Core tools II

Lecture 03

Version control

- Manages changes to code over time
- Enables easier collaborative development
- Provides a complete history of every change made to every file

git

https://git-scm.com/

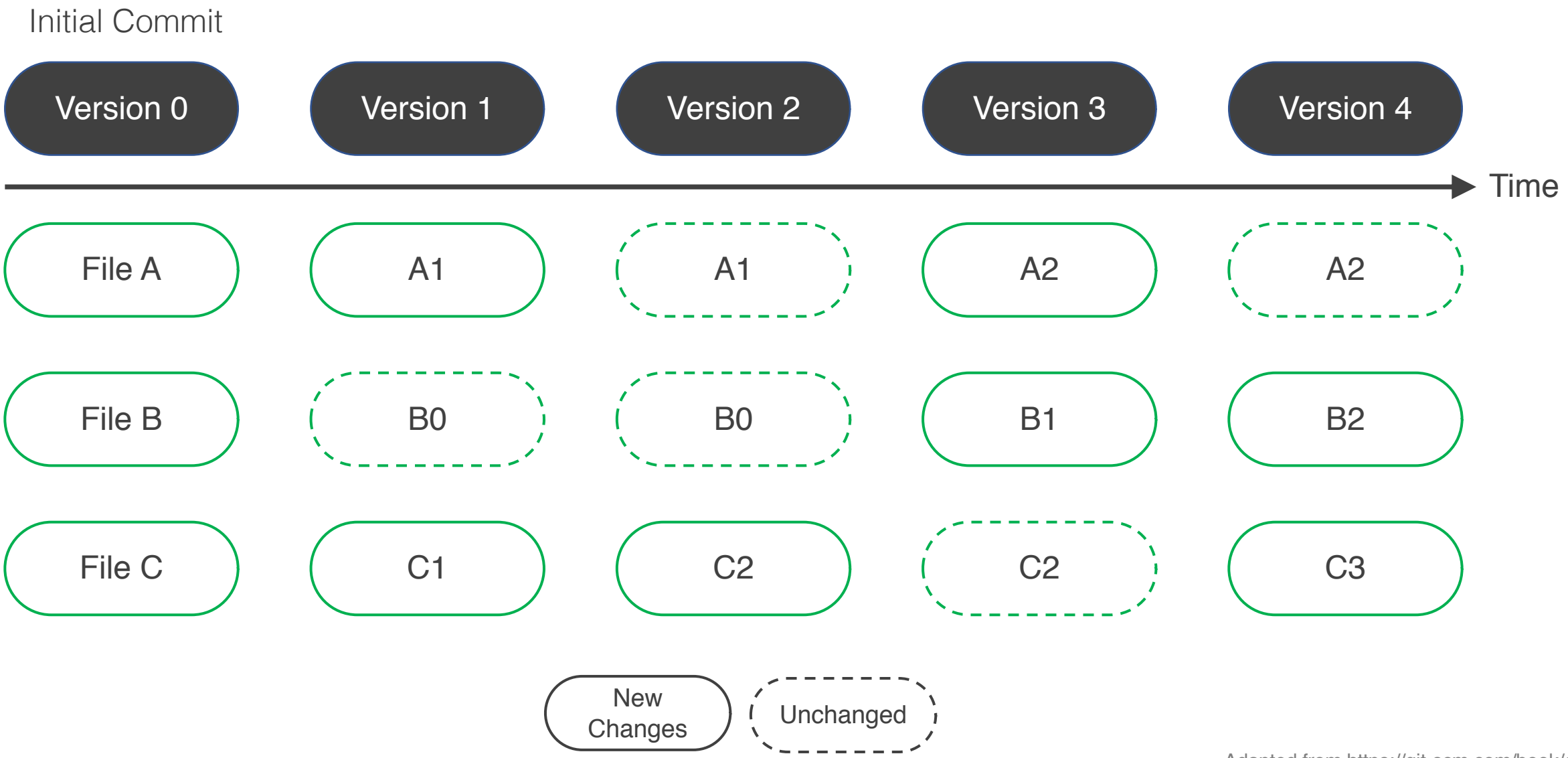
- Most used version control tool
- Distributed version control system
- Works across most platforms
- Isn't confused by file name changes
- **Creates and maintains repositories**

GitHub

https://github.com/

- Web-based hosting service for version control with git
- **Hosts git repositories on the web**

