Introduction to Git

Python Fundamentals for Engineers and Manufacturers

What is Git?

Workshop repo
http://bit.ly/2opFlZ6

From https://git-scm.com/

Why is version control important?

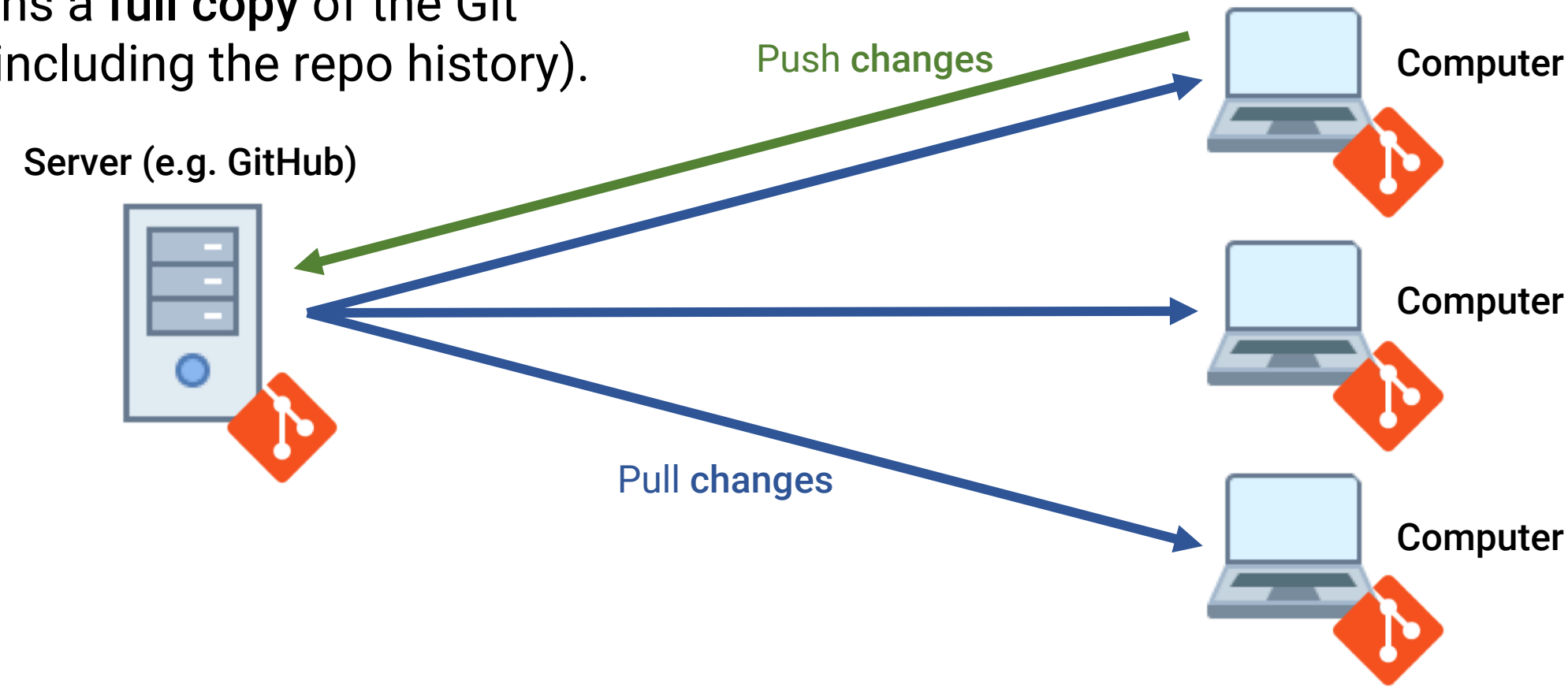
Workshop repo
http://bit.ly/2opFlZ6

- **No one is perfect** – you need a way to roll-back changes and explore lines of development.
- Developers can ”go back in time” and see every change to a file (e.g. to understand why a change was made).
- Provides a back-up for your repo (if you are using GitHub, for example).
- Supports multiple people working on a repo at the same time.

What makes Git “distributed”?

Workshop repo
http://bit.ly/2opFlZ6

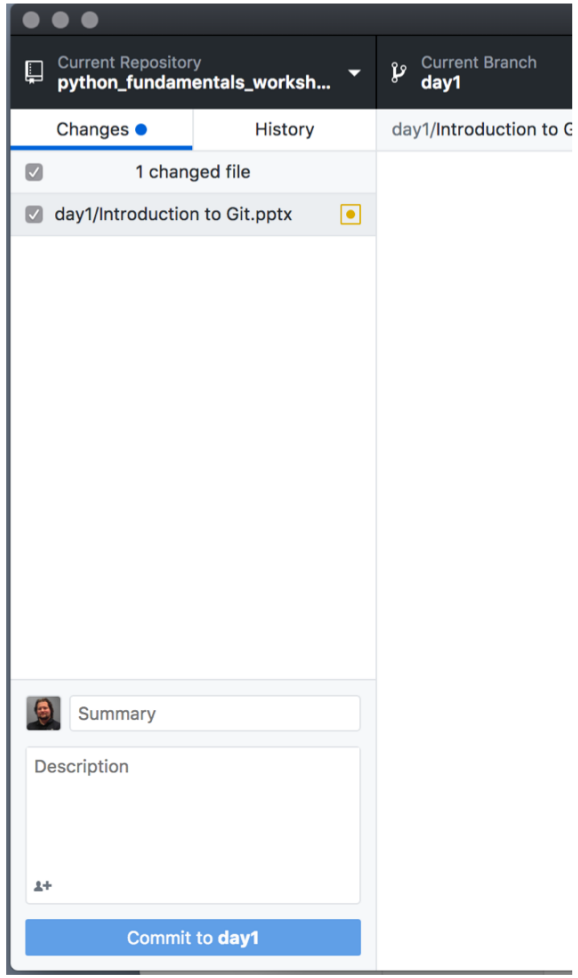
The Server and every Computer contains a **full copy** of the Git repo (including the repo history).



