CS 35L
Software Construction Laboratory

Lecture 10.1

12th March, 2019

Logistics

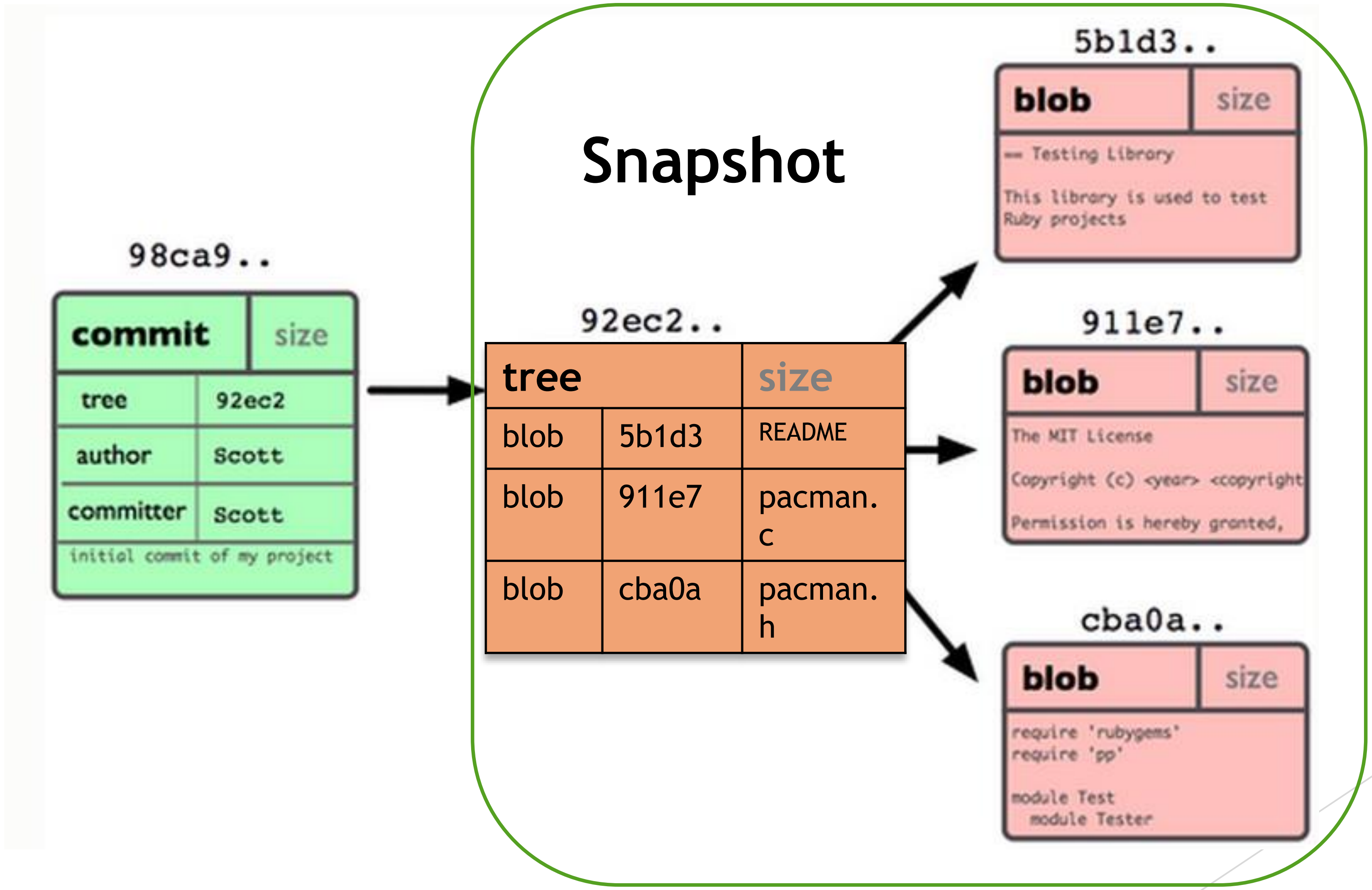
- Final Exam
  - Date: 17th March, 2019 (Sunday)
  - Time: 3pm to 6pm
  - Location: Franz 1178
- Presentations for Assignment 10
  - https://docs.google.com/spreadsheets/d/1o6r6CKCaB2du3klPflHiquymhBvbn7oP0wkHHMz_q1E/edit?usp=sharing
- Assignment 8 is due on *12th March, 2018* at 11:55pm
  - Limited Late Policy – Up till 15th March, 2018 only
- Assignment 9 is due on 15th March, 2018 at 11:55pm
  - NO Late Submissions accepted
- Instructor Evaluation

