Process for Contributing to the ONTAP REST API GitHub Repository

Version: March 25, 2020

This document describes a suggested process for how individuals can contribute to the NetApp ONTAP REST API code repository at GitHub. The process consists of a simple loop which is described below. There is also an appendix at the end with some common Git commands.

After the repository becomes public, Jacob (Jacob.A@netapp.com) will be able to administer the repository. Among other things, this involves reviewing changes and merging user branches to the master branch (based on pull requests).

Before you begin

- Make sure Git is installed on your local workstation
- Be familiar with the Git CLI (most of the commands are included in this document)
- Have an editor installed locally as needed to make content changes
- Have a GitHub account with access to the ONTAP REST API repository
- Access the repository at: *https://github.com/NetApp/ontap-rest-api*

General guidelines for branch, push, and merge operations

There are several ways that Git can be used and different ways to manage branches. Here are the suggested guidelines for the ONTAP REST API repository:

- Your local master branch should be in synch with the origin master branch
- Always create new branches from your local master for changes
- Local branches should:
    - Have a unique and identifiable name
    - Be pushed to the GitHub repository one time
    - Be temporary and relatively short lived
- After a branch is pushed to GitHub and merged to master, it can be deleted

Getting started

You'll need to copy the ONTAP REST API repository to your local Git environment.

```
git clone git@github.com:netapp/ontap-rest-api.git
cd ontap-rest-api
```

High level overview of the process

At a high level, the basic process for contributors consists of a simple loop. More detail for each step is provided below.

1. Create a new local branch:
    a. Refresh local master by pulling from origin master
    b. Create a new local branch with a unique name
    c. Check out the local branch
2. Prepare the content:
    a. Make changes in the local branch
    b. Commit changes
3. Push the branch to the GitHub repository
4. Create a pull request at GitHub for your new branch
5. Stop:
    a. Confirm that the new branch has been merged to master
    b. Optionally sign in to the GitHub web UI to see your changes
6. Delete the local branch
7. Go to #1

1. Create a new local branch

You should make all changes in a local branch (not master). To help track changes and administer the repository, every branch should have a unique name. It's a good idea to use your name or NetApp SSO ID, followed by some unique ID (such as, the creation date).

Before creating a new branch, it's a good idea to make sure that all previous branches have been pushed to the GitHub repository and have been merged and deleted.

```
git branch
git checkout master
git pull origin master
git status
```

```
git branch my-branch
git checkout my-branch
```

2. Prepare the content

You will most likely be adding or updating code, but there could be other content as well. After making changes, you need to commit them locally. In general, you should commit frequently, which is a little bit like saving a Word file as you edit it. This means you'll typically perform multiple commits before pushing a branch to the production repository. But how often you commit local changes is up to you. To commit:

```
git checkout my-branch
git status
git add *
git status
git commit -m 'Some comment'
```

If needed, you can make more changes and commit again. Also, if you left a branch open for a long time, you can refresh your local files after a commit:

```
git checkout my-branch
git status
git pull origin master
```

3. Push the branch to GitHub

When your local branch is ready, you can push it to GitHub. Before pushing a branch, all local changes must first be committed.

```
git checkout my-branch
git status
git push origin my-branch
```

4. Create a pull request

After a branch is pushed to GitHub, you can make a formal request to have the branch merged into master. After you make the pull request, Jacob is notified.

1. Sign in to GitHub.
2. Click the **ontap-rest-api** repository.
3. Click **branches** at the top of the page.

4. Locate your branch and on the right side click **New pull request**.
5. Confirm the merge goes from your branch to master.
6. Type a name for the request and optionally add a comment.
7. Click **Create pull request** (not draft pull request).

5. Stop

Wait for your pull request to be completed and the branch to be merged to master. Also, it's always a good idea to confirm that your changes are in the master branch. You can sign in to GitHub and click the **ontap-rest-api** repository to review the updated files.

6. Delete the local branch

Before creating another local branch for new changes, you should delete the old one.

```
git checkout master
git branch -D my-branch
git branch
```

Appendix: Common Git CLI commands

Here are some of the more common Git commands.

`git status`

Displays the status of the branch you currently have checked out. You can see files that are unstaged (including new files) as well as files that are staged and ready for the next commit.

`git branch`

Lists all the branches in the local repository, with an asterisk next to the branch that is currently checked out.

`git branch my-branch`

Creates a new branch named *my-branch*.

`git checkout my-branch`

Checks out a branch so that you can edit files as well as and add/remove files.

`git branch -D my-branch`

Deletes a local branch. You cannot delete a branch that you currently have checked out. In general, you should have the master branch checked out. You'll issue this command when a branch is no longer needed, such as after pushing it to the GitHub repository.

`git add file-name`

Adds or stages a single file so it will be included in the next commit.

`git add *`

Adds or stages all files and other changes that are currently unstaged, so they will be included in the next commit.

`git rm file-name`

Deletes a file.

`git mv file-name-1 file-name-2`

Renames a file.

```
git commit -m 'Some message'
```

Updates the local branch by committing all staged files.

```
git push origin my-branch
```

Pushes the checked out local branch to branch *my-branch* at the GitHub repository.

```
git pull origin master
```

Updates the checked out local branch from the *master* branch at the production repository.