Git “user interface options”

Workshop repo
http://bit.ly/2opFlZ6

GitHub Desktop



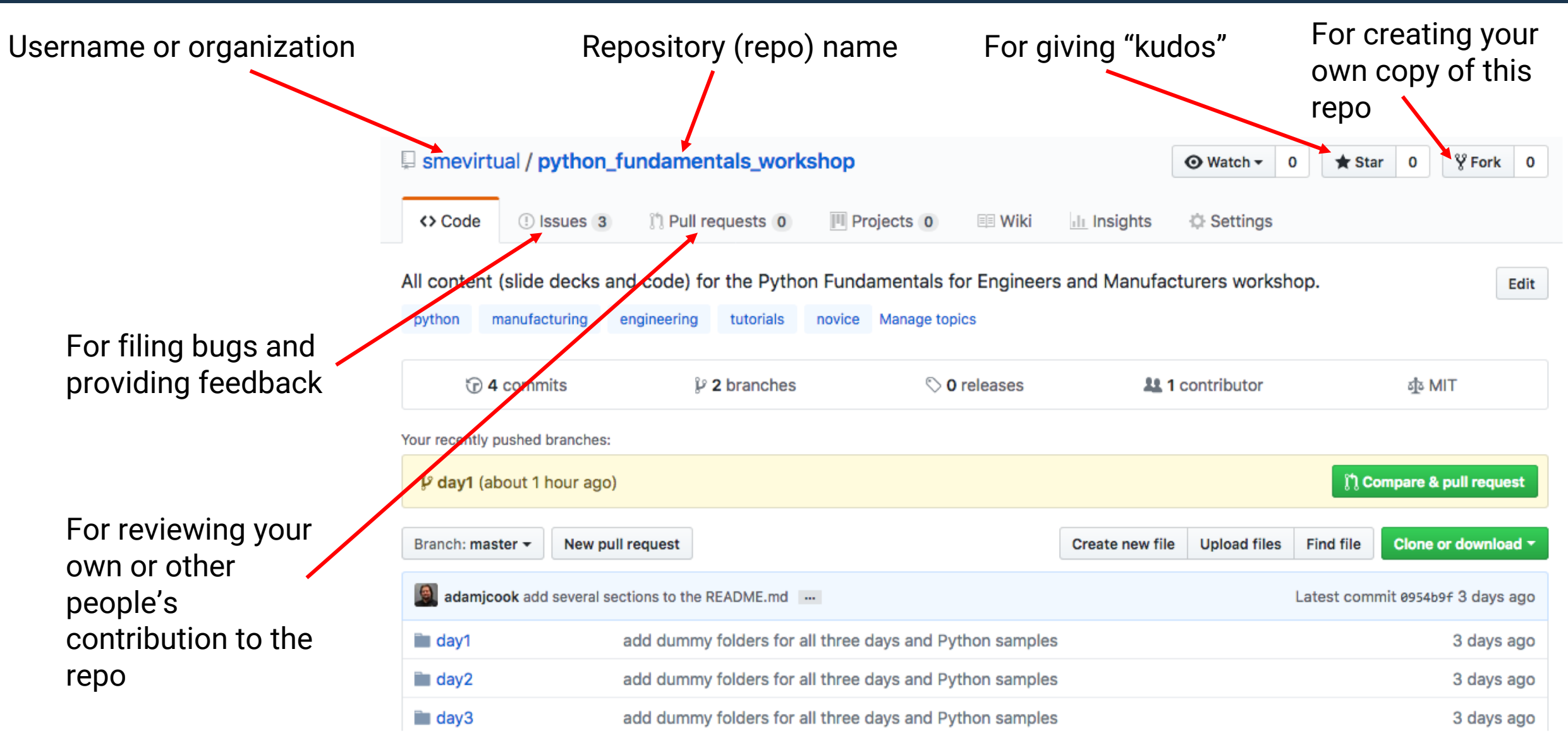
Git in terminal

```
adamjcook at rhazes in ~/projects/python_fundamentals_workshop on day1
$ git status
On branch day1
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   day1/Workshop Housekeeping.pdf
        modified:   day1/Workshop Housekeeping.pptx

no changes added to commit (use "git add" and/or "git commit -a")

adamjcook at rhazes in ~/projects/python_fundamentals_workshop on day1 [!]
$ git add --all

adamjcook at rhazes in ~/projects/python_fundamentals_workshop on day1 [+]
$ git status
On branch day1
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   day1/Workshop Housekeeping.pdf
        modified:   day1/Workshop Housekeeping.pptx


adamjcook at rhazes in ~/projects/python_fundamentals_workshop on day1 [+]
$ git commit
[day1 a9a0cba] slightly adjust title style on housekeeping slide deck
 2 files changed, 0 insertions(+), 0 deletions(-)

adamjcook at rhazes in ~/projects/python_fundamentals_workshop on day1
$ git push origin day1
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 126.92 KiB | 1.35 MiB/s, done.
Total 5 (delta 4), reused 0 (delta 0)
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To github.com:smevirtual/python_fundamentals_workshop.git
   b280828..a9a0cba  day1 -> day1

adamjcook at rhazes in ~/projects/python_fundamentals_workshop on day1
$ git status
On branch day1
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   day1/Introduction to Git.pptx

no changes added to commit (use "git add" and/or "git commit -a")

adamjcook at rhazes in ~/projects/python_fundamentals_workshop on day1 [!]
$
```