Review - Previous Lab

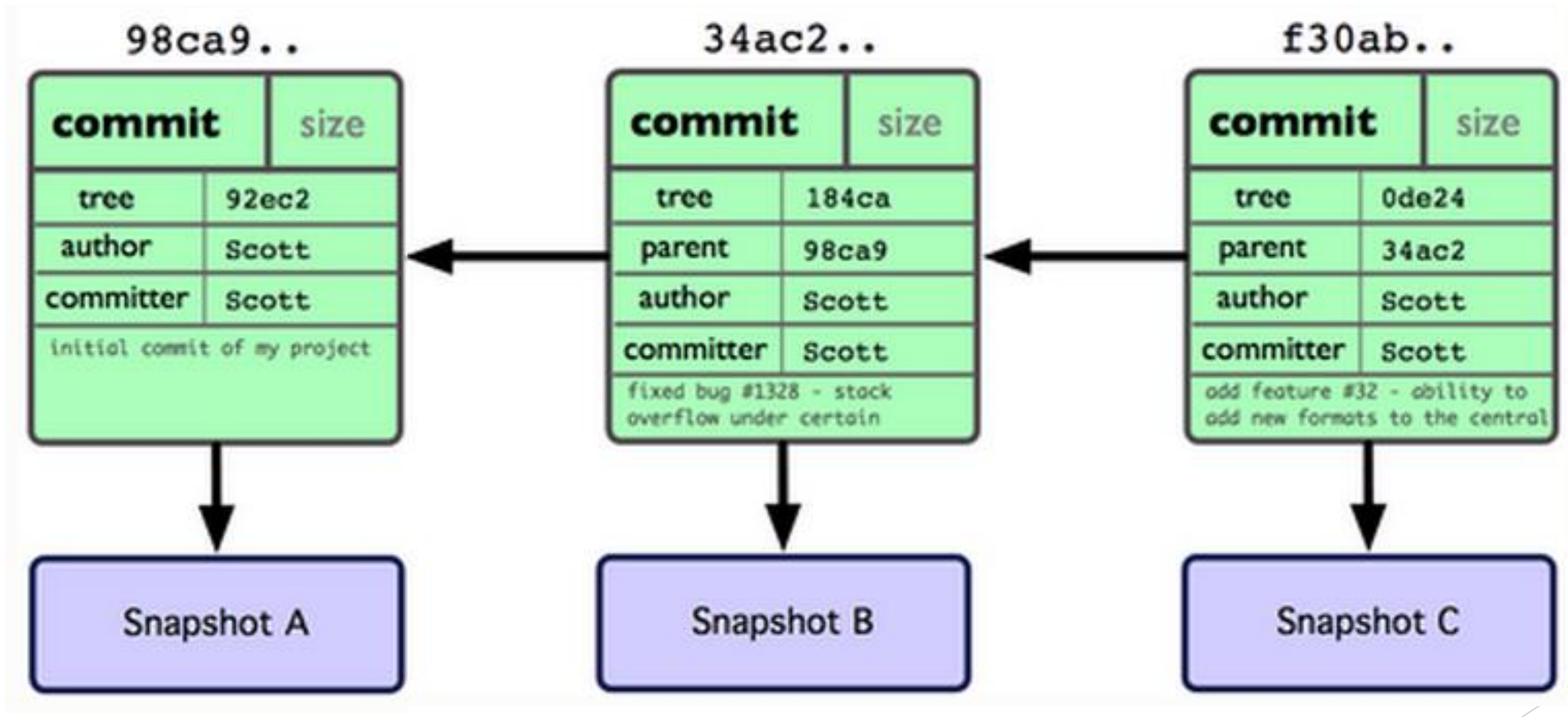
- Change Management
  - Why change management?
- Version Control System
  - Local, Centralized and Distributed
- Git
  - Git Repository Objects
    - Blobs, Trees, Commits, Tags
  - Git States
    - Working directory, staging area, git directory
- Initial git commands

GIT Source Control

Git Repo Structure



After 2 More Commits…



First Git Repository

- Mkdir gittest
- Cd gittest
- Git init
  - Creates an empty git repo (.git directory with all necessary directories)
- Echo “Hello World” > hello.txt
- Git add .
  - Adds content to the index
  - Must be run prior to a commit
- Git commit –m “First Commit”

