Code Yourself In

An Introductory Exercise in Open Source

Courteney Ervin

@courteneyervin

Write/Speak/Code

- Focused on professional women coders

- Focused on skills, mentorship & results

- NYC meetup group & annual conference

Getting Started

- Input == Output
- Ask. Every. Question.
- Give good feedback:
    - ASK: Actionable, Specific, Kind
    - About the work, not the person
    - Within the scope of recipient's skills

What’re we talking here?

- Introductions
- What is Open Source?
- Imposter Syndrome
- Identifying Your Goals
- Finding A Project
- Submitting Helpful Code
- Next Steps

!!!

Who are you?

Groups

!!!

- Find 3-5 other people.
- Introduce yourself using the following activity…

Rose, Bud, Thorn

!!!

- Name
- Rose: *an achievement you're proud of*
- Bud: *something you're excited to learn or work on*
- Thorn: *an area you're trying to improve*

Let’s talk open source.

What is OSS?

- Open Source Software
- Source code is publicly available.
- Source code can be modified or enhanced by anyone.
- Source code is licensed.

What is OSS?

- Linux / Ubuntu
- Apache
- Python & Django
- Ruby & Rails
- PostgreSQL
- MySQL
- Git
- Homebrew
- Mozilla Firefox
- Chromium
- WordPress
- Drupal
- OpenOffice
- GIMP

Open source is awesome.

- New solutions and perspectives
- Better, DRY-er code
- Sharing, collaboration & peer review practices
- Greater accessibility through open infrastructure
- Exponential impact - multiplies knowledge, effort, inspiration, and creativity

"When you work with open source and you discover new requirements not met by the software, it's your shining opportunity to give something back. Rather than just sit around idle waiting for some vendor to fix your problems, you get the unique chance of being a steward of your own destiny. To become a participant in the community rather than a mere spectator."

–David Heinemeier Hansson

Why contribute?

- Learn & improve skills
- Get feedback on your code
- Build credibility & reputation
- Find a job
- Give back

?

Why haven't you contributed?

Why not contribute?

- I'm not good enough.
- I don't have time.
- I don't know where to start.
- I don't want my code to be criticized.

Why not contribute?

I’m not good enough.

- I don’t have time.
- I don’t know where to start.
- I don’t want my code to be criticized.

Imposter Syndrome

“Oh crap!

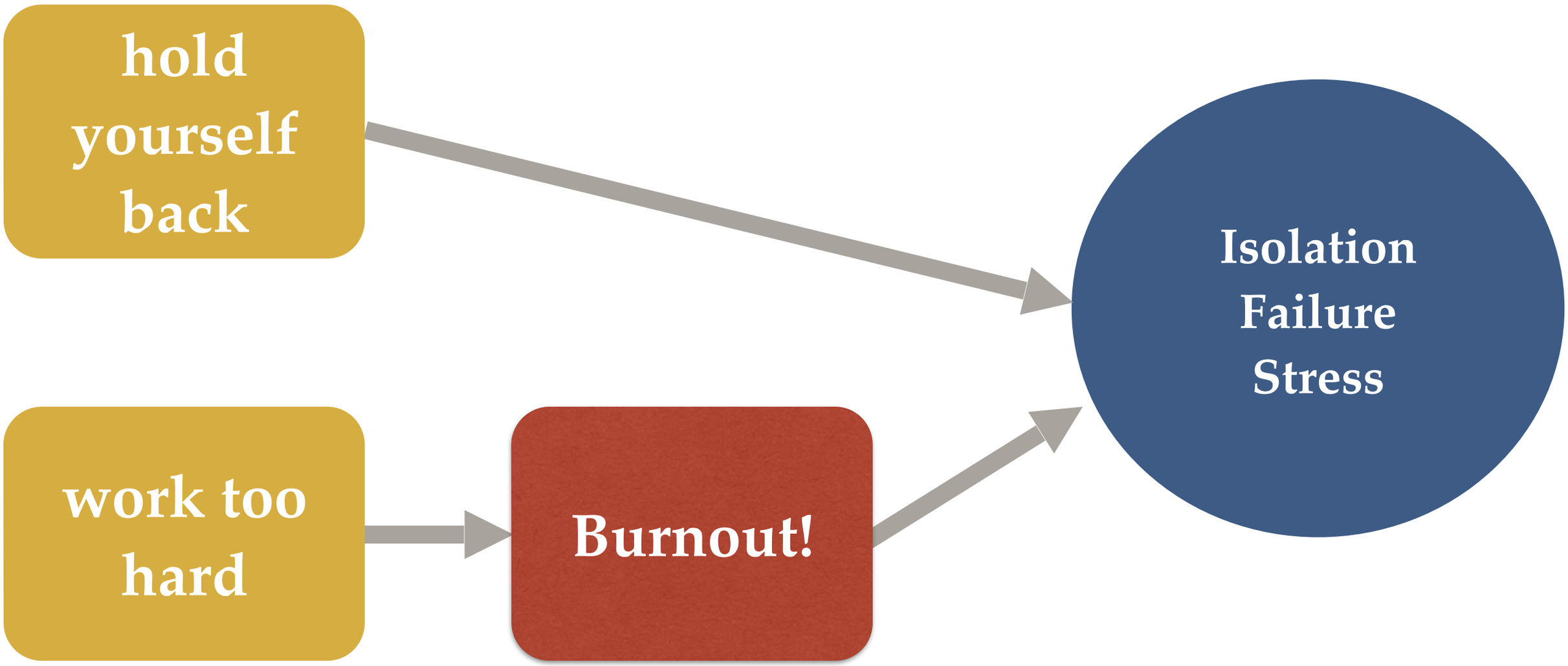
Everyone is going to realize I have no idea what I’m doing.”