Version control with Git



Adapted from https://git-scm.com/book/en/v2

How Git works

Local | Remote

Working Directory

A directory on your local file system

Staging Area

Tell Git what you want to commit to the repository and what you do not

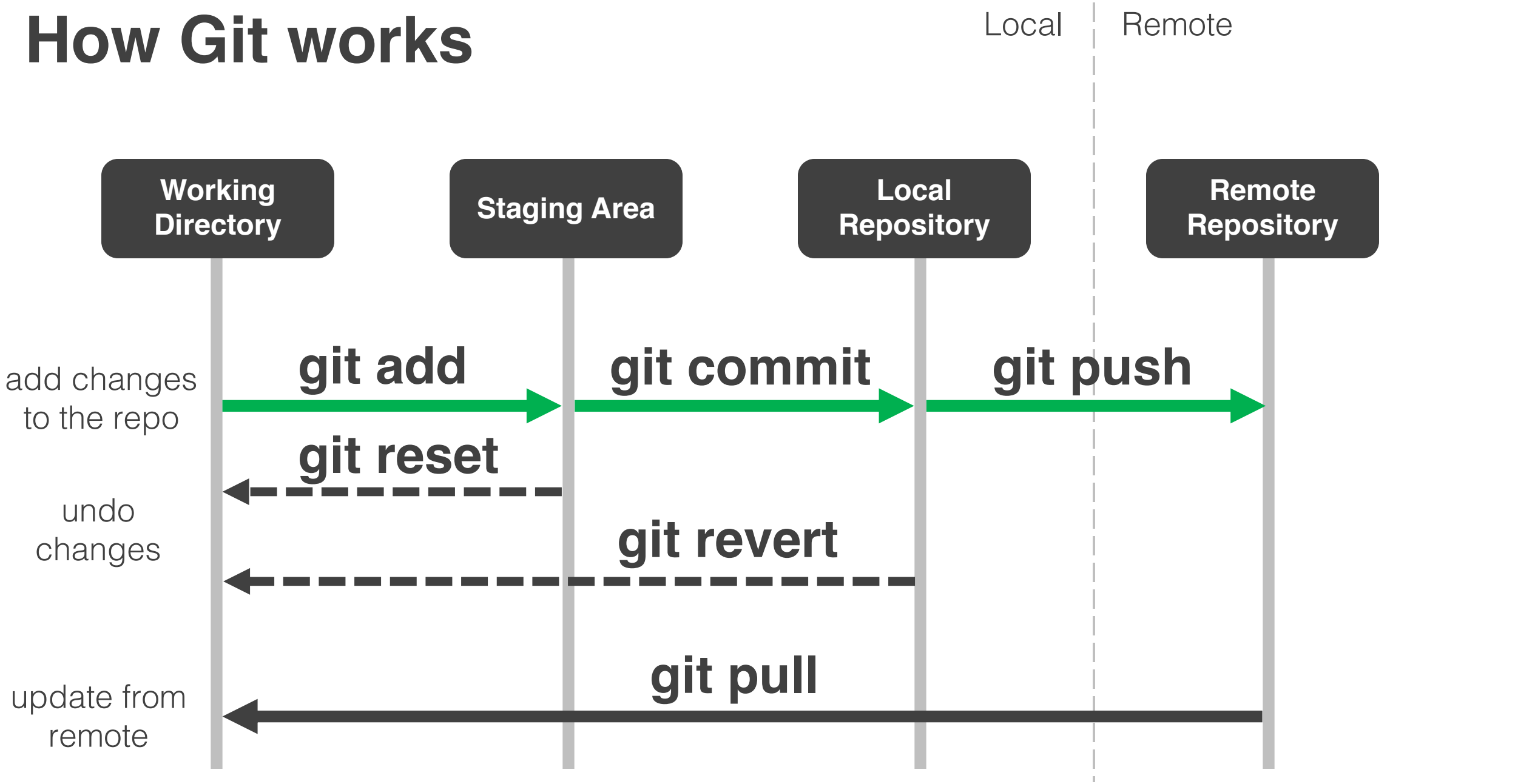
Local Repository

The repository on your local system

Remote Repository

A remote repository, for example, **Github**

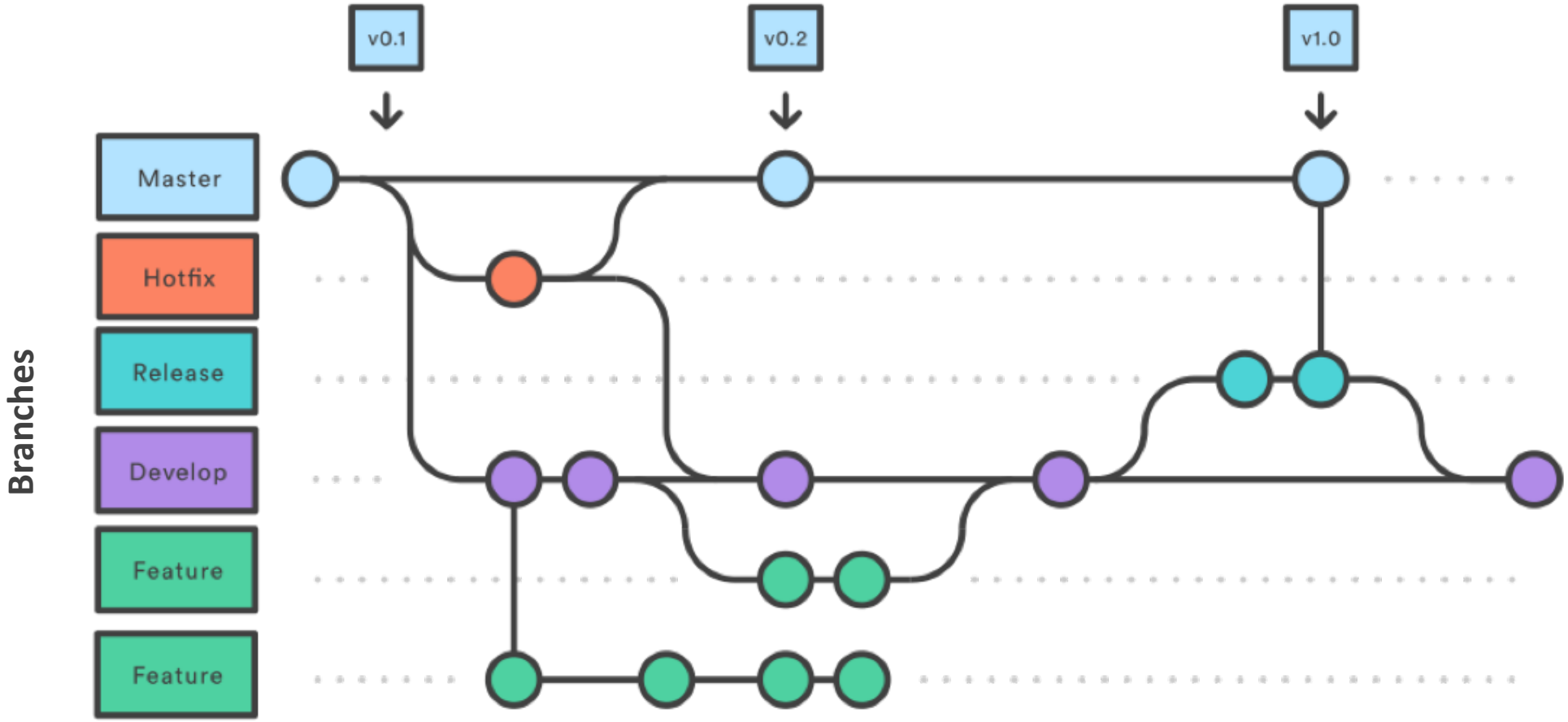
How Git works



Adapted from https://git-scm.com/book/en/v2

Workflow

The Gitflow workflow model is by Vincent Driessen



Modified from https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

Getting started

1. install Git — https://git-scm.com/downloads

2. create a new local repo in a local directory

   git init — initialize a new repo in a directory

   OR

   git clone — initialize a new local repo from an existing remote repo such as one on Github

3. create your project and commit it to your repo — **git add, git commit, etc.**

4. share your work and/or collaborate — **git push** — transfer and sync your local repo with a remote repo, such as Github

git status

inspect the staging area

When there are no changes

On branch master
nothing to commit, working tree clean

When there are changes

On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

These have been added with the git add command

new file: filea.txt
modified: myfilechanged.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)

These have not yet been added to the staging area

anotherfile.txt
fileb.txt

