Contributing to jQuery

Note: This is the code development repository for *jQuery Core* only. Before opening an issue or making a pull request, be sure you're in the right place.

- jQuery plugin issues should be reported to the author of the plugin.
- jQuery Core API documentation issues can be filed at the API repo.
- Bugs or suggestions for other jQuery organization projects should be filed in their respective repos.

Getting Involved

API design principles

We're always looking for help identifying bugs, writing and reducing test cases, and improving documentation. And although new features are rare, anything passing our guidelines will be considered.

More information on how to contribute to this and other jQuery organization projects is at contribute.jquery.org, including a short guide with tips, tricks, and ideas on getting started with open source. Please review our commit & pull request guide and style guides for instructions on how to maintain a fork and submit patches.

When opening a pull request, you'll be asked to sign our Contributor License Agreement. Both the Corporate and Individual agreements can be previewed on GitHub.

Questions and Discussion

Forum and IRC

jQuery is so popular that many developers have knowledge of its capabilities and limitations. Most questions about using jQuery can be answered on popular forums such as Stack Overflow. Please start there when you have questions, even if you think you've found a bug.

The jQuery Core team watches the jQuery Development Forum. If you have longer posts or questions that can't be answered in places such as Stack Overflow, please feel free to post them there. If you think you've found a bug, please file it in the bug tracker. The Core team can be found in the #jquery-dev IRC channel on irc.freenode.net.

Weekly Status Meetings

The jQuery Core team has a weekly meeting to discuss the progress of current work. The meeting is held in the #jquery-meeting IRC channel on irc.freenode.net at Noon EST on Mondays.

jQuery Core Meeting Notes

How to Report Bugs

Make sure it is a jQuery bug

Most bugs reported to our bug tracker are actually bugs in user code, not in jQuery code. Keep in mind that just because your code throws an error inside of jQuery, this does *not* mean the bug is a jQuery bug.

Ask for help first in the Using jQuery Forum or another discussion forum like Stack Overflow. You will get much quicker support, and you will help avoid tying up the jQuery team with invalid bug reports.

Disable browser extensions

Make sure you have reproduced the bug with all browser extensions and add-ons disabled, as these can sometimes cause things to break in interesting and unpredictable ways. Try using incognito, stealth or anonymous browsing modes.

Try the latest version of jQuery

Bugs in old versions of jQuery may have already been fixed. In order to avoid reporting known issues, make sure you are always testing against the latest build. We cannot fix bugs in older released files, if a bug has been fixed in a subsequent version of jQuery the site should upgrade.

Simplify the test case

When experiencing a problem, reduce your code to the bare minimum required to reproduce the issue. This makes it *much* easier to isolate and fix the offending code. Bugs reported without reduced test cases take on average 9001% longer to fix than bugs that are submitted with them, so you really should try to do this if at all possible.

Search for related or duplicate issues

Go to the jQuery Core issue tracker and make sure the problem hasn't already been reported. If not, create a new issue there and include your test case.

Tips For Bug Patching

We *love* when people contribute back to the project by patching the bugs they find. Since jQuery is used by so many people, we are cautious about the patches we accept and want to be sure they don't have a negative impact on the millions of people using jQuery each day. For that reason it can take a while for any suggested patch to work its way through the review and release process. The reward for you is knowing that the problem you fixed will improve things for millions of sites and billions of visits per day.

Build a Local Copy of jQuery

Create a fork of the jQuery repo on github at https://github.com/jquery/jquery

Change directory to your web root directory, whatever that might be:

```
$ cd /path/to/your/www/root/
```

Clone your jQuery fork to work locally

```
$ git clone git@github.com:username/jquery.git
```

Change directory to the newly created dir jquery/

```
$ cd jquery
```

Add the jQuery main as a remote. I label mine "upstream"

```
$ git remote add upstream git://github.com/jquery/jquery.git
```

Get in the habit of pulling in the "upstream" main to stay up to date as jQuery receives new commits

```
$ git pull upstream main
```

Run the build script

```
$ npm run build
```

Now open the jQuery test suite in a browser at http://localhost/test. If there is a port, be sure to include it.

Success! You just built and tested jQuery!

Test Suite Tips...

During the process of writing your patch, you will run the test suite MANY times. You can speed up the process by narrowing the running test suite down to the module you are testing by either double clicking the title of the test or appending it to the url. The following examples assume you're working on a local repo, hosted on your localhost server.

Example:

http://localhost/test/?module=css

This will only run the "css" module tests. This will significantly speed up your development and debugging.

ALWAYS RUN THE FULL SUITE BEFORE COMMITTING AND PUSHING A PATCH!

Loading changes on the test page

Rather than rebuilding jQuery with `grunt` every time you make a change, you can use the included `grunt watch` task to rebuild distribution files whenever a file is saved.

```
$ grunt watch
```

Alternatively, you can **load tests as ECMAScript modules** to avoid the need for rebuilding altogether.

Click "Load as modules" after loading the test page.

Repo organization

The jQuery source is organized with ECMAScript modules and then compiled into one file at build time.

jQuery also contains some special modules we call "var modules", which are placed in folders named "var". At build time, these small modules are compiled to simple var statements. This makes it easy for us to share variables across modules. Browse the "src" folder for examples.

Browser support

Remember that jQuery supports multiple browsers and their versions; any contributed code must work in all of them. You can refer to the browser support page for the current list of supported browsers.