Getting Started with Git and GitHub

What you'll need

This is a *lunchtime computing seminar*! As we proceed, you can follow along with the instructions and get some hands-on experience with using Git.

To start - go to http://github.com/ and create an account if you don't have one already. Fill out the information about yourself in the profile.

You will also need to download Git.

Installing Git

Git is software accessed through the **command line / terminal / console / shell**

- For MacOS: open terminal and run **git --version**
- For Linux: Visit https://git-scm.com/download/linux
- For Windows: Visit https://gitforwindows.org/
    - Contains download, as well as *GitBash*, a special terminal making it easier to use Git

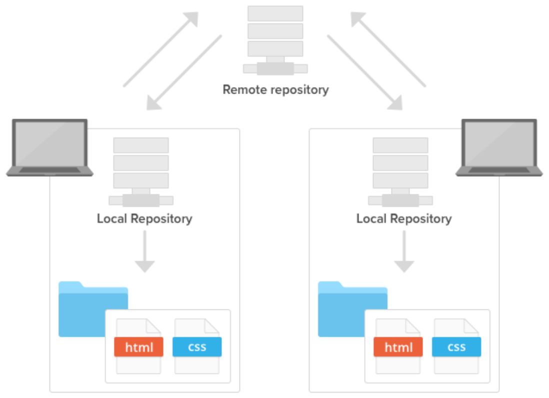
What is GitHub?

GitHub is a software development platform

Three major aspects of GitHub:

1. Project organization and version control
2. Collaboration between programmers
3. Self-promotion

How does GitHub improve project organization?



GitHub organizes project files in a **repository**

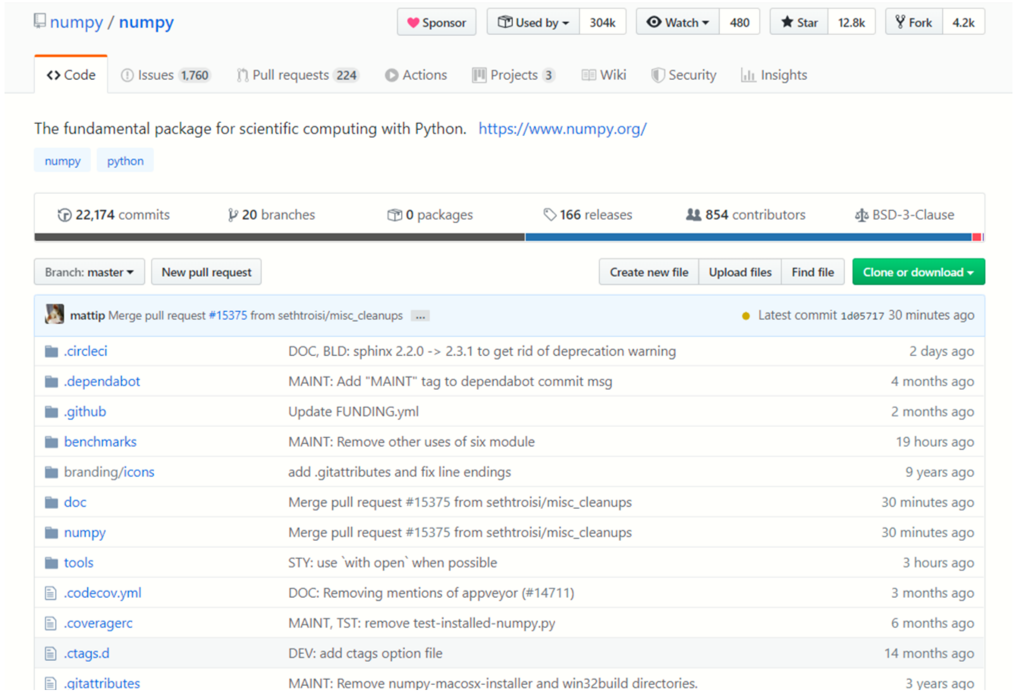
- Stores all files and folders related to project (.py, README.md, etc.)
- Repository is typically stored *remotely*
- As you program, you periodically update the repository with your progress (“push” changes)

Example of a repository

Numpy:
https://github.com/numpy/numpy

GitHub provides many tools for managing repositories

Repositories are created and managed using **Git**.