Integrated into code editor

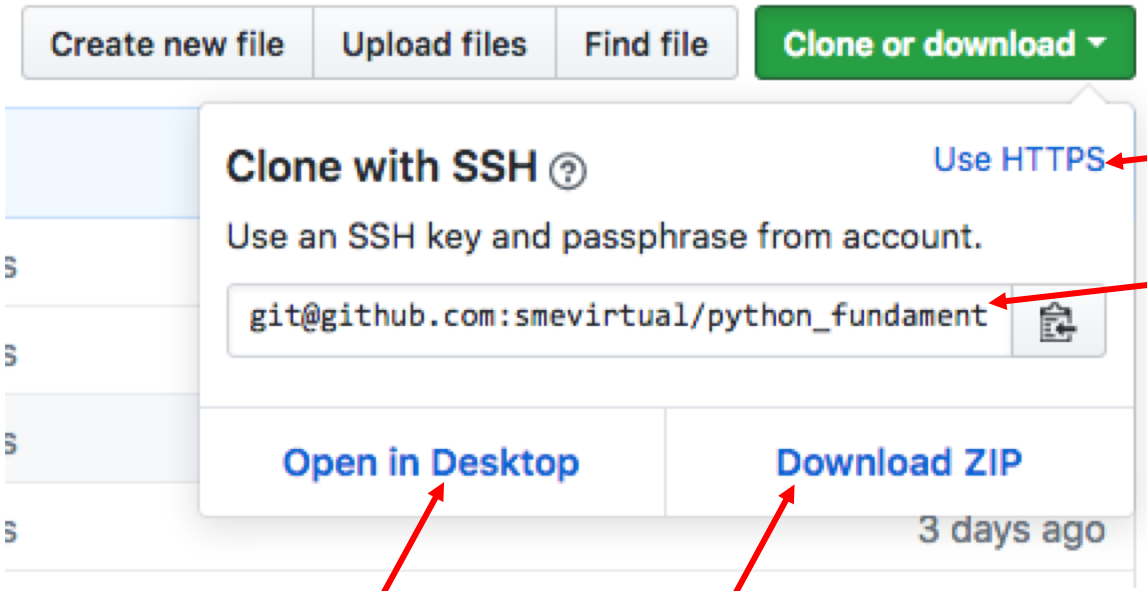
Remote Repositories (GitHub UI)

Workshop repo
http://bit.ly/2opFlZ6



Remote Repository URL

Workshop repo
http://bit.ly/2opFlZ6



Git over HTTPS (for repo cloning via terminal)

Git over SSH (for repo cloning via terminal)

Repo cloning via GitHub Desktop

Mostly used if you just want a copy of the source code.

