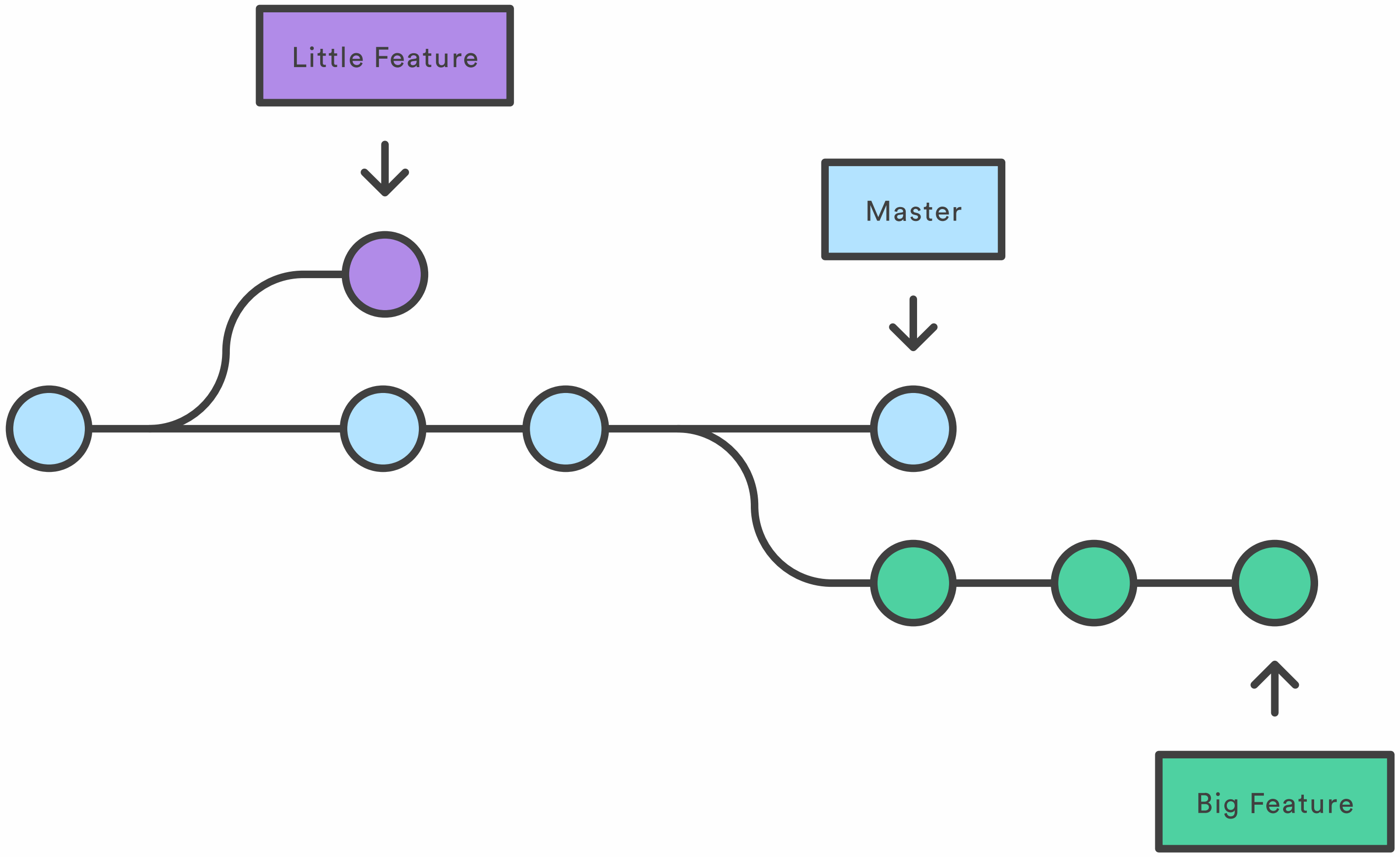
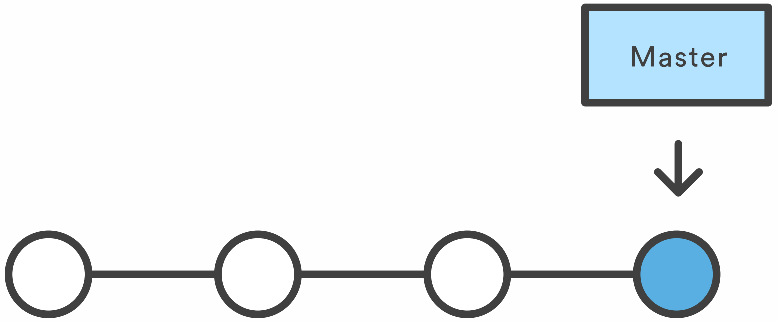
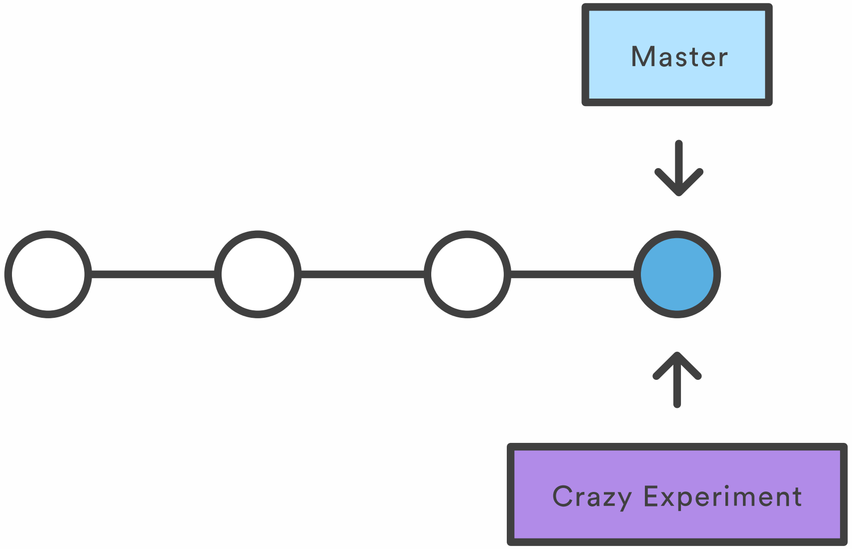
more git


