cisco DEVNET

Introduction to Git

Ashley Roach -- Principal Engineer Evangelist -- DevNet

Twitter: @aroach
Email: asroach@cisco.com

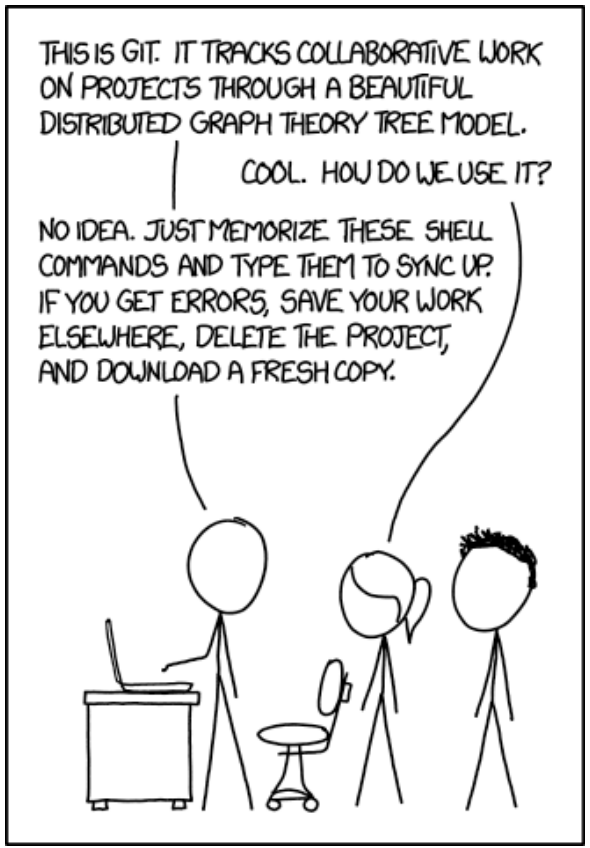
Agenda

- Introduction
- Quick overview of Git
- Workshop
- Resources

How is this relevant to networking?

- Programmability
- Scripts
- Infrastructure as code
  - Ansible
  - Terraform
- Even if you're solo, version control is critical

WHY ARE WE HERE?



Terraform: Infra layer

```
/*
  Public Subnet
*/
resource "aws_subnet" "us-east-1-public" {
    vpc_id = "${aws_vpc.default.id}"

    cidr_block = "${var.public_subnet_cidr}"
    availability_zone = "us-east-1a"

    tags {
        Name = "Public Subnet"
    }
}

resource "aws_route_table" "us-east-1-public" {
    vpc_id = "${aws_vpc.default.id}"

    route {
        cidr_block = "0.0.0.0/0"
        gateway_id = "${aws_internet_gateway.default.id}"
    }

    tags {
        Name = "Public Subnet"
    }
}

resource "aws_route_table_association" "us-east-1-public" {
    subnet_id = "${aws_subnet.us-east-1-public.id}"
    route_table_id = "${aws_route_table.us-east-1-public.id}"
}

/*
  Private Subnet
*/
resource "aws_subnet" "us-east-1-private" {
    vpc_id = "${aws_vpc.default.id}"

    cidr_block = "${var.private_subnet_cidr}"
    availability_zone = "us-east-1a"

    tags {
        Name = "Private Subnet"
    }
}
```

