# The basics of git

Jean-Claude Passy

ZWE Software Workshop

IMPRS-IS S4 Workshop, October 28, 2021



### Outline

- Standards for Software Development
- 2 What is git?
- Tutorial
- Advanced topics
- Conclusion

Jean-Claude Passy ZWE Software Workshop The basics of git 2/35

•00

- Standards for Software Development
- 2 What is git?

Standards for Software Development

- 3 Tutoria
- Advanced topics
- Conclusion

Jean-Claude Passy

ZWE Software Workshop
The basics of git

3 / 35

Standards for Software Development

000

### We often try to compare/oppose academia and industry:

	Academia	Industry
End-goals	Paper, code, start-up	Product (software, car, rocket), \$
Pressure	Supervisor, deadline,	N+1, N+2,, customer, credibility
Tasks	Read papers, write code, write paper	idem + talk to customer, business analyst, activity report
Keywords	Freedom, flexibility, unpredictability, discovery	Process, structure, validation, traceability

The Software Workshop believes that academia can benefit from some of the standards/principles used in the industry, for instance:

- Process comes with some overhead but helps you to stay focused
- Software should be verified and validated (see session about testing)
- During development, one should keep track of who has done what and when

Standards for Software Development

One of our most important mission is to teach researchers about good software development practices and new technologies.



Sofware Development Life Cycle (SDLC) is the process followed for the development of a software product. Its goal is to produce a software:

- with the **highest quality** (bug-free, stable, robust, modular, working)
- for the **lowest cost** (money, time, man power, ...).

The goal of this workshop is to introduce you to these standards and techniques.

Jean-Claude Passy

ZWE Software Workshop
The basics of git

5 / 35

### Outline

- Standards for Software Development
- 2 What is git?
- 3 Tutorial
- 4 Advanced topics
- Conclusion

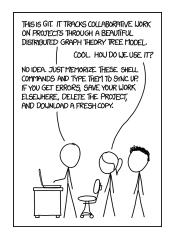
Jean-Claude Passy

ZWE Software Workshop
The basics of git

6 / 35

# How people talk about git

### A lot of the internet if about git related questions:



	COMMENT	DATE
Q	CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
ø	ENABLED CONFIG FILE PARSING	9 HOURS AGO
\dot \	MISC BUGFIXES	5 HOURS AGO
φ	CODE ADDITIONS/EDITS	4 HOURS AGO
Q.	MORE CODE	4 HOURS AGO
ΙÒ	HERE HAVE CODE	4 HOURS AGO
	ARAAAAA	3 HOURS AGO
6	ADKFJ5LKDFJ5DKLFJ	3 HOURS AGO
φ	MY HANDS ARE TYPING WORDS	2 HOURS AGO
þ	HAAAAAAANDS	2 HOURS AGO

AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.



### Version control

**Version control** is the process of managing and organizing information changes. It is done using **V**ersion **C**ontrol **S**ystems (**VCS**). The most popular (and really the only option) is git.

It's main advantages are:

- complete code base is stored on everyone's computer
- work collaboratively
- work simultaneously on several files
- work simultaneously on several tasks
- traceability
- rollback
- reproducibility

### git (unlike svn) is distributed (DVCS):

- complete code base is stored on everyone's computer
- not dependent on network connection
- faster
- allows private work

git is **complex**, we advise to use a **client**: SourceTree (macOS/Win), GitKraken, or solutions integrated to your IDE.