What is Imposter Syndrome?

- Inability to internalize accomplishments
- Feeling of "I shouldn't be here"
- Convinced you're a fraud
- Dismissing successes

It matters.

A Self-fulfilling Prophecy

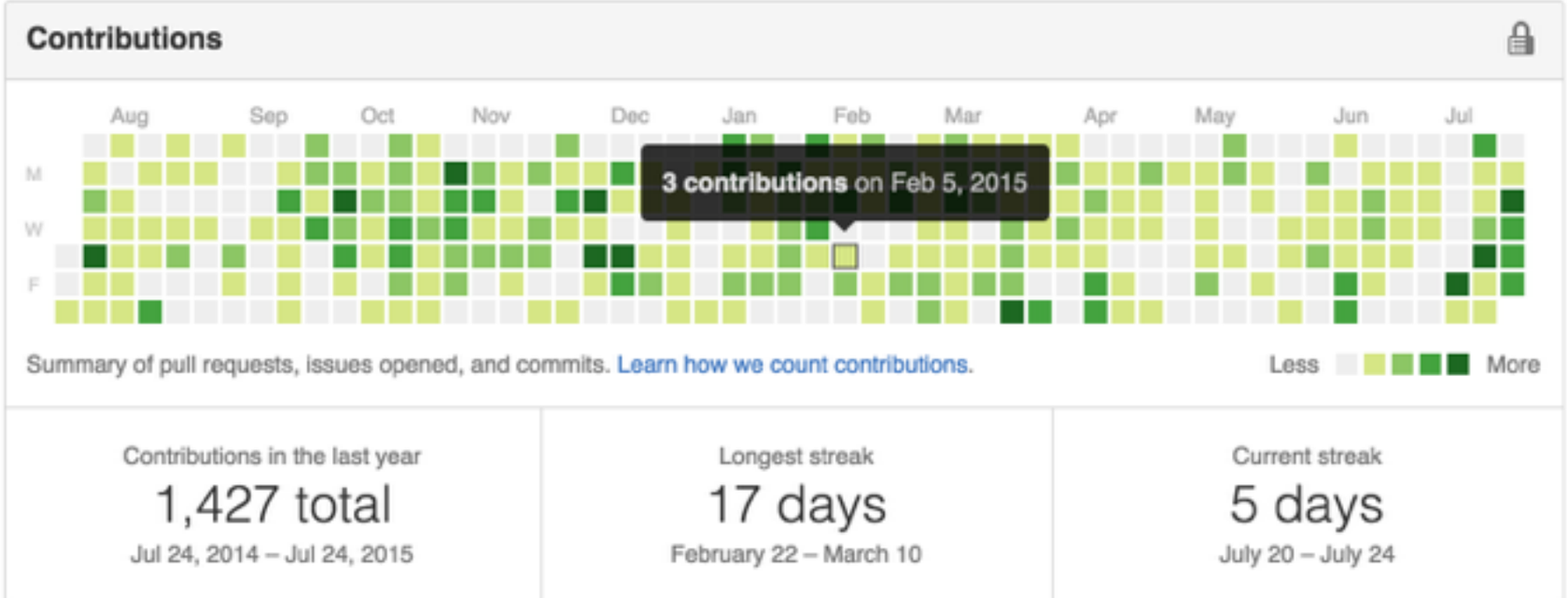


Tech is communal.

