# Week 6: Git Merge Conflict and Branches

Computational Tools and Techniques in STEM

Mar 5-7, 2019

# Learning Goals

- **L1**: Resolving merge conflicts.
- **L2**: Forking a Git branch.
- **L2**: Switching between branches.
- **L3**: Merging the branches.

# Git Branches

To see the exisiting branches and current branch:

```
$git branch
```

To create a new branch:

```
$git branch <branchname>
```

To checkout a branch:

```
$git checkout <branchname>
```

To delete a branch:

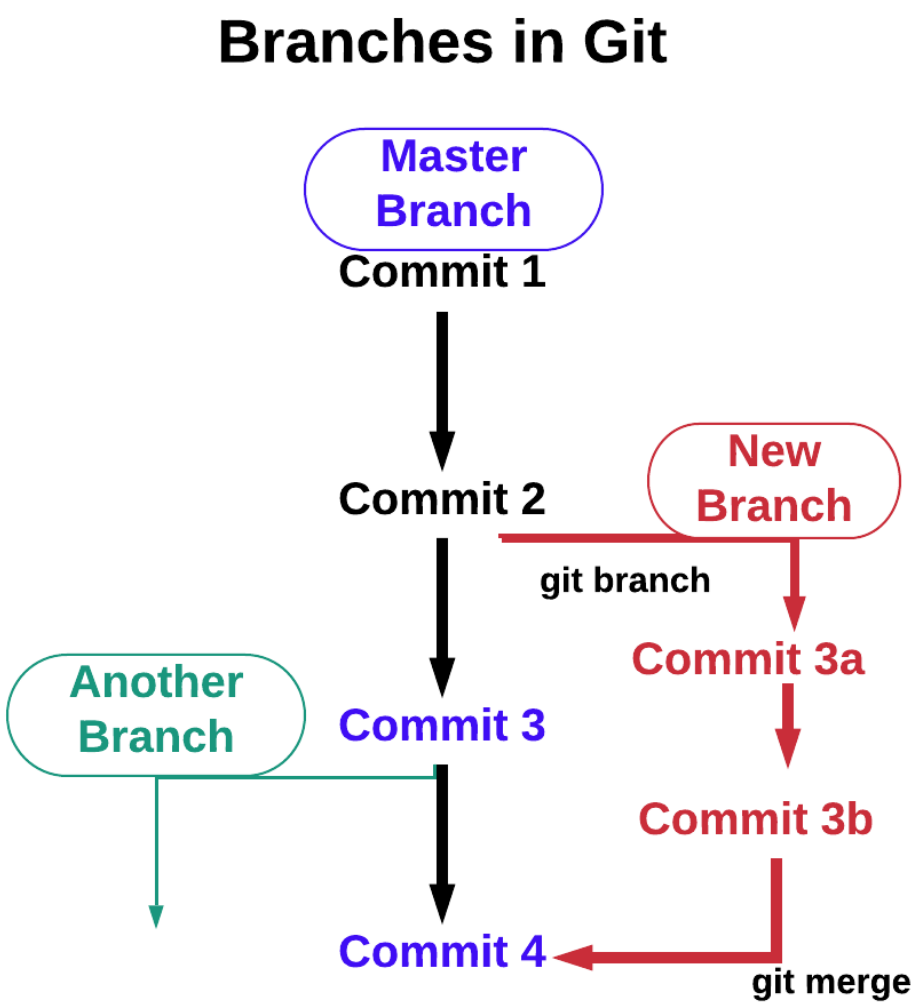
```
$git branch -d <branchname>
```

To merge a branch into a receiving branch, first go to the receiving branch:

```
$git checkout <receivingbranch>
```

Then, merge

```
$git merge <mergingbranch>
```



# Git Fetch and Merge

**git pull = git fetch + git merge**

## What does git fetch do?

It gets the updates from the remote, but does not update the working copy.

## What does git merge do?

Merges the changes from the remote into the local branch. This is the step at which the famous “merge conflicts” arise! Sometimes the changes have to be incorporated by hand.

## Syntax

```
$git fetch origin master

$git merge origin master
```

# Terminology

**What does origin refer to?**

It refers to the remote repo, from which the local repo originated.

**What does master refer to?**

Master is the default branch of any repo. Every repo starts with a master branch but, it's just a naming convention.

**How to explicitly refer to a particular branch in remote repo?**

`origin/<remotebranchname>`

**Example**

```
git push origin/<remotebranchname> master
```

# More Concepts and Commands

**What is HEAD?**

It refers to the latest commit on the current branch (but it can be made to point to any commit!).

**How to see all the branches?**

```
$git branch -a
```

**How to see all the remote branches?**

```
$git branch -r
```