Usually, called **`origin`**

Two types of Git URLs

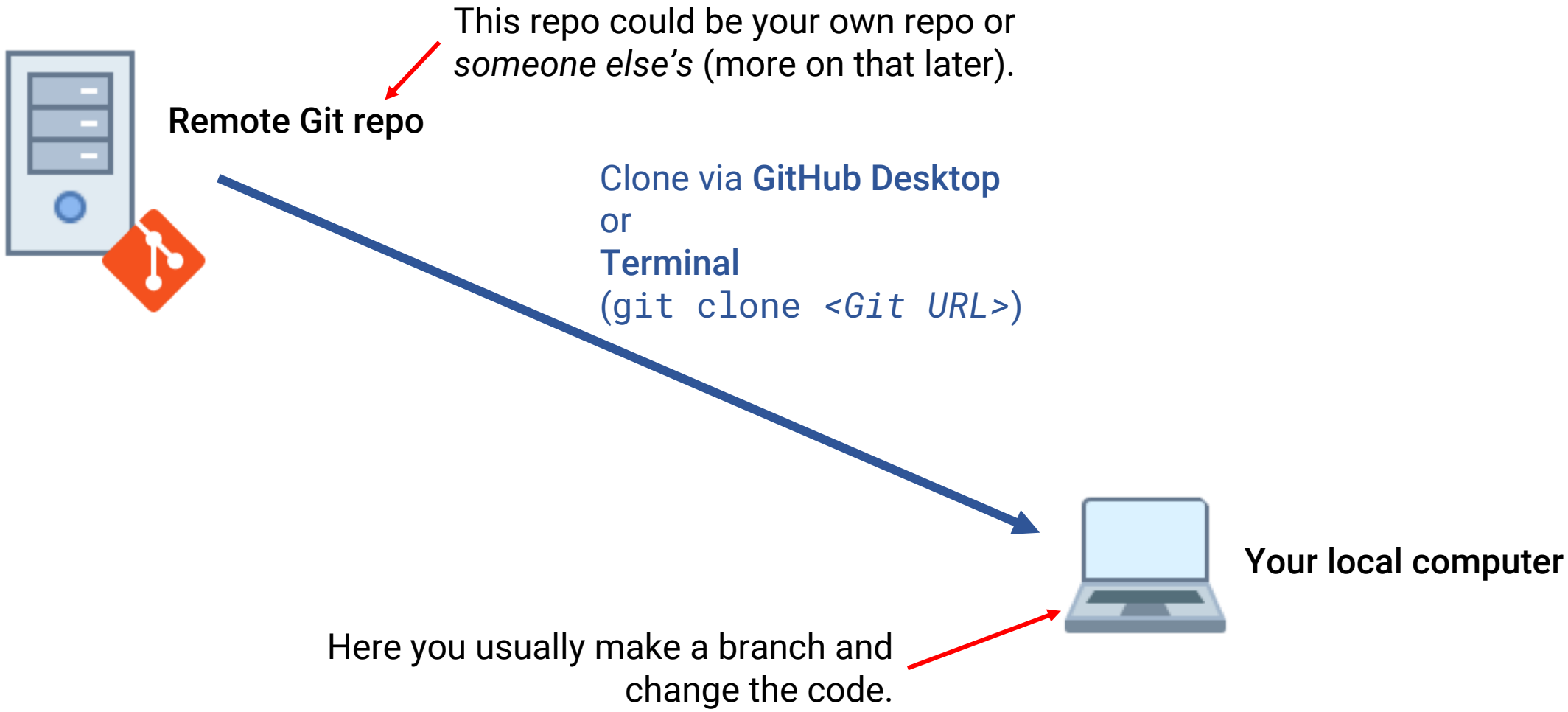
`git@github.com:smevirtual/python_fundamentals_workshop.git`

or

`https://github.com/smevirtual/python_fundamentals_workshop.git`

Cloning

Workshop repo
http://bit.ly/2opFlZ6

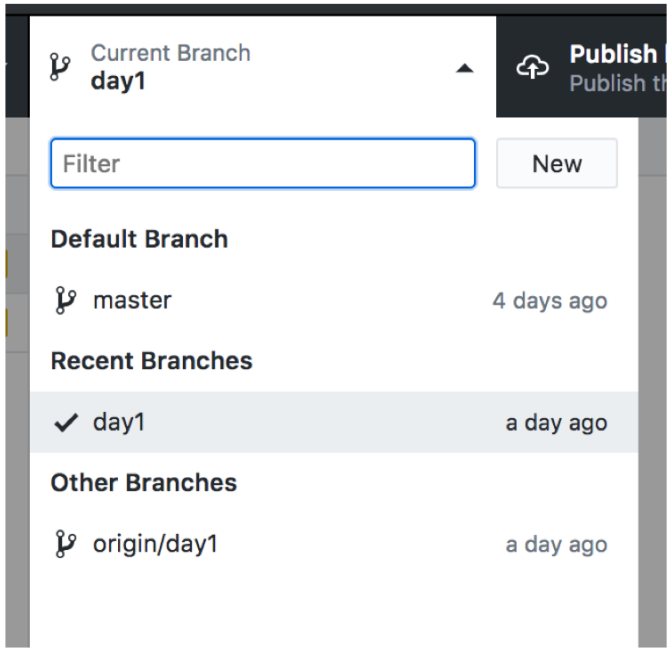


All icons from [Icons8]

