GIT 201 - A DEEPER LOOK AT GIT

Git Me With Your Best Shot

Arthur Doler

@arthurdoler

arthurdoler@gmail.com

We'll be using the Git command line…

Source: Giphy

“Fundamental particles” of Git

🟩 Blob

```
artdoler@machine$ git ls-files --stage
100644 ac461d89cb0a217c2ee2bfcac33e1c27df7739a7 0 path/to/source/file.js
```

△ Tree

```
artdoler@machine$ git write-tree
cf09c37fdb1e4ba320d2ae08e9eb32d6913979e3

artdoler@machine$ git ls-tree cf09
040000 tree c98ef89a1fc979e593927666c058d15f8111c0bf    path
```

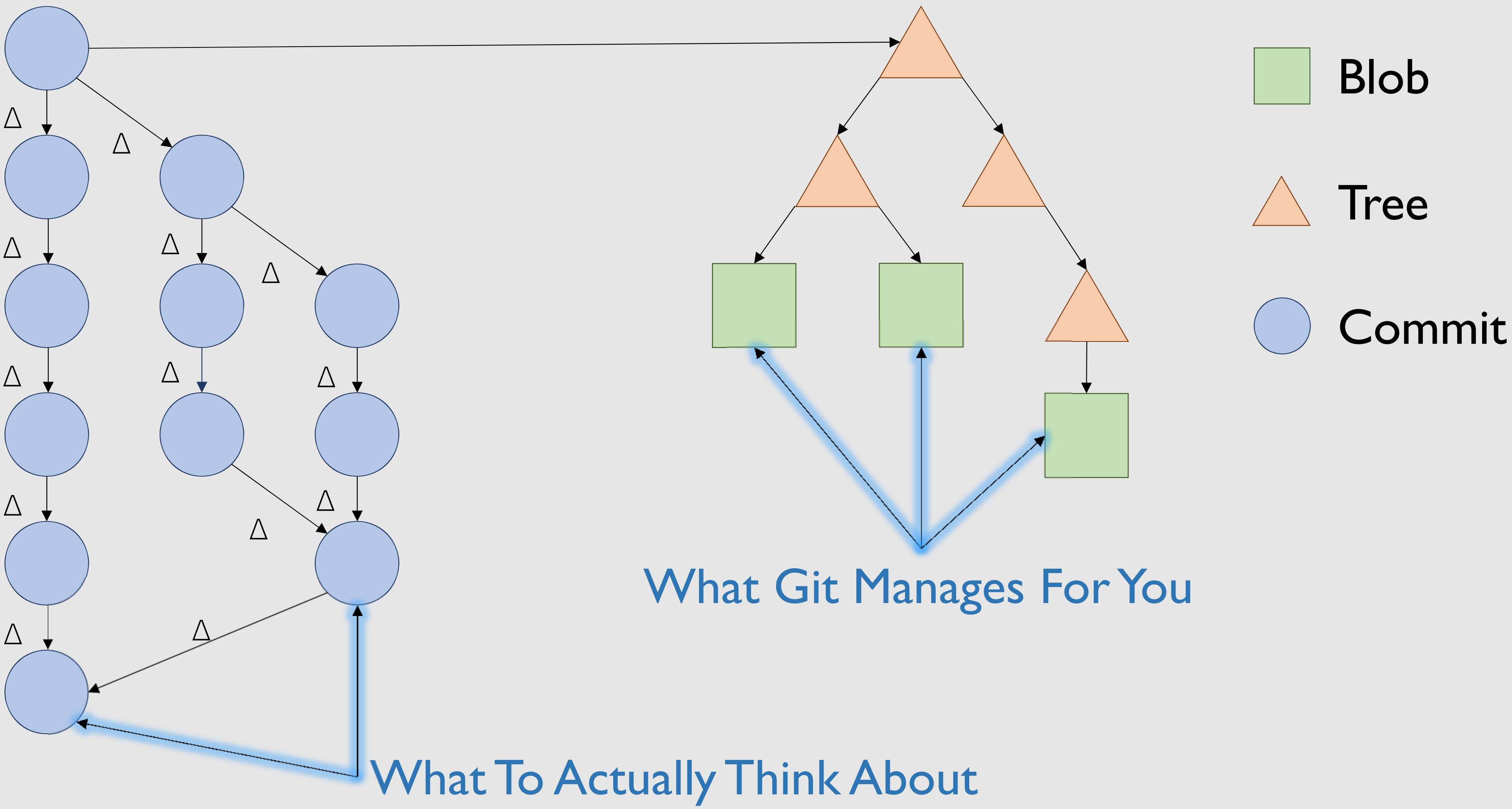
Commit

```
artdoler@machine$ echo "Manual commit." | git commit-tree cf09
c26db2aec27b78aeaed82173ec5f5b2303931b29

artdoler@machine$ git log c26d
commit c26db2aec27b78aeaed82173ec5f5b2303931b29
Author: Art Doler art.doler@aviture.us.com
Date:   Thu May 20 08:59:12 2016 -0500
    Manual commit.
```

