GitHub Workflows Training-Mini Sandbox

Mini V.01 | 09-04-2019

michael.vincerra@intel.com

practice repo: sandbox

1. Bookmark this link to access the git-workflows-training repo.
2. In the upper right, select "Download" to view this PDF and select hyperlinks.

git workflows sandbox

This repo offers you a place to learn Git Workflows. This repo is provided as a safe environment where you're free to attempt git commands, and make mistakes, without any major consequences. In this Sandbox, modify only the `.md` files. This training is meant to be interactive where an instructor guides participants.

Clear Linux Documentation

When you're ready to contribute to Clear Linux documentation, engage an instructor and use this PDF training document for more comprehensive training and instruction.

prerequisites

This tutorial assumes that you:

- Use SSH (thought https is acceptable for training only)
- Use a Linux environment (e.g., Window Subsystem for Linux, Ubuntu, etc.)
  - Follow this guide: https://document-publishing.gitlab-pages.devtools.intel.com/tcs-template/howtos/tech/set-up-ubuntu-on-windows.html
- Have **GitHub** account. If not, register here.
- May have little or no experience using the command line interface (CLI)

configure, create SSH key for GitHub

Check if you already have an SSH key.

1. Enter: *ls -la ~/.ssh*
2. Enter: cat ~/.ssh/id_rsa.pub
3. Copy the contents of: *id_rsa.pub*
4. Skip to "Adding a new SSH key to your GitHub account" below.

OTC recommends that you follow Connecting to GitHub with SSH in order to :

- Check for existing SSH keys
- Generate a new SSH key and adding it to the ssh-agent
- Add a new SSH key to your GitHub account

Note: You do not need to use a passphrase when setting up SSH.

Fork. Clone. Add remote upstream.

1. In the CLI, navigate to your home directory.
2. In GitHub, go to: https://github.com/mvincerx/git-workflows-training
3. Select **Fork** button in the far upper right. (Icon shows it's forked to your account.)
4. Go to *your own GitHub account*, navigate to YOUR FORKED repo.
5. Press **Clone or download**.
6. In the dialogue box, select "Use SSH". Copy the SSH address.
7. In CLI, enter the command below and assure that *[yourusername]* appears.

```
git clone git@github.com:[yourusername]/git-workflows-training.git
```

8. Return to the CLI and change directory into the forked repo:

```
$ cd git-workflows-training
```

9. Repeat Step 2. Repeat Step 5. Enter command to copy upstream repo's URL:

```
git remote add upstream git@github.com:mvincerx/git-workflows-training.git
```

10. Enter command to ensure **origin** and **upstream** exist: *git remote -v*

synchronize

After you fork and clone a repo, your forked copy is immediately up-to-date. Yet a repo is dynamic. Other contributors make pull requests. A repo maintainer merges those pull requests, and then your forked copy becomes out of date.

Your responsibility as a contributor is to keep your forked copy of the repo up to date. After you fork and clone, you contribute to an ever-changing river of continuous development.

PAUSE: Allow the instructor to merge a PR in this Sandbox repo. The purpose of this is to show you how to keep your forked copy up-to-date.

Terms:

- **pull request**: A change proposed to a branch (e.g., master); abbreviated as a "PR".
- **repo**: A GitHub repository.

synchronize after merge on upstream master

1. On the CLI, enter:

```
git checkout master
```

2. Next, enter:

```
git pull --rebase upstream master
```

NOTE: We recommend using ‘--rebase’ in this command for a cleaner commit history.

3. Push the downloaded changes to your local **origin**—to assure that your **forked copy** is up to date.

```
git push origin master
```

Congratulations! Your forked repo is now up to date. Make a habit of doing this every time that you start to work. Do this before you start a new branch.

create a branch: add, commit, and push.

1. Assure you complete steps in Synchronize before proceeding.
2. While on the master branch in the CLI, create a new branch. Enter:

```
git checkout -b [xx]-[filename]
```

   Branch name = **[xx]-[filename] – where [xx] are your initials**

3. While on new branch, make edits to the document in the Editor (e.g., Sublime).
4. In the CLI, enter command to **add** the revised [filename] and commit your changes.

```
git add [relative/path/filename]
```

5. Enter command. In the editor write a descriptive commit message. Save and exit.

```
git commit –s
```

   See: How to write a good commit message

6. Push your new branch. Then complete a Pull Request.

```
git push origin [xx]-[filename]
```

create a pull request (PR)

1. Visit https://github.com/mvincerx/git-workflows-training. This is 'upstream/master'.
2. "*Your recently pushed branches* :" will appear as the last branch you pushed.
3. Select the button "Compare & pull request. "Open a pull request" appears.
4. Find the **gear icon** beside Reviewers: Select the Repo Maintainer as reviewer.
5. Review the pull down menus below "Open a pull request."
6. Assure *compare* *(right)* shows your branch and *base fork* (left) shows upstream.
7. Select the button "Create pull request."
8. You will receive email notifications on the status of your pull request.
9. The Repo Maintainer will ask for changes or will merge your pull request.

Note: The repo maintainer may request additional changes. If so, continue to push commits to your branch until it's approved.

synchronize your fork to upstream

After your PR is merged, sync your fork with upstream master. **You've come full circle.**

1. In the CLI, enter:

```
git checkout master
```

2. Pull changes from upstream master to update your local forked copy:

```
git pull --rebase upstream master
```

We recommend using '--rebase' in this command for a cleaner commit history.

3. Push the changes to your local **origin**—to assure that your **forked copy** is up to date.

```
git push origin master
```

Do these steps every time that you start to work. Do this before you start a new branch.

Congratulations! Your forked repo is now up to date.