Pushing a HotSpot change

Deprecated

This page is deprecated and it's content is either old or being moved to the OpenJDK Developers' Guide. Please update your links.

Before pushing

In order to push a change to the HotSpot source code you need to complete all the steps in the list below. This is done in order to ensure code quality and reduce the risk of introducing bugs into the code base.

1. You must be a Committer in the JDK project
2. You need a non-JEP JBS issue for tracking
3. Your change must have been available for review at least 24 hours to accommodate for all time zones
4. Your change must have been approved by two Committers out of which at least one is also a Reviewer
5. Your change must have passed through the hs tier 1 testing provided by the submit-hs repository with zero failures
6. You must run all relevant testing to make sure your actual change is working
7. You must be available the next few hours, and the next day and ready to follow up with any fix needed in case your change causes problems in later tiers

There is a notion of ***trivial changes*** that can be pushed sooner than 24 hours. It should be clearly stated in the review mail that the intention is to push as a trivial change. How to actually define "trivial" is decided on a case-by-case basis but in general it would be things like fixing a comment, or moving code without changing it. Backing out a change is also considered trivial as the change itself in that case is generated by mercurial.

Relevant testing

Please note that the submit repository will only run a set of smoke-tests to ensure your change compiles and runs on a variety of platforms. It will not do any targeted testing on the particular code you have changed. Running through the submit repository is only the minimum requirement. You must also make sure your change works as expected before pushing using targeted testing. Consider writing a few JTREG tests for your change, or some unit tests using the GTest framework. Including the new tests (in the right places) with your push to the submit repository will ensure your tests will be run as part of your testing on all platforms and in the future. Look for tier1 in test/hotspot/jtreg/TEST.groups to see which tests and directories that are included in the submit repo testing.

The push

NOTE: This section is out of date since the move to using Git and GitHub for the main OpenJDK development project.

Pushing a change is fairly straight forward. Make sure your commit has a proper description. The JBS bug id and the Rewiewed-by lines are mandatory.

```
8197844: JVMTI GetLoadedClasses should use the Access API
Summary: Make sure the holder of a class loader is accessed during iteration of CLDG
Reviewed-by: eosterlund, rkennke
```

Always make sure there are no new changes in the repository you are pushing to before pushing

`hg in` should return an empty set of changes when you push. Always verify this the last thing you do! If there are new changes in there you must pull these changes and rebase your patch on top of these new changes. We use rebase to avoid merge changesets as we want to keep the mercurial history as linear as possible. This makes it easier to do binary searches through the history when trying to determine the cause of some changed behavior for instance (read debugging).

```
hg pull -u --rebase
```

Look through the new changes before you push your change. Did something recently pushed conflict with your change? There may be a need to rerun some testing or at least recompile before pushing.

When you feel confident enough, push using **`hg push`**.

After pushing

As noted in the list above, you are expected to be around after having pushed a change in case there are any issues with it. A change that causes failures in later tiers may be backed out if a fix can not be provided fast enough, or if the developer is not responsive when noticed about the failure. Note that #7 above should be interpreted as "it is a really bad idea to push a change the last thing you do before bedtime, or the day before going on vacation".