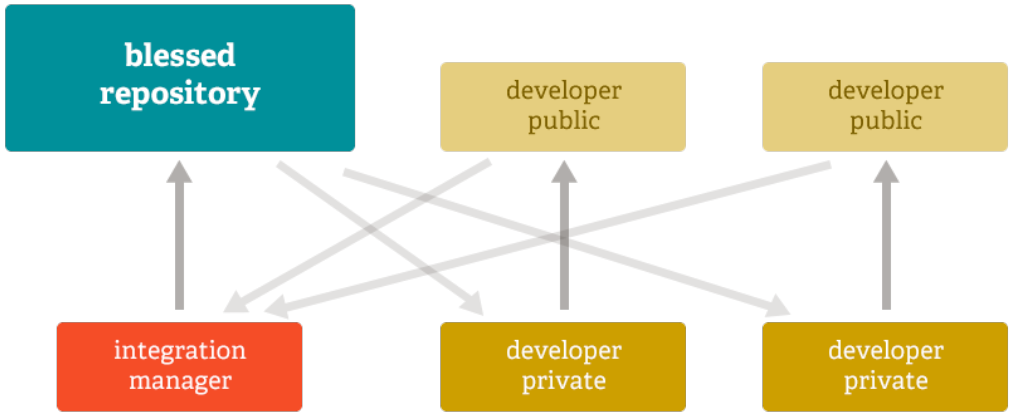
Ansible: OS/App Layer

```
---
# inspiration from https://github.com/belmarca/caddy-ansible

- name: Create Caddy user
  become: yes
  user:
    name=caddy
    system=yes
    createhome=yes

- name: Get all Caddy releases
  become: yes
  get_url:
    url=https://api.github.com/repos/mholt/caddy/git/refs/tags
    dest=/home/caddy/releases.txt
    force=yes
  when: caddy_update
  register: caddy_releases_cache

- name: Set Caddy features
  become: yes
  copy:
    content="{{caddy_features}}"
    dest=/home/caddy/features.txt
  when: caddy_update
  register: caddy_features_cache

- name: Download new Caddy version or build
  become: yes
  get_url:
    url=https://caddyserver.com/download/linux/amd64?plugins=tls.dns.gandi
    dest=/home/caddy/caddy.tar.gz
    force=yes
  when: caddy_releases_cache.changed or caddy_features_cache.changed
  register: caddy_binary_cache

- name: Download Caddy
  become: yes
  get_url:
    url=https://caddyserver.com/download/linux/amd64?plugins=tls.dns.gandi
    dest=/home/caddy/caddy.tar.gz

- name: Extract new Caddy version or build
  become: yes
  unarchive:
    src=/home/caddy/caddy.tar.gz
    dest=/usr/bin/
    copy=no
  when: caddy_binary_cache.changed
```

Why version control? To Protect yourself and others

DISTRIBUTED VERSION CONTROL

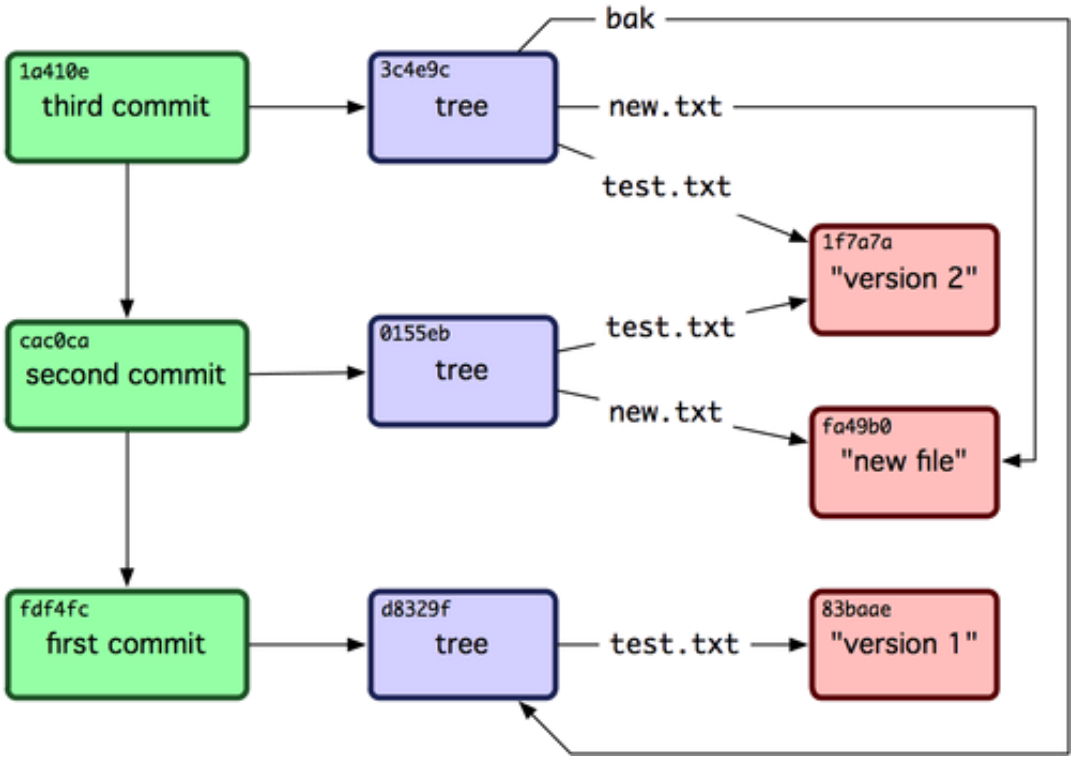
- Opens up to new workflows: git flow
- Each system has an exact replica of the repo as other collaborators.



