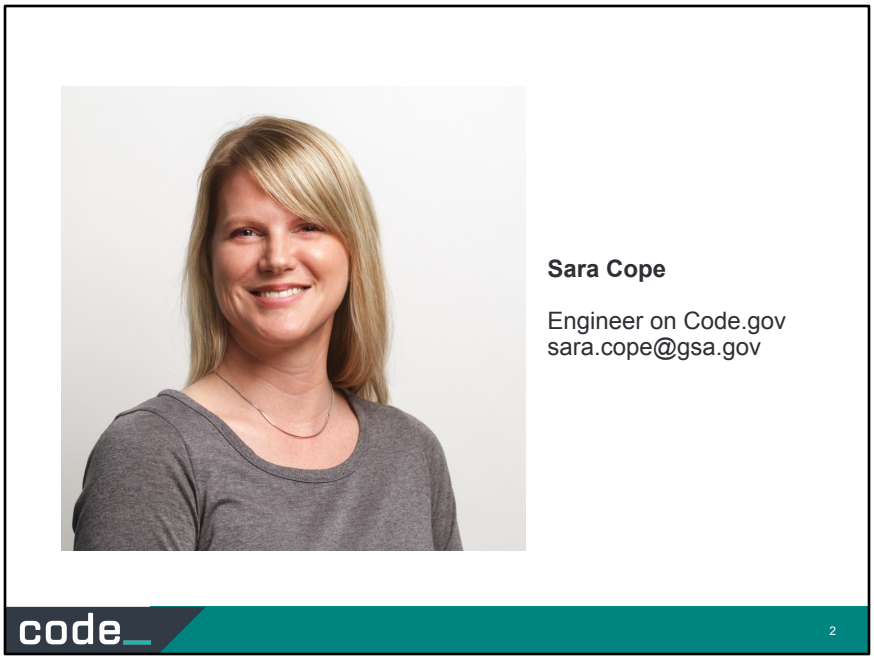
Thanks for joining me today, in the session we're going to take a tour of GitHub for non-developers .



My name is Sara Cope, I'm an engineer in TTS on the code.gov team
if you're not familiar with that it's actually an open source project which is a clearinghouse of other open source projects. So we actually collect and inventory open source code from GSA and other agencies.
Because of that, I spend a lot of time in open source code and on GitHub and often see people struggle with the github interface and sometimes feel intimidated by it. So I wanted to give this Tech talk today to hopefully demystify the GitHub website a bit and encourage more folks to use open source.

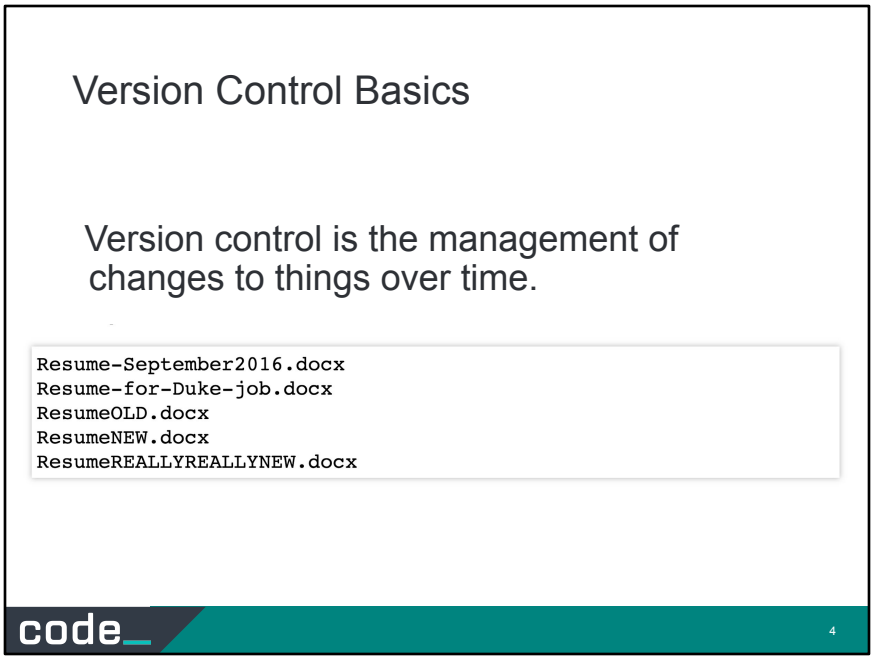
Agenda

- Version control basics
- Git vs. GitHub
- A tour of the GitHub interface
- Basic concepts
- Contributing to projects on GitHub without code
- Additional resources