Working with git

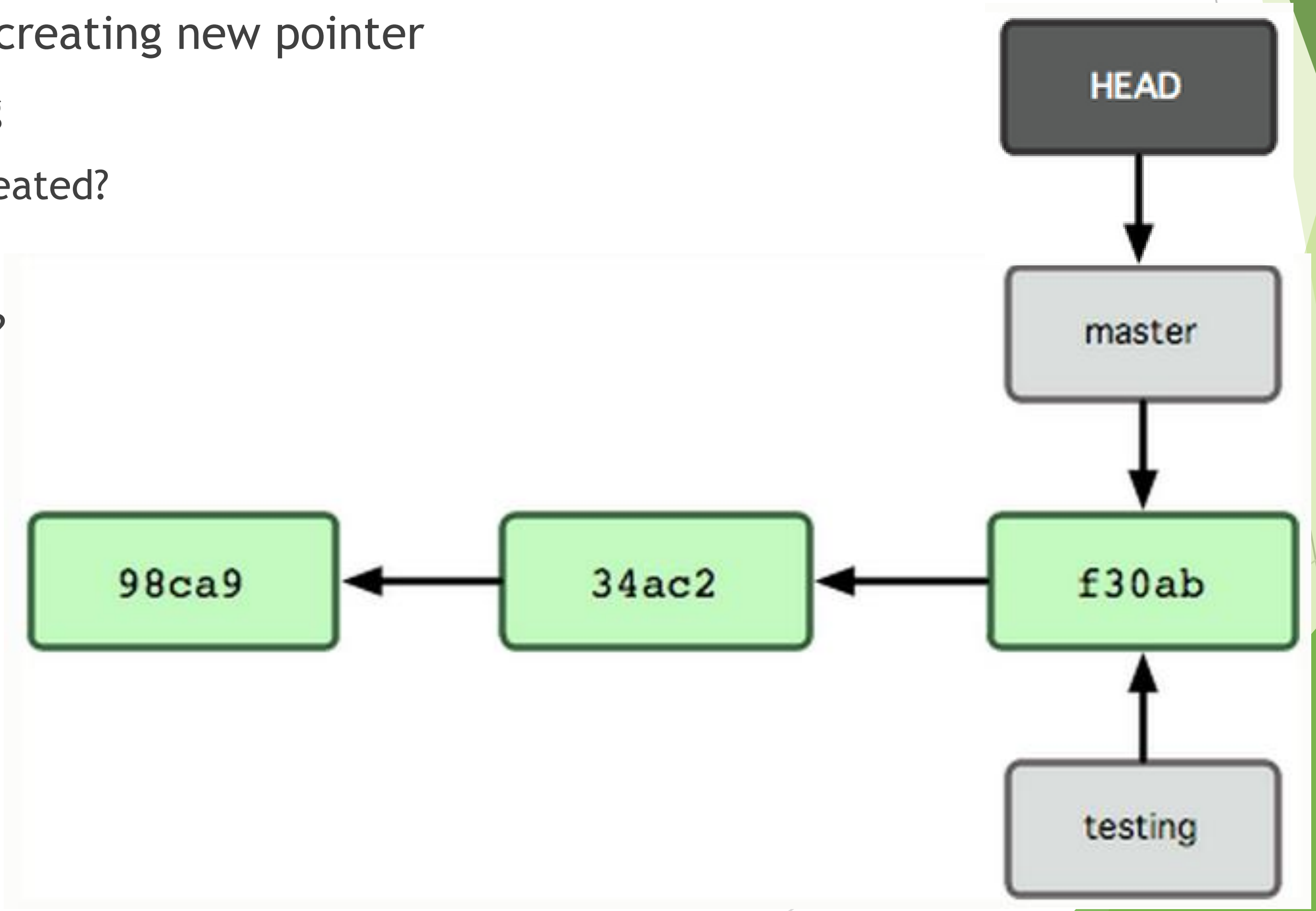
- Echo “I love git” >> hello.txt
- Git status
  - Shows list of modified files
- Git diff
  - Shows changes we made compared to the original index
- Git add hello.txt
- Git diff
- Git diff HEAD
- Git commit –m “Second commit”

What is a Branch?

- A pointer to one of the commits in the repo (head) + all ancestor commits
- When you first create a repo, are there any branches?
  - Default branch named ‘master’
- The default master branch
  - points to last commit made
  - moves forward automatically, every time you commit