- Teams
- Pair programming
- Stack Overflow
- Conferences
- Books & blogs
- OSS

OSS > Imposter Syndrome

- Help others
- Look at data



- If not you, who?

"Open source is like being an adult

It's magical until you realize no one knows what the **hell they're doing**"

–Zach Holman

http://zachholman.com/talk/open-source-misfeasance/

Thinking Beyond Code

- Promotion
- Star/Watch on GitHub
- Feature ideas
- Documentation edits
- Design reviews
- Typo removal
- Bug reports
- Syntax fixes
- Seed data
- Translations
- Library updates
- Tests
- Code reviews
- Code?

!!!

YES.

You, too, can open source.

Why not contribute?

~~I'm not good enough.~~

I don't have time.

- I don't know where to start.
- I don't want my code to be criticized.

Why are you here today?

What are your goals?

- Visibility
- Reputation
- Learning
- Job hunting
- Networking
- Supporting the community

Identify Your Goals

!!!

- Find someone to pair with.
- Ask: *Why do you want to write open source code?*
- 5 Whys
- Switch it up

!!!

SMARTen Up

- Specific
- Measurable
- Achievable
- Realistic
- Timed

?

Share

Why not contribute?

~~I'm not good enough.~~

~~I don't have time.~~

I don't know where to start.

- I don't want my code to be criticized.

Choosing a Project

Where do I look?

- Your own tools
- Your communities and networks
- github.com/explore
- CodeMontage
- OpenHatch
- CodeTriage

Brainstorm Projects

!!!

- Don’t use the internet.
- List 10 projects or leads.
- Use these questions:
  - What technologies do I already benefit from?
  - Does my preferred project have any clear constraints?
  - Do any communities or devs I know have projects?
  - Is there a particular company I admire or want to work for?

What do I look for?

- recent and consistent activity
- documentation on contributing, setting up, and running the codebase
- tests and tests coverage
- community — google groups, IRC, forums, email
- labeled issues

Creating a Project

- Look for similar projects to support.
- License your code: choosealicense.com
- Create documentation.
- Be kind.

Find a Project

!!!

Reminder of online resources:

- github.com/explore
- CodeMontage
- OpenHatch
- CodeTriage

How can I contribute?

- Listen to mailing list and look at issues.
- Reach out to maintainer and ask for suggestions.
- Install, run, test, and improve documentation as a result.
- Review or QA existing pull requests, especially on different browsers / OS / devices.

Contributing Code

- QA bugs - add a detailed bug report (or close)
- Test beta versions
- Fix bugs
- Write tests
- Deal with dependencies

Contribute Documentation

- Update, install, run, test documentation
- Write an example or tutorial for documentation
- Write documentation for areas where it doesn't exist (*hint: check the tests!*)
- Prettify it! Documentation or project website (*hint: check GitHub Pages*)

Other Contributions

- Copyediting - fix typos, freshen copy
- Marketing - twitter, social media, blog posts
- Design - the website, the logo, the app interface
- Share - answer questions on Stack Overflow, give a talk

Find an issue

!!!

- Go to your project's GitHub page.
- Look for an issue that you can contribute to.
- Consider why you chose the project. Is there:
  - a feature you'd like to add?
  - a bug you want to fix?
  - a design element you hate?
  - a strong desire to see it succeed?