code_

3

Here's our agenda of what we're going to be doing today
First we're going to talk about the basics of Version Control
Then we'll go over the differences between git and GitHub
After that we'll dive into a tour of the actual GitHub interface and talk about all the moving parts
And I'll show how to contribute to a project within GitHub, without going to the command line
Then finally we'll wrap up with a few resources for you to checkout if you're interested in getting to know GitHub a more in-depth.



Version Control lets you track your files over time. So when you mess up you can easily get back to a previous working version.

You've probably cooked up your own version control system without realizing it had such a geeky name. Got any files like this?

It's why we use "Save As". You want the new file without obliterating the old one. It's a common problem.

Create a thing
Save the thing
Edit the thing
Save again

Why Version Control Software

- Backup and restore
- Synchronization
- Track & Undo changes
- Sandboxing

code_ 5

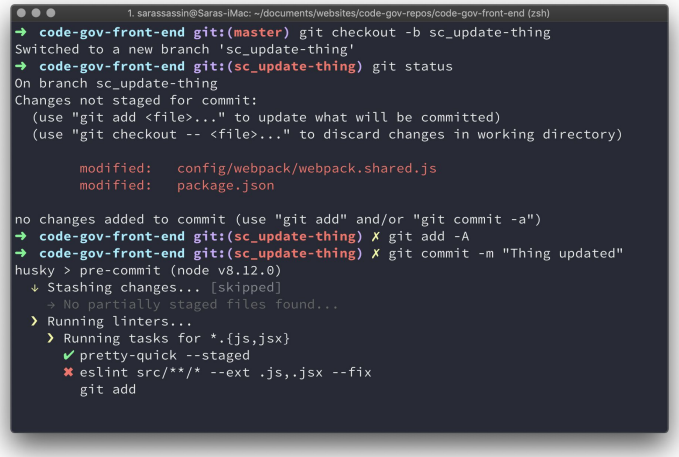
So Why Do We Need A Version Control Software?
Our shared naming system is fine for one-time documents maybe. But it's not something you want to use for software projects.
Version control software have a lot of benefits.

- **Backup and Restore.** Files are saved as they are edited, and you can jump to any moment in time. Need that file as it was on Feb 23, 2007? No problem.
- **Synchronization.** Lets people share files and stay up-to-date with the latest version.
- **Undo.** Monkeying with a file and messed it up? (That's just like you, isn't it?). Throw away your changes and go back to the "last known good" version of the file
- **Track Changes**. As files are updated, you can leave messages explaining why the change happened (stored in the VCS, not the file). This makes it easy to see how a file is evolving over time, and why.
- **Sandboxing**, or insurance against yourself. Making a big change? You can make temporary changes in an isolated area, test and work out the kinks before "checking in" your changes.

Git and GitHub

- Version control systems that let you take snapshots of a project
- Are 2 distinct pieces of software
- Both are awesome