A Git repository

is just graphs of commits



🟩 Blob

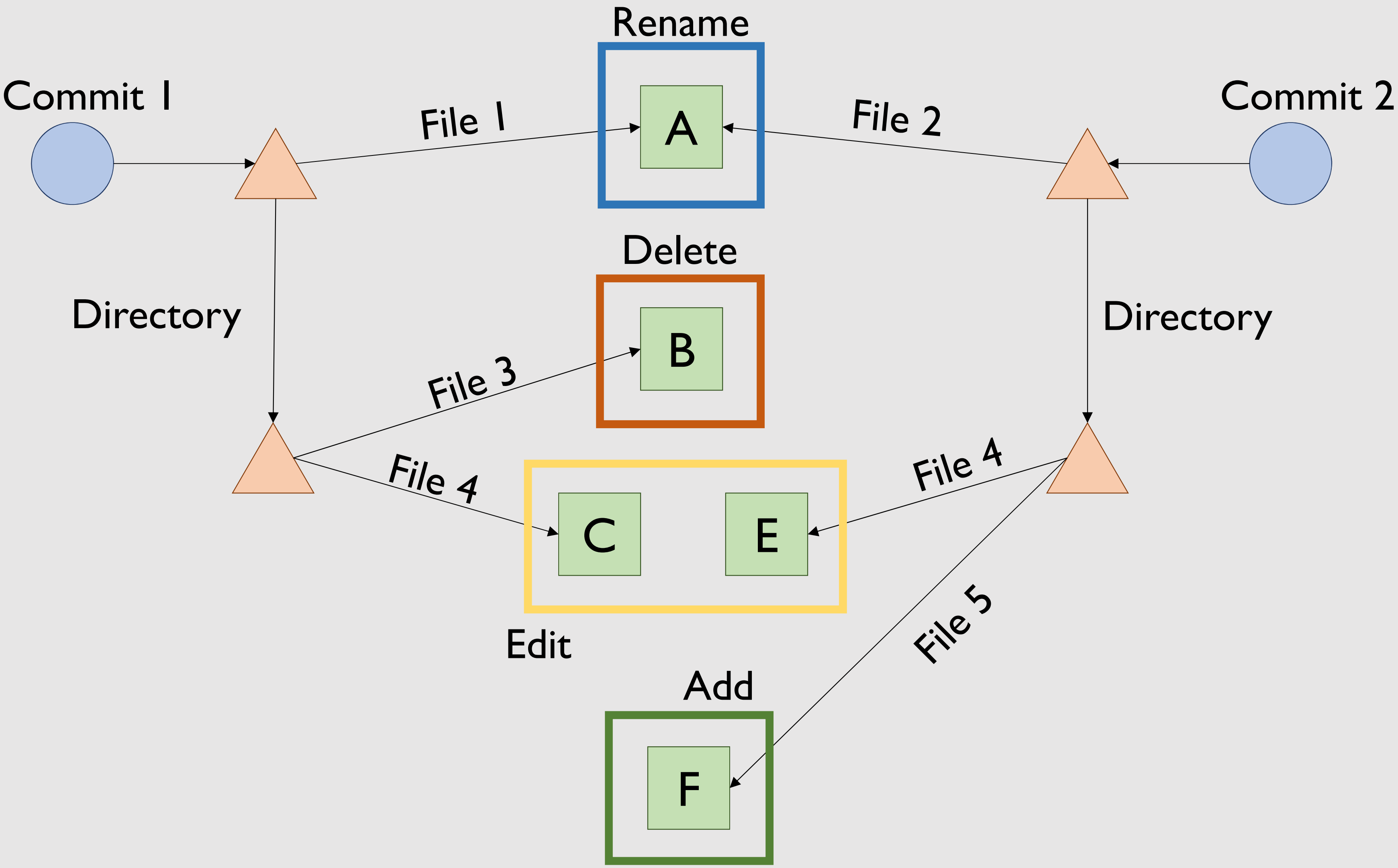
△ Tree

● Commit

What Git Manages For You

What To Actually Think About

How does Git make diffs?


Rename
Commit 1
File 1
A
File 2
Commit 2
Delete
Directory
B
Directory
File 3
File 4
C
E
File 4
Edit
File 5
Add
F

Branches & Tags

Source: Giphy

When I realize "DVCS" means I always get my own branch

What's in a name?

