

# git use in IEEE 802.1

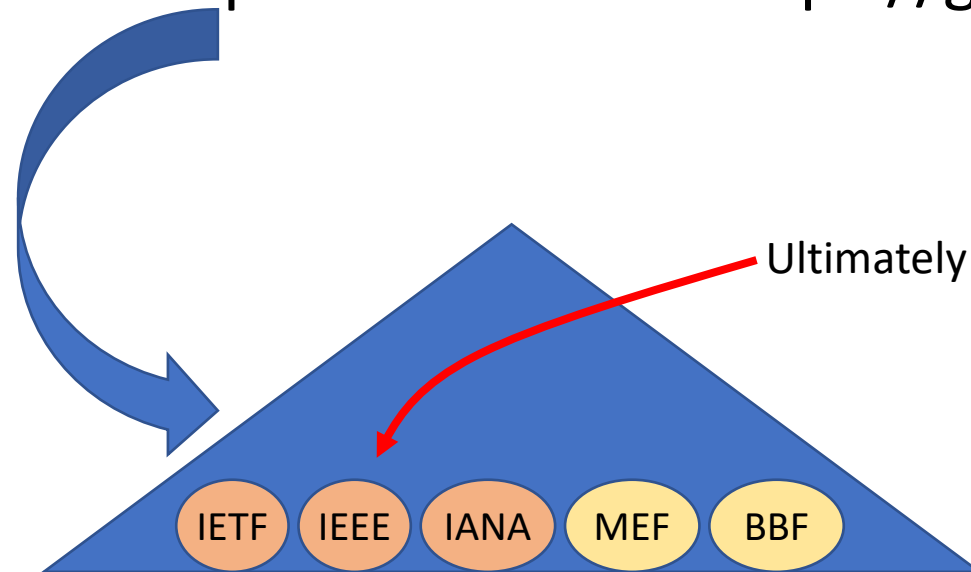
yangsters-smansfield-git-examples-1122-v03

# Executive Summary

- Desire is to use git the git was intended
  - Leverage git to make merging easier
- Decisions to make
  - Keep what we currently do
    - All Editors fork and clone the IETF's YangModels/yang github repository
  - Create a single IEEE 802 repository for YANGsters, that is used by the IEEE project editors
    - Have designated YANGster “yang leaders” interact with the IETF's YangModels/yang github
- YANGsters needs to decide if we want to have our own “common repository” or if we want to continue to have the editors interact with the IETF's github repository directly

# Overview

- Most SDO including IEEE 802.1 use git for Yang
- git is an industry standard way to update files
- Models are published here: <https://github.com/YangModels/yang>



Ultimately a copy of Published YANG goes here

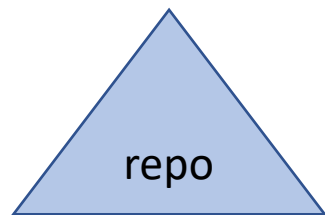
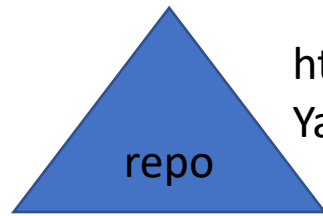
We need a low overhead process for IEEE 802.1 authors to write, validate and publish YANG

# Options for interacting with git repos

- Option 1: Fork of a common repository
  - This is known as the “Fork and Pull model”
    - See <https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/getting-started/about-collaborative-development-models#fork-and-pull-model>
- Option 2: Local clone only of a common repository
  - This is known as the “Shared repository model”
    - See <https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/getting-started/about-collaborative-development-models#shared-repository-model>

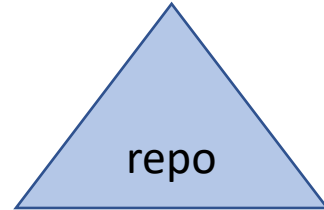
# This tutorial discusses the shared model

## Today's mode of operation (fork and pull)

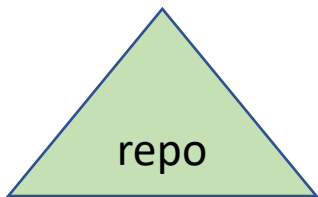


Editor 1's  
Personal  
Fork

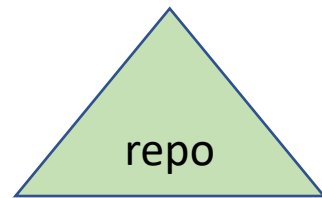
...