The fundamental package for scientific computing with Python. https://www.numpy.org/

22,174 commits · 20 branches · 0 packages · 166 releases · 854 contributors · BSD-3-Clause

mattip Merge pull request #15375 from sethtroisi/misc_cleanups … Latest commit 1d05717 30 minutes ago

.circleci	DOC, BLD: sphinx 2.2.0 -> 2.3.1 to get rid of deprecation warning	2 days ago
.dependabot	MAINT: Add "MAINT" tag to dependabot commit msg	4 months ago
.github	Update FUNDING.yml	2 months ago
benchmarks	MAINT: Remove other uses of six module	19 hours ago
branding/icons	add .gitattributes and fix line endings	9 years ago
doc	Merge pull request #15375 from sethtroisi/misc_cleanups	30 minutes ago
numpy	Merge pull request #15375 from sethtroisi/misc_cleanups	30 minutes ago
tools	STY: use `with open` when possible	3 hours ago
.codecov.yml	DOC: Removing mentions of appveyor (#14711)	3 months ago
.coveragerc	MAINT, TST: remove test-installed-numpy.py	6 months ago
.ctags.d	DEV: add ctags option file	14 months ago
.gitattributes	MAINT: Remove numpy-macosx-installer and win32build directories.	3 years ago

What is Git?

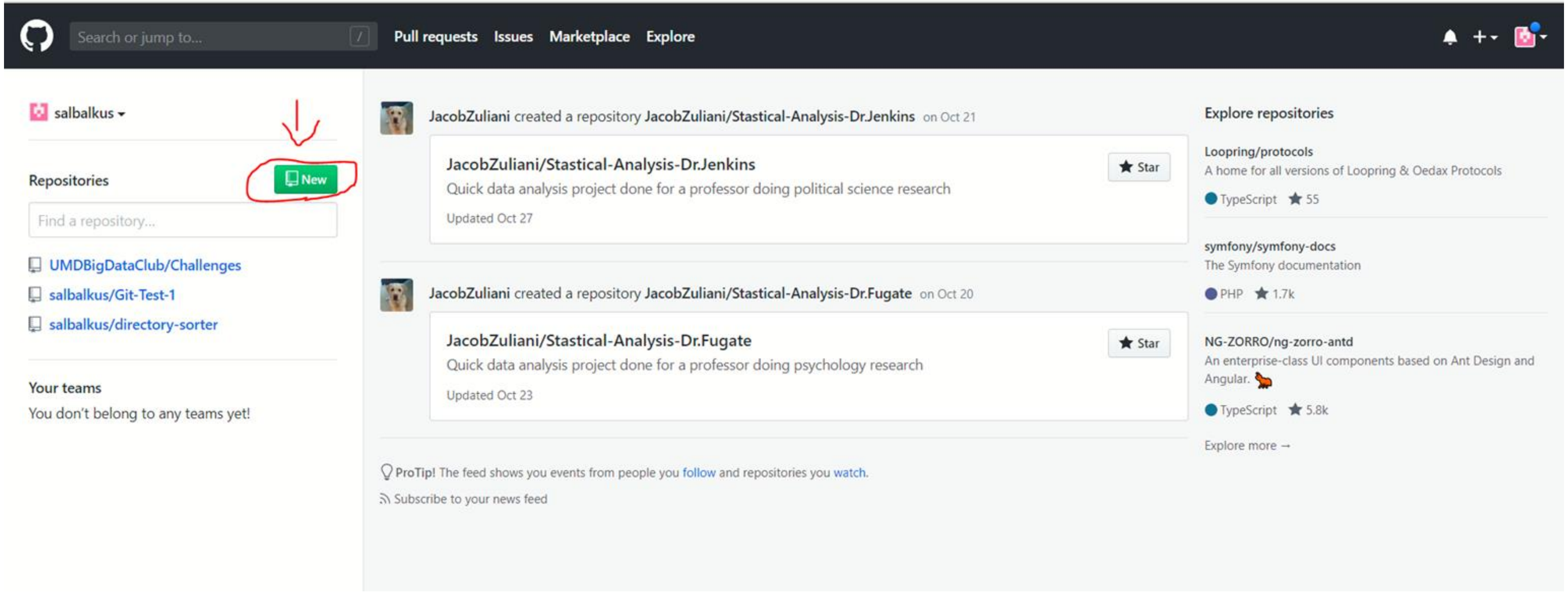
Git is the most commonly used *Version Control System* (VCS)

- Tracks changes made to a repository
- Create branches to work on different features
- Allows collaboration between developers
- Manage project using **command line**

How do I set up my own repository on GitHub?

