

# Programming with Python

Konstantins Tarasjuks

# Agenda

- ▶ **Version Control**
- ▶ GIT
- ▶ GIT clone
- ▶ GIT commit
- ▶ GIT create branch
- ▶ Exercises

# Version Control

- ▶ Version control systems are a category of software tools that help a software team manage changes to source code over time.
- ▶ Version control software keeps track of every modification to the code in a special kind of database
- ▶ Version control helps teams solve these kinds of problems, tracking every individual change by each contributor and helping prevent concurrent work from conflicting

# Benefits

- ▶ Developing software without using version control is risky, like not having backups. Version control can also enable developers to move faster and it allows software teams to preserve efficiency and agility as the team scales to include more developers

# History

- ▶ A complete long-term change history of every file. This means every change made by many individuals over the years.
- ▶ Changes include the creation and deletion of files as well as edits to their contents.
- ▶ Different VCS tools differ on how well they handle renaming and moving of files. This history should also include the author, date and written notes on the purpose of each change.

# Branching

- ▶ Branching and merging. Having team members work concurrently is a no-brainer, but even individuals working on their own can benefit from the ability to work on independent streams of changes.
- ▶ Creating a "branch" in VCS tools keeps multiple streams of work independent from each other while also providing the facility to merge that work back together, enabling developers to verify that the changes on each branch do not conflict
- ▶ Many software teams adopt a practice of branching for each feature or perhaps branching for each release, or both

# Traceability

- ▶ Being able to trace each change made to the software and connect it to project management and bug tracking software such as Jira, and being able to annotate each change with a message describing the purpose and intent of the change can help not only with root cause analysis and other forensics
- ▶ Having the annotated history of the code at your fingertips when you are reading the code, trying to understand what it is doing and why it is so designed can enable developers to make correct and harmonious changes that are in accord with the intended long-term design of the system

# Agenda

- ▶ **Version Control**
- ▶ **GIT**
- ▶ GIT clone
- ▶ GIT commit
- ▶ GIT create branch
- ▶ Exercises



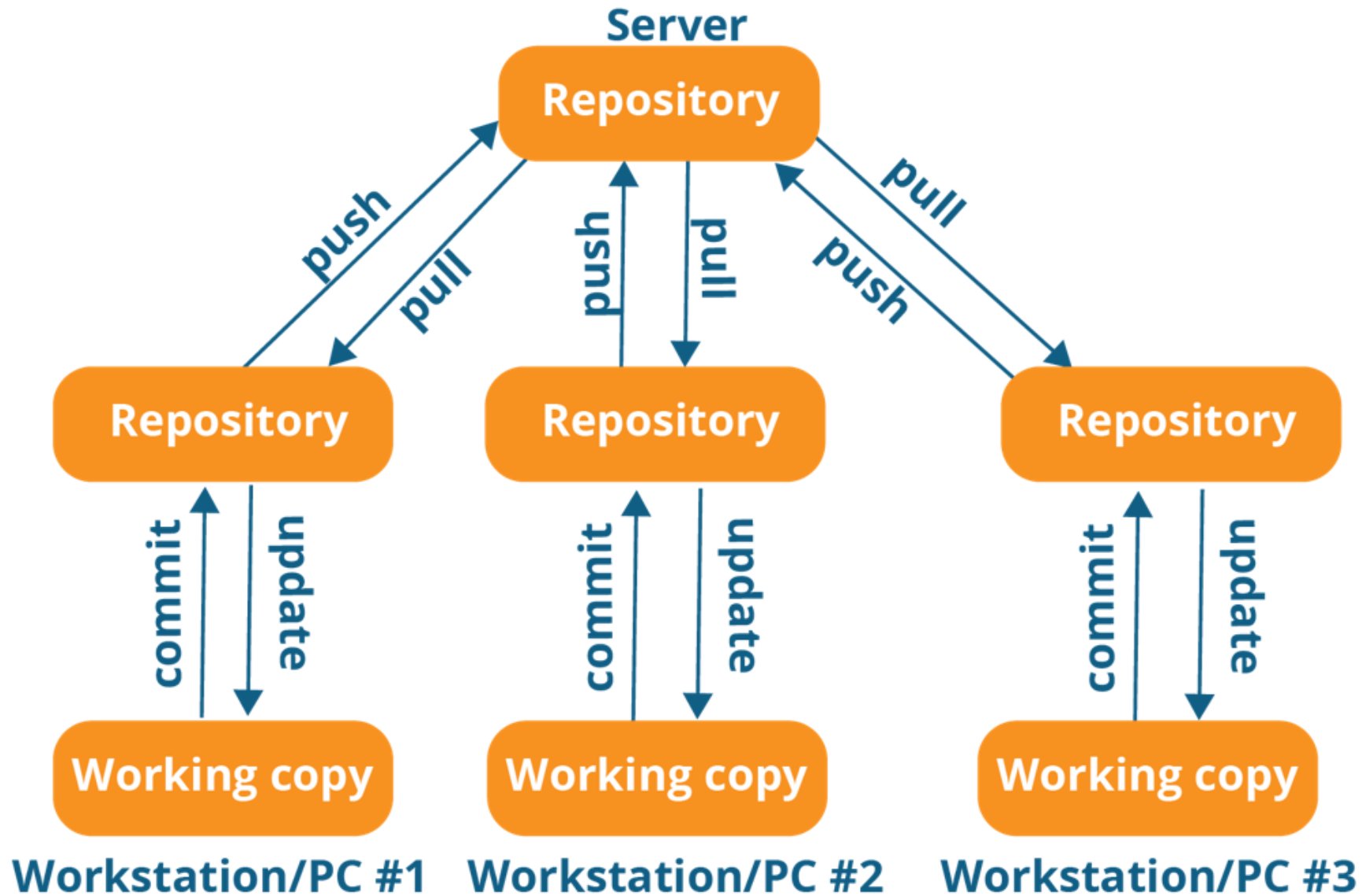
# What is a GIT?