Editor N's  
Personal  
Fork

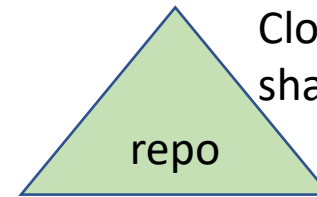
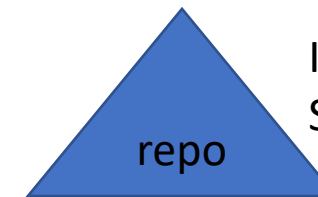
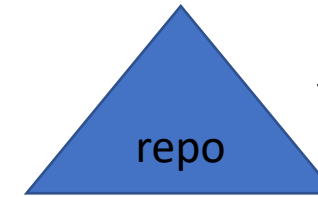


Editor 1's  
Clone of  
their fork



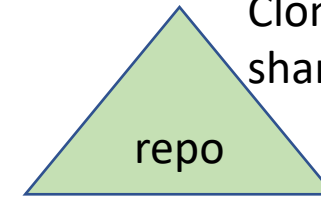
Editor N's  
Clone of  
their fork

## Possible mode of operation (shared)

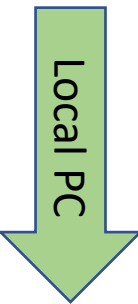


Editor 1's  
Clone of the  
shared fork

...



