Version Control

Git

What is version control?

- Also known as source control
- It is the practice of tracking and managing changes to software code
- **V**ersion **C**ontrol **S**ystems are software tools that help software teams manage changes to source code over time.
- As development environments have accelerated, version control systems help software teams work faster and smarter.
- It’s not just Git...
  - Tortoise SVN
  - BitBucket
  - CVS
  - ..... and many more.... https://www.softwaretestinghelp.com/version-control-software/

What does it do?

Version control software keeps track of every modification to the code in a special kind of database. (called a repository)

If a mistake is made, developers can turn back the clock (undo)

Developers can compare earlier versions of the code to help correct any errors that were added to the code base, with minimal disruption to the team

What is Version Control? - Git Guides (2020)

https://www.youtube.com/watch?v=xQujH0ElTUg

The Code!

For almost all software projects, the **source code** is like the crown jewels - a precious asset whose value must be **protected**.

The source code is a **repository** of a teams invaluable knowledge and understanding of an application/system.

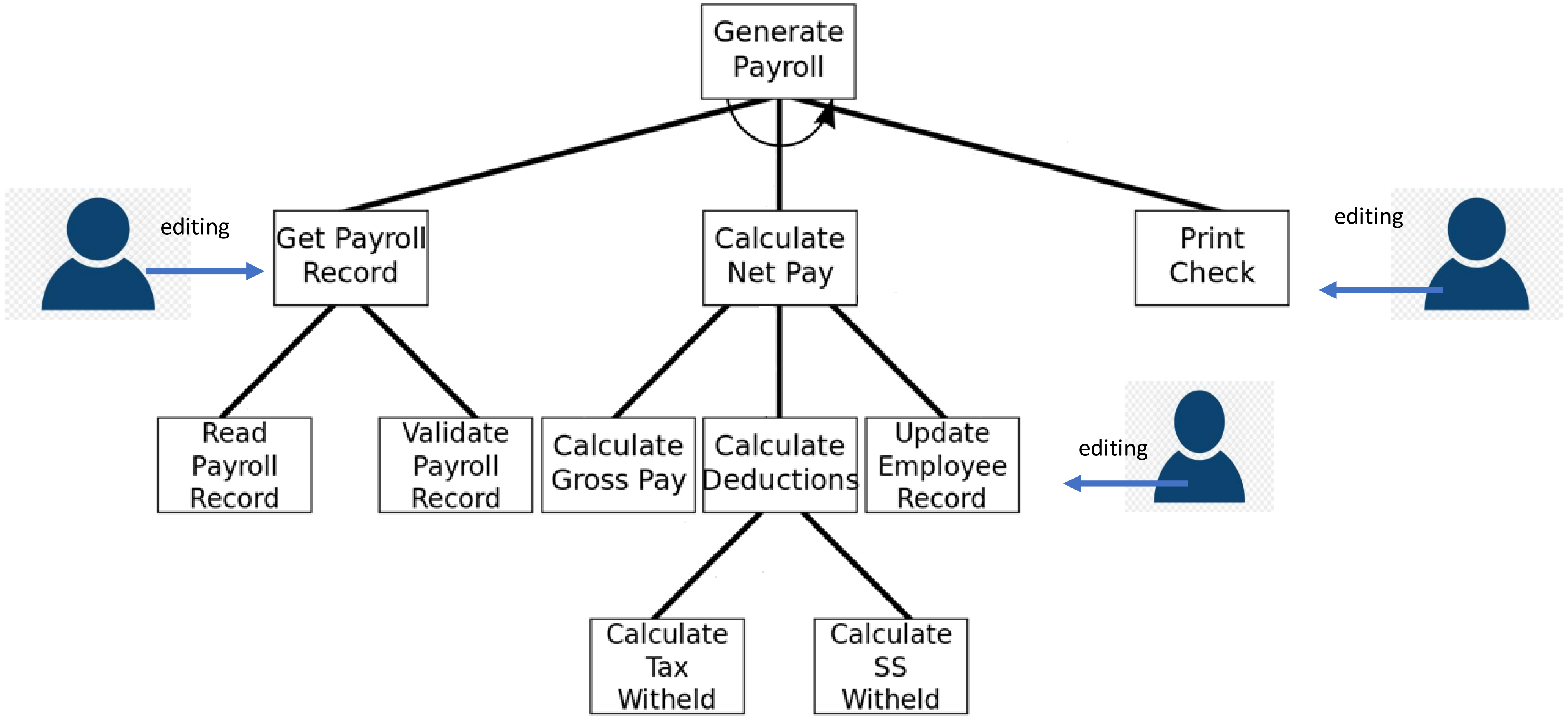
Software developers working in teams are continually writing new source code and changing existing source code. **Version control** protects the code from loss or **damage** through either human error or unintended consequences.