https://git-scm.com/images/about/workflow-b@2x.png

Under the hood

- Changes are stored in trees
- Trees contain changed files
- Commits contain trees



http://git-scm.com/figures/18333fig0903-tn.png

GIT CONFIG

- So you can be held accountable, configure git

```
$ git config --global user.name "Your Name Comes Here"
$ git config --global user.email you@yourdomain.example.com
```

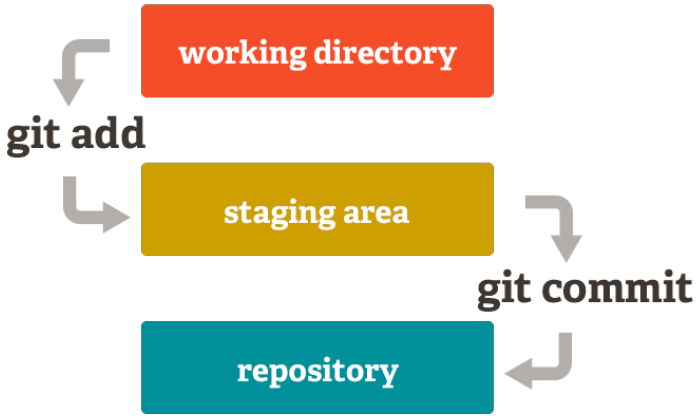
GIT INIT

- Initializes a project directory with a hidden directory /.git/

```
$ tar xzf project.tar.gz
$ cd project
$ git init
```