Create from GitHub (Recommended Method)

First, go to your dashboard and select “new”



UMDBigDataClub/Challenges
salbalkus/Git-Test-1
salbalkus/directory-sorter

Your teams
You don't belong to any teams yet!

JacobZuliani created a repository JacobZuliani/Stastical-Analysis-Dr.Jenkins on Oct 21

JacobZuliani/Stastical-Analysis-Dr.Jenkins
Quick data analysis project done for a professor doing political science research
Updated Oct 27

JacobZuliani created a repository JacobZuliani/Stastical-Analysis-Dr.Fugate on Oct 20

JacobZuliani/Stastical-Analysis-Dr.Fugate
Quick data analysis project done for a professor doing psychology research
Updated Oct 23

ProTip! The feed shows you events from people you follow and repositories you watch.
Subscribe to your news feed

Loopring/protocols
A home for all versions of Loopring & Oedax Protocols
TypeScript ★ 55

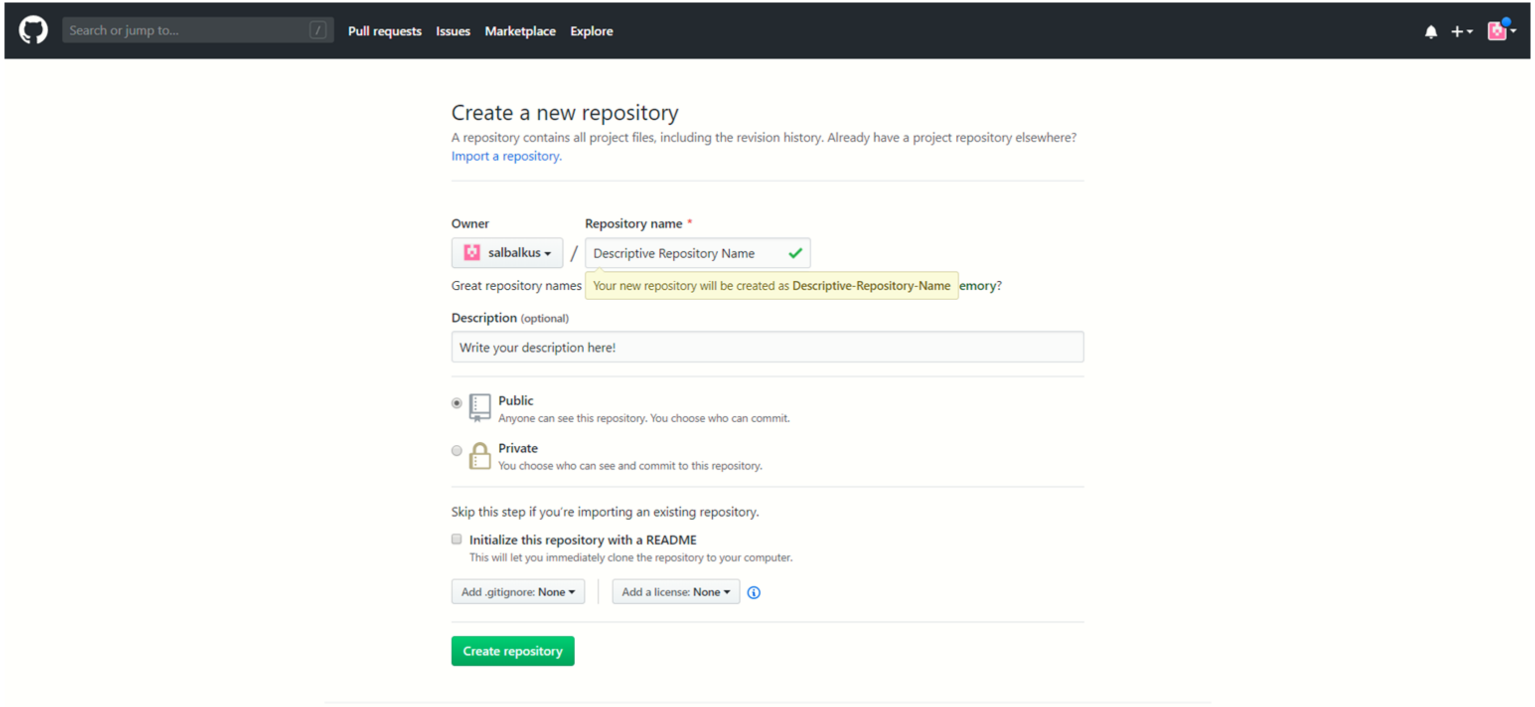
symfony/symfony-docs
The Symfony documentation
PHP ★ 1.7k

NG-ZORRO/ng-zorro-antd
An enterprise-class UI components based on Ant Design and Angular.
TypeScript ★ 5.8k

How do I set up my own repository on GitHub?