The Code

- The code for a project, app or software component is typically organized in a folder structure or "file tree".

- One developer on the team may be working on a new feature while another developer fixes an unrelated bug by changing code, each developer may make their changes in several parts of the file tree.

Code Structure Example

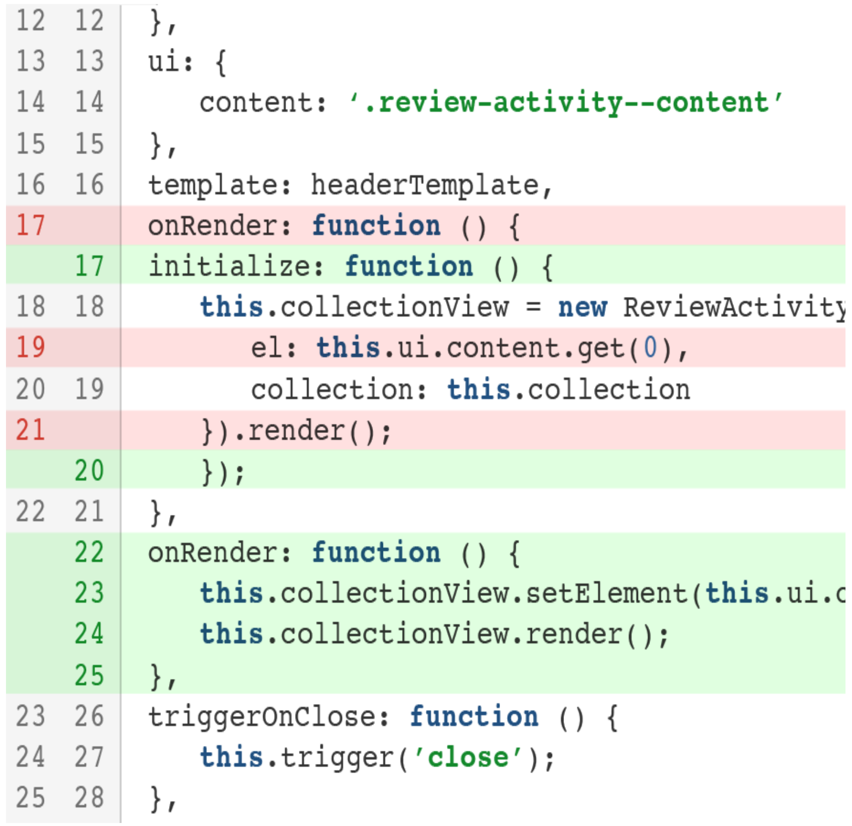


Code Changes

VERSION CONTROL HELPS TEAMS SOLVE ANY CODE PROBLEMS OR **CONFLICTS**

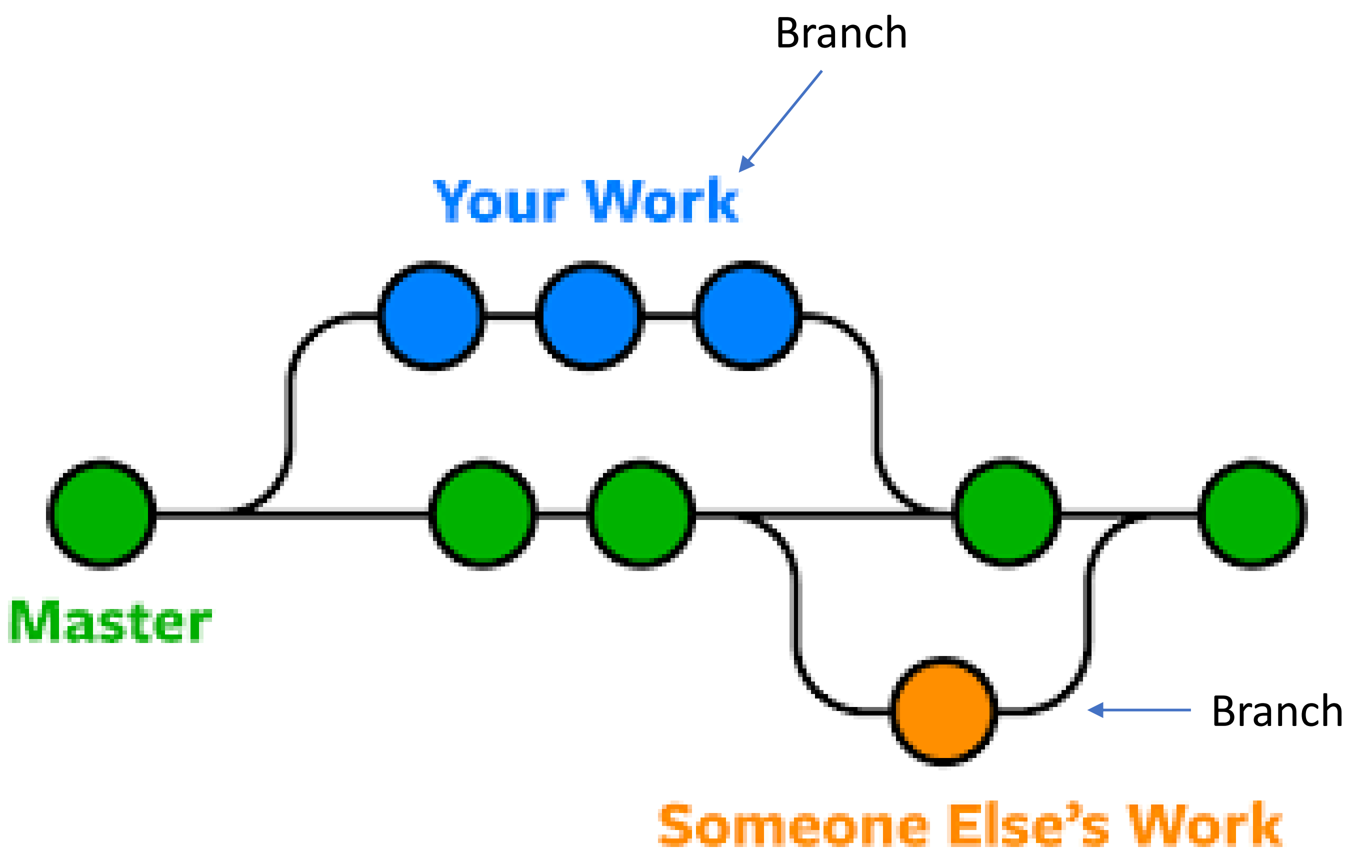
IT **TRACKS** EVERY INDIVIDUAL CHANGE BY EACH CONTRIBUTOR AND HELPING PREVENT CONCURRENT WORK FROM **CONFLICTING**.

CHANGES MADE IN ONE PART OF THE SOFTWARE CAN BE INCOMPATIBLE WITH THOSE MADE BY ANOTHER DEVELOPER WORKING AT THE SAME TIME.



```
},
ui: {
   content: '.review-activity--content'
},
template: headerTemplate,
onRender: function () {
initialize: function () {
   this.collectionView = new ReviewActivity
      el: this.ui.content.get(0),
      collection: this.collection
   }).render();
   });
},
onRender: function () {
   this.collectionView.setElement(this.ui.c
   this.collectionView.render();
},
triggerOnClose: function () {
   this.trigger('close');
},
```