GIT ADD

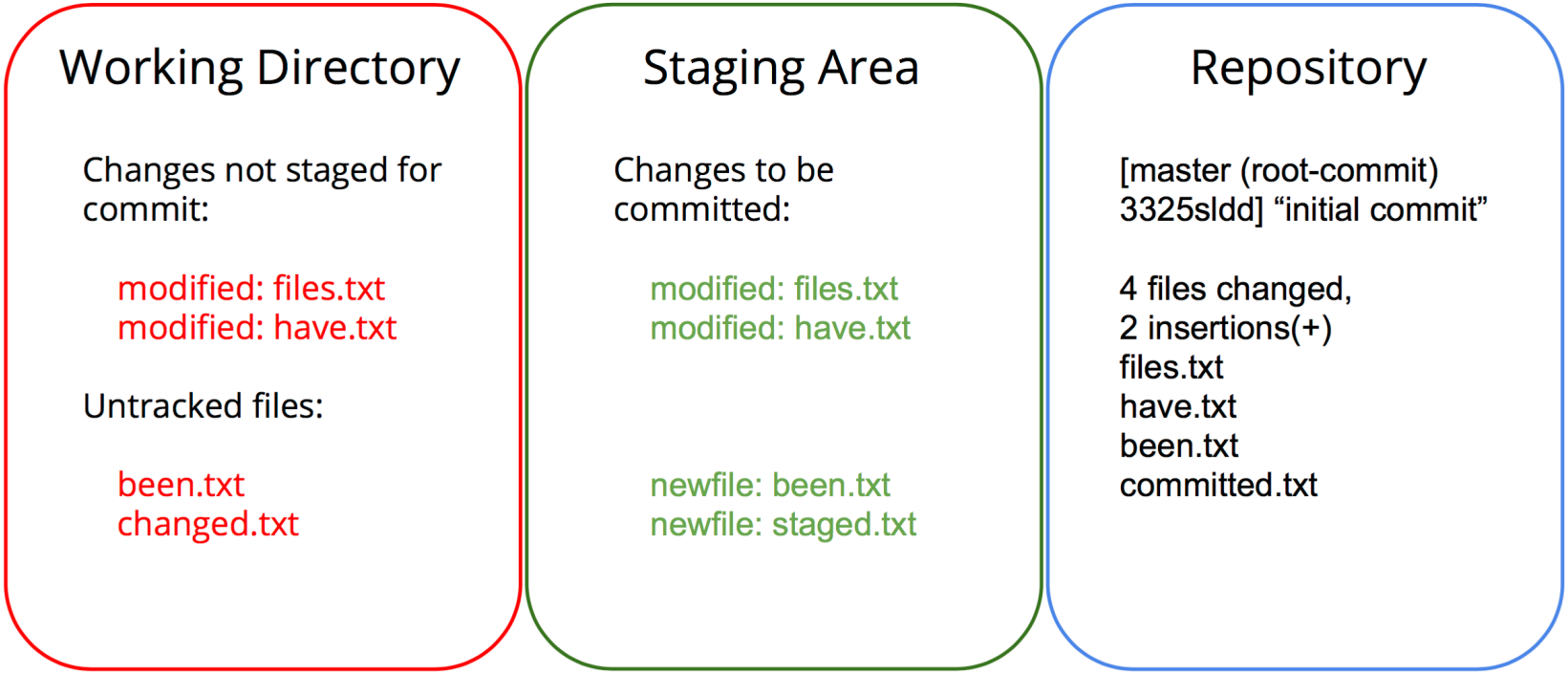
- Add any files in your repository to git “stage”



https://git-scm.com/images/about/index1@2x.png

```
$ git add .
```

STAGING AREA



Hat tip: @GeorgiaReh

GIT COMMIT

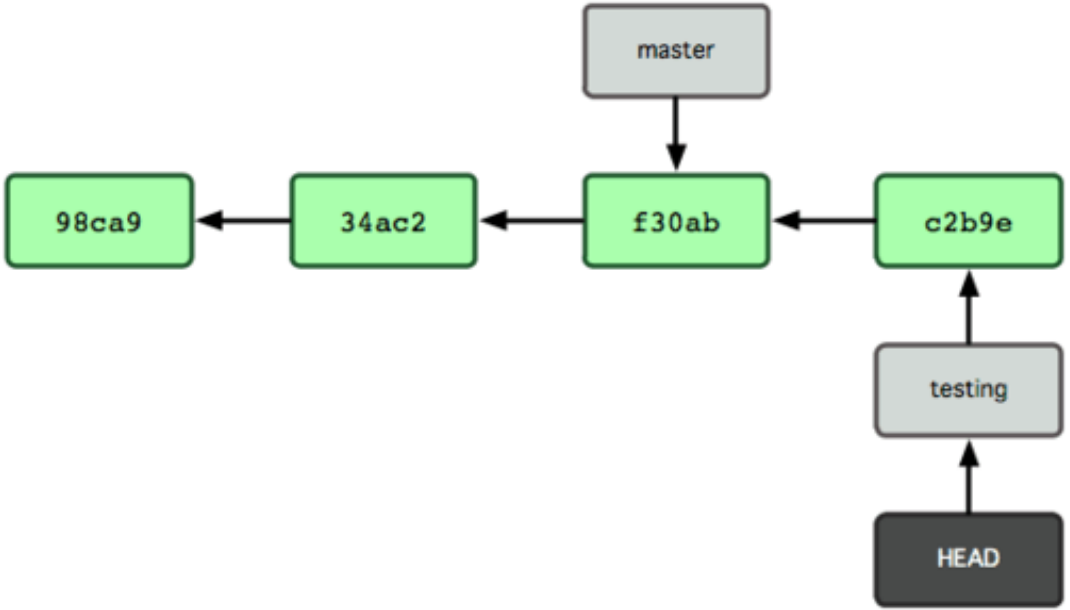
- Store your changes into a commit
- Saves all of your changes together / save point
- Commit does NOT push

```
$ git commit –m ’Initial commit’
```