- ▶ By far, the most widely used modern version control system in the world today is Git.
- ▶ Git is a mature, actively maintained open source project originally developed in 2005 by Linus Torvalds, the famous creator of the Linux operating system kernel.
- ▶ A staggering number of software projects rely on Git for version control, including commercial projects as well as open source

# What is a GIT?

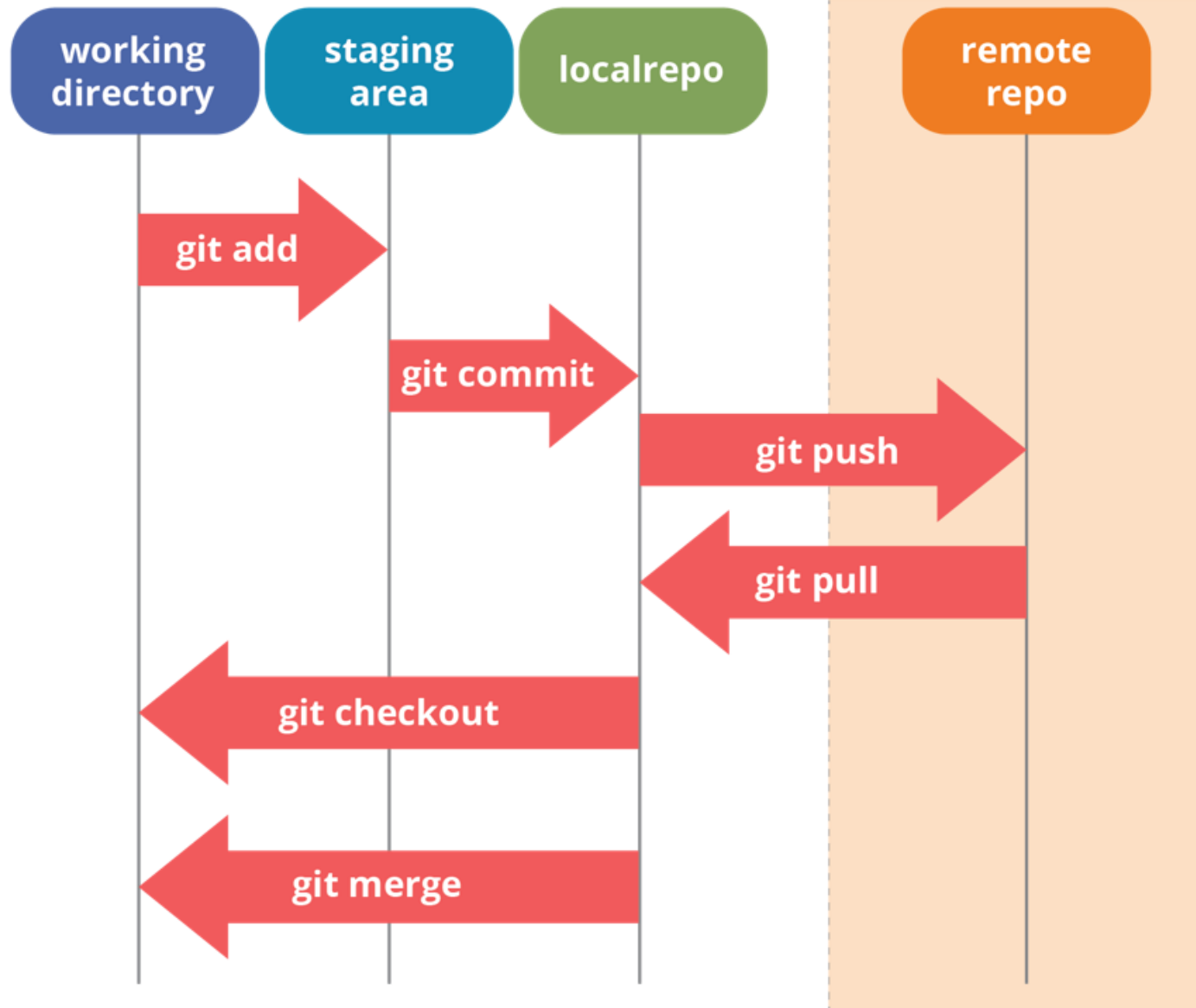
- ▶ Git is a very well supported open source project with over a decade of solid stewardship. The project maintainers have shown balanced judgment and a mature approach to meeting the long term needs of its users with regular releases that improve usability and functionality. The quality of the open source software is easily scrutinized and countless businesses rely heavily on that quality.
- ▶ Git enjoys great community support and a vast user base. Documentation is excellent and plentiful, including books, tutorials and dedicated web sites. There are also podcasts and video tutorials