Branching



Remember

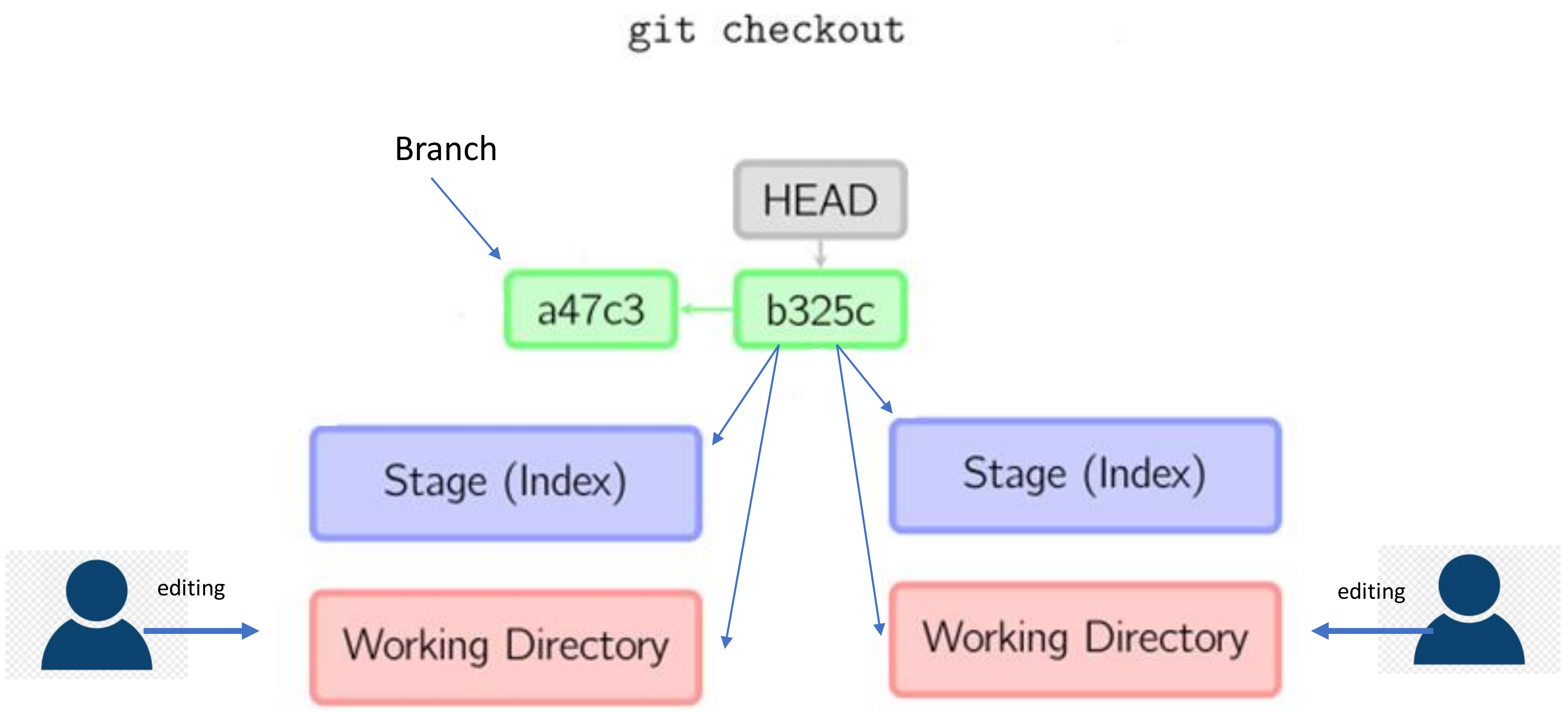
You Cannot Break Things

The first thing that you should keep in mind is that you can always **undo** a merge and go back to the state before the conflict occurred. **You're always able to undo and start fresh**.

Also, a conflict will only ever affect ***you***. It will *not* bring your complete team to a halt or cripple your central repository!!

- This is because, in Git, conflicts can only occur on ***your*** local machine – and not on the server.