Editor N's  
Clone of the  
shared fork





# High Level Description of Lifecycle

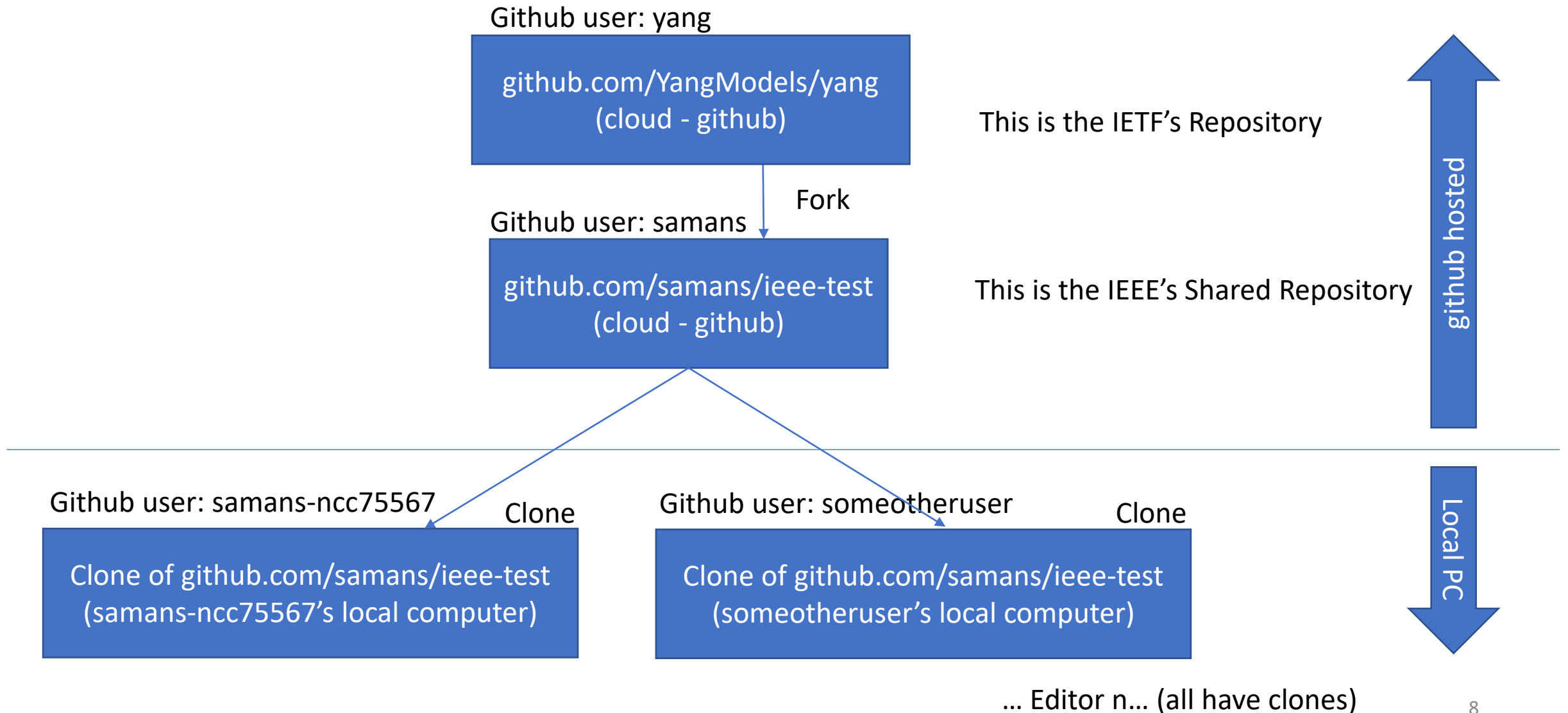
- What we do now ([Fork and Pull style](#))
  - Each editor creates a fork of IETF's YangModels/yang repository in the editor's personal github
  - Each editor then clones the fork to their local development environment
  - Develops their changes in a branch on their local development environment
  - Commits locally, then pushes to their remote
  - Goes to their github and requests a pull request to the IETF's Yang/YangModel
- Alternative ([Shared style](#))
  - IEEE YANGster "yang leader" forks the IETF's YangModels/yang repository
  - The "yang leader" enables access to the shared repository for all IEEE yang editors
  - Each editor clones the shared repository to their local development environment
  - Develops their changes in a branch on their local development environment
  - Commits locally, then pushes their branch to the shared repository, then creates a pull request to merge their branch into the main branch of the shared repository
  - The "yang leader" will review and approve the pull request
  - If merged, the "yang leader" can submit the pull request to the IETF's YangModels/yang repository

Focus of the rest of this presentation

# Benefits of the Shared Approach

- There is only one Fork of the IETF's YangModels/yang repository
  - Easier to keep the IEEE's share repository synced
- The IEEE editors are only interacting with IEEE yang people
  - Validation can be more IEEE focused
  - Merging is handled by people that understand the IEEE Yang Models and structure
  - Should be more responsive than waiting for YangCatalog people to merge
  - Editors can assist one another easily. The “yang leader” can update editors branches if needed to resolve conflicts.
  - Easier for all editors to stay in sync with IEEE Yang Models and dependencies
- Number of commands and amount of syncing is reduced for the IEEE Yang Editors

# Setup of Tutorial (Shared Example)





# Clone locally

- For this example, here is a clone of samans/ieee-test:main
  - `git clone https://github.com/samans/ieee-test.git`
- The pointer back to samans/ieee-test is called “origin”
  - `git remote -v`
- If you haven’t done this, you need to
  - [Creating a personal access token - GitHub Docs](#)

samans-ncc75567’s local computer

```
scott@Cosima:~/rws1/gits/tutorial-shared$ git clone https://github.com/samans/ieee-test.git
Cloning into 'ieee-test'...
remote: Enumerating objects: 20, done.
remote: Counting objects: 100% (20/20), done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 20 (delta 6), reused 4 (delta 2), pack-reused 0
Unpacking objects: 100% (20/20), 4.89 KiB | 8.00 KiB/s, done.
scott@Cosima:~/rws1/gits/tutorial-shared$ cd ieee-test
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ ls
README.md  ieee802-dot1q-common-types-test.yang  ieee802-dot1q-foo-test.yang
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$
```

```
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git remote -v
origin  https://github.com/samans/ieee-test.git (fetch)
origin  https://github.com/samans/ieee-test.git (push)
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$
```