Managing the Code

Version Control

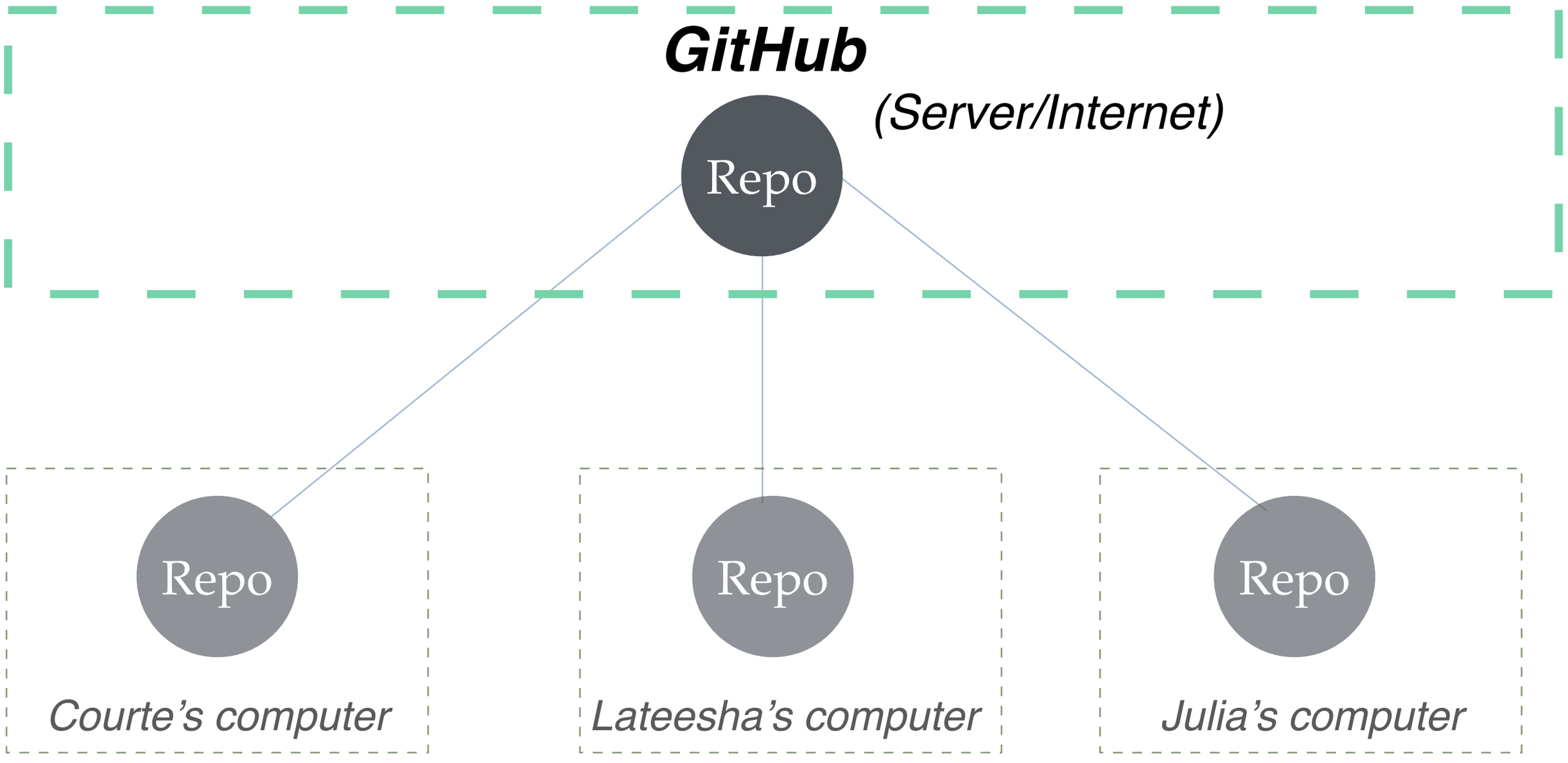
- Multiple developers work on one codebase
- Ensures that work won't be overwritten
- Stores a history of code changes
- Git & Subversion (svn)

Git

- Free & OSS
- Decentralized version control
- Fast & low memory
- Easy branching