Checkout Code

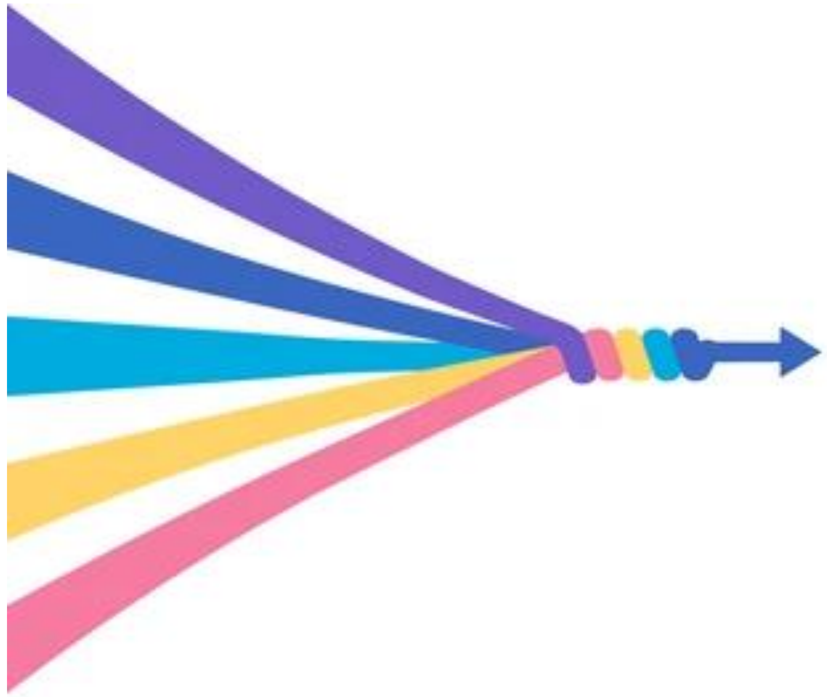


Merge

In Git, “merging” is the act of integrating another branch into your current working branch. You’re taking changes from another context (that’s what a branch effectively is: a context) and combine them with your current working files.

Have a look at this [introduction to branching]() if you’re new to the concept in general.

A great thing about having Git as your version control system is that it makes merging extremely easy: in most cases, Git will figure out how to integrate new changes.



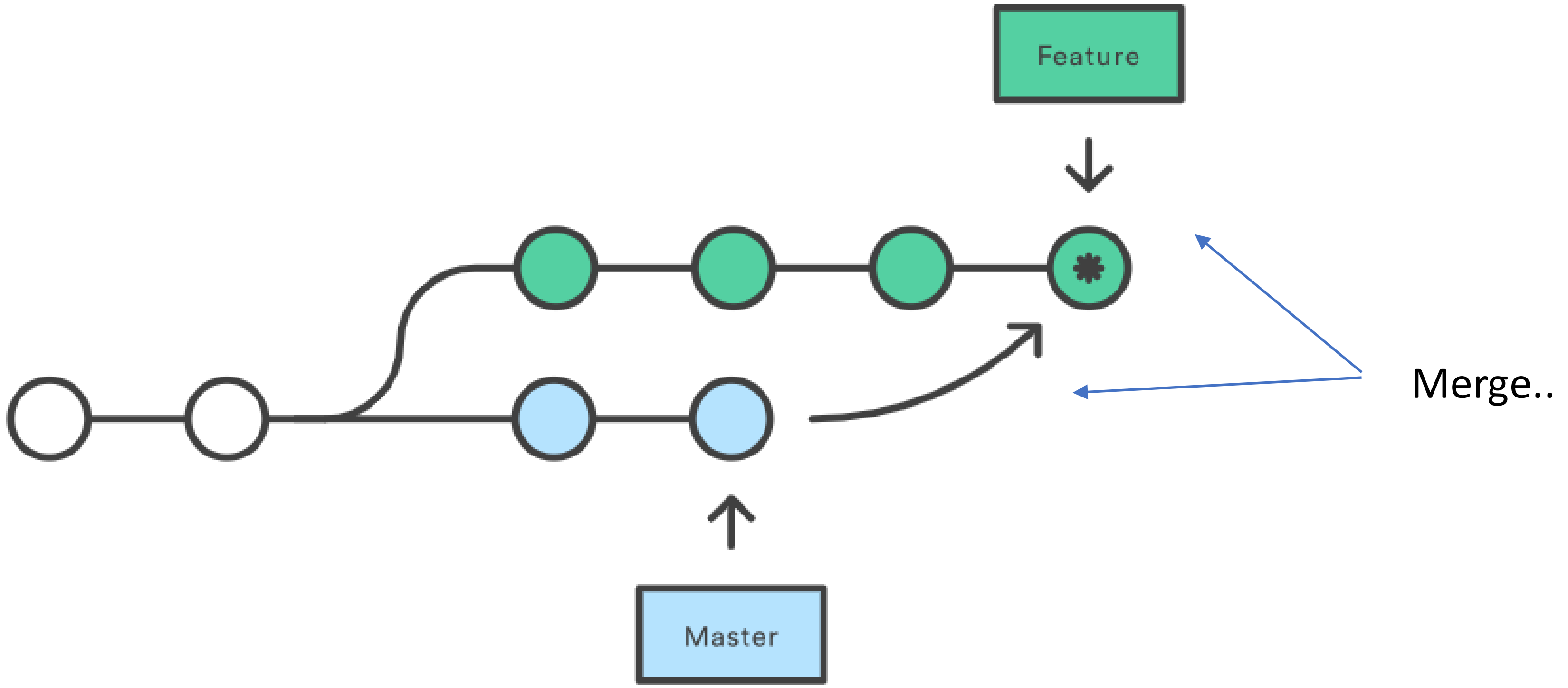
Merge

However, there's a handful of situations where you might have to step in and tell Git what to do. Most commonly, this is when there are changes to the same file on both branches.

