

Safari on Mac: Settings → Advanced → tick *Show features for web developers*. A Develop menu appears.

iPhone: turn on Settings → Safari → Advanced → Web Inspector, connect to a Mac, then use Safari's Develop menu.

Android: turn on USB debugging, connect, open `chrome://inspect` on your computer.

Command Menu actions Ctrl+Shift+P, then type

Capture full size screenshot: the whole page as one image.

Capture node screenshot: just the selected element.

Emulate CSS prefers-color-scheme: dark: test dark mode.

Show Rendering → **Emulate vision deficiencies**: see the page as a colour-blind user.

Show Sensors: fake a location or time zone.

Show Recorder: record a user flow and replay it.

Request conditions block or slow one URL

Command Menu → **Show Request conditions**, or right-click a request → block it.

Block one API or image and see how the page copes. Does it show an error, or break?

Lighthouse a quick audit

Scores Performance, Accessibility, Best practices and SEO.

Run it in a private window, on Mobile, and compare across builds.

Treat it as a starting point. A good score does not mean no bugs.

Things you will find, and what to report

WHAT YOU SEE	WHAT IT USUALLY MEANS	PUT IN THE BUG
Red error: Cannot read properties of undefined	A front-end code bug	Full error text, the file and line, your steps
404 on a .js, .css or image file	A file missing after a deploy	The file URL and the page it broke
CORS error in the Console	The API does not allow calls from this site	Request URL and the site you were on
Mixed content warning	An http:// file on an https:// page	The http:// URL
401 after some minutes on the page	Token expiry is not handled	How long after login it happened
The same API called 5+ times	Wasted calls, often a loop	Count and a Network screenshot
Long Waiting for server response	A slow API or database query	Endpoint, time taken, test data used

What to attach to a bug report

- ☐ Screenshot or screen recording
 ☐ Console error text, copied in full
 ☐ The failing request: URL, status, response
- ☐ Copy as cURL, or a sanitized HAR file
 ☐ Browser version (`chrome://version`)
 ☐ Window size or device you used