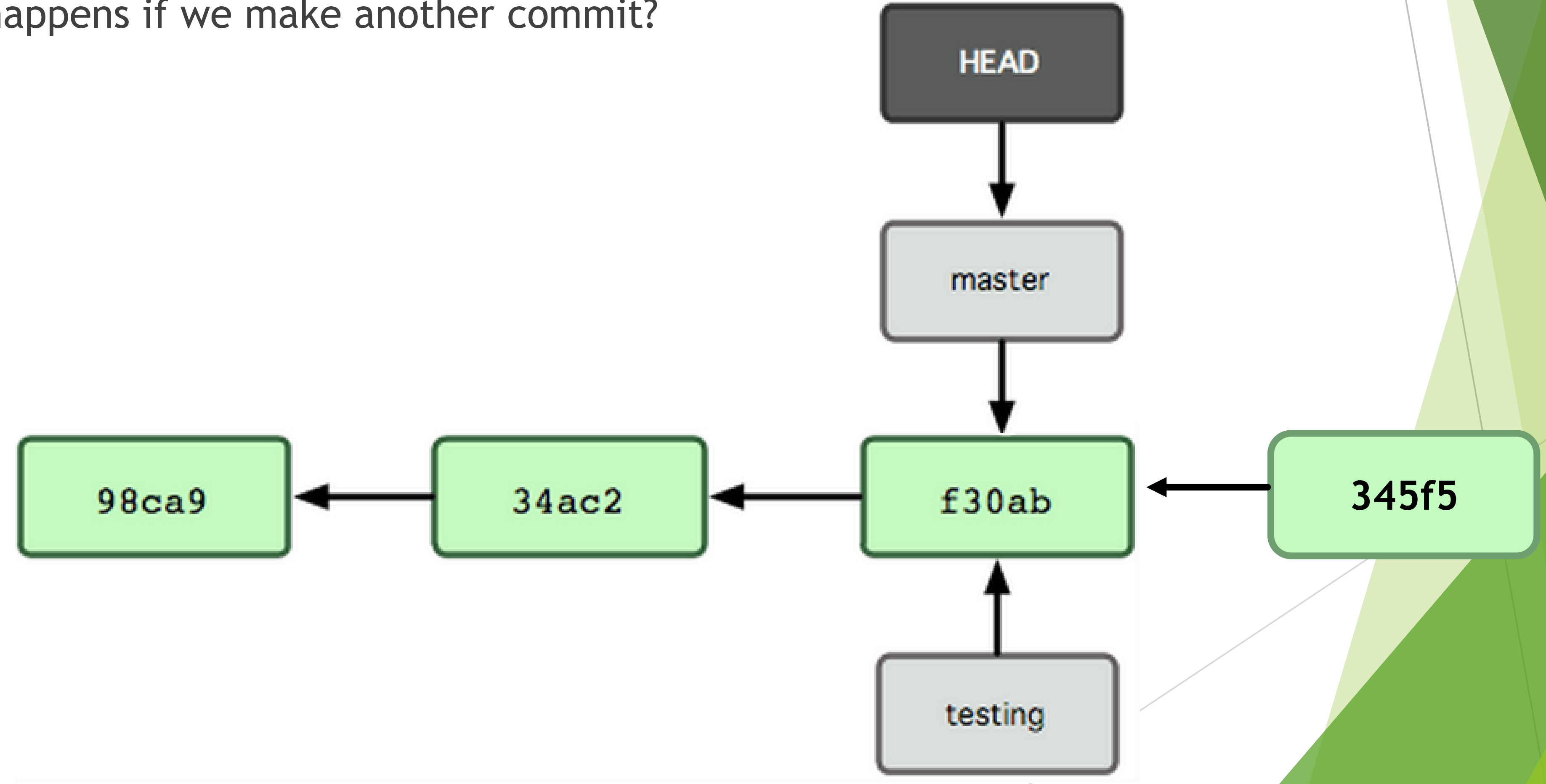
New Branch

- Creating a new branch = creating new pointer
  - `$ git branch` testing
  - Where is new branch created?
    - Current commit
- Where is current commit?
  - HEAD



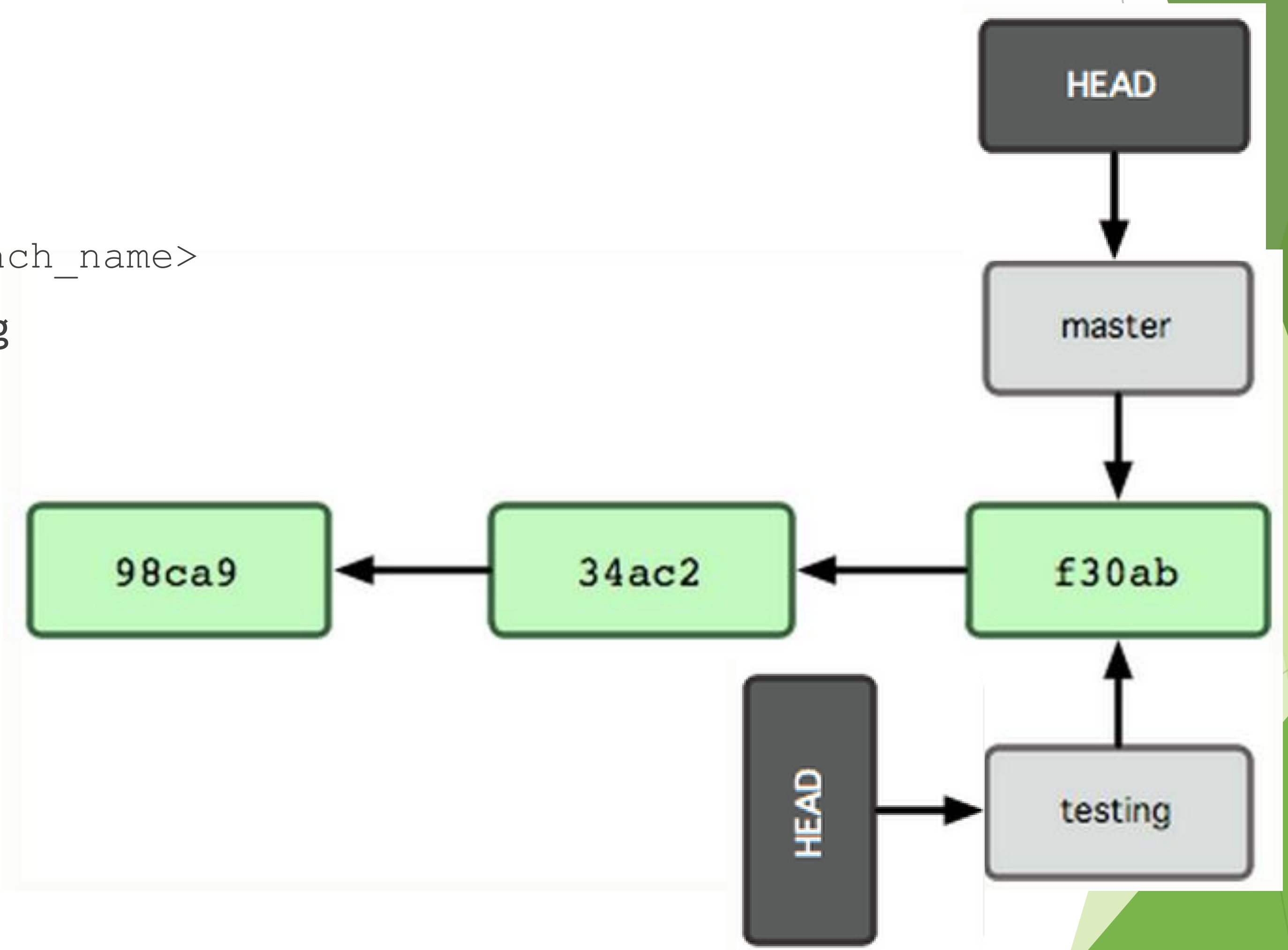
New Commit

- What happens if we make another commit?