Git



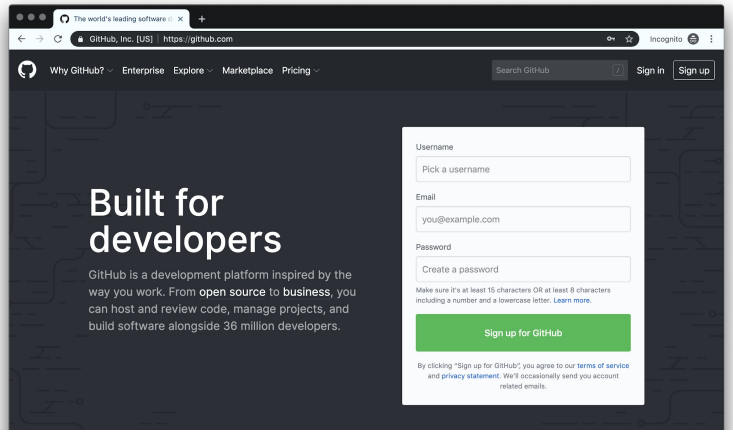
code_ 7

Git is one flavor of version control system that is free and open source and one we commonly use here in GSA and many other agencies do as well.

It's a distributed system, which means each person using a project has a local copy of it on their own machine.

You interact with Git on your computer by using the terminal or command line interface.

GitHub



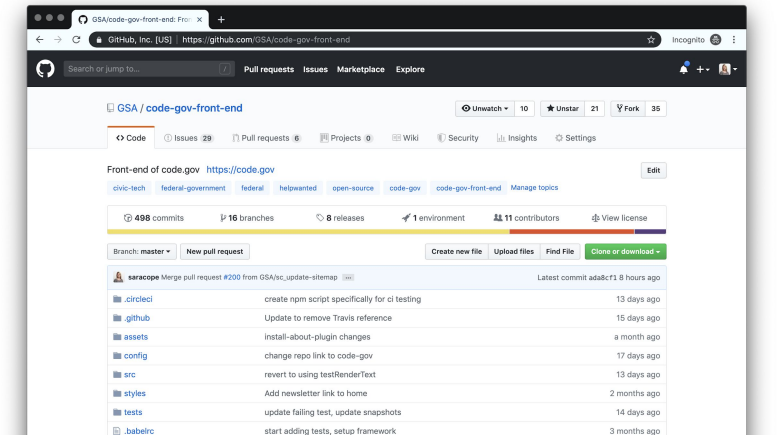
code_

8

GitHub

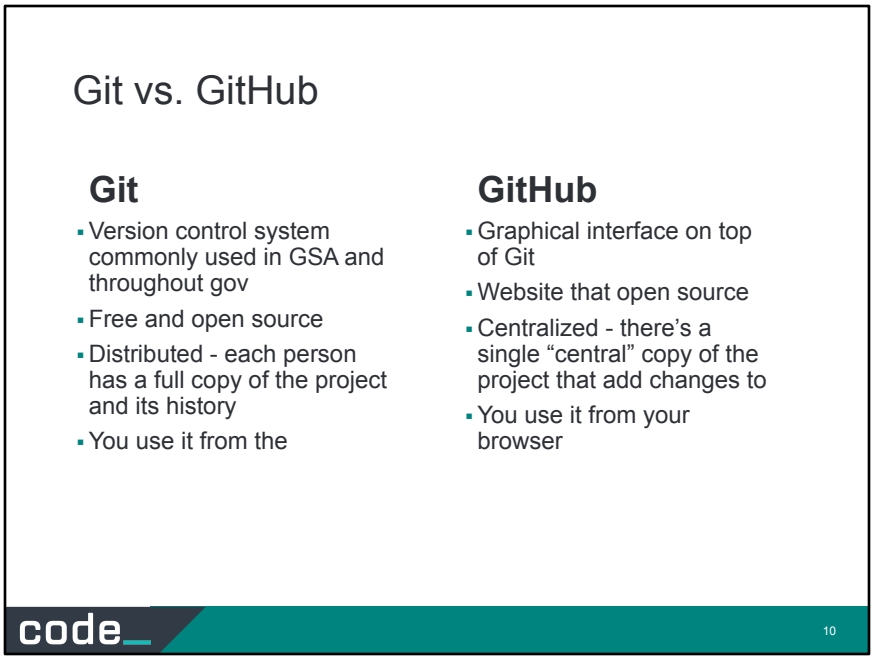
- Website that gives you a
- graphical interface on top of Git
- so it's like the icing on the cake (of git)
- You use it from your browser (or the desktop app they have)