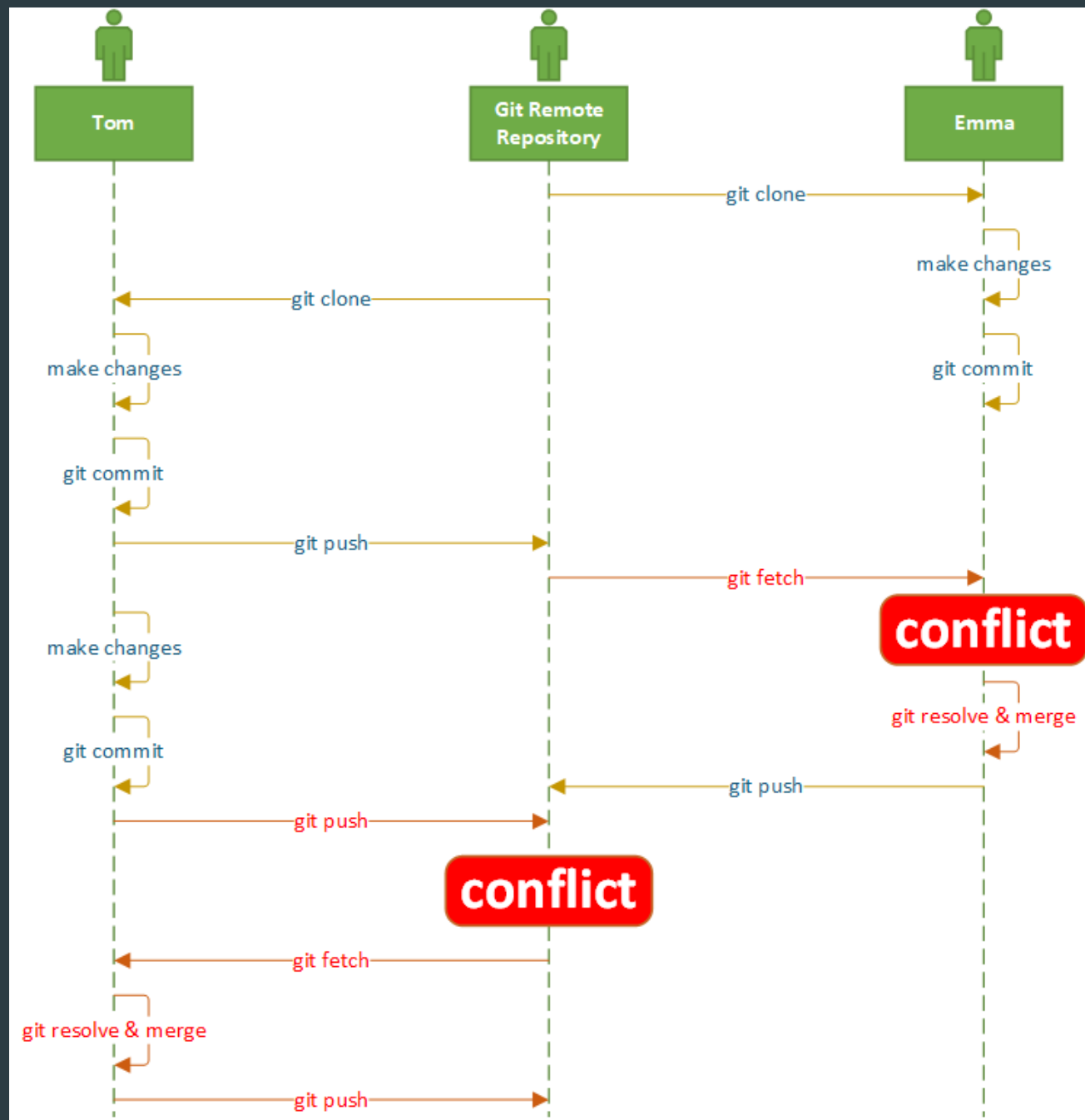
# Distributed version control system

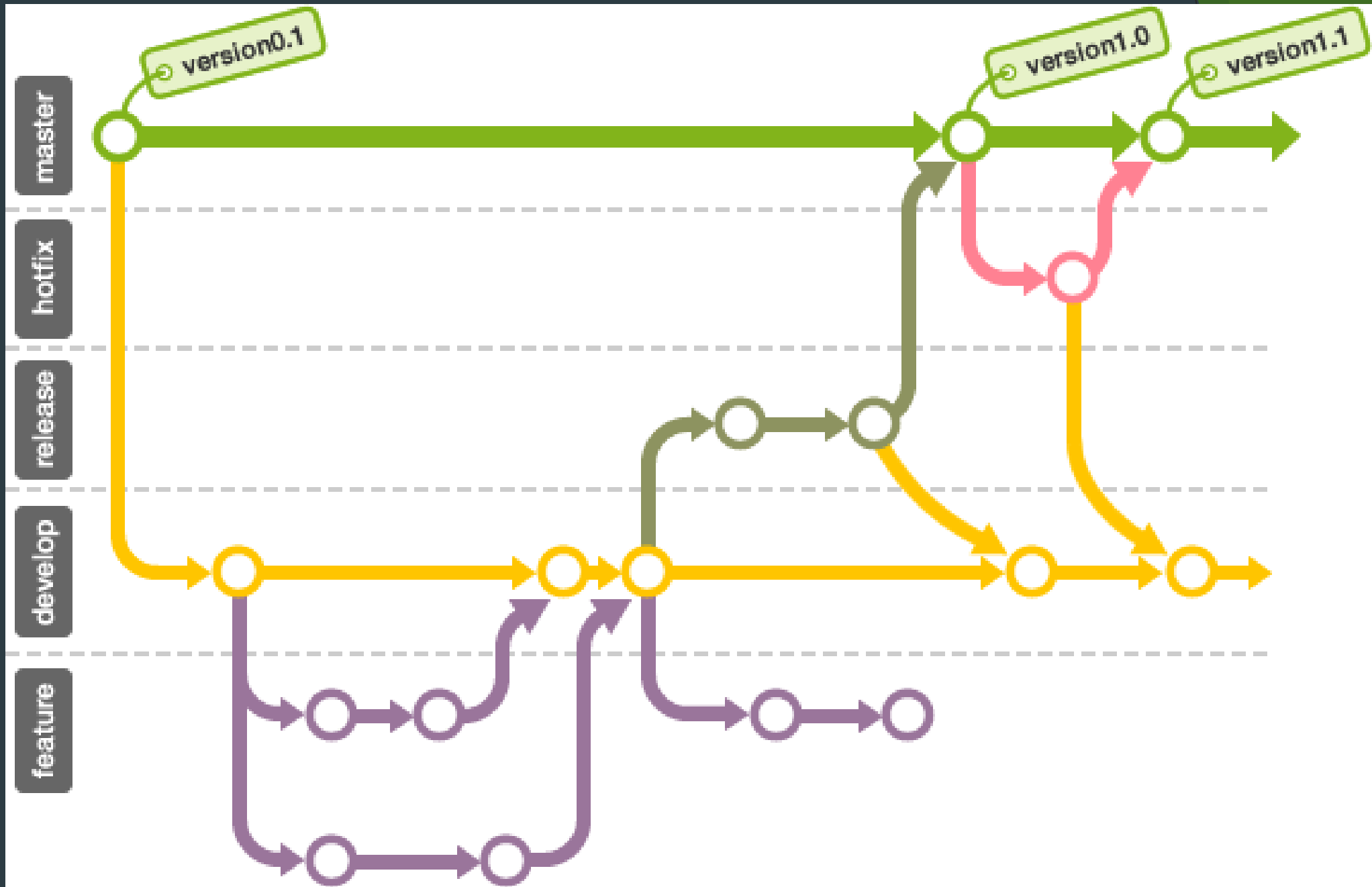


## Local

## Remote







# Agenda

- ▶ **Version Control**
- ▶ **GIT**
- ▶ **GIT clone**
- ▶ **GIT commit**
- ▶ **GIT create branch**
- ▶ **Exercises**

# GIT create repo

- ▶ 0. Choose repo name - camelCase or hyphens - sectionSection2 or section1-section2
- ▶ 1. Public - anybody can view your repo
- ▶ 2. Only you and collaborators can view your repo
- ▶ 3. Readme file - description and how to launch your project
- ▶ 4. .gitignore - ignore specific files and folders for git - like automatically build files or IDE profile configuration


## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

---

Owner \*

Repository name \*

 ktarasjuks


 /

Great repository names are short and memorable. Need inspiration? How about [furry-waffle?](#)


Description (optional)

---

1

☒  **Public**  
Anyone on the internet can see this repository. You choose who can commit.