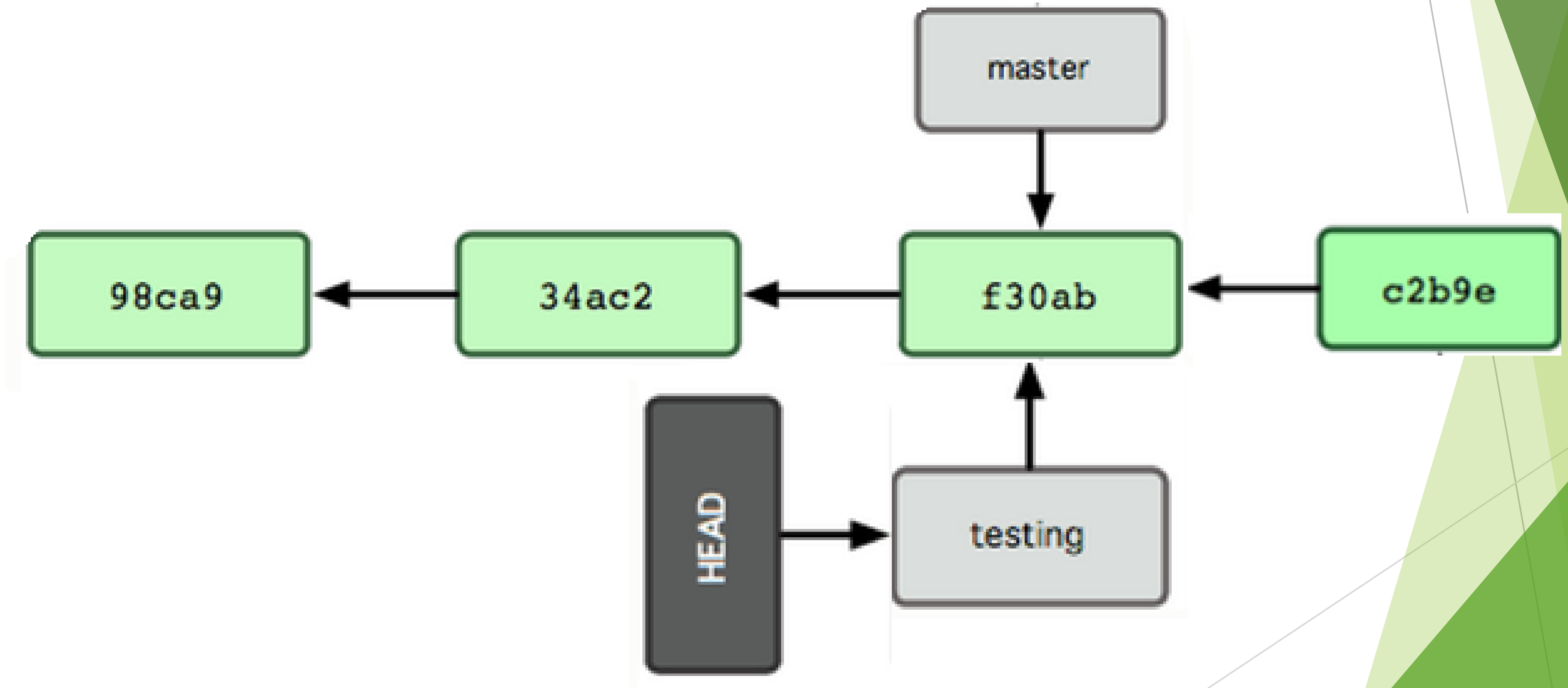


Switching to New Branch

- Check out new branch
  - `$ git checkout <branch_name>`
  - `$ git checkout testing`



Commit After Switch



Why Branching?

- Experiment with code without affecting main branch
- Separate projects that once had a common code base
- 2 versions of the project