Little Feature
Master
Big Feature

A branch represents an independent line of development.


Master

```
$ git branch crazy-experiment
```


Master
Crazy Experiment

Why Use Branches?

Use Branches to Encapsulate Changes

- Avoids new unstable code in main code base
- Clean up history before merging
- Work in parallel

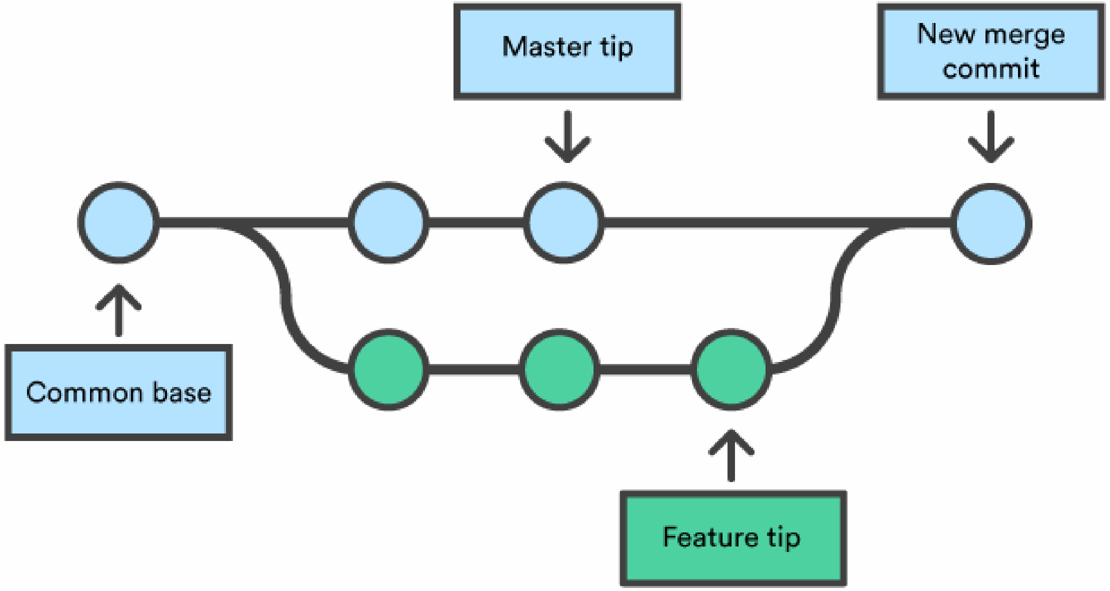
How Do We Use Branches?

```
$ git checkout -b <new-branch>
$ git checkout <existing-branch>
```