2

☐  **Private**  
You choose who can see and commit to this repository.

---

**Initialize this repository with:**  
Skip this step if you're importing an existing repository.

3

☐ **Add a README file**  
This is where you can write a long description for your project. [Learn more.](#)

4


**Add .gitignore**  
Choose which files not to track from a list of templates. [Learn more.](#)  

.gitignore template: None ▾

**Choose a license**  
A license tells others what they can and can't do with your code. [Learn more.](#)  

License: None ▾

---

 You are creating a public repository in your personal account.

---

Create repository



# GIT create repo

- ▶ 1. Menu - buttons
- ▶ 2. Https - anybody can download git using this link SSH - if ssh key is generated and save to github can use this link
- ▶ 3. Commands to copy to console to init Git repo in existing project
- ▶ 4. if git repo has been init - link it to your Git repo
- ▶ 5. Import to your IDE
- ▶ 6. in Settings option you can delete your repo

The screenshot shows the GitHub repository page for 'ktarasjuks / Python-sample-git'. The repository is public. The navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings (annotated with a red '6'). The main content area has four sections for repository creation, each with a red annotation:

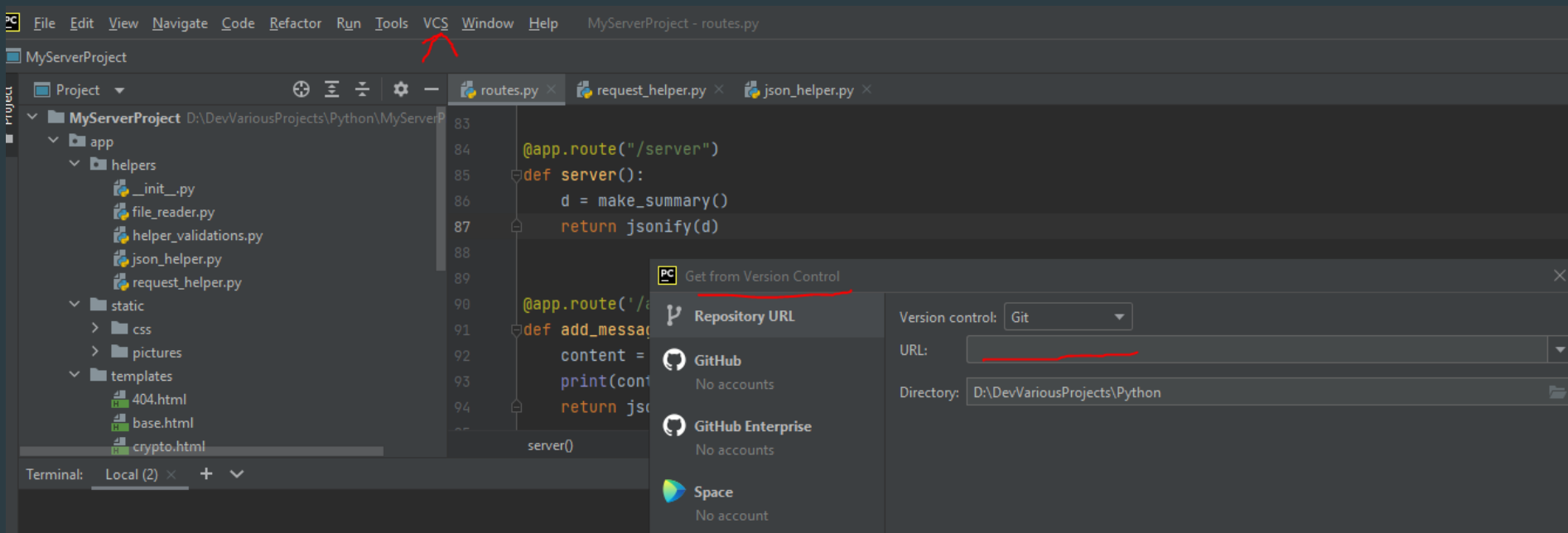
- 1** (under the navigation bar): The repository name and public status.
- 2** (under 'Quick setup'): The 'HTTPS' button and the repository URL.
- 3** (under '...or create a new repository on the command line'): The command line instructions for creating a new repository.
- 4** (under '...or push an existing repository from the command line'): The command line instructions for pushing an existing repository.
- 5** (under '...or import code from another repository'): The 'Import code' button.