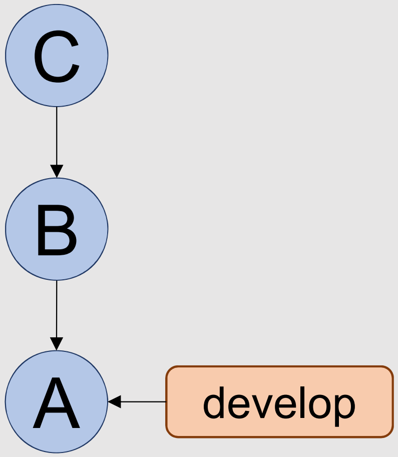
HEAD

feature/my_feature_branch

ac461d89cb0a217c2ee2bfcac33e1c27df7739a7

^

~



A = A^0

B = A^ = A~ = A^1 = A~1

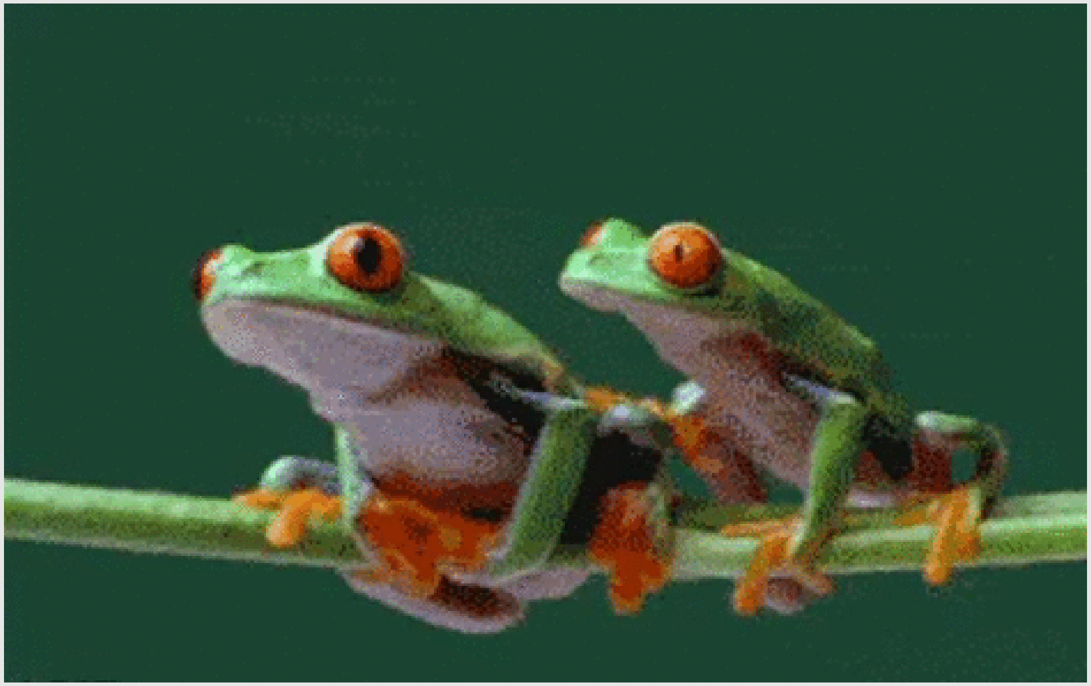
C = B^ = A^^ = A~~

C = A^1^1 = A~2

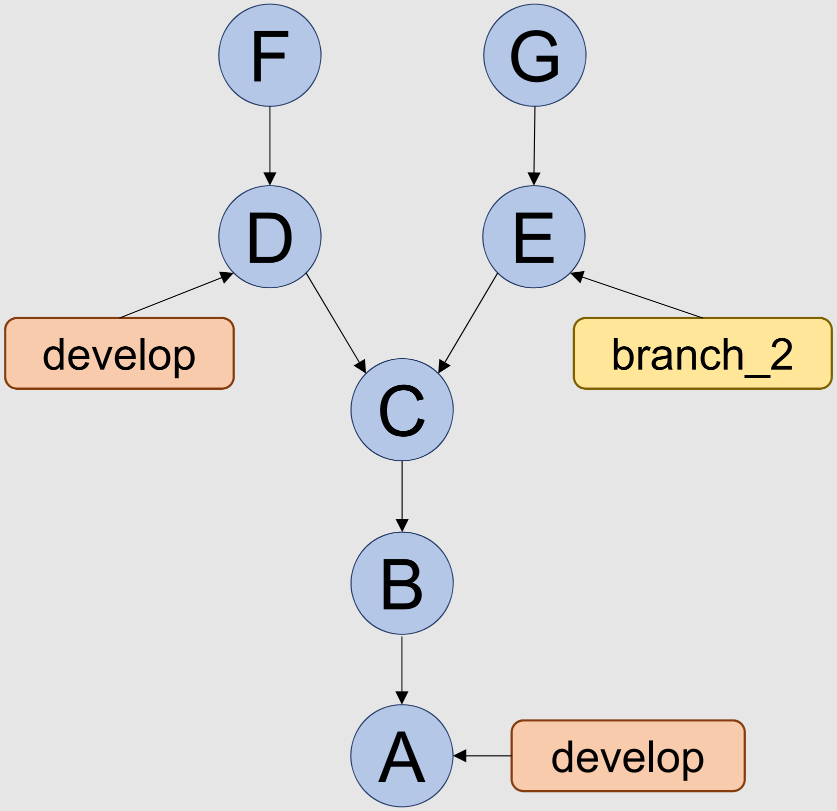
C ≠ A^2

MERGING

Source: Giphy