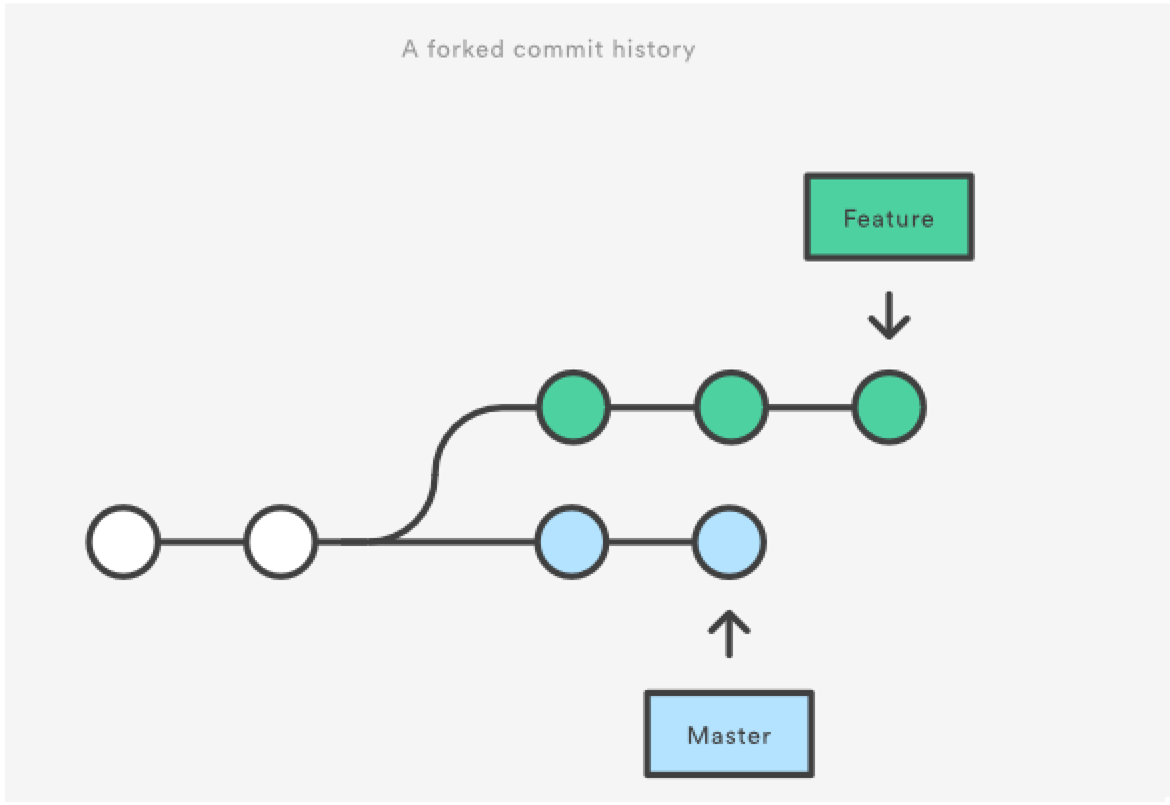
Working with branches

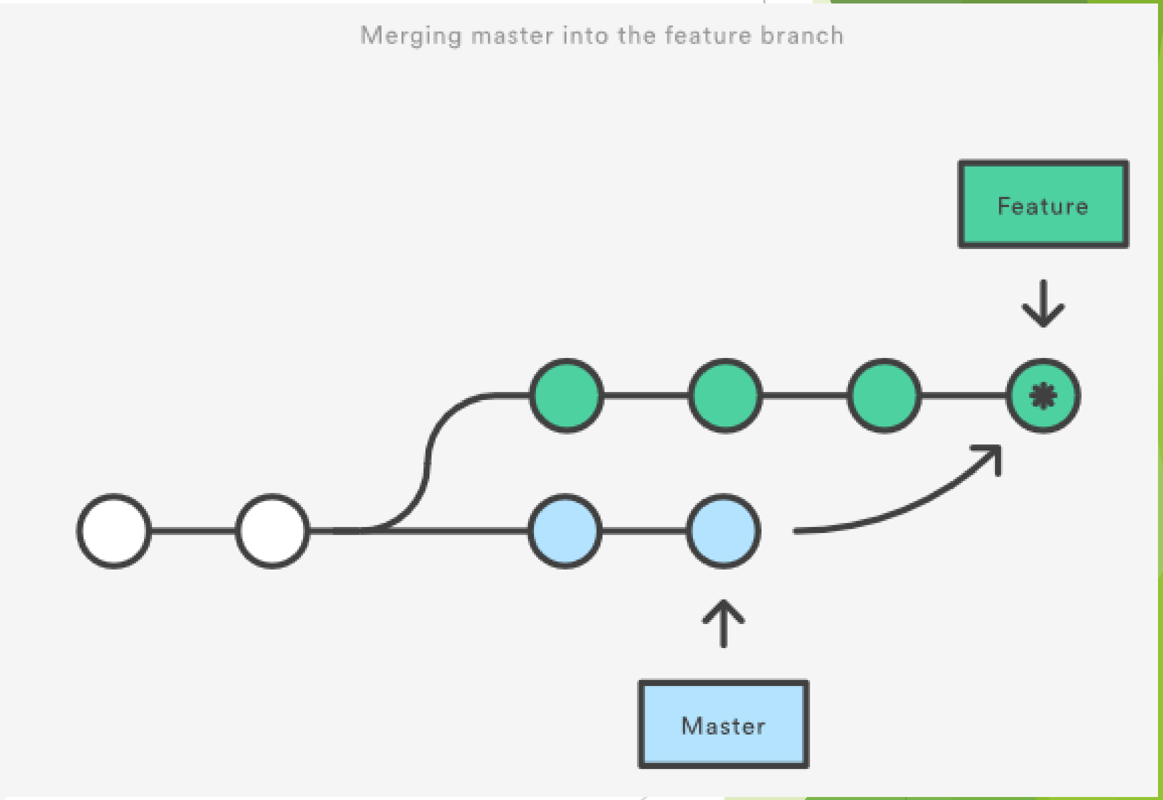
- Git branch test
  - Create new branch
- Git branch
  - List all branches
- Git checkout test
  - Switch to test branch
- Echo “Hello World!” > hw
- Commit the change in new branch
- Git checkout master
  - Back to master branch
- Git log
- Git merge test
  - Merge commits from test branch to current branch