At the bottom, there is a 'ProTip!' message: 'Use the URL for this page when adding GitHub as a remote.'

- ▶ <https://www.theserverside.com/blog/Coffee-Talk-Java-News-Stories-and-Opinions/How-to-push-an-existing-project-to-GitHub>

# GIT CLONE - existing project

- ▶ VCS -> Get from Version control
- ▶ Copy HTTPS url
- ▶ Enter your Github credentials



# Agenda

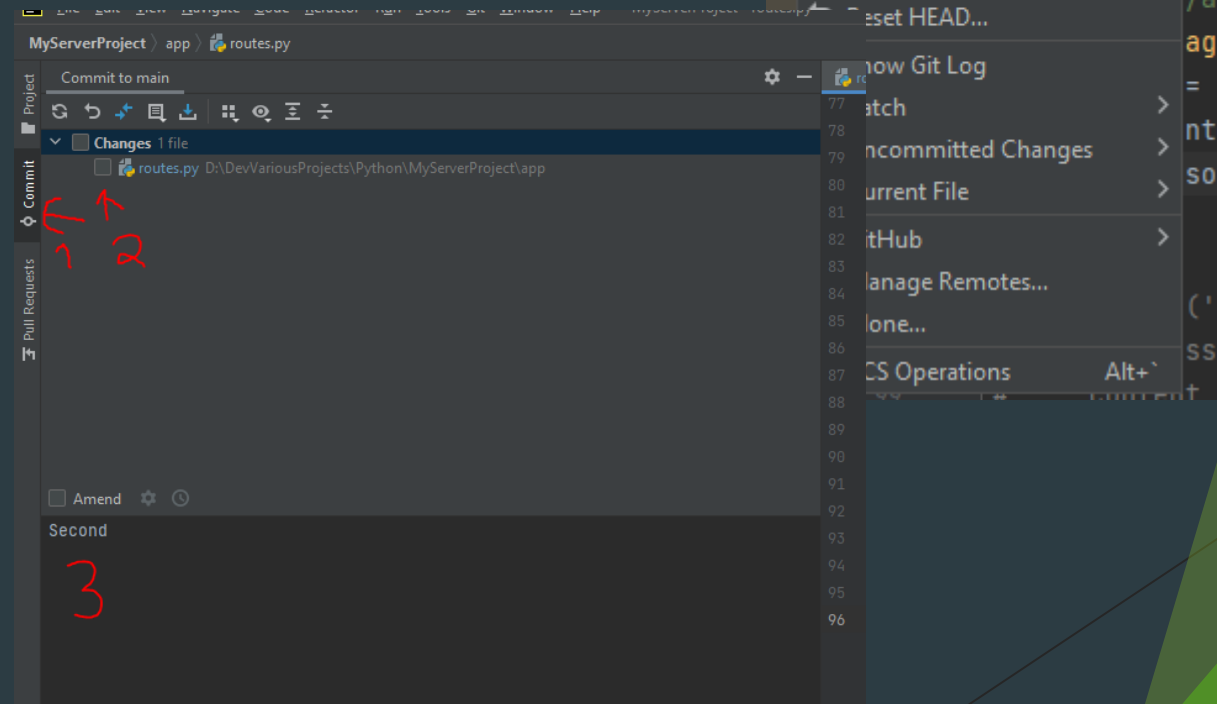
- ▶ **Version Control**
- ▶ **GIT**
- ▶ **GIT clone**
- ▶ **GIT commit**
- ▶ **GIT create branch**
- ▶ **Exercises**

# GIT commit

- ▶ Commit is required to save your GIT changes to Local repo
- ▶ Push is required to save your GIT changes to Remote repo

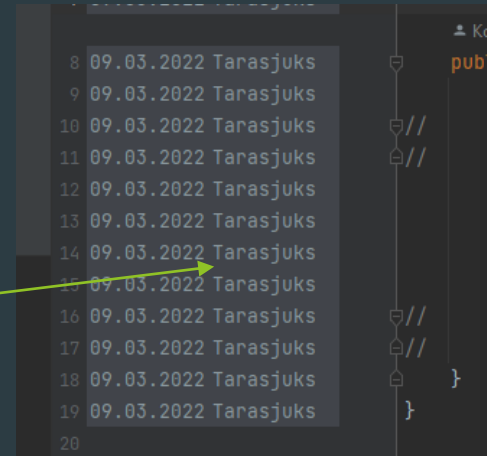
1. Option 2 - where to find changes in your GIT to commit changes
2. Files that have been changed
3. Commit message - List all the changes that were done

Option 1. - where to find GIT commit command



# GIT global config

Username that will displayed when you commit something



Username to be displayed in github



```
$ git config --global user.name "John Doe"
$ git config --global user.email johndoe@example.com
```

