GitHub Workflow Activity

Form teams

Find a partner, then pair up with another pair. These 2 pairs will form a team.

Each pair will sometimes be a maintainer and sometimes a contributor. Rotate roles as necessary to ensure everyone in your team gets a chance to experience each role.

Overview

In this activity your team will play out several scenarios following steps described in *Workflow Reference*.

- Complete each part in order.
- Complete each step in each part in order.

Part 0: Create an account on GitHub

If you currently have an account on GitHub, use that for this lab.

If you do not currently have an account on GitHub, create one now:

- Go to github.com
- Follow these WikiHow instructions. Select the free plan (for public and open source repositories).

Part 1: Create an organization on GitHub

- Make sure at least 2 members of your team are logged into GitHub.
- Have one member of your team click on the button with their username/photo at the top left of the page, in the feed. Select "Create organization" from the drop-down menu that appears.
- Name the organization whatever you like.
- Set the default permissions so that all organization members can create projects and write to any project in the organization.
- Add all team members to the organization.

Part 2: Create an official upstream repository in organization

- Name it `ourfavorites`
- Give it a default README.md file.

Part 3: Contributor-1 setup

- Select a team member to be Contributor-1
- Help Contributor-1 to follow *Setup: (1-4)* in the *Workflow Reference* to prepare his/her local and

remote repositories.

Part 4: First contribution

- Help Contributor-1 to follow *Starting your contribution: (5-11)* to add a new file `favorite-foods.txt` that contains a couple of Contributor-1's favorite foods.
- Select someone to play Maintainer (not Contributor-1).
- Help the Maintainer to accept Contributor-1's pull-request on GitHub.
- Help Contributor-1 to follow *Update your master (25-26)* and *Delete unneeded branches (27-29)* to clean up.

Congratulations, your team has made its first contribution! Celebrate. :clap: :clap:

Part 5: Contributor-2 setup and second contribution

- Help Contributor-2 to follow *Setup: (1-4)* to prepare his/her local and remote repositories.
- Repeat the steps above to have contributor-2 contribute a new file `favorite-movies.txt` with a couple of his/her favorite movies.
- Make sure that the maintainer has accepted contributor-2's pull-request and contributor-2 has updated their master and cleaned up.

Celebrate again. :clap: :clap:

Part 6: First synchronization

- Contributor-1's repositories are out of synch. Help Contributor-1 follow *Keep your repositories up-to-date (16-21)* to update his/her repositories.

Celebrate. But keep it small. :clap: Don't worry, there will be bigger celebrations later.

Part 7: Contribute conflicting changes

- Have Contributor-1 and Contributor-2 independently follow the contribution workflow to add another favorite food to the end of `favorite-foods.txt`.
- Maintainer, accept one of the pull-requests. Try to accept the other. You won't be able to because changes in the pull-request conflict with the other that you already accepted.
- Help the contributor with the unresolved pull-request to follow *Keep your repositories up-to-date (16-21)* to synchronize his/her repositories and resolve the conflicts.
- Maintainer, note that the conflicted pull-request is automatically updated and should be acceptable. Accept the pull-request.
- Have contributors clean up.

Celebrate enthusiastically. :clap: :clap: :clap: That was challenging.

Part 8: Multi-round contribution

- Have contributor-1 add another food, and contributor-2 another movie.
- Have the maintainer ask for a modification through the pull-request (e.g., "Please pick another flavor. I don't like chocolate.").
- Have contributors make, commit, and push the new changes.
- If the maintainer is satisfied, accept the pull-requests.
- Contributors, don't forget to clean up.

Notice how pull-requests provide a way for a contributor and a maintainer to communicate about a proposed change. Also notice how the pull-request updates automatically as new changes are pushed to the same branch.

Part 9: Squash

- Repeat the multi-round contribution until both contributors have made multiple commits.
- Maintainer, through the pull-request, ask contributors to squash their work into a single commit.
- Help contributors to follow *Squash your commits (22-23)* to squash their commits into a single commit and push it.
- Maintainer, accept the pull-requests once it contains the same work, but only a single commit.
- Contributors, don't forget to clean up.

This is another moment for an enthusiastic celebration. :clap: :clap: :clap: :clap: :clap: Well done!

Copyright and Licensing