PhabricatorCodeReview

Out of date

This page is out of date, use Review Board instead.

This page explains how to use Phabricator for code review of contributions to Apache Hive.

Setup

1. Visit https://reviews.facebook.net and connect your existing github or Facebook account there.
2. The Phabricator command-line tool arc requires a PHP interpreter available on your development machine. If you can run the command **php --version**, you already have it. If not, google for "php command line" to see how to install it for your system.
3. Next, install arc. If you follow the Arcanist Quick Start guide, only the first section ("Install Arcanist") is applicable.
4. Run **ant arc-setup** to install an extra module needed for JIRA integration. (This ant target was added with the commit of HIVE-2486.)
5. In your Hive checkout (either git or svn), run **arc install-certificate** and follow the instructions given.

That's it, you're ready to use Phabricator!

Submitting Patches (git)

From a git checkout (e.g. from **git clone git://github.com/apache/hive.git**), a typical usage sequence is as follows:

1. **ant arc-setup**
2. **git checkout -b HIVE-123-dev-branch**
3. edit some files to implement HIVE-123
4. **git commit -a -m "initial description of how I have gone about making DDL code suck less"**
5. **arc diff --jira HIVE-123**
6. optionally, visit diff URL provided by Phabricator to do things like add reviewers and CC's

Note that at this point, your diff has already been submitted as a patch against JIRA, along with a corresponding comment in JIRA containing a link back to the new diff.

If you just want to see what your diff looks like, without actually creating it, use **--preview** to the **arc diff** command line (but don't proceed with creating the diff when you visit it in the UI since it will be missing the JIRA information).

Updating Patches (git)

At this point, reviewers can add comments and request changes. To address them:

1. **git checkout HIVE-123-dev-branch**
2. edit some more files
3. **git commit -a --amend --reset-author -C HEAD** (this will keep the existing log message, which is important to Phabricator)
4. **arc diff** (you'll be required to provide a description of your changes here; this will be shown to reviewers in the UI)

Repeat until all reviewers are satisfied. Note that you only need to specify **--jira** on the very first call to **arc diff**; after that, arc will get it from the revision ID stored in the git log.

Submitting Patches (svn)

From an svn checkout (e.g. from **svn checkout http://svn.apache.org/repos/asf/hive/trunk hive-trunk**), a typical usage sequence is as follows:

1. **ant arc-setup**
2. edit some files to implement HIVE-123
3. **arc diff --jira HIVE-123** (you will be prompted for a description)
4. optionally, visit diff URL provided by Phabricator to do things like add reviewers and CC's

Note that at this point, your diff has already been submitted as a patch against JIRA, along with a corresponding comment in JIRA containing a link back to the new diff.

If you just want to see what your diff looks like, without actually creating it, omit **--jira** on the **arc diff** command line. (The svn workflow does not support the --preview option.)

Updating Patches (svn)

At this point, reviewers can add comments and request changes. To address them:

1. edit some more files
2. **arc diff** (you'll be prompted to pick the correct existing diff against which to apply your changes)

Unlike git, the svn workflow does not have the ability to associate the existing diff with your checkout, which is why you have to help it out.

Reviewing Patches

When using the Differential UI to review a diff:

- Click on a line number in code to add a line-specific comment
- You can also add general comments in the box at the bottom
- When done, be sure to choose an action (e.g. Request Changes) from the dropdown at the bottom, and then click the Clowncopterize button. Without this, the contributor+JIRA won't see your changes.

Committing Patches (svn)

If you are a committer, you can apply a patch from revision D123 to be committed with:

1. **svn update**
2. **ant arc-setup**
3. **arc patch --revision 123**

Amazingly, it will actually **svn add/remove** files for you automatically, so you don't have to (as you would with a raw **patch** command). Wow! Progress!

Then to commit, run **arc commit --revision 123** once all necessary testing has passed. This will perform an **svn commit** for you with the correct log message filled in automatically.

Limitations

1. there's currently no way for contributors to grant the ASF rights on their patch except to go to JIRA and re-add the last version of the patch there; we're working on a solution for this
2. it would be nice if submitting a diff and requesting changes automatically updated the JIRA "Patch Available" status

How Arcanist works (git)

Arcanist stores information about your current workflow in the commit message.
If you amend your message it's important that you don't remove this information.
Important things that you should not touch:

1. "HIVE-123 jira" prefix on the first line - removing this will break JIRA integration, this is the only thing that ties Differential revision to a JIRA issue.

2. "Test plan:" - by default Arc-JIRA provides "EMPTY" as the test plan. You can change it if you want to, but Arcanist will refuse to work if you remove it completely.

3. "Differential revision: 123" - this is id of the revision you are currently working on, if you remove it Arcanist will create a new revision when you run "arc diff" instead of updating the old one.

4. "Reviewers: JIRA" - if you remove JIRA from reviewers, comments and patches won't be propagated to JIRA.

FAQs

1. If you run into an error like the following on arc:

```
libphutil v1 libraries are no longer supported
```

git checkout this particular tag in your libphutil repository:

```
git checkout 870bcc76434410344d27a3fa4604ac96200bf7f6
```

2. If you run into an error like this with Arcanist:

```
Call to undefined method ArcanistGitAPI parseRelativeLocalCommit()
  OR
Failed to load symbol ...
```

Checkout this tag for arcanist:

```
git checkout 6f6fde84cc530c2b51f095d9636b9e15301519a1
```