Checkout Updates `HEAD`

- `HEAD` refers to the current snapshot
- Branch (which points at a Commit) or Commit
- New Commits are part of a Branch's line of development
- New Commits can't exist outside of a line of development

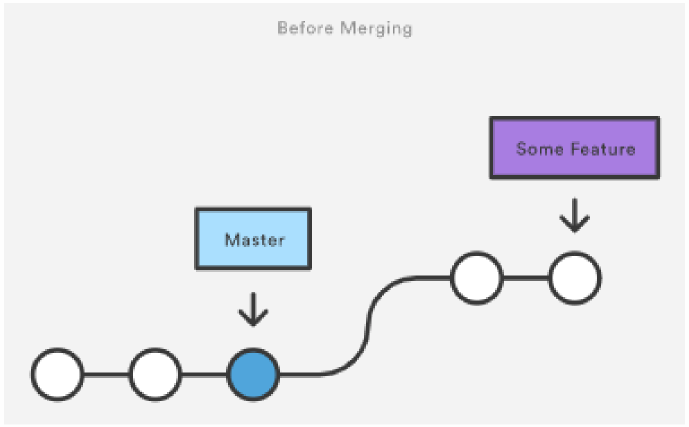
When you are in a Detached HEAD, there's no way to reference new Commits.

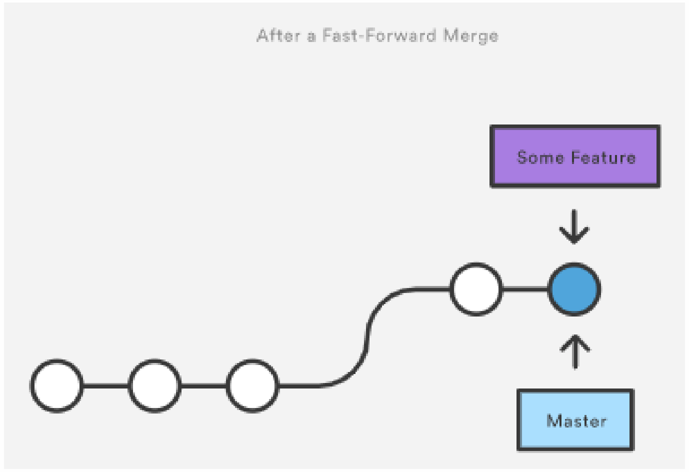

Master tip
New merge commit
Common base
Feature tip

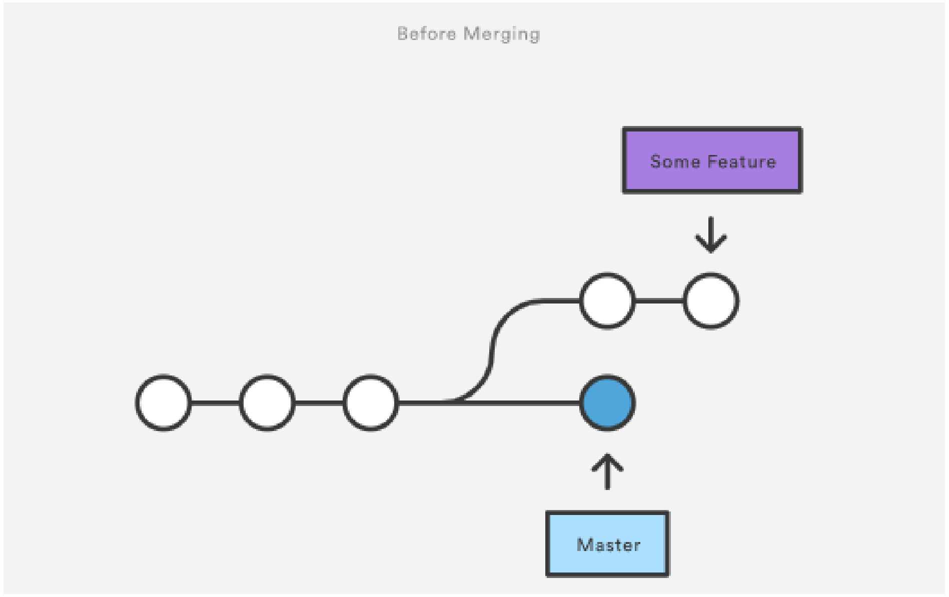
Merging Branches is merging two different lines of development.