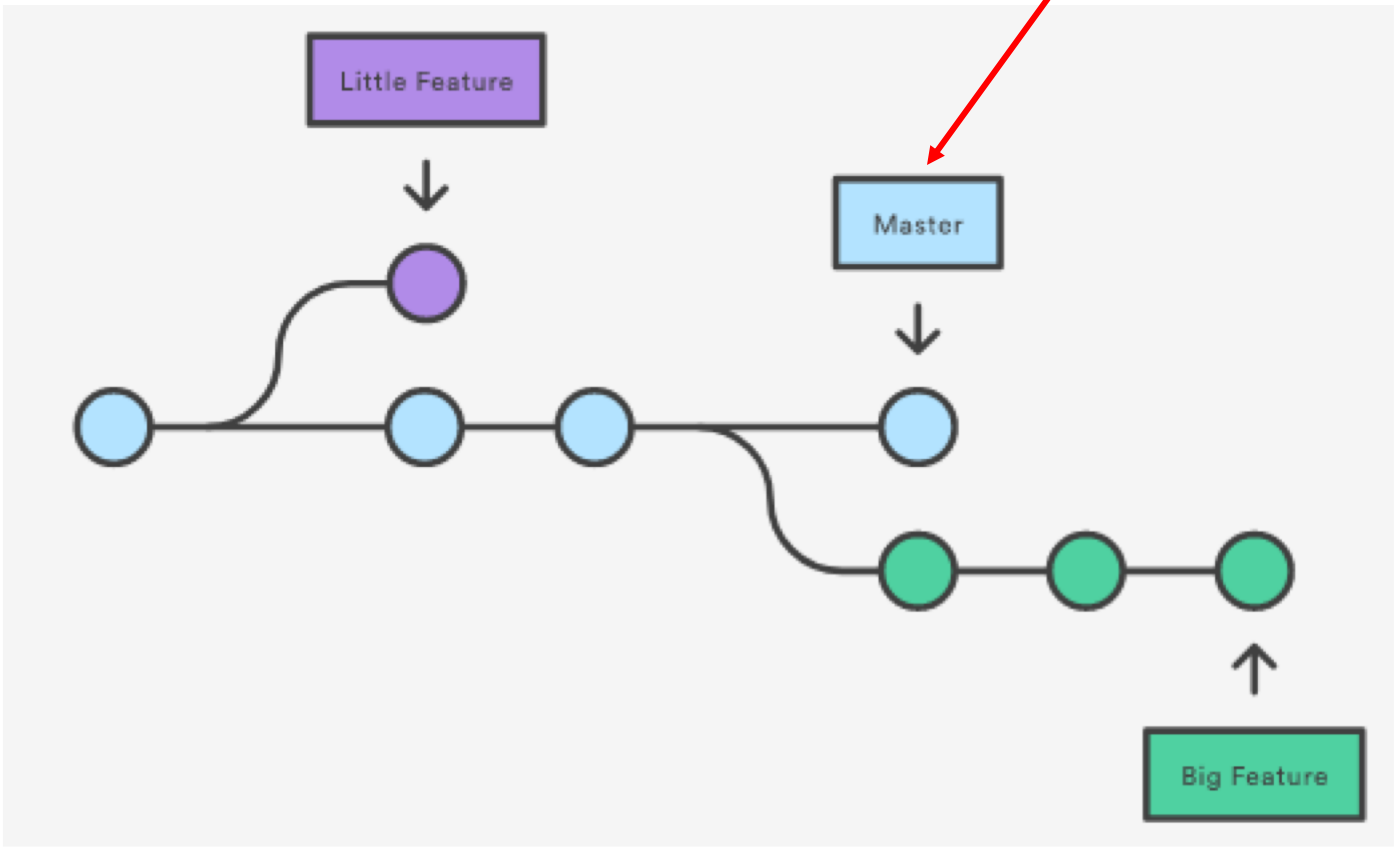
Branching

Workshop repo
http://bit.ly/2opFlZ6

Terminal command
`git branch <branch name>`
Terminal command
`git checkout <branch name>`



Usually, the `master` branch is the main development branch.



All icons from Icons8

Image source: http://bit.ly/2ocHJ2Z

Diffs

Workshop repo
http://bit.ly/2opFlZ6

Terminal command
`git diff`

Fix lint issues

develop (#2154) @blueprintjs/webpack-build-scripts@0.5.1 ... @blueprintjs/core@2.0.0-rc.2

Browse files

adidahiya committed 6 days ago — 1 parent 78d39ee — commit cefd2dd0521bd3987e064f864a4e9196c408806e

Showing 3 changed files with 18 additions and 17 deletions. — Unified | Split

17 packages/core/src/components/callout/callout.tsx — View

```
@@ -36,14 +36,15 @@ export class Callout extends React.PureComponent<ICalloutProps & React.HTMLAttri
        const iconName = this.getIconName();
        const classes = classNames(Classes.CALLOUT, Classes.intentClass(intent), className);

-       const maybeIcon = iconName === undefined ? undefined : (
-           <span className={Classes.CALLOUT_ICON}>
-               <Icon icon={iconName} iconSize={Icon.SIZE_LARGE} />
-           </span>
-       );
-       const maybeTitle = title === undefined ? undefined : (
-           <h5 className={Classes.CALLOUT_TITLE}>{title}</h5>
-       );
+       const maybeIcon =
+           iconName === undefined ? (
+               undefined
+           ) : (
+               <span className={Classes.CALLOUT_ICON}>
+                   <Icon icon={iconName} iconSize={Icon.SIZE_LARGE} />
+               </span>
+           );
+       const maybeTitle = title === undefined ? undefined : <h5 className={Classes.CALLOUT_TITLE}>{title}</h5>;

        return (
            <div className={classes} {...htmlProps}>
```

Removed lines

Added lines

All icons from [Icons8]

.gitignore File

Workshop repo
http://bit.ly/2opFlZ6

gitignore.io


```
# Created by https://www.gitignore.io/api/python

### Python ###
# Byte-compiled / optimized / DLL files
__pycache__/
*.py[cod]
*$py.class

# C extensions
*.so

# Distribution / packaging
.Python
build/
develop-eggs/
dist/
downloads/
eggs/
.eggs/
lib/
lib64/
parts/
sdist/
var/
wheels/
*.egg-info/
.installed.cfg
*.egg
```

Example of a Python `.gitignore` file contents.

Certain operating systems, code editors, development environments and runtimes create files that you **do not** want to commit (track changes).

Put a `.gitignore` file in your repo root.

Adding and Committing

Workshop repo
http://bit.ly/2opFlZ6

Terminal command
`git add --all`

Add change to staging area.

Terminal command
`git commit`

Commits the staged snapshot to the repo history (you can add a description of the changes made).