GitHub



code_ 9

- It’s centralized - so there’s a single “central” copy of the project everyone adds changes to



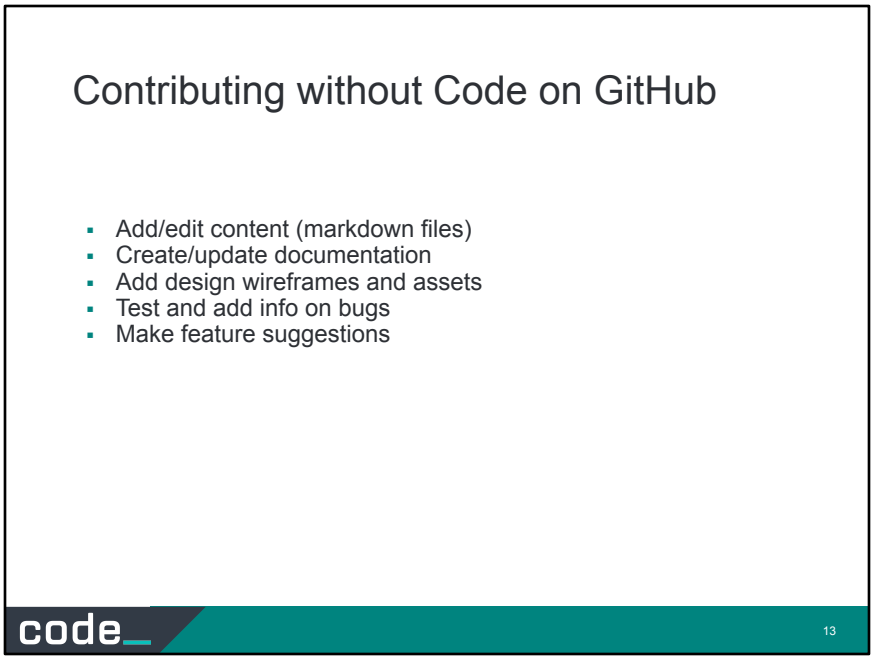
So just to recap…
GitHub is a gui that's build on top of Git. It holds a central copy of your project and you interact with it through the browser, which is what we're going to do today.

Demo Time!