Create from GitHub

Next, fill out the required details



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner: salbalkus / Repository name *: Descriptive Repository Name

Great repository names Your new repository will be created as Descriptive-Repository-Name emory?

Description (optional): Write your description here!

[✓ Public] Anyone can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

Initialize this repository with a README
This will let you immediately clone the repository to your computer.

Add .gitignore: None | Add a license: None

Create repository

How do I set up my own repository on GitHub?

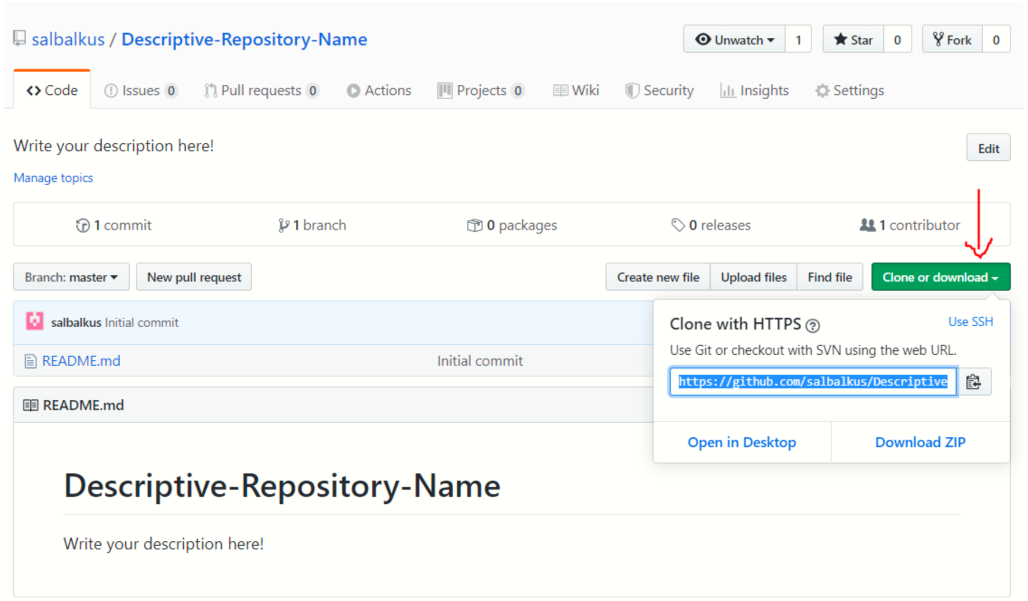
Create from GitHub

Click “clone or download” to obtain URL

Navigate to where you want the repository to be stored locally and run the command

```
git clone "url"
```

Can also clone a team’s repository



Write your description here!

salbalkus Initial commit

README.md Initial commit

Clone with HTTPS

Use Git or checkout with SVN using the web URL.

https://github.com/salbalkus/Descriptive

README.md

Descriptive-Repository-Name

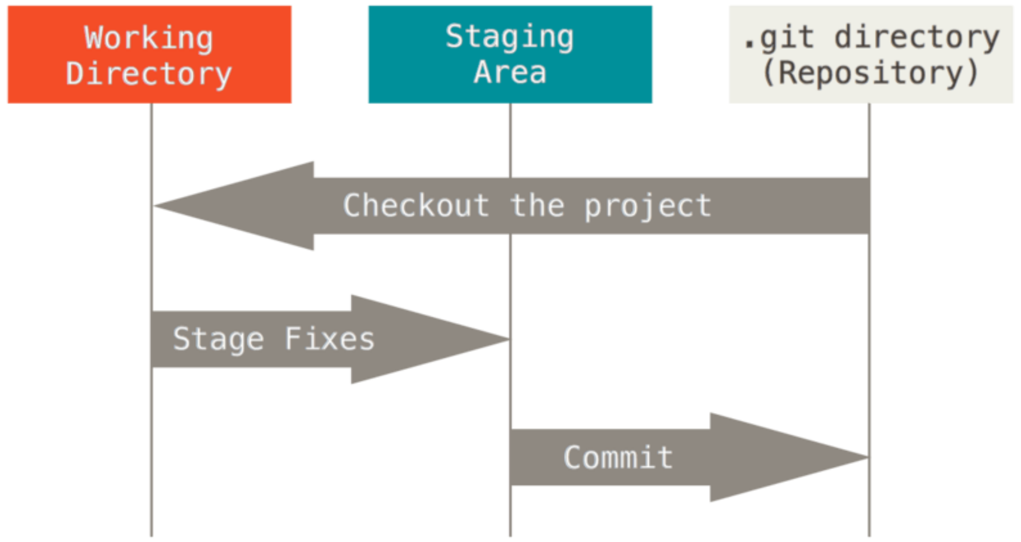
Write your description here!