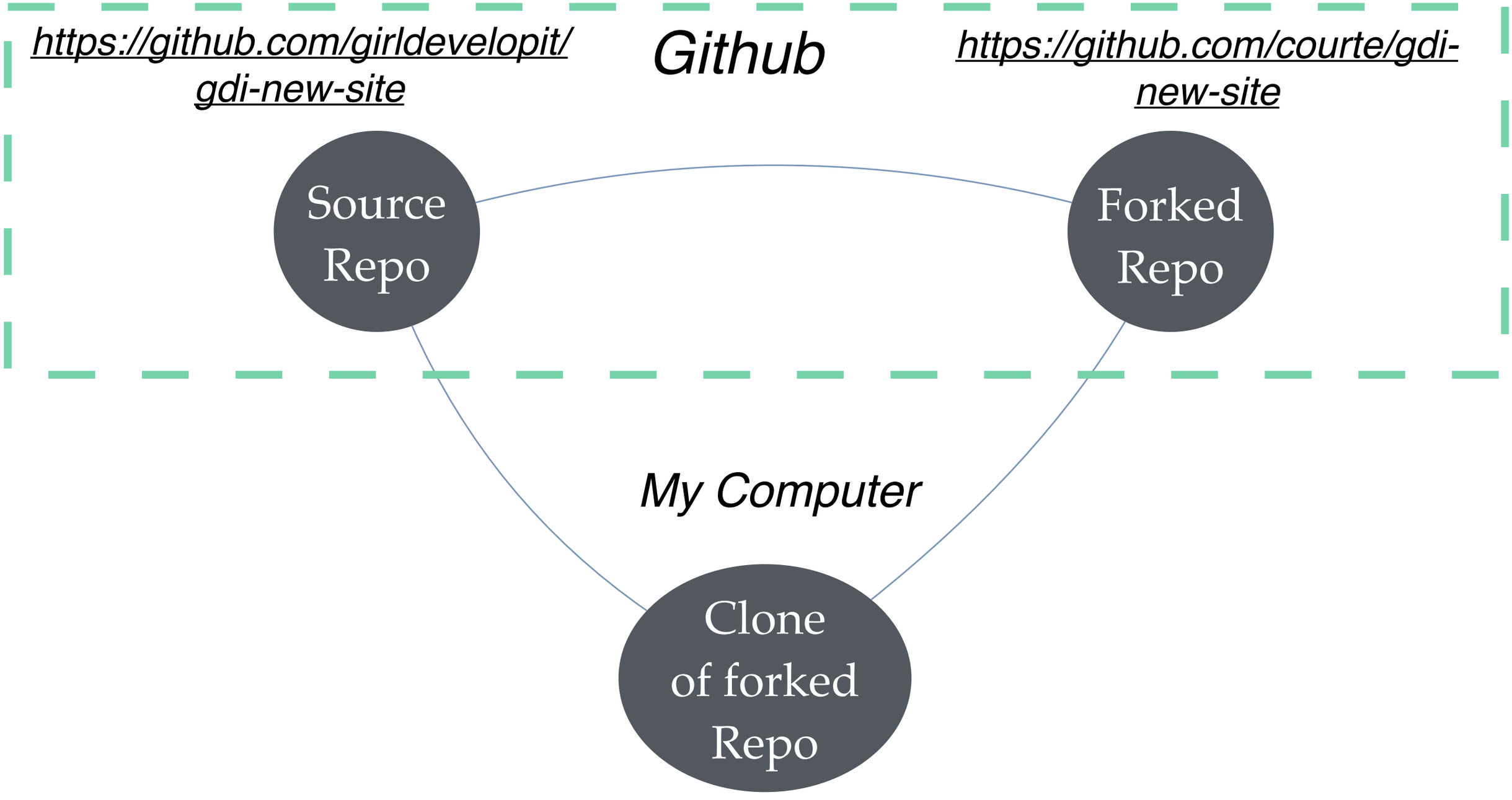
- Hosted source control, based on git
- Lives in the cloud
- Used by open source projects, personal projects, and proprietary projects
- Demo time!