When my coworker’s merge breaks my feature



A = A^0

B = A^ = A~ = A^1 = A~1

C = B^ = A^^ = A~~

C = A^1^1 = A~2

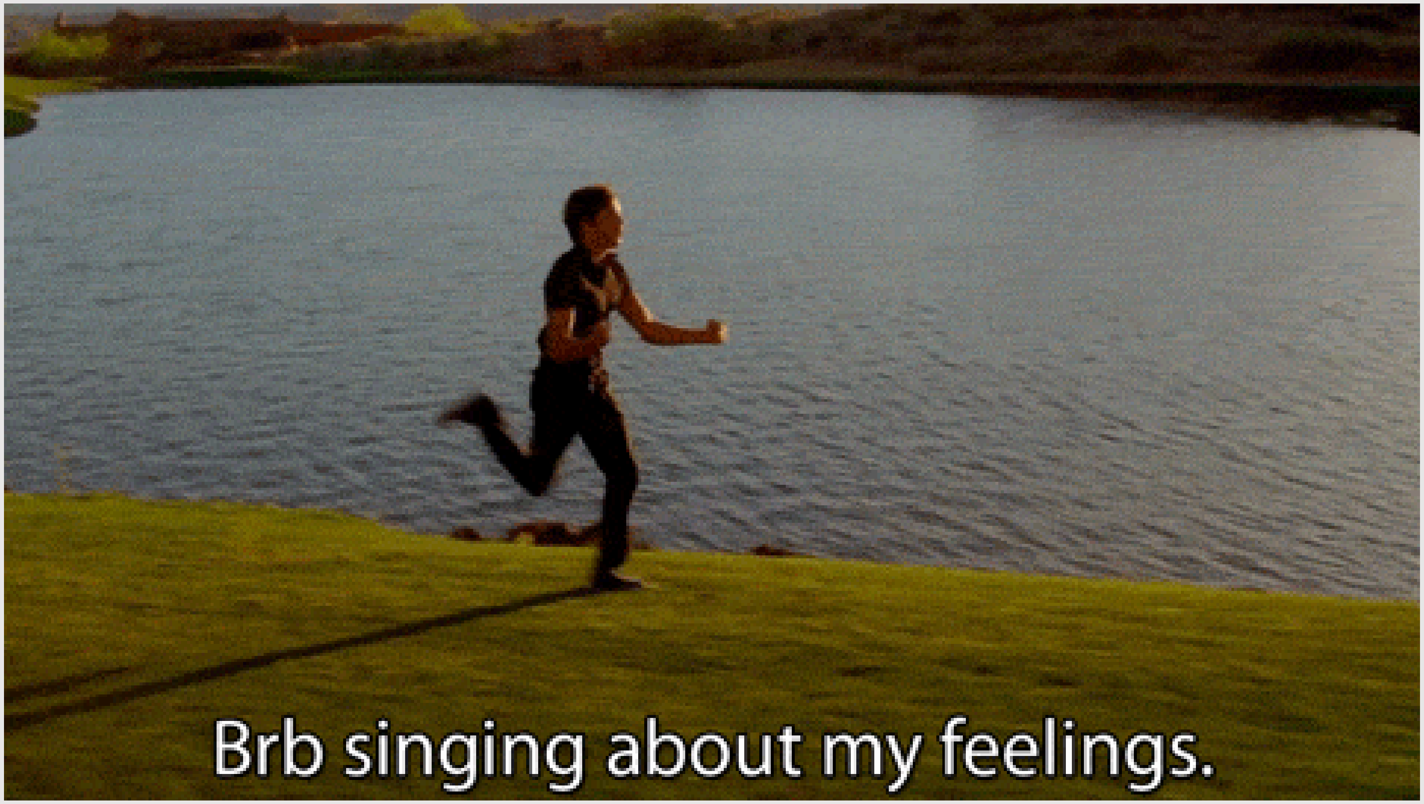
C ≠ A^2

D = C^1 = C~ = B~2

E = C^2 = B^^2 = A~2^2

F = C^^ = A~4

G = E~ = C^2^ = A~2^2~



Source: Giphy

Opinion Time

If the best way to think about Git is commit topologies...