Git integrating changes

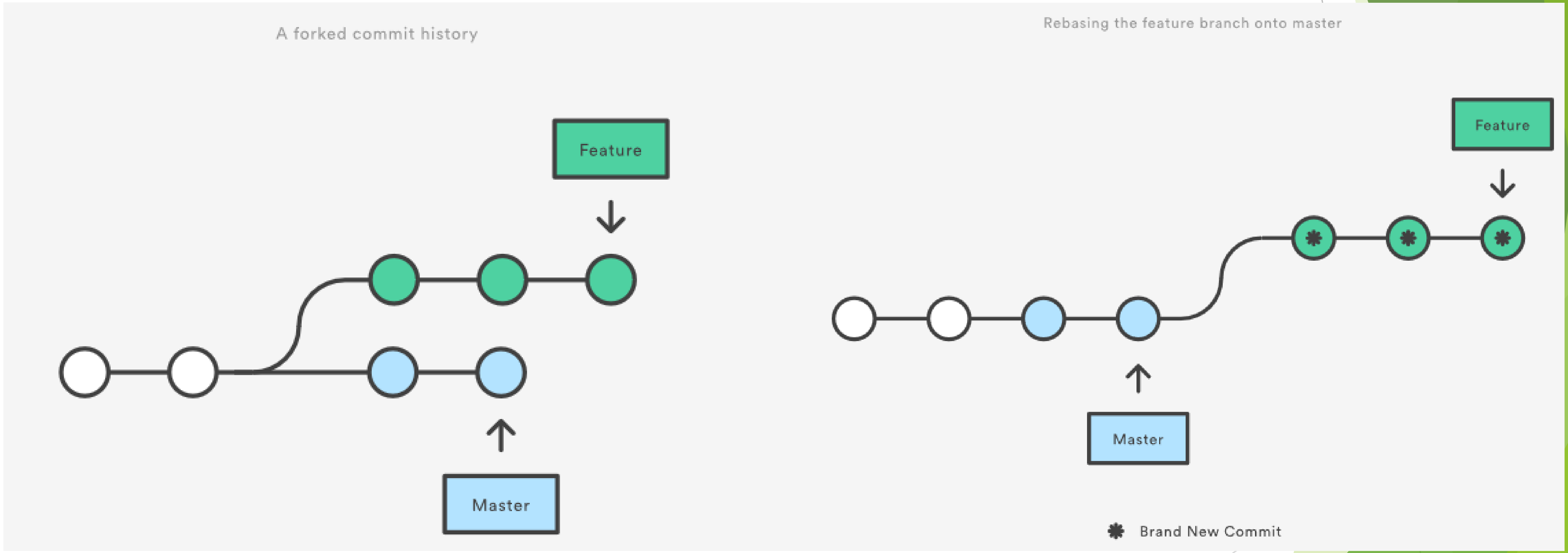
- Required when there are changes in multiple branches
- Two main ways to integrate changes from one branch to another
  - merge
  - rebase
- Merge is simple and straightforward
- Rebase is much cleaner

Git merge


A forked commit history
Feature
Master


Merging master into the feature branch
Feature
Master

Git rebase


A forked commit history
Feature
Master
Rebasing the feature branch onto master
Feature
Master
✱ Brand New Commit

Merge Conflicts

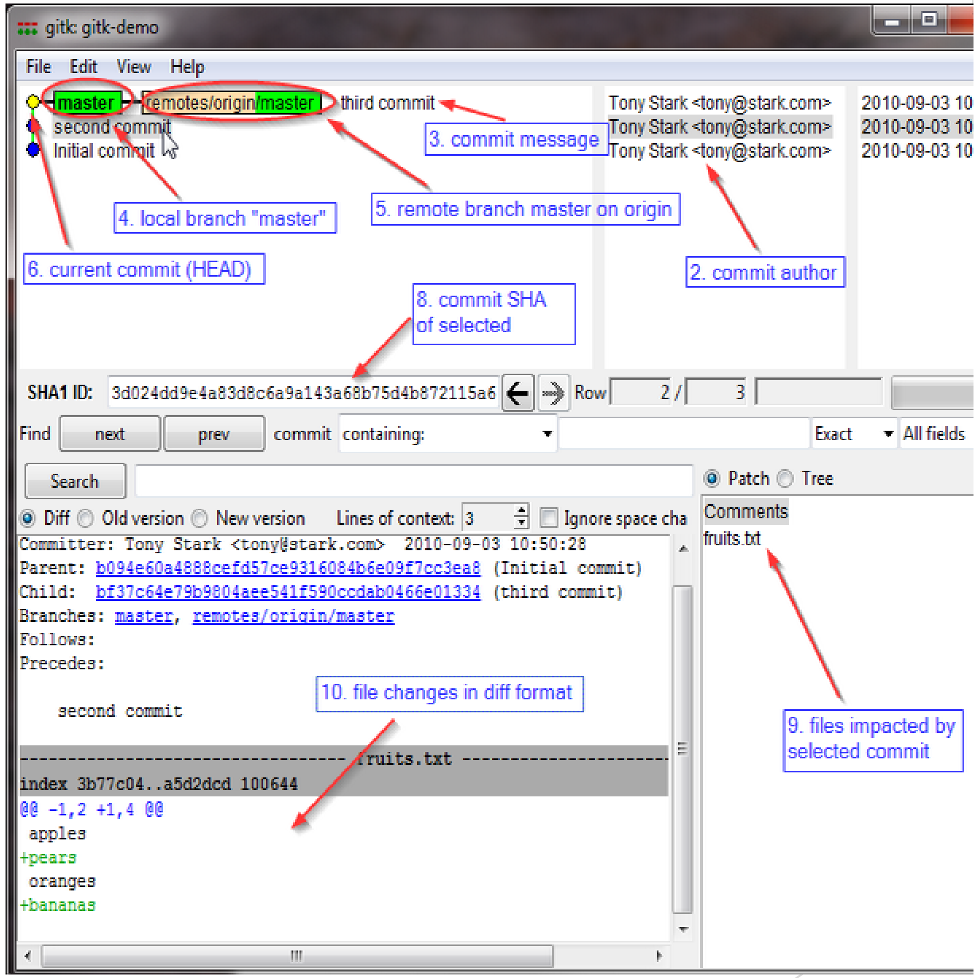
- Usually git will do merge automatically
- Conflict arises when you changed the same part of the same file differently in the two branches you’re merging together
- The new commit object will not be created
- You need to resolve conflicts manually by selecting which parts of the file you want to keep