BRANCHING: Your safe place

- Makes a pointer to your code
- Moves HEAD around

```
$ git branch <name>
$ git branch testing
$ git commit -m "new"
$ git checkout master
```



http://git-scm.com/figures/18333fig0307-tn.png

MERGING

- git merge <topic>
- You must be on the branch you want to merge INTO when you execute this command (e.g. master)

```
$ git merge <branch>
```

GOING BACKWARDS

- Generate a new commit that undoes all of the changes introduced in <commit>, then apply it to the current branch.

```
$ git revert <commit>
```

SHARE YOUR CHANGES

- git push <destination> <branch>
- git push origin master

```
$ git remote add <name> <url>
$ git push <name> <branch>
$ git push origin master
```

CLONING PROJECTS (SSH OR HTTPS)

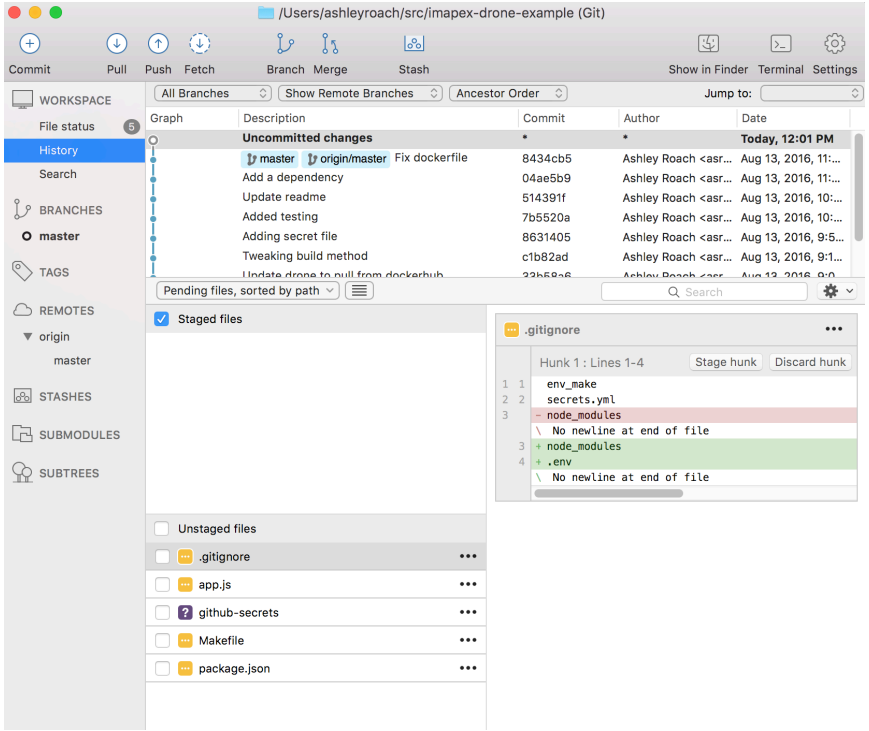
- No password
- Set up SSH key on remote server (e.g. ssh-keygen)

```
$ git clone git@github.com:aroach/upgraded-guacamole.git
```

```
$ git clone https://github.com/aroach/upgraded-guacamole.git
```