How does Git track changes?

Git File Stages (Three Types):

Git stores changes to a repository as *snapshots*

- serves as a “mini filesystem”
- entire project stored locally in .git directory
- Adding to repository called “**committing**”



The Git Workflow

When you make changes to your project locally, you update your remote repository with three commands:

`git add “filename”` OR **`git add “foldername”`** → tells Git that this is the part of your local work that you want to update (staging)

`git commit -m “commit message”` → commits the changes you have staged

`git push` → push the local commits to the remote repository

Obtaining remote changes

If others add changes to the remote repository, you will need to update your own.

`git pull` → commit remote changes after your own (do not use if you have uncommitted changes)

`git pull --rebase` → takes remote changes and commits BEFORE your own; ensures your changes are committed after

`git fetch` → only updates repository, not working files; use to avoid overwriting your changes

Diagnostic Commands

Useful commands for tracking repository

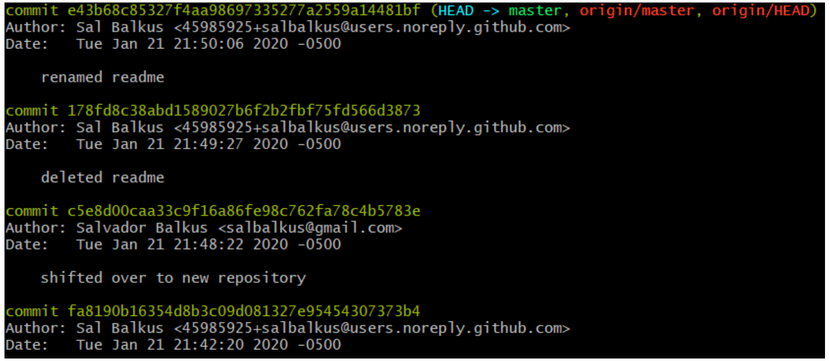
`git status` → provides some information about the changes currently staged

`git diff` → shows difference between remote and your own

`git --help` → lists all available git commands in case you forget

Git provides a vast array of tools to manage repositories and track changes; if there is something specific you want to do, search the documentation.

Accessing Past Changes


```
commit e43b68c85327f4aa98697335277a2559a14481bf (HEAD -> master, origin/master, origin/HEAD)
Author: Sal Balkus <45985925+salbalkus@users.noreply.github.com>
Date:   Tue Jan 21 21:50:06 2020 -0500

    renamed readme

commit 178fd8c38abd1589027b6f2b2fbf75fd566d3873
Author: Sal Balkus <45985925+salbalkus@users.noreply.github.com>
Date:   Tue Jan 21 21:49:27 2020 -0500

    deleted readme

commit c5e8d00caa33c9f16a86fe98c762fa78c4b5783e
Author: Salvador Balkus <salbalkus@gmail.com>
Date:   Tue Jan 21 21:48:22 2020 -0500

    shifted over to new repository

commit fa8190b16354d8b3c09d081327e95454307373b4
Author: Sal Balkus <45985925+salbalkus@users.noreply.github.com>
Date:   Tue Jan 21 21:42:20 2020 -0500
```

Using Git, you can examine commits that have been made and work with them.