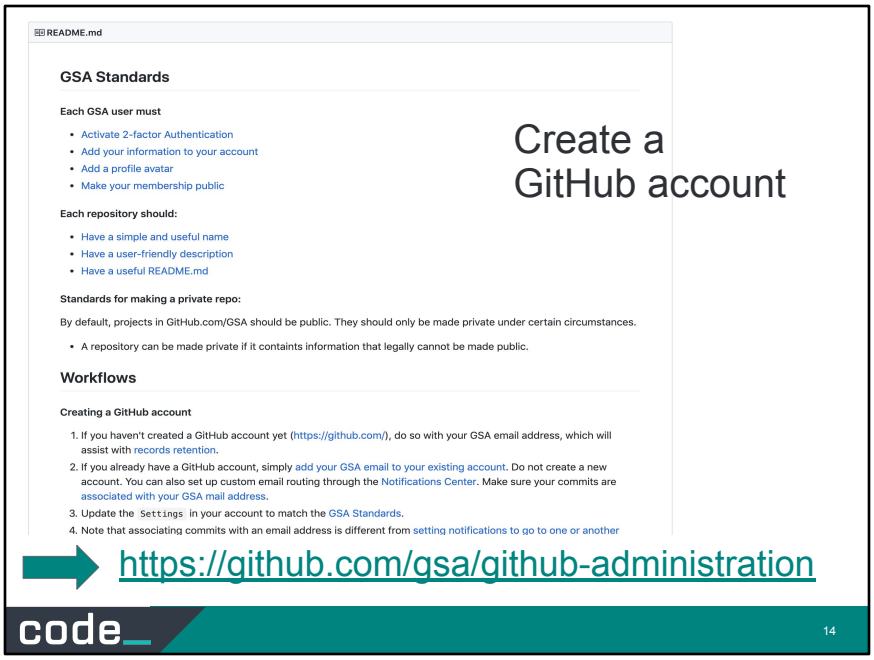
Learn the Lingo

- **Repository (repo)** - the place where your project lives
- **Issues -** start or engage in a convo about a project, often are but reports or feature requests
- **Branch** - copy of the code in the repo that can be modified without worry (like doing “save as”)
- **Master (branch) -** commonly the main branch used in a project, this is often deployed to production
- **Commit -** save the current version of your work
- **Pull request** - Ask that your changed branch be integrated back into the main branch
- **Merge -** take the changes from one branch and apply them to another branch
- **Fork** - copy of the repo that you can modify for yourself
- **Markdown -** a way to format text which is later compiled to code

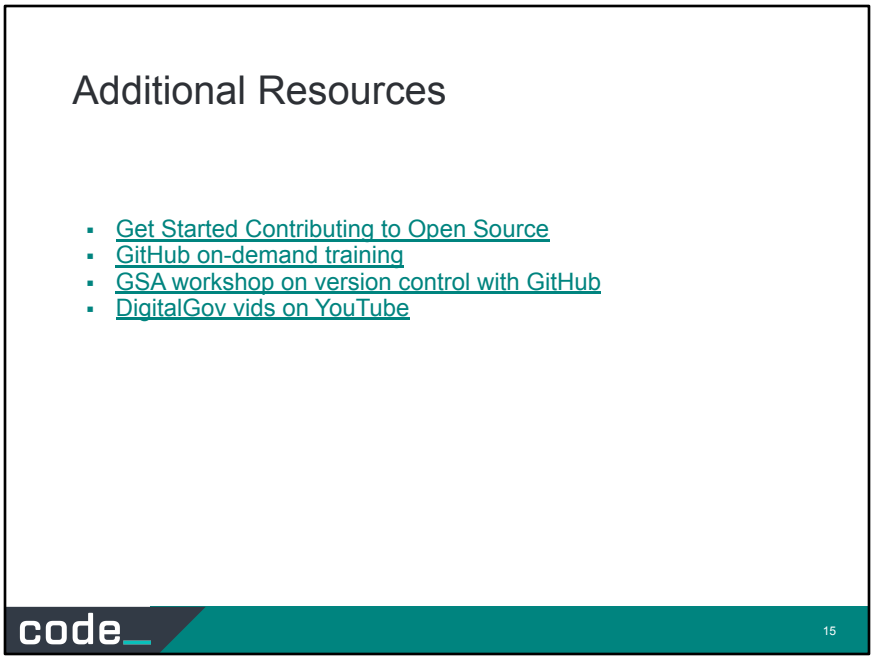
Here’s a recap of some of the base terminology that I just mentioned.



There are several ways to contribute to projects on GitHub without necessarily coding. One of those is what I just demonstrated with adding or editing content and markdown files. You could also create or update documentation for a project that you use if you find that a feature might be lacking or you have an extra tip to add and that's always appreciated. If you are a designer you could add design wireframes are assets for a new feature or add to a discussion.
It's always great if folks see bugs and add those as issues or could go in and test & verify bugs that have been reported by someone else.
Also if you have open source software that you're using like Federalist and you want to make suggestions you can do that as well through opening an issue in their project repository.



