Contributing to the Spring Framework

First off, thank you for taking the time to contribute! :+1: :tada:

Table of Contents

Code of Conduct

This project is governed by the Spring Code of Conduct. By participating you are expected to uphold this code. Please report unacceptable behavior to spring-code-of-conduct@pivotal.io.

How to Contribute

Ask questions

If you have a question, check Stack Overflow using this list of tags. Find an existing discussion, or start a new one if necessary.

If you believe there is an issue, search through existing issues trying a few different ways to find discussions, past or current, that are related to the issue. Reading those discussions helps you to learn about the issue, and helps us to make a decision.

Create an Issue

Reporting an issue or making a feature request is a great way to contribute. Your feedback and the conversations that result from it provide a continuous flow of ideas. However, before creating a ticket, please take the time to ask and research first.

If you create an issue after a discussion on Stack Overflow, please provide a description in the issue instead of simply referring to Stack Overflow. The issue tracker is an important place of record for design discussions and should be self-sufficient.

Once you're ready, create an issue on GitHub.

Many issues are caused by subtle behavior, typos, and unintended configuration. Creating a Minimal Reproducible Example (starting with https://start.spring.io for example) of the problem helps the team quickly triage your issue and get to the core of the problem.

Issue Lifecycle

When an issue is first created, it is flagged `waiting-for-triage` waiting for a team member to triage it. Once the issue has been reviewed, the team may ask for further information if needed, and based on the findings, the issue is either assigned a target milestone or is closed with a specific status.

When a fix is ready, the issue is closed and may still be re-opened until the fix is released. After that the issue will typically no longer be reopened. In rare cases if the issue was not at all fixed, the issue may be re-opened. In most cases however any follow-up reports will need to be created as new issues with a fresh description.

Submit a Pull Request

1. If you have not previously done so, please sign the Contributor License Agreement. You will be reminded automatically when you submit the PR.
2. Should you create an issue first? No, just create the pull request and use the description to provide context and motivation, as you would for an issue. If you want to start a discussion first or have already created an issue, once a pull request is created, we will close the issue as superseded by the pull request, and the discussion about the issue will continue under the pull request.
3. Always check out the `main` branch and submit pull requests against it (for target version see settings.gradle). Backports to prior versions will be considered on a case-by-case basis and reflected as the fix version in the issue tracker.
4. Choose the granularity of your commits consciously and squash commits that represent multiple edits or corrections of the same logical change. See Rewriting History section of Pro Git for an overview of streamlining the commit history.
5. Format commit messages using 55 characters for the subject line, 72 characters per line for the description, followed by the issue fixed, e.g. `Closes gh-22276` . See the Commit Guidelines section of Pro Git for best practices around commit messages, and use `git log` to see some examples.
6. If there is a prior issue, reference the GitHub issue number in the description of the pull request.

If accepted, your contribution may be heavily modified as needed prior to merging. You will likely retain author attribution for your Git commits granted that the bulk of your changes remain intact. You may also be asked to rework the submission.

If asked to make corrections, simply push the changes against the same branch, and your pull request will be updated. In other words, you do not need to create a new pull request when asked to make changes.

Participate in Reviews

Helping to review pull requests is another great way to contribute. Your feedback can help to shape the implementation of new features. When reviewing pull requests, however, please refrain from approving or rejecting a PR unless you are a core committer for the Spring Framework.

Build from Source

See the Build from Source wiki page for instructions on how to check out, build, and import the Spring Framework source code into your IDE.

Source Code Style

The wiki pages Code Style and IntelliJ IDEA Editor Settings define the source file coding standards we use along with some IDEA editor settings we customize.

Reference Docs

The reference documentation is in the src/docs/asciidoc directory, in Asciidoctor format. For trivial changes, you may be able to browse, edit source files, and submit directly from GitHub.

When making changes locally, execute `./gradlew asciidoctor` and then browse the result under `build/docs/ref-docs/html5/index.html`.

Asciidoctor also supports live editing. For more details see AsciiDoc Tooling.