git log

history of commits

```
commit 4e7d29f337f1678624acd253007877a7fe7d41db
Merge: ad7f3b1 38485bb
Author: Kyle Bradbury kjb17@duke.edu
Date:   Tue Jan 23 16:34:23 2018 -0500

    merged dev into master

commit
38485bb1690e821f6d8ed02206bb1504c7a4bf9b
Author: Kyle Bradbury kjb17@duke.edu
Date:   Tue Jan 23 15:29:51 2018 -0500

    changed file name

commit 8914856cffd5a910ef9c7a2ae7a0cc9e4c796889
Author: Kyle Bradbury kjb17@duke.edu
Date:   Tue Jan 23 15:13:47 2018 -0500

    initial commit
```

git diff

inspecting differences

myfile.txt
(on branch master)

```
1  I have some text in myfile.
2
3  This
4  is
5  the
6  text
7  I
8  did not
9  write.
10
11 # No more comments
```

myfilechanged.txt
(on branch dev)

```
1  I have some text in myfile.
2
3  This
4  is
5  the
6  text
7  I
8  never
9  wrote.
10
11 # No comments but my own
```

git diff master dev

```
diff --git a/myfile.txt b/myfilechanged.txt
similarity index 56%
rename from myfile.txt
rename to myfilechanged.txt
index 847c142..1716537 100644
--- a/myfile.txt
+++ b/myfilechanged.txt
@@ -5,7 +5,7 @@ is
 the
 text
 I
-did not
-write.
+never
+wrote.

-# No more comments
+# No comments but my own
```

- Comparing files a/b
- File names were different, but Git knew these were the same files
- File metadata
- File markers to indicate differences
- File chunk header
  @@ [file a range] [file b range] @@
  where each range is given by:
  -<start line>,<number of lines>
- Identical text in both a and b
- Differences only in file a
- Differences only in file b

Demo