One Concept, One Commit

Source: Wikipedia

"I don't know how many people look at Al's progression of patches, but they are stand-alone patches on their own, while at the same time _also_ being part of a larger migration to the inscrutable goals of Al - ie namespaces etc.

You may not realize just _how_ impressive that is, and what a absolute wonder it is to work with the guy.

Poetry in patches, indeed."

- Linus Torvalds
  On fa.linux.kernel, 27 Dec 2001

Clean code ↔ Clean history

Pick “too many commits” over “too few”

“History is the version of past events that people have decided to agree upon.”

- Attr. Napoleon Bonaparte

Source: Wikipedia

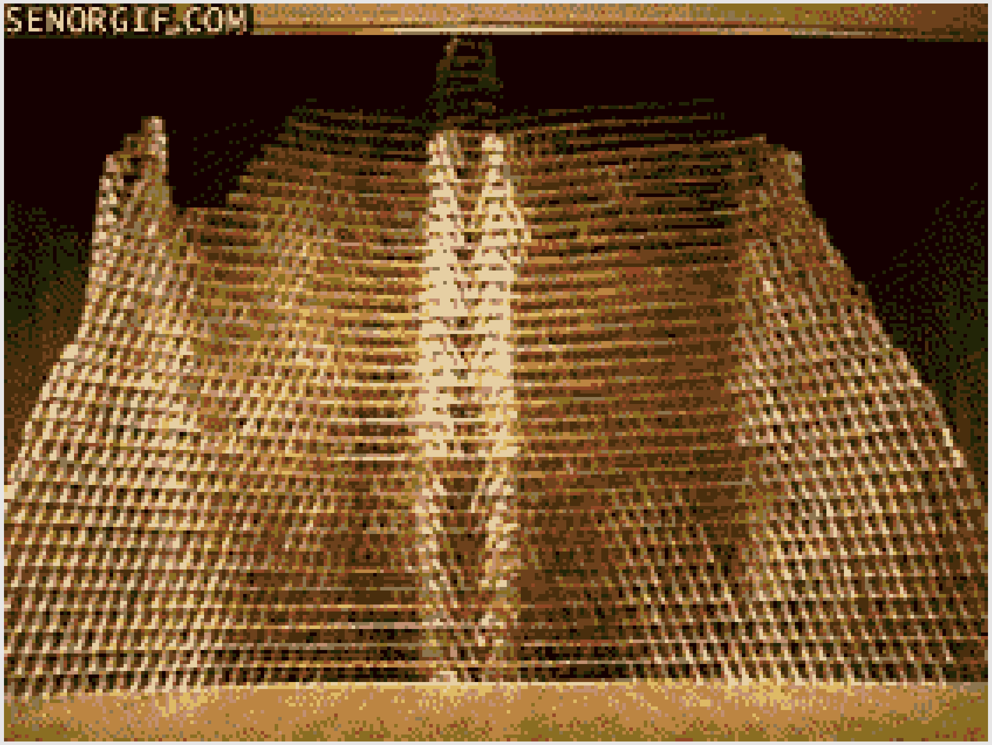
REBASING

Source: Giphy

When you learn to rebase like a boss

Public versus Private Branches

When you force push to a public branch



Source: Giphy

Garbage Collection

Source: Giphy

Finding out git has a garbage collector when it deletes your accidentally untracked branch

History Methodologies

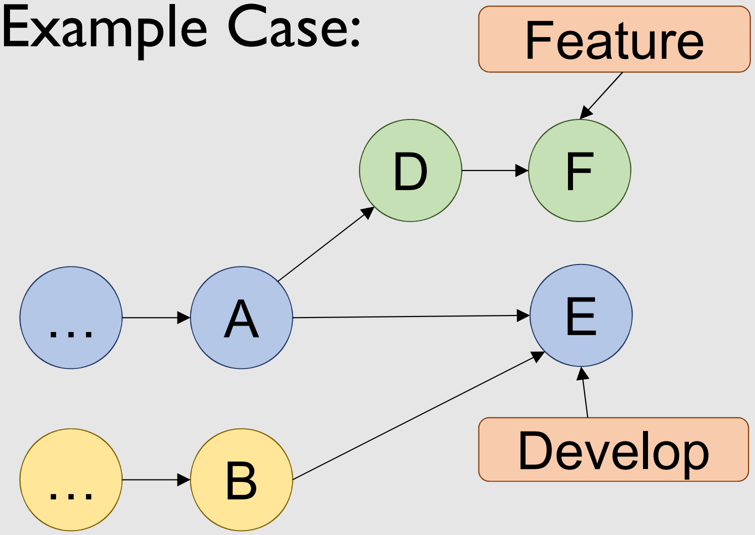
Pure Merge (i.e. gitflow)