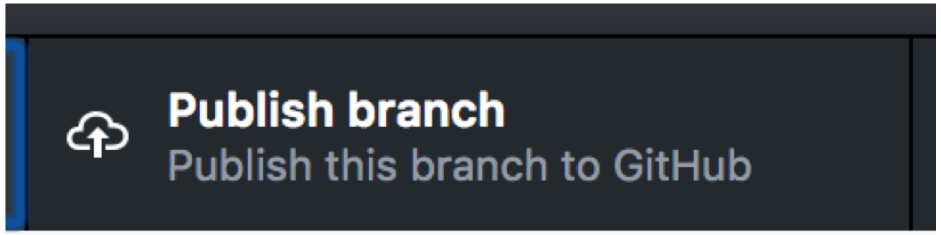
All icons from [Icons8]

Pushing or Publishing

Workshop repo
http://bit.ly/2opFlZ6

Terminal command
`git push origin <local branch name>`

Pushes commits made on **your local branch** to the **remote repo**.



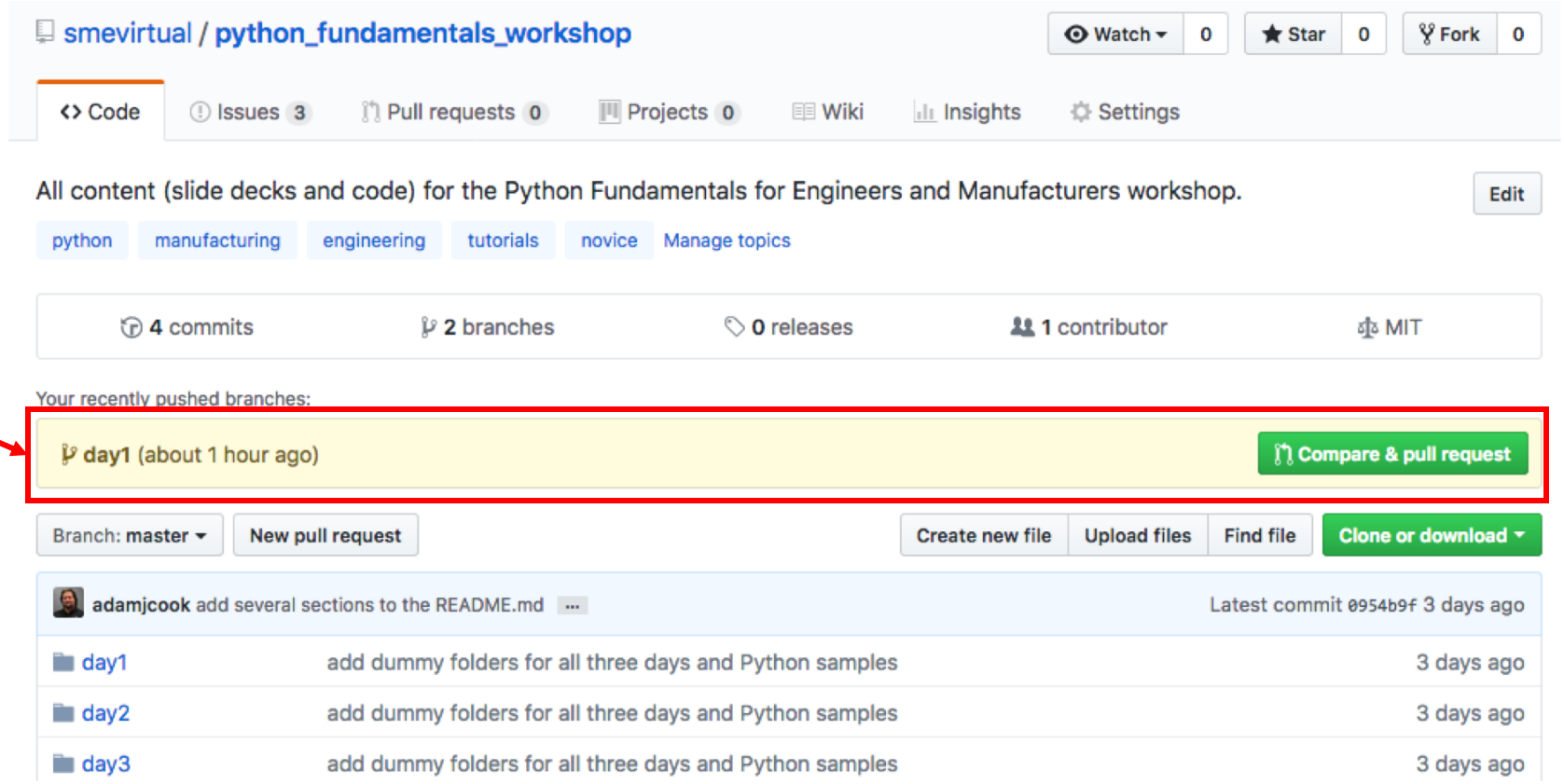
All icons from [Icons8]

Pull Requests

Workshop repo
http://bit.ly/2opFlZ6

Once you push your local branch to your remote repo (e.g. forked GitHub repo), you can create a **pull request**.

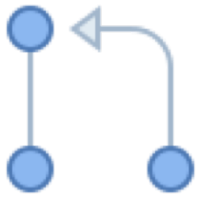
A **pull request** gives you or the owner of the remote repo the opportunity to review the changes before accepting them **officially** into the repo.