`git log` → displays commits

`git checkout “commit”` → access files from a previous commit

`git checkout -- “`*`file name`*`”` → obtain old version of a specific file

`git tag “tag name”` → labels a commit so you can access it more easily

Resetting Changes

Git provides many options for resetting changes:

`git rm “file name” -f` → remove a file from both working directory and repo

`git rm “file name” --cached` → remove a file from just repo

`git reset --hard “commit”` → roll back repository to the given commit

`git revert “commit” -m “message”` → re-commit a previous commit; use to target a specific commit to reset

Git: Branching

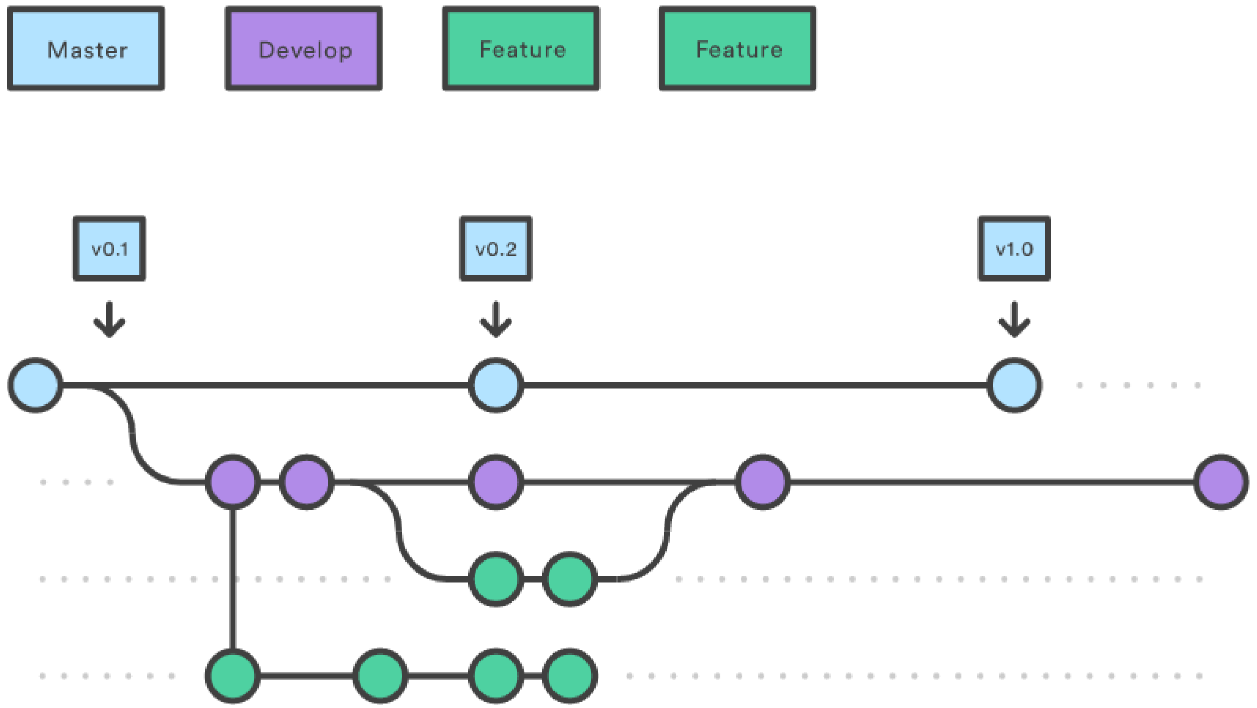
Suppose you want to develop or test a part of code separately and not have to worry about breaking existing code. This requires the use of **branching**.

`git checkout -b “branch name”` → creates a new “branch” where changes will not affect the master branch

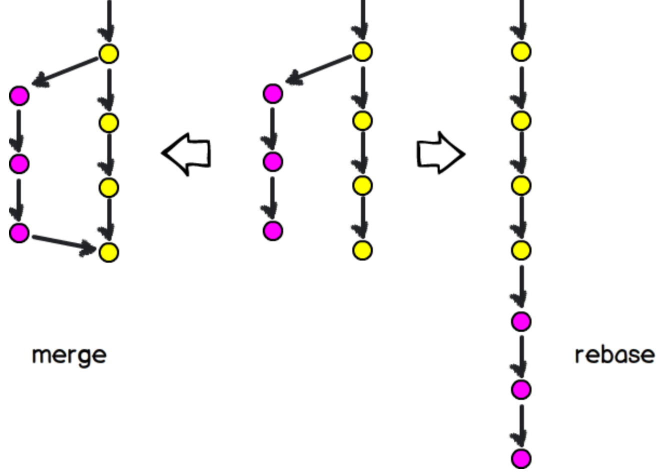
`git checkout “branch name”` → switch to an existing branch

`git branch` → lists current branches

Git: Branching



Git: Branching



Once you've finalized the code you are working on, you will want to combine the branch with master.

`git merge "branch name"` → merges the specified branch with the branch you are currently on.