Pure Rebase (Keep Branch or Delete Branch)

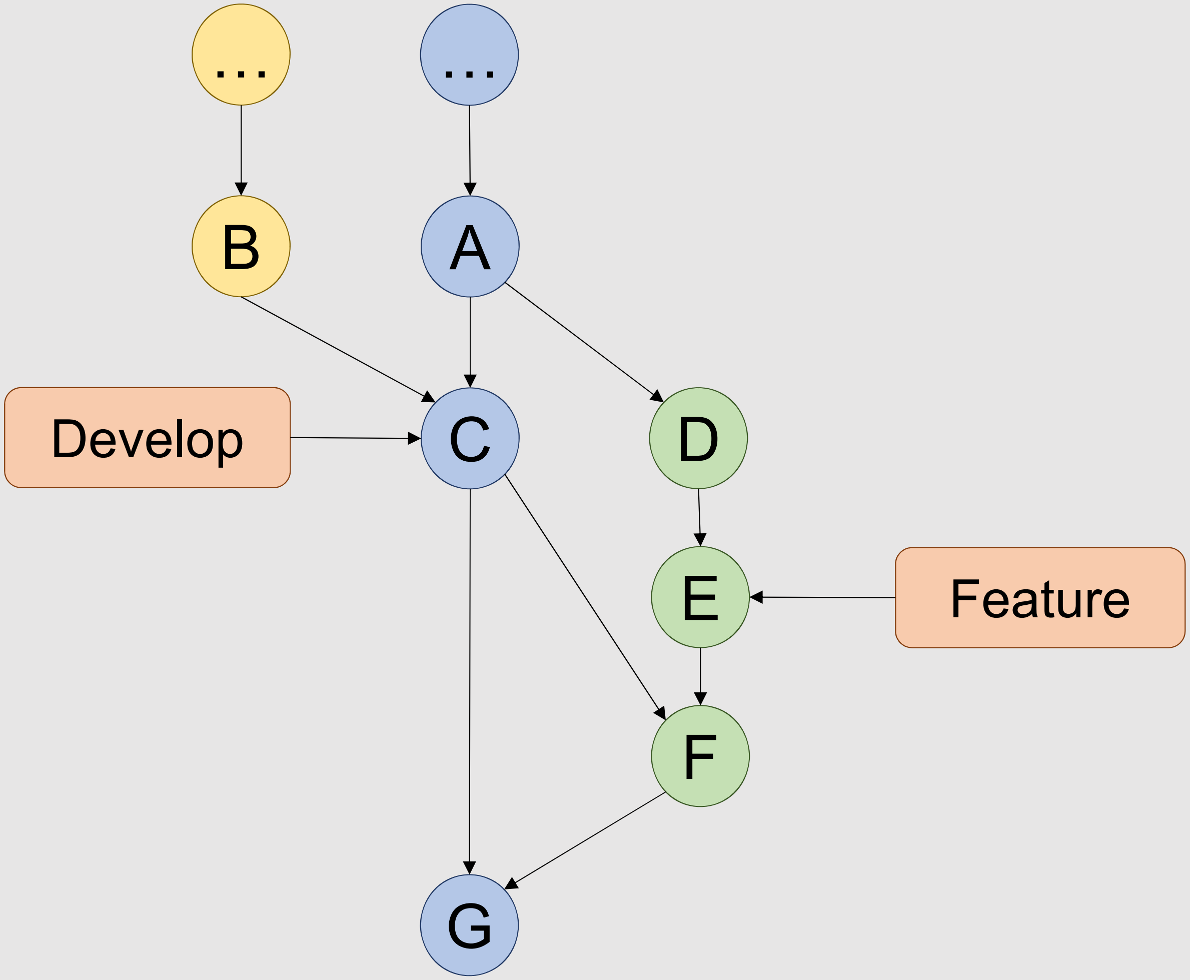
Rebase Then Merge

Rebase With Squash (Mild/Major)