GitHub Repo

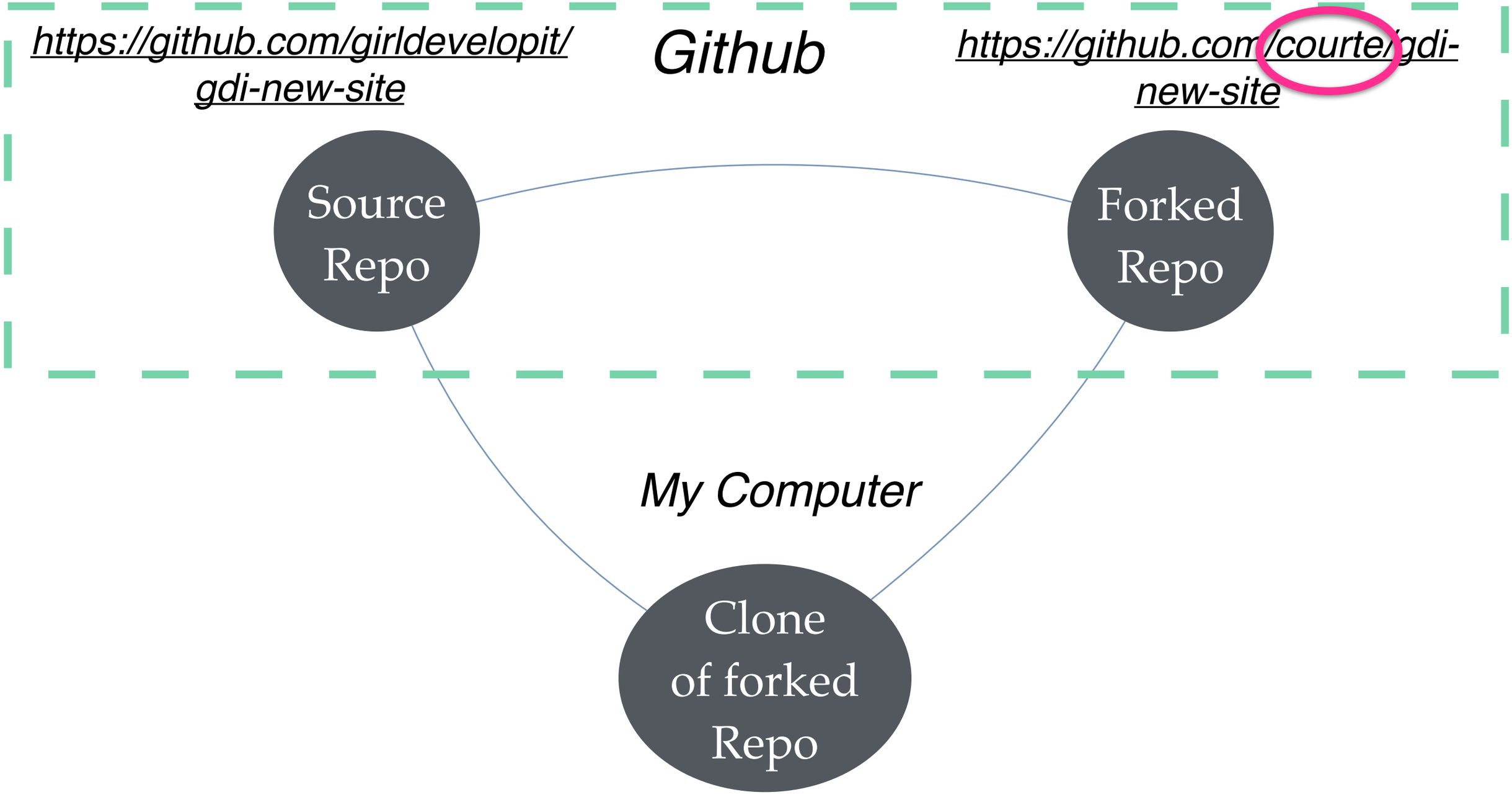


https://speakerdeck.com/lornajane/git-githu

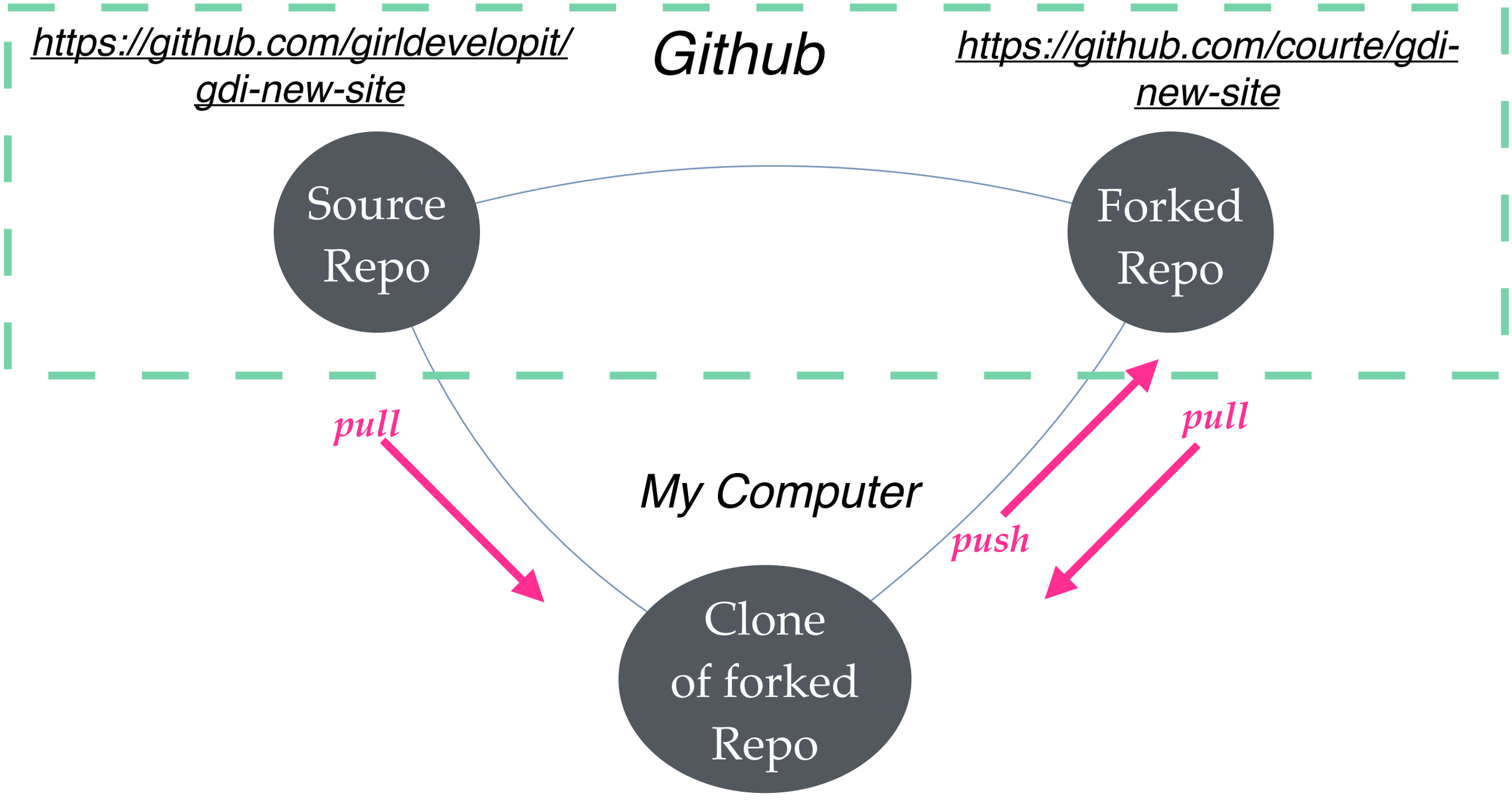
Git & GitHub for OSS



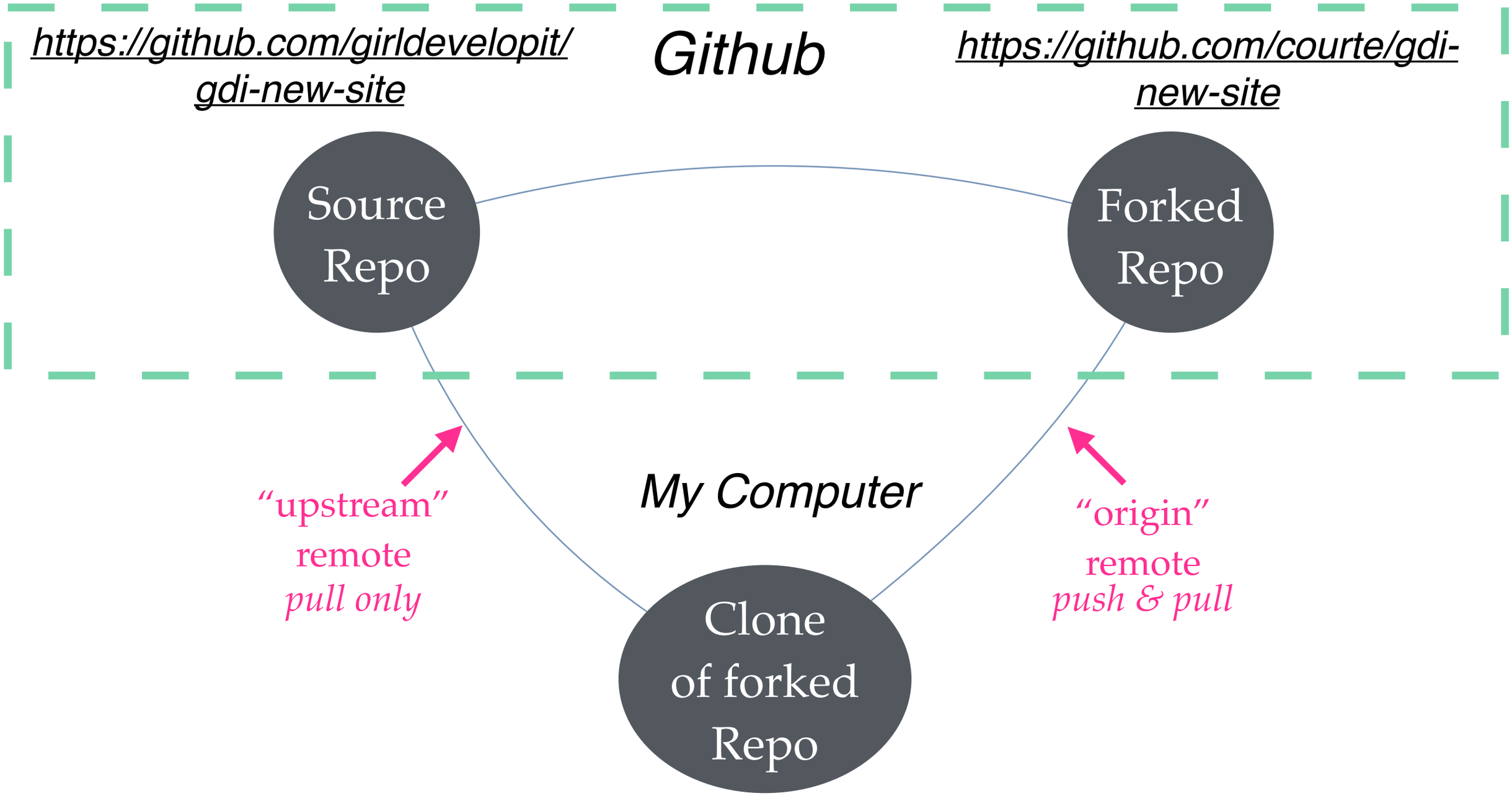
Git & GitHub for OSS



Git & GitHub for OSS



Git & GitHub for OSS



Clone Your Project!

!!!

- Go to project's GitHub page
- Fork project
- Clone project to your computer:

```
git clone <URL>
```

- Add "upstream" remote:

```
git remote add upstream <URL>
```

Writing the Code

Getting Started

1. Clone your project
2. Install your project
3. Get help when you need it
4. Make your contribution

Pro tip:

Record your errors & fixes!

Set Up Your Project

- Follow documentation
- Install dependencies
- Run the code
- Run the tests
- 😭 😭 😭 as needed

Set Up Your Project

- Follow documentation
- Install dependencies
- Run the code
- Run the tests
- ~~😭 😭 😭 as needed~~

Getting Help

- Embrace a growth mindset
- Ask the internet: research, research, research
- Ask your peers: find an accountability buddy
- Ask a mentor: pair through the tricky parts

Your First Contributions

1. Respond to issue/ticket telling maintainer you're working on it.
2. Create a branch for the issue.