Even in this case, Git will most likely be able to figure it out on its own. But if two people changed the same lines in that same file, or if one person decided to delete it while the other person decided to modify it, Git simply cannot know what is correct.

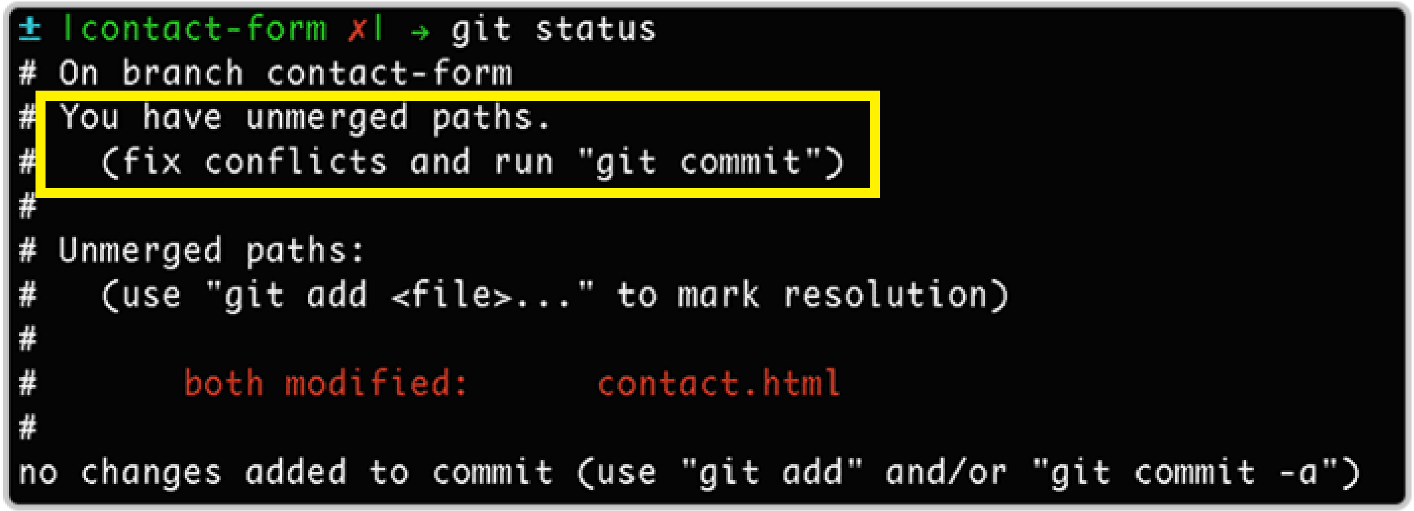
Git will then mark the file as having a conflict – which you'll have to solve before you can continue your work.

Merge



How to Solve a Merge Conflict

- When faced with a merge conflict, the first step is to understand what happened.
  - Did one of your colleagues edit the same file on the same lines as you?
  - Did they delete a file that you modified?
  - Did you both add a file with the same name?
- Git will tell you that you have *"unmerged paths"* (which another way of telling you that you have one or more conflicts) via "git status" command:



```
± |contact-form ✗| → git status
# On branch contact-form
# You have unmerged paths.
#   (fix conflicts and run "git commit")
#
# Unmerged paths:
#   (use "git add <file>..." to mark resolution)
#
#       both modified:      contact.html
#
no changes added to commit (use "git add" and/or "git commit -a")
```

Some videos…..

- Learn Git In 15 Minutes

https://www.youtube.com/watch?v=USjZcfj8yxE

- Git For Dummies (20 mins)

https://www.youtube.com/watch?v=mJ-qvsxPHpY

- Git and GitHub for Beginners - Crash Course (60 mins)

https://www.youtube.com/watch?v=RGOj5yH7evk