`git rebase "branch name"` → moves the specified branch to the end of another branch.

Other Notes About Git

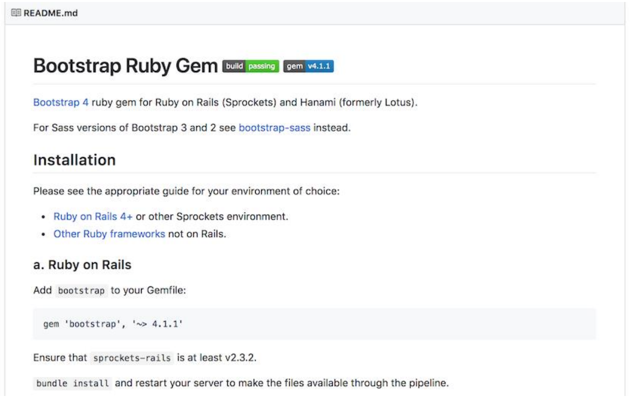
In addition to the commands discussed, Git offers a variety of tools for managing repositories and branches. Please see the documentation:

https://git-scm.com/doc

Most of your work will consist of *add, commit, push*

- branches used for large projects with different features OR collaboration
- **do not push directly to master!** (in general)

Some good practices


README.md

Bootstrap Ruby Gem build passing gem v4.1.1

Bootstrap 4 ruby gem for Ruby on Rails (Sprockets) and Hanami (formerly Lotus).

For Sass versions of Bootstrap 3 and 2 see bootstrap-sass instead.

Installation

Please see the appropriate guide for your environment of choice:

- Ruby on Rails 4+ or other Sprockets environment.
- Other Ruby frameworks not on Rails.

a. Ruby on Rails

Add `bootstrap` to your Gemfile:

```
gem 'bootstrap', '~> 4.1.1'
```

Ensure that `sprockets-rails` is at least v2.3.2.

`bundle install` and restart your server to make the files available through the pipeline.

- Include a README and other text files
  - explain purpose of project, contributors, how to use, etc.

- Consider including a license (see repo settings)
  - MIT license - allows reuse with citation, but retains your copyright

- Avoid pushing private data, changing files, or “auxiliary” files to repo
  - Data generally not pushed; excessive storage, may change
  - Do not want files specific to your machine (ex: .pynb checkpoints)
  - use **.gitignore** file

.gitignore File

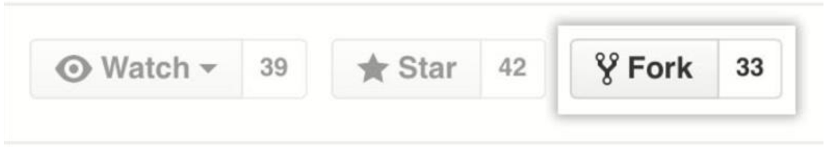
Defines what will NOT be pushed to the git repository:

- byproduct/auxiliary files; these are specific to your machine
- Data files that you do not want to push
- LaTeX intermediates

Uses patterns to define what files NOT to add; see how to define: https://git-scm.com/docs/gitignore

Or copy one from a compilation: https://github.com/github/gitignore

Open Source Project Models



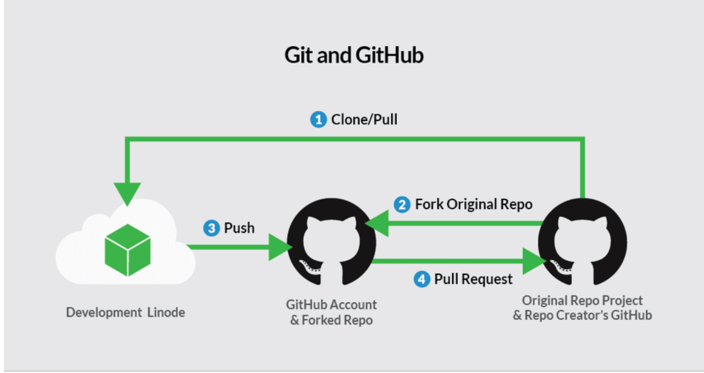
Shared Repository: Administrators explicitly designate certain individuals or teams as contributors

- Can give different individuals different access (read, write, admin)

Fork and Pull: allows anyone to contribute to a project

- Individuals "fork" a copy of the project and work in their individual fork
  - Puts the repository on your account
- Suggest additions via pull request
- Allows you to contribute to open-source projects!