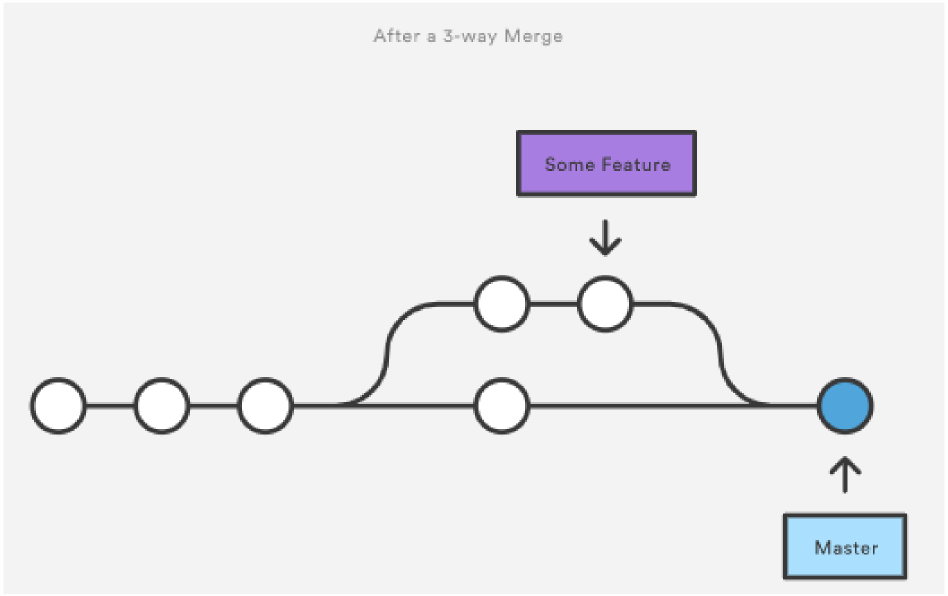
Two Types of Merge

- Fast Forward Merge: linear update to target branch
- Three Way Merge: non-linear combination of source and target branches


Before Merging
Some Feature
Master


After a Fast-Forward Merge
Some Feature
Master


Before Merging
Some Feature
Master


After a 3-way Merge
Some Feature
Master

But what if the merge breaks?

Conflict Resolution

Practice fixing merge conflicts

What Makes A Good Commit Message?

- made some edits
- changed h1 to h2
- worked on new search feature
- oops

The commit is all of your code changes.

The commit message tells you why.

Proper Commit Messages

- Short imperative summary (Email Subject)
- Followed by detailed explanation (Email Body)
- Separated by a blank line!

Tim Pope: A Note About Git Commit Messages

Why do I keep seeing .DS_Store files?

`git rm`

- The opposite of `git add`
- Remove files tracked by git
- Or remove files from both the staging area and the working directory

Let's Remove Them

If you already commited a file and want to remove it from history AND delete it:

```
$ git rm path/to/file
```

If you want to remove from git but keep the file on your machine:

```
$ git rm --cached path/to/file
```

Uh... I still see .DS_Store files.

Let's Ignore Them

In your terminal:

```
$ git config --global core.excludesfile ~/.gitignore_gl
```

Let's Ignore Them

- A gitignore file specifies intentionally untracked files to ignore
- Usually project specific in a file at `my-project/.gitignore`

Github's gitignore templates

Practice Merge Conflicts in Remote Branches

Practice Viewing Commit History