GIT CLIENTS

- CLI Client
- IDE Clients
  - VIM: airblade/vim-gitgutter
  - Emacs: magit
- GUI Clients
  - SourceTree
  - Git Kracken

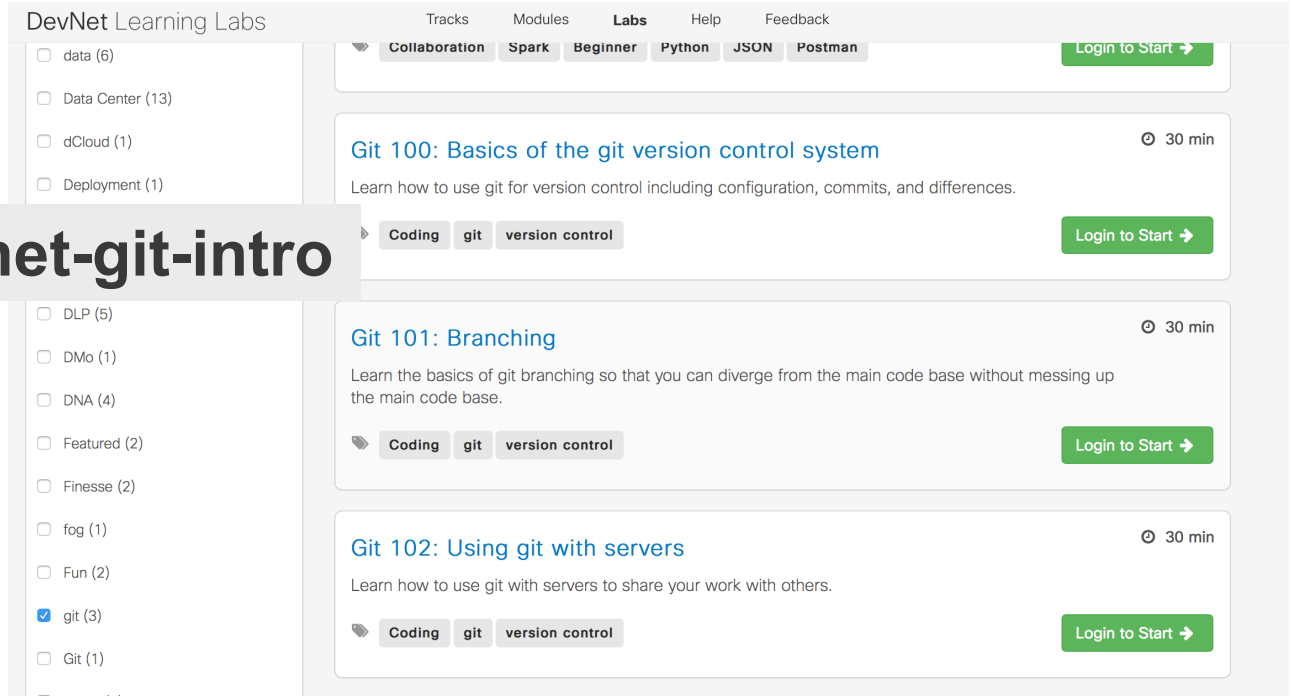


Workshop

Git Workshop Resources
http://cs.co/git-us2017

Let’s get started

http://bit.ly/devnet-git-intro



Forking and Pull Requests

- Login to GitHub
- Navigate to: https://github.com/aroach/git-intro-fork-example
- Click the fork button
- Create a new file with some dummy content
- Save it
- Submit a Pull Request (PR)

Come find me after today… I'll be waiting

Ashley Roach

asroach@cisco.com

@aroach

http://github.com/aroach

http://linkedin.com/in/ashleyroach

Cisco DEVNET

@CiscoDevNet

http://github.com/CiscoDevNet

DEVNET


cisco
DEVNET