More git commands

- Reverting
  - git checkout HEAD main.cpp
    - Gets the HEAD revision for the working copy
  - git checkout – main.cpp
    - Reverts changes in the working directory
  - git revert
    - Reverting commits (this creates new commits)
- Cleaning up untracked files
  - git clean
- Tagging
  - Human readable pointers to specific commits
  - git tag –a v1.0 –m ‘Version 1.0’
    - This will name the HEAD commit as v1.0

Gitk

- A repository browser
  - Visualizes commit graphs
  - Used to understand the structure of the repo
  - Tutorial: http://lostechies.com/joshuaflanagan/2010/09/03/use-gitk-to-understand-git/


gitk: gitk-demo
File Edit View Help
3. commit message
4. local branch "master"
5. remote branch master on origin
6. current commit (HEAD)
2. commit author
8. commit SHA of selected
10. file changes in diff format
9. files impacted by selected commit
Find next prev commit containing: Exact All fields
Search
Patch Tree
Diff Old version New version Lines of context: 3 Ignore space cha
Comments
fruits.txt

```
master  remotes/origin/master  third commit     Tony Stark <tony@stark.com>  2010-09-03 10
second commit                                    Tony Stark <tony@stark.com>  2010-09-03 10
Initial commit                                   Tony Stark <tony@stark.com>  2010-09-03 10

SHA1 ID: 3d024dd9e4a83d8c6a9a143a68b75d4b872115a6   Row 2 / 3
```

```
Committer: Tony Stark <tony@stark.com>  2010-09-03 10:50:28
Parent: b094e60a4888cefd57ce9316084b6e09f7cc3ea8 (Initial commit)
Child:  bf37c64e79b9804aee541f590ccdab0466e01334 (third commit)
Branches: master, remotes/origin/master
Follows:
Precedes:

    second commit

-------------------------------- fruits.txt ---------------------
index 3b77c04..a5d2dcd 100644
@@ -1,2 +1,4 @@
 apples
+pears
 oranges
+bananas
```

Gitk

- SSH into the server with X11 enabled
  - ssh -X for OS with terminal (OS X, Linux)
  - Select “X11” option if using putty (Windows)
- Run gitk in the ~eggert/src/gnu/emacs directory
  - Need to first update your PATH
    - $ export PATH=/usr/local/cs/bin:$PATH
  - Run X locally before running gitk
    - Xming on Windows, Xquartz on Mac

Assignment 9

- Deadline
  - 15th March, 2018, 11:55pm
  - NO late submissions accepted

Assignment 9 - Laboratory

- Fix an issue with diff diagnostic
  - Apply a patch to a previous version
- Installing Git
  - Ubuntu: sudo apt-get install git
  - SEASNet: git is installed in /usr/local/cs/bin
  - Add it to PATH variable or use whole path
    - Export PATH=/usr/localcs/bin:$PATH
- Make a directry 'gitroot' and get a copy of the diffutils git repository
  - Mkdir gitroot
  - Cd gitroot
  - Git clone <url>
  - Follow steps given in the specs and use man git to find commands

Assignment 9 - Laboratory

- Hints
  - Git clone
  - Git log
  - Git tag
  - Git show <hash>
  - Git checkout v3.0 –b <branchname>

Assignment 9 - Homework

- Publish patch you made in lab 9
  - Create a new branch “quote” of version 3.0
    - Branch command + checkout command **(git branch quote v3.0; git checkout quote)**
    - `$ git checkout v3.0 -b quote`
  - Use patch from lab 9 to modify this branch
    - Patch command
    - `$ patch -pnum < quote-3.0-patch.txt`
  - Modify ChangeLog file in diffutils directory
    - Add entry for your changes similar to entries in ChangeLog
  - Commit changes to the new branch
    - `$ git add .    $ git commit -F <Changelog file>`
  - Generate a patch that other people can use to get your changes
    - `$ git format-patch -[num] --stdout > formatted-patch.txt`
  - Test your partner’s patch
    - Check out version 3.0 into a temporary branch `partner`
    - Apply patch with `git am` command: `$ git am < formatted-patch.txt`
    - Build and test with $ `make check`
    - Make sure partner’s name is in HW9.txt for #8

Presentations

- Today’s Presentation:
  - Junting Luo
  - Jefferson Lee
- Next up:
  - Felix Zhang
  - Karl Huang

Questions?