Using jcheck

The following applies only to Mercurial (hg) repositories. For GIT repositories, the Skara tooling provides jcheck.

The various JDK projects on OpenJDK, e.g., jdk9, use jcheck to check the content and commit message of each changeset that goes into the repo. See the jcheck page for more information about jcheck, and the OpenJDK Producing a Changeset page for the required format of commit messages.

OVERVIEW

OpenJFX is not presently enabled to use jcheck on the server, but we are working towards this as a goal. This work is tracked by JBS bug JDK-8145561. The three main benefits of using jcheck are:

1. Enforces white space rules to avoid gratuitous differences from creeping into a file
2. Enforces consistency in the formatting of changeset comments so you can find the JBS bug ID associated with the change
3. Prerequisite to enabling hgupdater, which will automate the now-manual process of resolving a bug as fixed, creating backports, and adding the changeset URL in the comments of the JBS bug

The jcheck work is far enough along that it is ready for use by OpenJFX Committers to check their changesets before they push to make sure that common mistakes in formatting commit messages are caught. This can be done now, prior to us actually enabling jcheck on the server, by using a version of jcheck that has been modified to recognize FX build tags. Until this is enabled on the server, the use of jcheck is a manual process – you have to remember to run 'hg jcheck' before you push – but at least this makes it very easy to do so.

All Committers are **strongly** encouraged to use jcheck to check their changeset before pushing. There should be no excuse for pushing a changeset with a mistake that that jcheck would have caught.

SETUP

There are three simple setup steps needed to enable running jcheck:

One-time setup:

1. Download the updated version of jcheck that includes support for FX tags and put it somewhere on your system:

http://cr.openjdk.java.net/~kcr/jcheck/bin/jcheck.py

2. Add the following entry to the extensions section of your ~/.hgrc file to point to the local copy of jcheck.py

```
[extensions]
....
jcheck = /PATH/jcheck.py
```

NOTE: replace "/PATH" with the path to the jcheck.py that you downloaded.

Per-repo setup:

3. Create a .jcheck directory in your repo (e.g., the 9-dev/rt repo), and copy the following conf file there:

http://cr.openjdk.java.net/~kcr/jcheck/conf

RUNNING JCHECK

Once you have done the above setup, you can run jcheck at any time.

To check the 'tip' changeset after you commit but before you push:

```
$ hg jcheck
```

To check a specific changeset or list of changesets:

```
$ hg jcheck -r REV
$ hg jcheck -r REV_FROM:REV_TO
```

If any errors are reported, then you will need to correct them before you can push the changeset. If you are an MQ user, then you should already know how to fix up your changeset comment prior to pushing – it's easy. Otherwise, you can do the following:

If you only have a single, committed but unpushed changeset:

```
$ hg rollback   # NOTE: this only undoes the last hg operation that modified your repo
<fix any reported white-space problems>
$ hg commit
<enter a corrected commit message>
```

If you have multiple changesets, or if you have done another write operation, then you will need to save off your patches and redo them, either with a new clone of the repo or by stripping the unpushed changesets from your local repo. For this reason, you should run "hg jcheck" right after each "hg commit".