# High Level Process

1. Local Repo Synced with samans/ieee-test:main
  - Make sure your local repository is up to date with the common repository
  - Make sure your “github id” is a collaborator of samans/ieee-test:main
2. Make a Change Locally
  1. Use branches
  2. Fix revision dates of YANG files (See discussion of date conflicts)
  3. Validate etc. (pretty print, pyang and yanglint)
  4. Commit your changes locally
3. Resync the local repo with samans/ieee-test:main again and merge any changes into your branch
  - Resolve any conflicts
  - Revalidate using the yang tooling (like check.sh or equiv.)
4. Push branch to samans/ieee-test
5. Pull request to merge your branch into samans/ieee-test:main
6. “yang leaders” take it from here

# STEP 1: Local Repo Synced

- This step syncs your local main with samans/ieee-test:main (origin)
- Command line
  - git checkout main
  - git pull
- This ensures you have the latest files when you create your branch

samans-ncc75567's local computer

```
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git checkout main
Already on 'main'
Your branch is up to date with 'origin/main'.
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git pull
Already up to date.
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ █
```

# STEP 2: Make a Change Locally

- git checkout -b foobar-04
- Edit the file(s)
- git status
- git commit -am "some commit message"
- If you want to save you branch into the repo
  - git push origin foobar-04
    - (using the "editors" github login)

samans-ncc75567's local computer

```
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git checkout main
Already on 'main'
Your branch is up to date with 'origin/main'.
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git pull
Already up to date.
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git checkout -b foobar-04
Switched to a new branch 'foobar-04'
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ ls
README.md  ieee802-dot1q-common-types-test.yang  ieee802-dot1q-foo-test.yang
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ vi ieee802-dot1q-common-type
s-test.yang

scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git status
On branch foobar-04
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   ieee802-dot1q-common-types-test.yang

no changes added to commit (use "git add" and/or "git commit -a")
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git commit -am "changed sign
ed int type"
```

# STEP 3: Resync and merge locally

- This step is needed to ensure you have the latest main in your branch
  - This is important if ready to publish a draft with these new files, do this before you do the pull request
  - This will make the “yang leaders” job easier by resolving any conflicts that arise from merges that other branches have done

- git checkout main
- git pull

if changes come down then...

- git checkout foobar-04
- git merge main foobar-04
  - In rare cases you will need to resolve any conflicts (none in this example)

samans-ncc75567's local computer

```
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 1.39 KiB | 18.00 KiB/s, done.
From https://github.com/samans/ieee-test
   bbf005f..fd23264  main      -> origin/main
Updating bbf005f..fd23264
Fast-forward
 README.md | 28 ++++++++-----
 1 file changed, 23 insertions(+), 5 deletions(-)
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git checkout foobar-04
Switched to branch 'foobar-04'
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git merge main foobar-04
Merge made by the 'recursive' strategy.
 README.md | 28 ++++++++-----
 1 file changed, 23 insertions(+), 5 deletions(-)
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$
```

# STEP 4: Push branch to samans/ieee-test

- git checkout foobar-04
- git push origin foobar-04
  - (using the “editors” github login)

```
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git checkout foobar-04
Already on 'foobar-04'
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ git push origin foobar-04
Username for 'https://github.com': samans-ncc75567
Password for 'https://samans-ncc75567@github.com':
Enumerating objects: 12, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 908 bytes | 45.00 KiB/s, done.
Total 8 (delta 5), reused 0 (delta 0)
remote: Resolving deltas: 100% (5/5), completed with 3 local objects.
remote:
remote: Create a pull request for 'foobar-04' on GitHub by visiting:
remote:   https://github.com/samans/ieee-test/pull/new/foobar-04
remote:
To https://github.com/samans/ieee-test.git
 * [new branch]      foobar-04 -> foobar-04
scott@Cosima:~/rws1/gits/tutorial-shared/ieee-test$ █
```