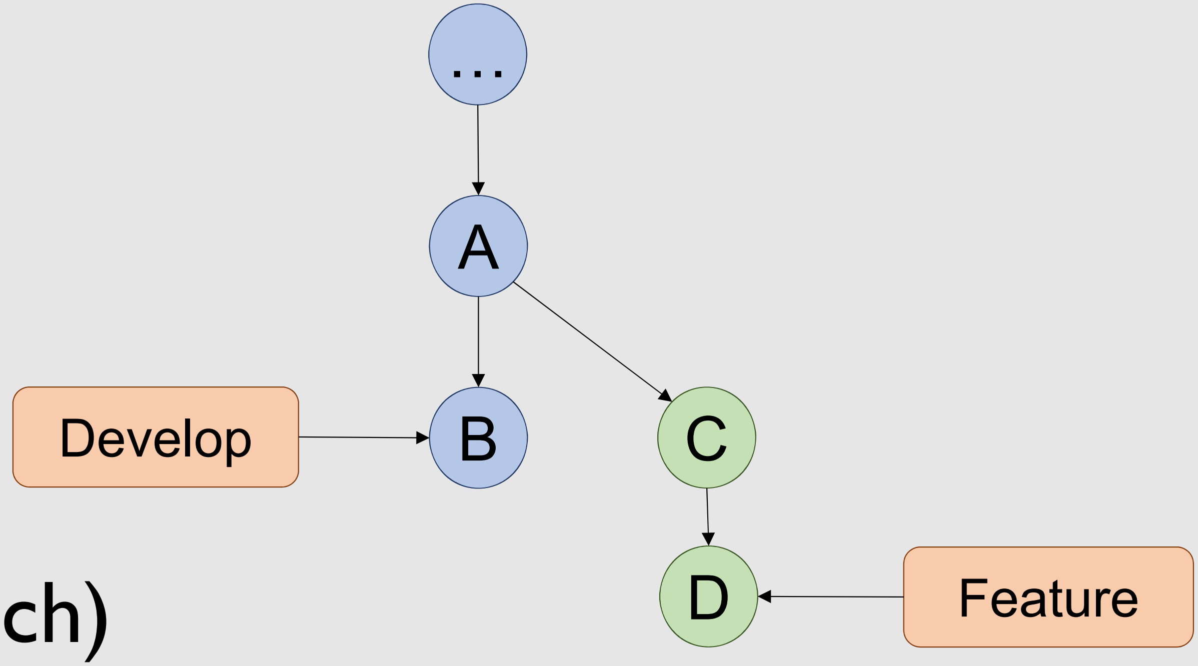
Example Case:



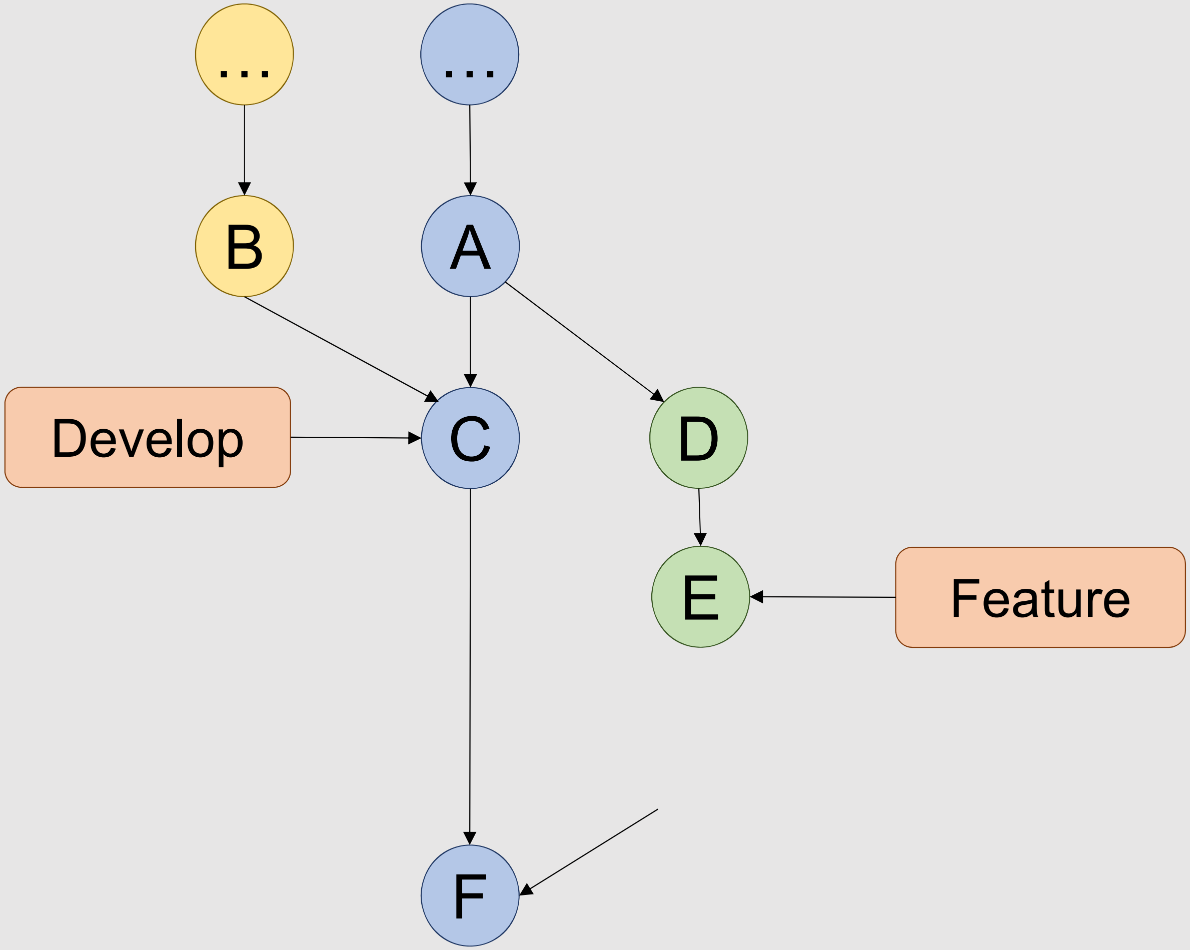
Pure Merge



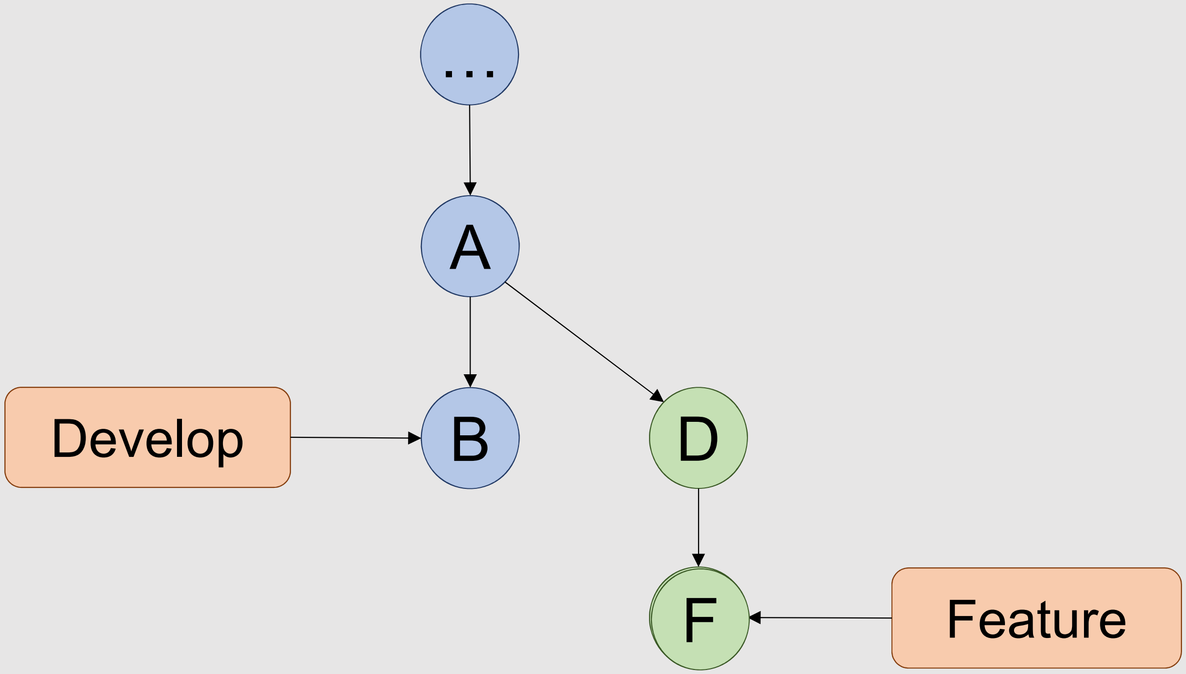
Pure Rebase
(Keep or Delete Branch)



Rebase Then Merge



Rebase With Squash



Source: Giphy

When I accidentally click ‘Merge pull request’ instead of ‘Comment’ in GitHub

THANKS

Arthur Doler

@arthurdoler

arthurdoler@gmail.com

Aviture™
ingenuity. engineered.

Resources

- Git from the Bottom Up (free!)
  - http://ftp.newartisans.com/pub/git.from.bottom.up.pdf
- Gitflow
  - http://nvie.com/posts/a-successful-git-branching-model/
- A Git Workflow for Agile Teams
  - http://reinh.com/blog/2009/03/02/a-git-workflow-for-agile-teams.html
- Git Team Workflows: merge or rebase?
  - http://blogs.atlassian.com/2013/10/git-team-workflows-merge-or-rebase/
- Git Pro Book (also free!)
  - http://git-scm.com/book
- Git pack files
  - http://git-scm.com/book/en/Git-Internals-Packfiles
  - http://stackoverflow.com/questions/5176225/are-gits-pack-files-deltas-rathe...