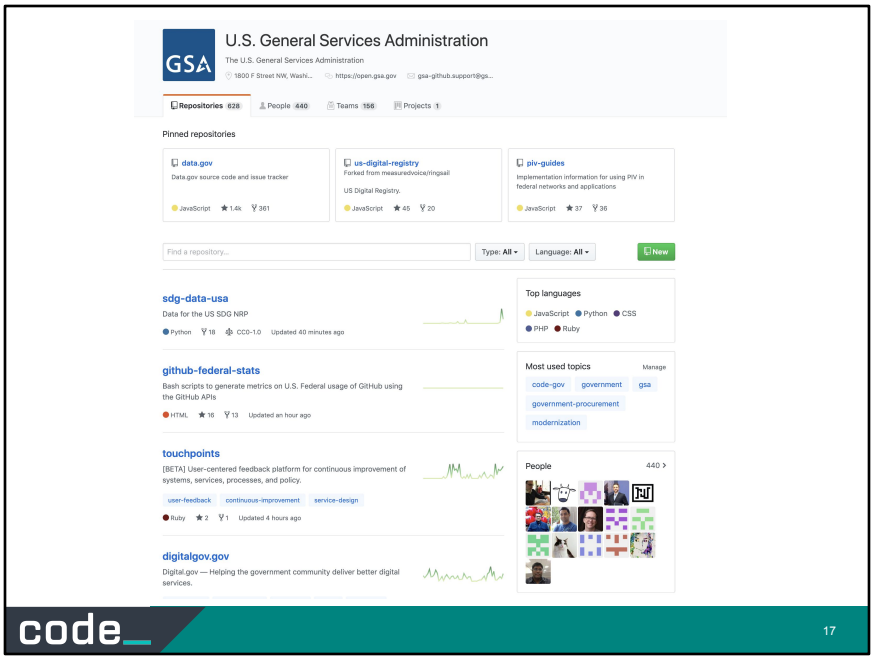
If you don’t already have a GitHub account and need to create one, follow the GSA guidelines for this.



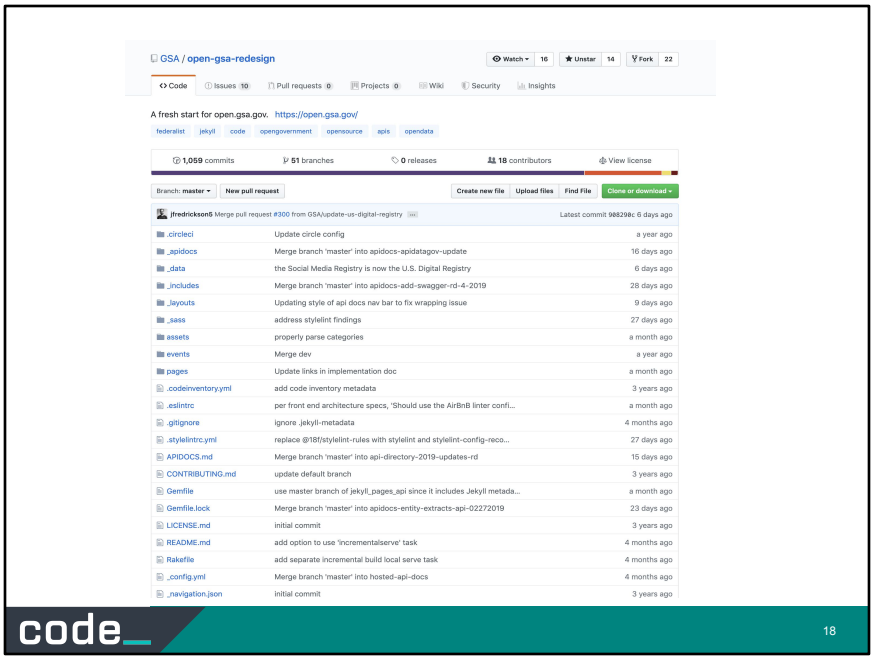
I also have for you here several resources that you can use if you're interested in getting to know GitHub a bit more.
The first one is another talk I did on getting started contributing to open source it mentions some additional things that are good to know like basic files you can expect to see on an open source project and how to make a good commit message and things like that.
The next one is the GitHub actual on-demand training that the company has produced which is a set of tutorials you can do.
The third item here we have is a GSA Workshop that we actually did in person a couple years ago and it takes you through some of the things I demonstrated today but, actually gives you exercises for doing those things yourself.
And then lastly or digitalgov program actually has a lot of good videos around the same topic if you're more of a video person you can go and check those out.

Thank you!

sara.cope@gsa.gov



The majority of GSA open source projects will be under the GSA organization on GitHub.
Here you can explore our projects or search for something specific.



The majority of GSA open source projects will be under the GSA organization on GitHub.
Here you can explore our projects or search for something specific.