Search or jump to... Pull requests Issues Marketplace Explore

samans / ieee-test Public

Watch 1 Fork 1 Star 0

Code Issues Pull requests Actions Projects Wiki Security Insights

foobar-04 had recent pushes 1 minute ago [Compare & pull request](#)

main 4 branches 0 tags [Go to file](#) [Add file](#) [Code](#)

samans Update README.md fd23264 10 minutes ago 17 commits

README.md	Update README.md	10 minutes ago
ieee802-dot1q-common-types-test...	added an unsigned int type	1 hour ago
ieee802-dot1q-foo-test.yang	Update ieee802-dot1q-foo-test.yang	yesterday

About  
No description, website, or topics provided.

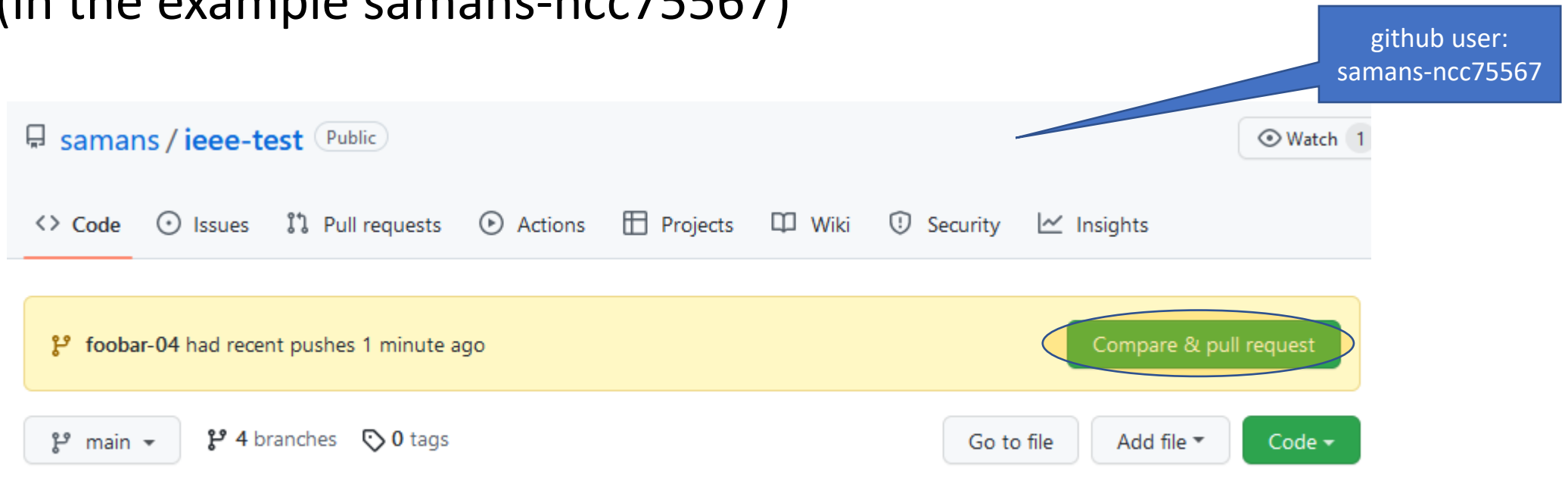
Readme  
0 stars  
1 watching  
1 fork

Releases  
No releases published

github user:  
samans-ncc75567

# STEP 5: Pull request

- Pull request to merge your branch into samans/ieee-test:main
- This is done on the github website logged in using the editor's github id (in the example samans-ncc75567)



The screenshot shows the GitHub interface for the repository 'samans / ieee-test'. At the top, the repository name is displayed with a 'Public' badge and a 'Watch 1' button. Below this is a navigation bar with links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, and Insights. A yellow notification bar indicates that 'foobar-04' had recent pushes 1 minute ago. A green button labeled 'Compare & pull request' is highlighted with a blue oval. A blue callout box points to the repository name, containing the text 'github user: samans-ncc75567'. At the bottom, there are buttons for 'Go to file', 'Add file', and 'Code', along with branch and tag information.