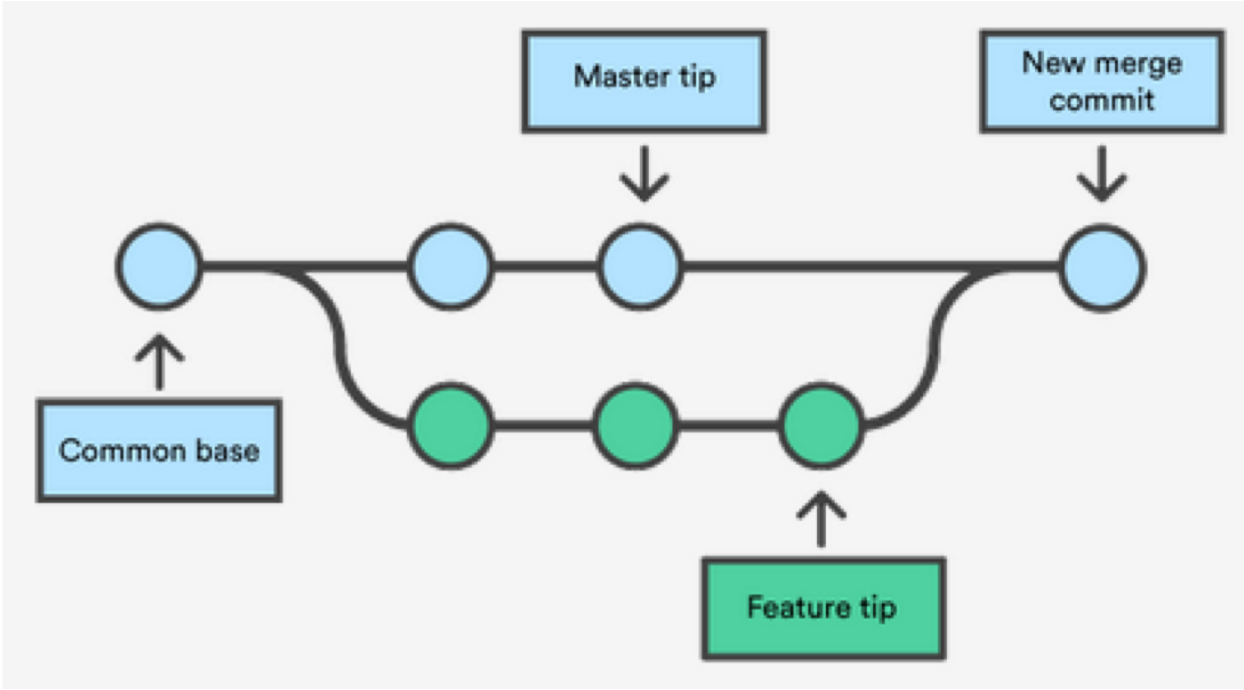
All icons from Icons8

Merging (after accepted Pull Request)

Workshop repo
http://bit.ly/2opFlZ6



Terminal command
`git pull <upstream remote branch> master`
Terminal command
`git checkout master`
Terminal command
`git merge master <upstream repo alias>/master`