How to make a pull request



Pull request: used to recommend and discuss changes

- Request merging a side branch with “upstream” branch such as master
  - Shows differences between branches
- Administrators can require pull requests to be reviewed

GitHub “pull requests” menu provides options for managing pull requests

- Gives users space to discuss changes
  - Once discussion complete, branches merged by admin

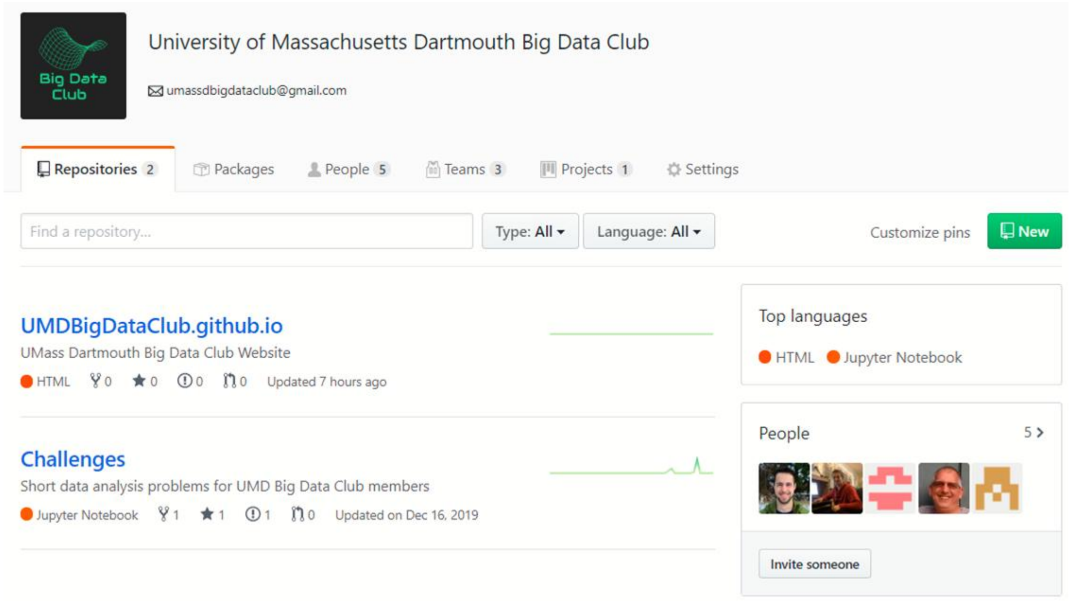
Organizations

Example: UMD Big Data Club

https://github.com/UMDBigDataClub

Organizations facilitate collaboration on GitHub

- Set up teams and projects to assign tasks
- Can create repos under the organization


University of Massachusetts Dartmouth Big Data Club

umassdbigdataclub@gmail.com

UMDBigDataClub.github.io

UMass Dartmouth Big Data Club Website

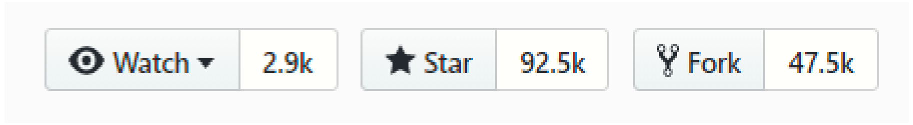
HTML 0 0 0 0 Updated 7 hours ago

Challenges

Short data analysis problems for UMD Big Data Club members

Jupyter Notebook 1 1 1 0 Updated on Dec 16, 2019

The Social Media Aspect



GitHub provides a variety of ways to interact with other users

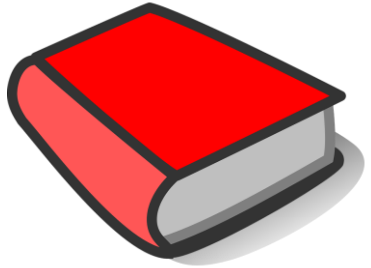
- **Watch** repositories to be notified of updates
- **Star** repositories you find useful
- **Follow** other users to be notified of their activities
- Create static **websites** using GitHub Pages

Your profile tracks where you are contributing, how often, and the statistics for your repositories you have created or forked

Other GitHub Features

GitHub offers many additional tools to manage repositories, including…

- **Automation** - set up “Actions” to manage repository and contributors automatically
- **Webhooks** - notify services when events occur
- **Issue Tracking** - allow others to report bugs
- **Wikis** - provide additional documentation



Any questions?