3. Write tests and code.
4. Get feedback.
5. Fix & commit *one thing at a time*.
6. Push branch to GitHub
7. Create a pull request & link to the issue/ticket.

Why not contribute?

~~I'm not good enough.~~

~~I don't have time.~~

~~I don't know where to start.~~

I don't want my code to be criticized.

Writing Better Code

@courteneyervin / @girldevelopit

Good Code

- Follows the code style guide (if available)
- Follows existing coding standards (naming, indentation, etc.)
- Is self-documenting: seek clarity over cleverness
- Has received feedback
- Fixes and commits *one. single. thing.* at a time
- Doesn't have comments or debugging statements

Good Debugging

- Read the error message (if available)
- Isolate the error, down to the exact line, using debugger or print statements
- Ask: What is the input? What is the output? What did you expect it to be?
- Google the error message
- Delete portions until the code works again.

Good Commit Messages

https://robots.thoughtbot.com/5-useful-tips-for-a-better-commit-message

- Short summary for the first line (50 chars)
- Word wrap at 72-80 characters
- One commit per change
- Answer:
    - Why is this change necessary?
    - How does it address the issue?
    - What side effects does it have?
- Include a link to the issue/ticket any any external sources.

Good Pull Requests

- Make after a pull from upstream

```
git pull --rebase upstream master
```

- Run tests without errors
- Include screenshots for ANY front end changes
- Receive a code review

Good Bug Reports

- Include the exact error you're getting, including the error message
- Share detailed, repeatable steps for the error
    - What did you click on? Did you use Tab, enter, command, or or another key?
    - What page did you start on? What page did you end up on?
- List technical details: browser, OS, device, language
- Show the error and/or steps taken through screenshots or video

Good Code Reviews

1. Read the issue/ticket.
2. Review contributing guidelines & code style guides.
3. Be thorough and give yourself time.
4. Focus on the code and not the author.
5. Ask questions.
6. Run the code.
7. Sign off & high-five! 🎉👐✨👍💯

Why not contribute?

~~I'm not good enough.~~

~~I don't have time.~~

~~I don't know where to start.~~

~~I don't want my code to be criticized.~~

!!!

3 Next Steps!

?

Aha Moments

Thank you!

@courteneyervin
courteney.ervin@gmail.com