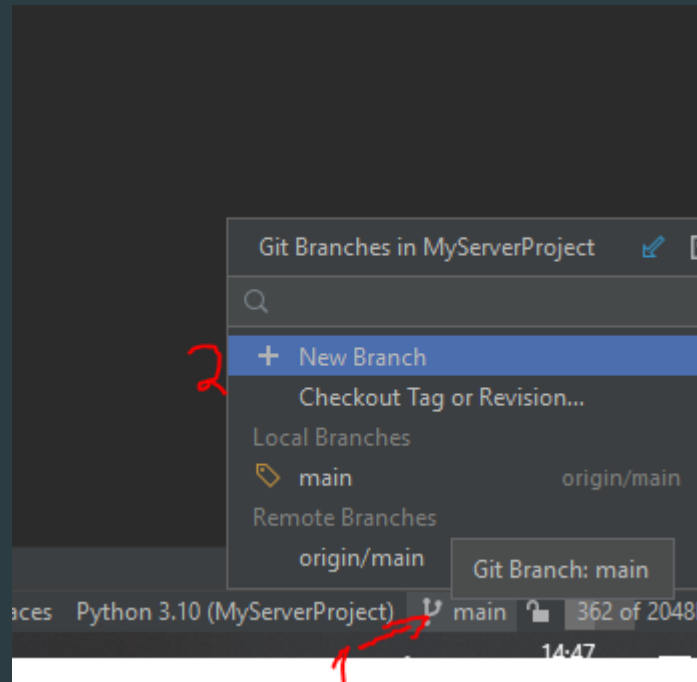
tasks for lecture 10	Konstantins	13.04.2022 20:49
tasks for lecture 9	Konstantins	06.04.2022 21:11
Task lecture 8 Update folders lecture 8	Konstantins	30.03.2022 20:30
Task for lecture 7 Cheat sheet Lecture 8	Konstantins Tarasjuks	24.03.2022 20:03
Task for lecture 6	Konstantins Tarasjuks	16.03.2022 22:01
Task for lecture 5	Konstantins Tarasjuks	09.03.2022 21:20
Tasks for lecture 4	Konstantins Tarasjuks	02.03.2022 22:04
Merge remote-tracking branch 'origin/main'	Konstantins Tarasjuks	23.02.2022 22:04
Tasks for lecture 3	Konstantins Tarasjuks	23.02.2022 22:04
Apple Pie doc	Konstantins*	18.02.2022 20:57

# Agenda

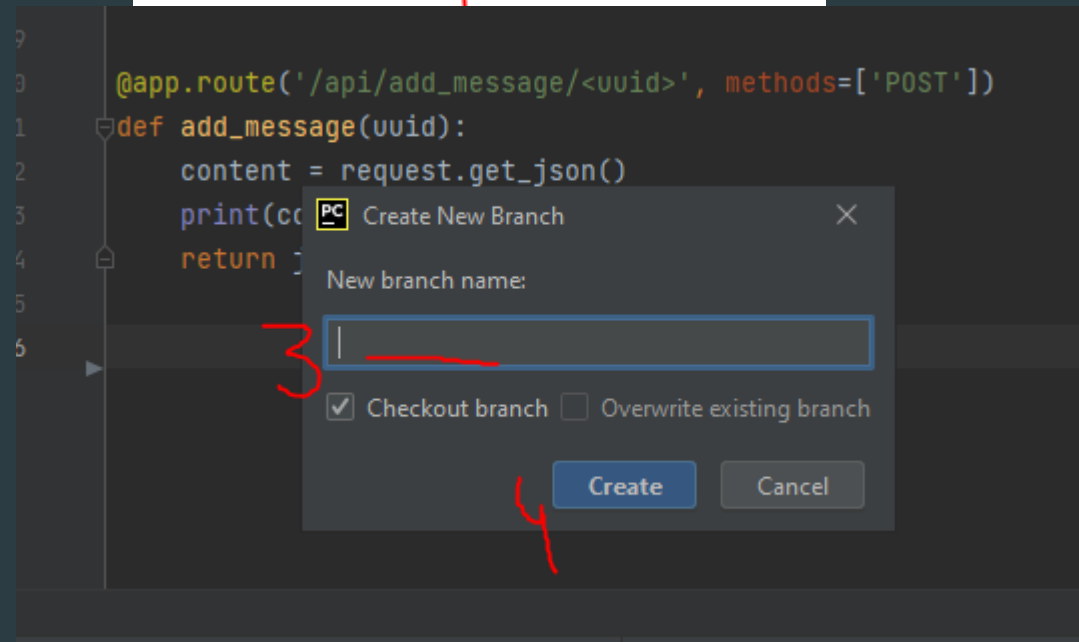
- ▶ Version Control
- ▶ GIT
- ▶ GIT clone
- ▶ GIT commit
- ▶ GIT create branch
- ▶ Exercises

