

HowToCommit

- [Guide for new Hama Committers](#)
 - [Update Team List](#)
- [Guide for Hama Committers](#)
 - [Discuss](#)
 - [Review](#)
 - [Reject](#)
 - [Commit](#)
 - [Keys](#)
 - [Verify Release](#)
 - [Verification Script](#)
 - [Eclipse Settings](#)

Guide for new Hama Committers

Please read this page <http://www.apache.org/dev/new-committers-guide.html> first.

Update Team List

Please feel free to add your name and Apache ID to [Team List page](#)

- Update the <http://svn.apache.org/repos/asf/hama/trunk/src/site/xdoc/team-list.xml> file
- Build the site

```
mvn site
```

- Deploy the website (See <http://svn.apache.org/repos/asf/hama/site/trunk/> and <http://www.apache.org/dev/project-site.html#svnpubsub>)

Guide for Hama Committers

This page contains guidelines for committers.

Discuss

Every ideas for improvements, new features, and suggestions are recommended to be discussed in polite terms before implementation on the dev@ list, and then its decisions must be listed on our [RoadMap](#) page. In simple improvement or bug type issues, you can skip the discussion and report directly on JIRA.

Review

Hama committers should, as often as possible, attempt to review patches submitted by others. Ideally every submitted patch will get reviewed by a committer within a few days. If a committer reviews a patch they've not authored, and believe it to be of sufficient quality, then they can commit the patch, otherwise the patch should be cancelled with a clear explanation for why it was rejected.

For non-trivial changes, it is best to get another committer to review your own patches before commit. Use "Submit Patch" like other contributors, and then wait for a "+1" from another committer before committing.

Reject

Patches should be rejected which do not adhere to the guidelines in [HowToContribute](#) and to the [CodeReviewChecklist](#). Committers should always be polite to contributors and try to instruct and encourage them to contribute better patches. If a committer wishes to improve an unacceptable patch, then it should first be rejected, and a new patch should be attached by the committer for review.

Commit

When you commit a patch, please:

- [Configuring the Subversion client](#)
1. Add an entry in CHANGES.txt, at the end of the appropriate section. This should include the Jira issue id, and the name of the contributor.
 2. Include the Jira issue id in the commit message, along with a short description of the change and the name of the contributor if it is not you. Be sure to get the issue id right, as this causes Jira to link to the change in Subversion (use the issue's "All" tab to see these).
 3. Resolve the issue as fixed, thanking the contributor. Always set the "Fix Version" at this point, but please only set a single fix version, the earliest release in which the change will appear.

Keys

You may want to add your KEY to <http://www.apache.org/dist/hama/KEYS>

To add your KEY, you can add text to <https://dist.apache.org/repos/dist/release/hama/KEYS> (svn) file. and recommend to use a 4096 bit RSA key.

See <http://www.apache.org/dev/release-signing.html> and <http://www.apache.org/dev/openpgp.html>

Then, you can create release candidate with following command:

```
mvn clean install package gpg:sign
```

Verify Release

To verify a release, which is normally done by the committers, you can do the following:

1. Download the RC's package asc, the url must be given by the release manager starting the vote thread.

```
wget <SOME URL>/hama-0.x.0.tar.gz.asc
```

2. Import the committers keys

```
wget http://hama.apache.org/files/KEYS
gpg --import KEYS
```

Note that the last argument after import is a file, not a url.

3. Verify

```
gpg --verify hama-0.x.0.tar.gz.asc
```

If everything is correctly verified, you have to post the result on the vote thread while casting your vote.

Verification Script

You can use a convenient shell script to check the signatures, it takes the release managers apache index file URL as argument (e.g. <http://people.apache.org/~edwardyoon/dist/0.6-RC3/>).

```
#!/bin/bash

function check_return {
    RETVAL=$?
    [ $RETVAL -ne 0 ] && exit 1
}

function check_md5 {
    real=`openssl md5 $1 | cut -d ' ' -f2`
    expected=`cat $1.md5 | cut -d ' ' -f1`
    if [ "$real" != "$expected" ]
    then
        echo "md5 sums mismatch" && exit 1
    else
        echo "md5 checksums OK"
    fi
}

function check_sha1 {
    real=`openssl sha1 $1 | cut -d ' ' -f2`
    expected=`cat $1.sha1 | cut -d ' ' -f1`
    if [ "$real" != "$expected" ]
    then
        echo "sha1 sums mismatch" && exit 1
    else
        echo "sha1 checksums OK"
    fi
}
```

```

        fi
    }

function import_keys {
    wget -q http://hama.apache.org/files/KEYS
    gpg --import KEYS
    rm KEYS
}

if [ $# -ne 2 ]
then
    echo "Usage: `basename $0` {people.apache.org index url of the release}"
    exit 1
fi

import_keys

wget --convert-links -q -L -O out.html $1
cat out.html | grep -o -E 'href="([^\#]+)([gz|asc|md5|sha1|zip])"' | cut -d'"' -f2 | sort | uniq > links

mkdir release

while read lnk; do
    echo "Downloading $lnk"
    wget --directory-prefix=release $lnk
done < links

for filename in `find release/ -iregex ".*\.(gz|zip)" -printf "%f\n"`
do
    echo "checking release/$filename"

    gpg --verify release/$filename.asc
    check_return

    check_md5 release/$filename

    check_sha1 release/$filename

done;

rm out.html
rm links

```

Eclipse Settings

You should set the **Active Maven Profiles** property for each of these projects: *hama-core*, *hama-examples*, *hama-graph* and *hama-ml*

Property	Value
Project Properties -> Maven -> Active Maven Profiles	add hadoop1