

# (DRAFT) Code Reviews

- [Code Review Policies](#)
  - [Overview](#)
  - [Details](#)
    - [1. The Reviewer role for the OpenJFX Project](#)
    - [2. Code review policies](#)
      - [A. Low-impact bug fixes.](#)
      - [B. Higher impact bug fixes or RFEs.](#)
      - [C. New features / API additions.](#)

KCR: Following are updated Code Review Policies reflecting the move of the official openjfx repo from HG at <https://hg.openjdk.java.net/openjfx/jfx-dev/rt> to GIT at <https://github.com/openjdk/jfx> using the Skara tooling. Once the switch is complete, this will be copied to the [Code Reviews](#) page.

## Code Review Policies

### Overview

All code must be reviewed before being pushed to the repository. The short version of the OpenJFX code review policy is:

1. We define a formal "Reviewer" role, similar to the JDK project.
2. The code review policies recognize the following different types of changes, with different thresholds for review:
  - a. Simple, low-impact fixes: 1 Reviewer
  - b. Higher-impact fixes: 2 Reviewers + allow sufficient time (1 day recommended) for others to chime in
  - c. Features / API changes: CSR for approving the change, including approval by a "lead"; implementation then needs 2 Reviewers for the code (as with other "higher-impact" fixes above)
3. A review must be submitted via a pull request against the <https://github.com/openjdk/jfx> repo.

### Details

Code reviews are important to maintain high-quality contributions, but we recognize that not every type of change needs the same level of review. Without lowering our standards of quality, we want to make it easier to get low-impact changes (simple bug fixes) accepted.

In support of this, the following review policies are in effect. Many of these will involve judgment calls, especially when it comes to deciding whether a fix is low impact vs. high-impact, and that's OK. It doesn't have to be perfect.

#### 1. The Reviewer role for the OpenJFX Project

We define a formal "Reviewer" role, similar to the JDK project. A [Reviewer](#) is responsible for reviewing code changes and helping to determine whether a change is suitable for including into OpenJFX. We expect Reviewers to feel responsible not just for their piece, but for the quality of the JavaFX library as a whole. In other words, the role of Reviewer is one of stewardship.

An experienced Committer can eventually become a Reviewer by providing good quality fixes and participating in code reviews over time, demonstrating the high-level of skill and understanding needed to be a competent reviewer. The JDK uses a threshold of 32 significant contributions. Without wanting to relax this standard too much, one thing we may consider is that a Committer with, say, 24 commits who regularly participates in reviews, offering good feedback, might be just as good a reviewer (or maybe even better) as someone with 32 commits who rarely, if ever, provides feedback on proposed bug fixes. This is meant to be a somewhat loose guideline. It is up to the Reviewers and the Project Leads to decide whether and when a new Committer is ready to become a Reviewer.

#### 2. Code review policies

All code reviews must be done via a pull request submitted against the <https://github.com/openjdk/jfx> repo -- even simple fixes. A JBS bug ID must exist before the pull request will be reviewed. See [CONTRIBUTING.md](#) for information on how to submit a pull request.

All fixes must be reviewed by at least one reviewer with the "Reviewer" role (aka a "R"eviewer). We have a different code review threshold for different types of changes. If there is disagreement as to whether a fix is low-impact or high-impact, then it is considered high-impact. In other words we will always err on the side of quality by "rounding up" to the next higher category. The contributor can say whether they think something is low-impact or high-impact, but it is up to a Reviewer to confirm this. This should be done and recorded in the same manner as the review comments.

Review comments can either be added directly to the GitHub pull request, or by replying to the auto-generated "RFR" (Request For Review) email thread. The Skara bot will cross-forward between them. To approve a pull request, a reviewer must do that in the PR itself. See the following [GitHub help page](#) for help on reviewing a pull request.

New code should be formatted consistently in accordance with the [Code Style Rules](#). However, please do not reformat code as part of a bug fix. The makes more changes for code reviewers to track and review, and can lead to merge conflicts. If you want to reformat a class, do that in a separate pull request (which will need its own unique JBS bug ID).

##### A. Low-impact bug fixes.

These are typically isolated bug fixes with little or no impact beyond fixing the bug in question; included in this category are test fixes (including new tests), doc fixes, and fixes to sample applications (including new samples).

One (1) "R"eviewer is sufficient to accept such changes. As a courtesy, and to avoid changes which later might need to be backed out, if you think there might be some concern or objection to the change, please give sufficient time for folks who might be in other time zones the chance to take a look. This should be left up to the judgment of the reviewer who approves it as well as the contributor.

## B. Higher impact bug fixes or RFEs.

These include changes to the implementation that potentially have a performance or behavioral impact, or are otherwise broad in scope. Some larger bug fixes will fall into this category, as will any fixes in high-risk areas (e.g., CSS).

Two (2) reviewers must approve to accept such changes, at least one of whom must be a "R"eviewer. Additionally, the review should allow sufficient time for folks who might be in other time zones the chance to review if they have concerns. A suggested wait time is 24 hours (not including weekends / or major holidays).

## C. New features / API additions.

This includes behavioral changes, additions to the FXML or CSS spec, etc.

Feature requests come with a responsibility beyond just saying "here is the code for this cool new feature, please take it". There are many factors to consider for even small features. Larger features will need a significant contribution in terms of API design, coding, testing, maintainability, etc.

A feature should be discussed up-front on the openjfx-dev mailing list to get early feedback on the concept (is it a feature we are likely to accept) and the direction the API and/or implementation should take.

To ensure that new features are consistent with the rest of the API and the desired direction of the Project, a [CSR](#) is required for a new Feature, API addition, or behavioral change. The CSR must be reviewed and approved by a "lead" of the Project. Currently this is either Kevin Rushforth or Johan Vos as indicated [above](#).

The review of the implementation follows the same "two reviewer" standard for higher-impact changes as described in category B. The two code reviewers for the implementation may or may not include the Lead who reviewed the CSR. The review / approval of the CSR is an independent step from the review / approval of the code change, although they can proceed in parallel.

### 3. Streamlined review process for changes developed on GitHub