# STEP 6: Merge into samans/ieee-test (“yang leader”)

github user:  
samans

- The IEEE “yang leader” can merge the pull request, comment on it, etc. (In the example, logged into “samans” github id)
- Then delete the branch

The screenshot shows a GitHub pull request interface for the repository 'samans / ieee-test'. The pull request is titled 'Foobar 04 #4' and is in an 'Open' state. It shows that 'samans-ncc75567' wants to merge 3 commits into the 'main' branch from the 'foobar-04' branch. The pull request details include a comment from 'samans-ncc75567' (1 minute ago) with the text 'No description provided.' and a commit history from 'samans' (41 minutes ago) showing three commits: 'added signed int type' (21df5d9), 'changed signed int type' (5067173), and 'Merge branch 'main' into foobar-04' (ecb9cec).

A good idea



**Pull request successfully merged and closed**

You're all set—the foobar-04 branch can be safely deleted.

Delete branch



## STEP 7: Pull request for IETF YangModels/yang (“yang leader”)

- If the pull request is merged into the common repository (samans/ieee-test)
  - Then the “yang leader” can to a pull request to the IETF’s YangModels/yang repository
- This follows the same “fork and pull” procedure we do now
  - Except only a small number of people need to worry about this

# Things To Do

- High Level
  - Setup repository
  - Secure the repository
  - Automated validation and testing support
- Create a github id for the IEEE “yang leader” called <<what???.>>
- Create a github repository called: <<what???.>>
- Assign the collaborators (IEEE YANG editors github ids) to the new github
- Identify a few allowed to merge pull requests
- Lock down the main branch
- Update check scripts and enable automation

# Summary of steps

- Step 1 (sync local)
  - `git checkout main`
  - `git pull`
- Step 2 (make local changes)
  - `git checkout -b foobar-04`
  - (make some changes to a file)
  - `git status`
  - `git commit -am "some commit message"`
- Step 3 (resync) (note: in the tutorial I introduced a change to the origin main, to illustrate what happens)
  - `git checkout main`
  - `git pull`
  - `git checkout foobar-04`
  - `git merge main foobar-04` (this is the command that would show conflicts that need to be resolved -- none in this example)
- Step 4 (push branch to origin)
  - `git checkout foobar-04`
  - `git push origin foobar-04`
- Step 5 (pull request to suggest your branch to origin)
  - This is done at <https://github.com/samans/ieee-test> while logged in as samans-ncc75567 (or whatever the editors github id is)
- Step 6
  - The "yang leader" logs in to <https://github.com/samans/ieee-test> (in this example as samans) and clicks the "Merge pull request" button
  - The branch can be deleted after it is successfully merged
- Step 7
  - The "yang leader" then follows the existing process to do a pull request of the changes to the IETF's YangModels/yang repository

# Backup Material

# Date Conflicts

- Because of the way pyang and yanglint process imports
  - If there are multiple revisions of the same module in the search path, the tooling will always use the module with the most recent revision date.
- In the IEEE directory structure there is the following structure (not exhaustive)
  - YangModels
    - yang
      - ieee
        - draft
          - 802.1
            - Qrev
            - Qcw
            - Qcz
        - Published
          - 802.1

