Github Repository Guide and Conventions (v0.2)

WayFinder
Group 12

Change Log

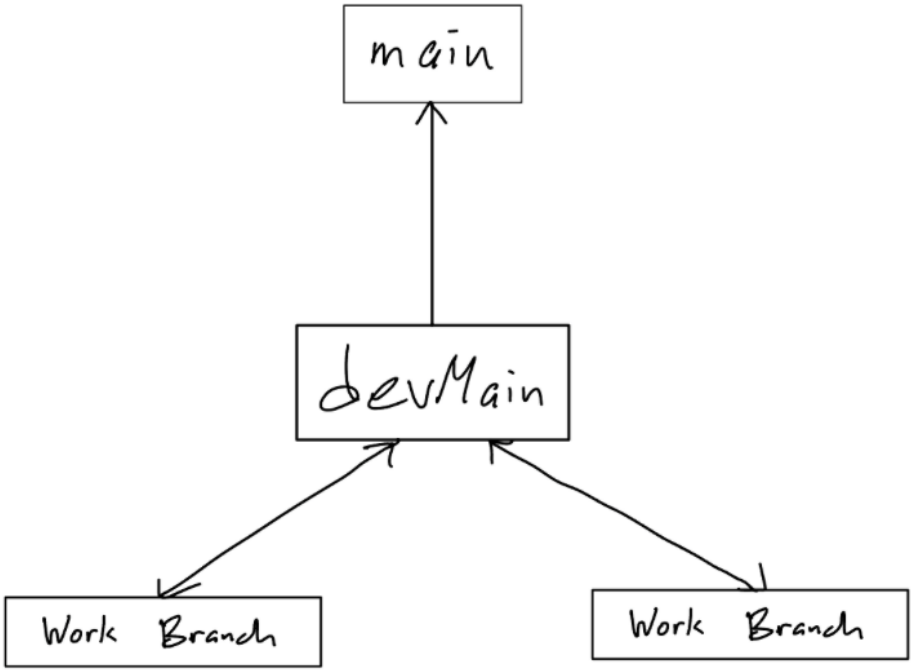
Date	Author	Version	Change Log
June 23, 2023	Nicholas Mah	v0.1	Initial iteration of document
July 2, 2023	Nicholas Mah	v0.2	-Add specific instructions to syncing branch with devMain -pull request related items

Table of Contents

Purpose

This document outlines the conventions and best practices to be followed for the CMPT-276 Github repository.

Overview of Repository Structure



Main: ‘main’ branch will be considered production ready and will contain the latest stable release.

devMain: The collective branch all developers will work off of. This is where end-to-end integration testing will be conducted.

Work Branch: Individual branches made for each assigned task/feature being worked on.

Branches

Main

This will be the branch the render server will pull its updates from. The only way to push changes onto the ‘main’ branch is to merge the ‘devMain’ branch into ‘main’ via a pull request. This branch may not have the most up-to-date code as we may hold changes on the ‘devMain’ branch that aren’t ready for release.

Note that this is not the default branch on GitHub.

Developer Branch (devMain)

This will be considered the main branch for developers to combine and integrate. This will also be the default branch to ease the cloning process for new work branches. Developers must ensure their code on the work branch has been merged with ‘devMain’ on local to ensure there are no conflicts during the pull request process.

Work Branches

Each branch will only be used for 1 feature/task being worked on. To ensure work amongst different branches being worked on by one individual is not mixed up a new local repo should be made to work on each branch.

1. Clone the repo onto your local computer.
2. Checkout a new branch in that repo. The naming convention is as follows for these branches:

WFDR-XX-Short_Description

The beginning is a tag representing our project. Following this will be a 2-digit numeric tag for our tasks. Lastly, a short, couple-word description of the task/feature being worked on in the branch with underscores in between each word. Here are a couple of examples:

- WFDR-06-Login_UI
- WFDR-23-Login_Controller
- WFDR-56-Setup_GPT_Requests

3. After finishing working on the code, merge or rebase with the current ‘devMain’ branch on local. Once all conflicts have been resolved, submit a pull request to merge the work branch into ‘devMain’

**Ensure all commits have a meaningful message.*

Syncing with ‘devMain’