# GIT branch

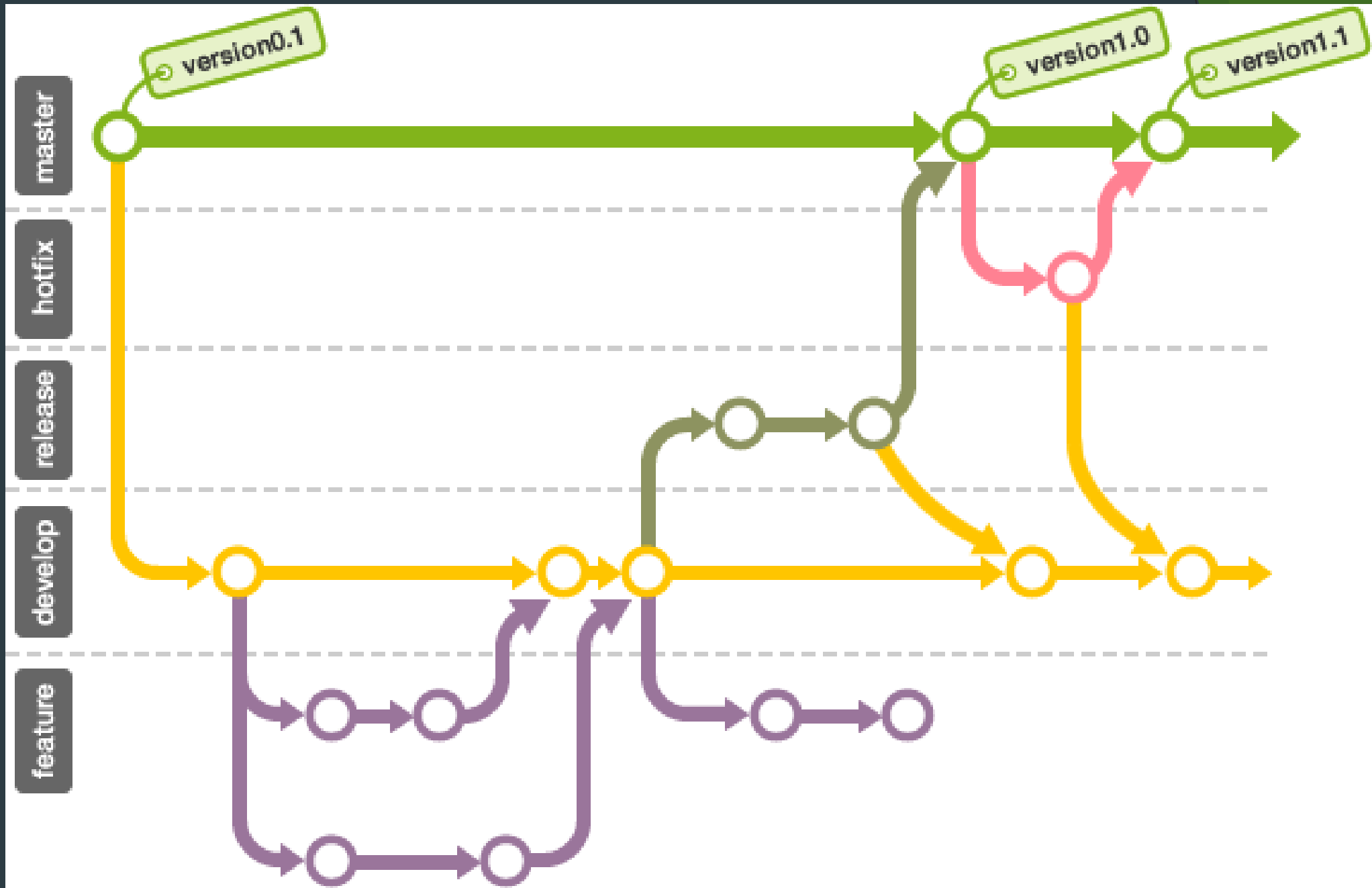
1. Current chosen branch
2. Press New Branch



3. Select branch name
4. Press Create







# Agenda

- ▶ Version Control
- ▶ GIT
- ▶ GIT clone
- ▶ GIT commit
- ▶ GIT create branch
- ▶ Exercises

# Exercise

1. Register Account <https://github.com/>
2. Create empty repository
3. Open Repo using Pycharm
4. Create first commit
5. Push your commit
6. Check changes in Github
7. Create branch development
8. Create second commit
9. Push your commit

**THANK YOU FOR YOUR  
ATTENTION!**

**...YOU ALWAYS HAVE MY  
ATTENTION.**