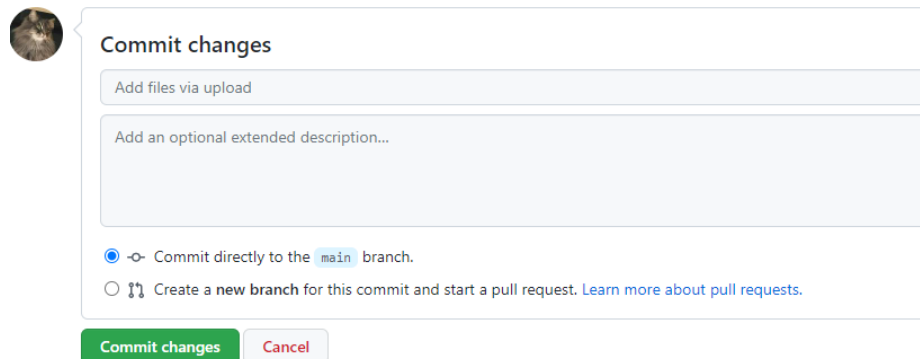
- We have a situation where (for example)
  - ieee802-dot1q-types.yang is found in
    - yang/ieee/draft/802.1/Qrev
    - yang/ieee/draft/802.1/Qcw
    - yang/ieee/draft/802.1/Qcz
    - yang/ieee/published/802.1
  - So it can be tricky to ensure each project is including the correct module to validate
- The revision date of a module in published should always be older than any module found in a draft directory

# Options

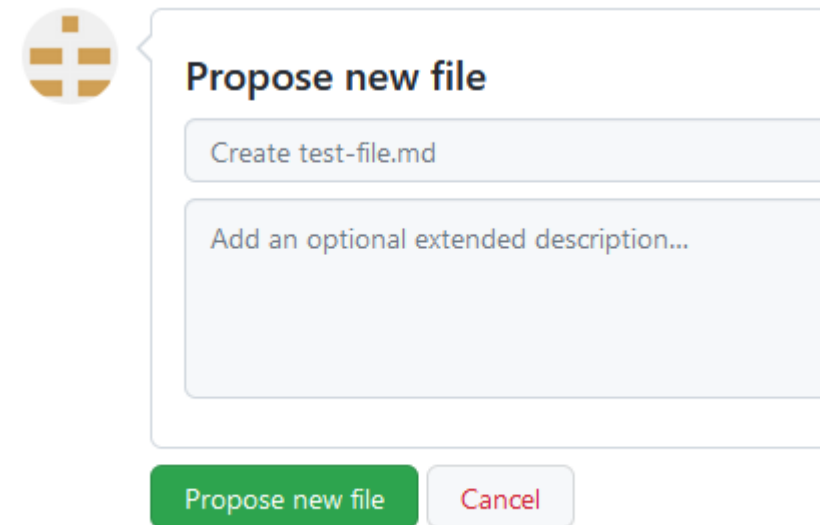
- Option 1: Fork of a common repository
  - This is known as the “Fork and Pull model”
    - See <https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/getting-started/about-collaborative-development-models#fork-and-pull-model>
- Option 2: Local clone only of a common repository
  - This is known as the “Shared repository model”
    - See <https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/getting-started/about-collaborative-development-models#shared-repository-model>

# Notes

- If you have the permission, you can commit directly to the main branch
- If you don't, you can propose a new file/changes or create a new branch and start a pull request.
- Keep in mind the branch is called "main" in both samans:ieee-test and in samans-ncc75567:ieee-test
  - So when it says "Commit directly to the main branch" you need to be aware of which user you are logged-in as, and which repository you trying to modify.



The screenshot shows a 'Commit changes' dialog box. It features a header with a globe icon and the title 'Commit changes'. Below the title is a text input field labeled 'Add files via upload'. Underneath is a larger text area for 'Add an optional extended description...'. At the bottom, there are two radio button options: the first is selected and labeled 'Commit directly to the main branch.', and the second is labeled 'Create a new branch for this commit and start a pull request. Learn more about pull requests.'. At the very bottom, there are two buttons: a green 'Commit changes' button and a white 'Cancel' button with a red border.



The screenshot shows a 'Propose new file' dialog box. It features a header with a cross icon and the title 'Propose new file'. Below the title is a text input field containing 'Create test-file.md'. Underneath is a larger text area for 'Add an optional extended description...'. At the bottom, there are two buttons: a green 'Propose new file' button and a white 'Cancel' button with a red border.