1. Using ‘git status’ ensure you are on your ‘Work Branch’ and ensure all of your changes are staged. The response should be as follows.

```
$ git status
On branch WFDR-111-Link_DataBase
Your branch is up to date with 'origin/WFDR-111-Link_DataBase'.

nothing to commit, working tree clean
```

2. Switch to the ‘devMain’ branch by checking-out ‘devMain’

```
$ git checkout devMain
Switched to branch 'devMain'
Your branch is up to date with 'origin/devMain'.
```

3. Update your local ‘devMain’ branch by pulling from remote. An example output is provided below.

```
$ git pull
remote: Enumerating objects: 258, done.
remote: Counting objects: 100% (79/79), done.
remote: Compressing objects: 100% (55/55), done.
remote: Total 258 (delta 23), reused 58 (delta 14), pack-reused 179
Receiving objects: 100% (258/258), 36.96 KiB | 970.00 KiB/s, done.
Resolving deltas: 100% (60/60), completed with 5 local objects.
From https://github.com/NickMoo5/CMPT-276-Project
   4939dbd..d735a98  devMain                 -> origin/devMain
 * [new branch]      WFDR-102-Signup/profile_creation ->
origin/WFDR-102-Signup/profile_creation
 * [new branch]      WFDR-103.1-Landing_Page_User ->
origin/WFDR-103.1-Landing_Page_User
 * [new branch]      WFDR-103.2-AdminLanding ->
origin/WFDR-103.2-AdminLanding
 * [new branch]      WFDR-104.1-SetPrefsNewUsers ->
origin/WFDR-111-Link_DataBase
 * [new branch]      WFDR-112-UserController ->
origin/WFDR-112-UserController
Updating 4939dbd..d735a98
```

4. Switch back to your work branch by checking-out your ‘Work Branch’

```
$ git checkout WFDR-111-Link_DataBase
Switched to branch 'WFDR-111-Link_DataBase'
Your branch is up to date with 'origin/WFDR-111-Link_DataBase'.
```

5. Merge the updated local ‘devMain’ branch into your ‘Work Branch’. A vim editor may open up. Enter a message indicating you are updating your work branch with ‘devMain’ changes. If you have merge conflicts you must resolve these before finishing the merge. VSCode and Intellij have built-in merge-conflict tools.
   https://learn.microsoft.com/en-us/visualstudio/version-control/git-resolve-conflicts?view=vs-2022
   https://www.jetbrains.com/help/idea/resolving-conflicts.html

```
$ git merge devMain
hint: Waiting for your editor to close the file...       0 [sig] bash 1182!
sigpacket::process: Suppressing signal 18 to win32 process (pid 23604)
Merge made by the 'ort' strategy.
 src/main/resources/css/addUser.css      |  31 ++++++
 src/main/resources/js/addUser.js        | 154 ++++++++++++++++++++++++++
 src/main/resources/static/addPrefs.html | 188
++++++++++++++++++++++++++++++++
 src/main/resources/static/addUser.html  | 124 ++++++++++++++++++++
 4 files changed, 497 insertions(+)
 create mode 100644 src/main/resources/css/addUser.css
 create mode 100644 src/main/resources/js/addUser.js
 create mode 100644 src/main/resources/static/addPrefs.html
 create mode 100644 src/main/resources/static/addUser.html
```

6. Push your changes from the merge to remote

```
$ git push
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 272 bytes | 272.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/NickMoo5/CMPT-276-Project.git
   1c45979..295cfb7  WFDR-111-Link_DataBase -> WFDR-111-Link_DataBase
```

Pull Requests

Initiating a Pull Request

See the steps to initiate a pull request at the following link:
https://scribehow.com/shared/How_to_Initiate_a_Pull_Request__acJ2ClzOT36FqntwBpYeYA

Merging a Pull Request

On the Pull Request page, after at least 2 reviewers have approved the request and no additional changes are required, click the green button “Squash and Merge”. Remove any commit messages that appear in the text box that are redundant or irrelevant. Click “Confirm Squash and Merge” to finish and close the pull request.

Reviewing a Pull Request

See the steps to review a pull request at the following link:
https://scribehow.com/shared/Reviewing_a_Pull_Request__0-a-ZArVQcKwkr-PDzXA7g

**If changes were requested then repeat the same process once the individual has made the updates to their code.*