How to file a good browser bug

- Verify it is a bug
  - What's the correct behavior?
  - Does it work in a different browser?
  - Did it break in a recent release?
  - Are others seeing the same problem?
- Has it been reported before?
- Report the bug
  - Create a minimized test case {: #minified-test-case }
    - Hosting a minified test case
- Filing your issue
  - Provide a clear description and the steps required to reproduce the issue
    - Bonus: Add a screenshot or screencast of the issue
  - Include details of the environment
  - Submit the bug

Filing a good bug isn't hard, but takes a little work. The goal is to make it easy to find what's broken, reach the root cause and, most importantly, find a way to fix it. Bugs that make fast progress tend to be easy to reproduce with a clear expected behavior.

Verify it is a bug

The first step is to figure out what the "correct" behavior should be.

What's the correct behavior?

Check the relevant API docs on MDN, or try to find related specs. This information can help you decide which API is actually broken, where it's broken, and what the expected behavior is.

Does it work in a different browser?

Behavior that differs between browsers is generally prioritized higher as an interoperability issue, especially when the browser containing the bug is the odd one out. Try to test on the latest versions of Chrome, Firefox, Safari and Edge, possibly using a tool like BrowserStack.

If possible, check that the page isn't intentionally behaving differently due to user agent sniffing. In Chrome DevTools, try setting the `User-Agent` string to another browser.

Did it break in a recent release?

Did this work as expected in the past, but broke in a recent browser release? Such "regressions" can be acted upon much quicker, especially if you supply a version number where it worked and a version where it failed. Tools like BrowserStack can make it easy to check old browser versions and the bisect-builds tool (for Chromium) allows searching for the change very efficiently.

If an issue is a regression and can be reproduced, the root cause can usually be found and fixed quickly.

Are others seeing the same problem?

If you're experiencing problems, there's a good chance other developers are too. First, try searching for the bug on Stack Overflow. This might help you translate an abstract problem into a specific broken API, and it might help you find a short term workaround until the bug is fixed.

Has it been reported before?

Once you have an idea of what the bug is, it's time to check to see if the bug has already been reported by searching the browser bug database.

- Chromium-based browsers: https://crbug.com
- Firefox: https://bugzilla.mozilla.org/
- Safari & WebKit-based browsers: https://bugs.webkit.org/

If you find an existing bug that describes the problem, add your support by starring, favoriting, or commenting on the bug. And, on many sites, you can add yourself to the CC list and get updates when the bug changes.

If you decide to comment on the bug, include information about how the bug affects your website. Avoid adding "+1" style comments, as bug trackers typically send emails for every comment.

Report the bug

If the bug hasn't been reported before, it's time to tell the browser vendor about it.

Create a minimized test case {: #minified-test-case }

Mozilla has a great article on how to create a minimized test case. To make a long story short, while a description of the problem is a great start, nothing beats providing a linked demo in the bug that shows the problem. To maximize the chance of fast progress the example should contain the minimum possible code needed to demonstrate the problem. A minimal code sample is the number one thing you can do to increase the odds of your bug getting fixed.

Here are a few tips for minimizing a test case:

- Download the web page, add `<base href="https://original.url">` and verify that the bug exists locally. This may require a live HTTPS server if the URL uses HTTPS.
- Test the local files on the latest builds of as many browsers as you can.
- Try to condense everything into 1 file.
- Remove code (starting with things you know to be unnecessary) until the bug goes away.
- Use version control so that you can save your work and undo things that go wrong.

Hosting a minified test case

If you're looking for a good place to host your minified test case, there are several good places available:

- Glitch
- JSBin
- JSFiddle
- CodePen

Be aware that several of those sites display content in an iframe, which may cause features or bugs to behave differently.

Filing your issue

Once you've got your minimized test case, you're ready to file that bug. Head over to the right bug tracking site, and create a new issue.

- Chromium-based browsers - https://crbug.com/new
- Firefox - https://bugzilla.mozilla.org/
- Safari & WebKit-based browsers - https://bugs.webkit.org/

Provide a clear description and the steps required to reproduce the issue

First, provide a clear description to help engineers quickly understand what the problem is and help to triage the issue.

```
When installing a PWA using the `beforeinstallprompt.prompt()`, the
`appinstalled` event fires before the call to `prompt()` resolves.
```

Next, provide the detailed steps required to reproduce the issue. This is where your minified test case comes in.

```
What steps will reproduce the problem?
1. Go to https://basic-pwa.glitch.me/, open DevTools and look at the
   console tab.
2. Click the Install button in the page, you might need to interact with
   the page a bit before it becomes enabled.
3. Click Install on the browser modal install confirmation.
```

And finally, describe the *expected*, and *actual* result.

```
What is the expected result? In the console:
0. INSTALL: Available (logged when `beforeinstallprompt` event fired)
1. INSTALL_PROMPT_RESPONSE: {outcome: "accepted", platform: "web"}
   (logged when beforeinstallprompt.prompt()` resolves)
```

```
2. INSTALL: Success (logged when `appinstalled` event fired)

What is the actual result? In the console:
0. INSTALL: Available (logged when `beforeinstallprompt` event fired)
1. INSTALL: Success (logged when `appinstalled` event fired)
2. INSTALL_PROMPT_RESPONSE: {outcome: "accepted", platform: "web"}
   (logged when beforeinstallprompt.prompt()` resolves)
```

For more information, check out Bug report writing guidelines on MDN.

Bonus: Add a screenshot or screencast of the issue

Though not required, in some cases, it can be helpful to add a screenshot, or screencast of the issue. This is especially helpful in cases where bugs may require some odd steps to reproduce. Being able to see what happens in a screencast, or on a screenshot can frequently be helpful.

Include details of the environment

Some bugs are reproducible only on certain operating systems, or only on specific kinds of displays (for example, low-dpi or high-dpi). Be sure to include the details of any test environments you used.

Submit the bug

Finally, submit the bug. Then, remember to keep an eye on your email for any responses to the bug. Typically during investigation and when fixing the bug, engineers may have additional questions, or if they have difficulty reproducing the issue, they may reach out.

□□□ #2
□ □□□ □□□ 2026-10-31 21:56:42 UTC
□ □□□ □□□ 2026-10-31 21:56:44 UTC