All icons from Icons8

Image source: http://bit.ly/2jFfaNA

Issues

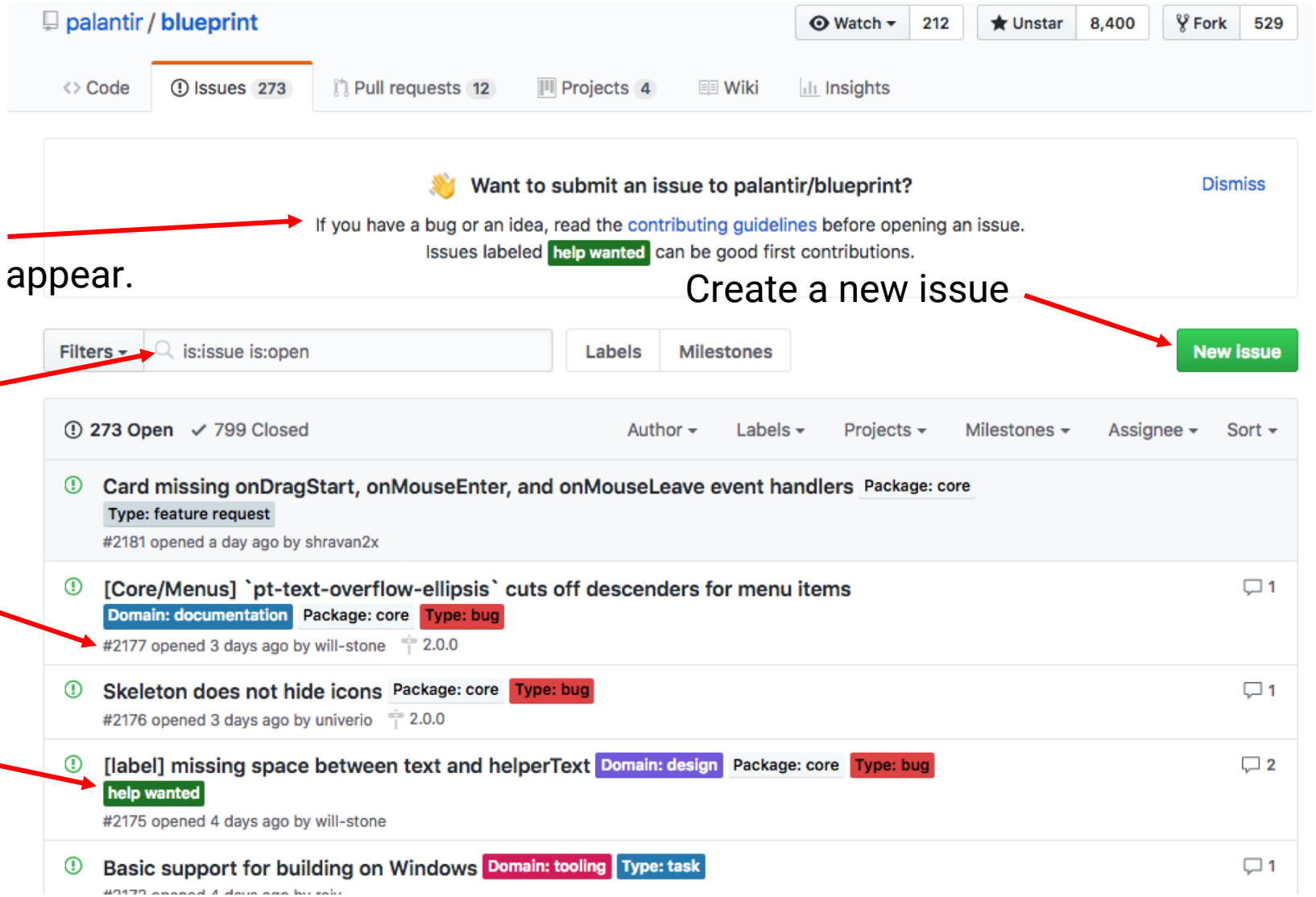
Workshop repo
http://bit.ly/2opFlZ6

Some repos will have contributing guidelines. If so, this message will appear.

Search for open and closed issues and/or issues by label.

Issue number

Issue label



All icons from [Icons8]

Common Workflow for Repos You Own

Workshop repo
http://bit.ly/2opFlZ6

1. Clone your GitHub repo to your local computer.
2. Create a new branch for your work.
3. Checkout the new branch that you created.
4. Make code changes.
5. Add and commit changes.
6. Push changes to your repo.
7. Create a pull request on GitHub.
8. Accept pull request to merge your changes.
9. Change back to your master branch and merge changes back into the master branch on your computer.

All icons from Icons8

Common Workflow for 3rd Party Repos

Workshop repo
http://bit.ly/2opFlZ6

1. Fork the repo on GitHub.
2. Clone the forked repo to your local computer.
3. Create a new branch for your work.
4. Checkout the new branch that you created.
5. Make code changes.
6. Add and commit changes.
7. Push changes to your repo.
8. Create a pull request on GitHub (the repo owner will review it and merge it in or reject it).

All icons from Icons8

Workshop repo
http://bit.ly/2opFlZ6

If you use Git, you are **practically guaranteed** to be safe. You really have to go out of your way to lose data, lose changes or screw up your codebase.

Popular Git Providers

Workshop repo
http://bit.ly/2opFlZ6

github.com

bitbucket.org

gitlab.com

Good choice if you want to host you own Git server.

Open-Source Licenses

Workshop repo
http://bit.ly/2opFlZ6

Most permissive

- Public Domain (Unlicense)
- MIT License
- BSD Licenses
- Apache 2.0
- Mozilla Public License 2.0
- GNU LGPLv2
- GNU GPLv2
- GNU LGPLv3
- GNU GPLv3
- No license or proprietary license

Least permissive

All icons from Icons8

There are other “open-source” licenses for non-software projects too.

If the license has been altered or it is not a standard open-source license or you do not understand the license, **contact a lawyer before using the code**!

- **Warning.** Some open-source communities are more inviting than others – particularly with beginners.
- Look for repos with a Code of Conduct and/or repos that support `good-beginner-issue` labels. See here for good ”beginner friendly” repos.
- **Always** check for contributing guidelines in the repo (there should be a **`CONTRIBUTING.md`** file in the repo root).
- **Also** check for the repo guidelines on code convention, issue formats and pull request procedures.
- **Note.** SME Virtual Network will **always** be inviting and supportive of programming beginners.

SME Virtual Network Code of Conduct

Workshop repo
http://bit.ly/2opFlZ6

SME Virtual Network Code of Conduct

Contact: chapters@sme.org

As leaders of the SME Virtual Network, we pledge to respect everyone who contributes by providing feedback, helping others, posting issues, updating documentation, submitting pull requests and any other activities.

Communication through any of SME Virtual Networks's channels (Facebook, LinkedIn, YouTube, GitHub, Slack, etc.) must be constructive and never resort to personal attacks, trolling, public or private harassment, insults or other unprofessional conduct.

We promise to extend courtesy and respect to everyone involved in this community regardless of gender, gender identity, sexual orientation, disability, age, race, ethnicity, religion or level of experience. We expect anyone participating in this community to do the same.

If any member of the community violates this code of conduct (including any SME Virtual Network leaders), the SME Virtual Network leadership may take action - up to and including permanently banning said member from all SME Virtual Network communication channels and accounts.

If you are subject to or are a witness to unacceptable behavior (even by the SME Virtual Network leadership), or have any other concerns, please email us at chapters@sme.org.

Adapted from the Google code of conduct.

Live document: http://bit.ly/2ov3YD0

Want to learn more about Git?

Workshop repo
http://bit.ly/2opFlZ6

Two **very good** resources are available

http://bit.ly/1fUdUiy
For terminal beginners

and

http://bit.ly/1MRoX7u
For advanced users