ren	on Service Drawnian C - Street or Co Description		Correct	Autor	James Inc.
		Marga branch Tosses/CSD-Mill-bog-add-keyword-dising irro develop	THEATH	Africa Street commissional involvery	71 Fee 2003 of 30
1		e. Marga Branch Yannes, C.S.C. 100-100-100-100-100-100-100-100-100-100		Adres Stock - carbot direct insulants Adries Stock cambrid direct insulants	V. Our 2000 at 100
,		Charles reministry to develop	Contract county a same. Contract	han Destroker streetstandbudtsen nas der	1. Dec 2020 of Till
		The removing Trying to cognitive long and other removies.	States	has Charleson viano marteliana tradicione no	1 to 100 a 100
ţ		Mineratoring Typing to imparatiology and shart versions.  (Mil. amphantion, conflict into disposition.	5945445 2964548	han Oraninasi kiran Jasahnika etiabogan tera dan	7. Day 2009 at 10:0
1		no process rong mr overs skelerards Red De enal skeet is dre.	79000	has Destroker stan are first before me.	3. Dec 2000 at 16.0
		ndurbon weilig (Fland the entail subject in dex. (ing environments and settings.	1304048	han Deshrikar stran areahrikan@lastingen.np han Deshrikar stran areahrikan@lastingen.np	7, Dec 2000 at 1616 7, Dec 2001 at 1915
ţ		ing environments and settings. 155 carrier deployment into develop	2000ala	han Oraninkov kivan prachrikovat usongan, ng., han Oraninkov kivanskrikovat prakrikovan man den.	7. Dec 2008 at 10.0
1		over-deployment Various server loughters.	7038416	han Deshrikar siran ansulmkur@hakingen.mp.	
l†		movement are adjust and but one group always to experimental are adjust and but accordingly	849002	han Drednikar kiran arednikar@bebrgen.np	18, Nov 2020 at 12
l†		graphile Using stiging instance for the submission server.	2346616	han Oreshnikov viran oreshnikováltustingan mp	18, Nov 2020 et 19
l†	Trying to separate long an		MARKET	han Deshelau siran anaholau (iliadiingen np	W. New 2020 at 10
l†	Find the small subject in		settino	han Osednikar kiran arednikar@bebrigen.np	18, Nov 2020 at 19
l†		ing environments and settings.	6:6:00	han Osobnikov viran prodnikovátuobingan mp	18. Nov 2020 or 10
l†		ely of the confirmation email.	Master	Irum Oranbeilaur siram anasheilaur@isustringan.mp	W. New 2020 at 10
l†		ION, POOT TWIN the environment (	835,339	han Oreshnikar kiran areshnikar@belongen.np	18. Nov 2020 of 10
l <del>t</del>	Stable certificate relided	n. He will be using self-signed certs.	1000499	han Osehnikov viran sreshnikovätusbingan, trp	18. Nov 2020 et 10
l <del>t</del>	Semaning the MANIFEST.		m0907	han Oreshnikar viran sneshnikar@halbingan.mp	W. New 2020 at 10
l <del>t</del>	E turns out sweep expects.	every specific entrypoint.	65966	han Diednikar kiran ansdrikar@fusbingen.np	18. Nov 2020 at 10
l <del>l</del>	Adding parant directors to	summer path so we can import across without immaking.	conticin	han Orestnikov viran orestnikovátu abingan na	18. Nov 2020 at 16
l <del>t</del>	Proper Imports in veryl go		13479	han Osehnikov viran snedmiovijituotingan mp	18. New 2020 at 10
4	Address MARKETT In his we	can include server into source directory	BROOM	han Dieshnikar stran areshnikar@buildingen.np	18. Nov 2020 at 10
P	Marge branch BugflyCSC	XX-rising-use-sear into books	N00066	han Orazhekov -korazhekovátustingan zegyde-	18. Nov 2020 of 16
	1- organização/CSS-XX mi	sing user lidear. Fixed retroactive user token parenation.	5455e	Inan Oseshmäner einneshmännslitustringen myg den	W. New 2000 et 10
,		No one-sound: Yill develop	65003	After Book cardist@real.locatedy	18. Nov 2020 of 20
Social Notice   E =					
antinus/services.pg:		also jurametian, arabiting			
antinus/scinoless psy					Secure 1
Maybrismilion training		1 from anthous application input deplication			
train/ant, pindou plant	No. py				
		Anni action version layersversion     Anni action, version layers and long version.			
		II II fra attavoida igart Gatelianitier			
Displaying oil changes be	etween.				Books
TOGODINATO POR NOT THE	CONTRACTOR CONTRACTOR AND	Note: 1 Control 27-63			Roses
	a contraction and				
		71 70 Starguardes			
		in a principal (angularization			

- Tutorial

Jean-Claude Passy ZWE Software Workshop 10 / 35 The basics of git

# Create a git repository

Git repositories are usually created on the server using the web interface, then cloned:

### Cloning

```
$ git clone https://github.com/MPI-IS/workshops.git
```

```
$ git clone git@github.com:MPI-IS/workshops.git
```

You can also create a repo from an existing folder and update the remote:

### Init and push

```
$ git init
```

- \$ git branch -M master
- \$ git add/commit ...
- \$ git remote add origin https://github.com/MPI-IS/workshops.git
- \$ git push -u origin master

# Vocabulary

### **Local and Remotes**

- Two distinct environments: your local repo and the remote repo
- They can evolve independently and the remote is shared among users
- origin denotes what is on the remote repo

### Commit

- A commit is a snapshot of your repository
- Properties: datetime, author, UID, description
- At least one parent commit (except for the first commit)

### **Branch**

- Branches are simply pointers to commits
- They can be created, moved, reset

### **Branches**

- Isolate implementation changes
- Switch contexts

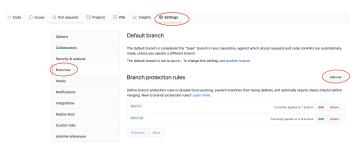
### Workflow

- New dev. starts on feature branch
- Usually start on reference stable state
- When ready merge the feature branch
- Always merge from stable into less stable
- Develop almost always stable
- Master always stable (releases)

All Branches 0	Show Remote Branches C Date Order C		Jun
ph.	Description	Commit	Author
	Diorgan/asses/908-467-workers-and-assignments Finally everything works through the serializ	ad73961	Ivan Oreshnikov «ivan ores
	WP	7907104	Ivan Oreshnikov coreshnik
	No need for a separate command for preview link collection.	199cb98	Ivan Oreshnikov kivan.ores
	> origin(s/808-467-deploy) Minor changes to make things work given the current staging depl	ab45a07	Jean-Claude Passy «jean»
	Reed the imports.	4/28641	Ivan Cheshnikov kivan ores
	Added a synchronization command.	b45d576	Ivan Oreshnikov «ivan.ores
	Trying a newer version of MeshAnnotator	78c97bd	Ivan Oreshnikov civan ores
	Added a command for HIT expiration.	219106	Ivan Oreshnikov «Ivan.ores
	Random line folds ()	c859949	Ivan Oreshnikov koreshnik
	Atypo.	3031fcf	Ivan Cheshnikov kivan ores
	Cosmetic changes.	c4cadb8	Ivan Oreshnikov vivan ores
	Some more minor status changes.	cba7ff8	Ivan Oreshnikov kivan.ores
	Use Turk's HIT Status as the database value.	818ad7b	Ivan Oreshnikov kivan ores
	Moving all the machinery into a separate module.	a1095e5	Ivan Oreshnikov «ivan ores
	HT title should be the same for the whole dataset	1639002	Ivan Cheshnikov kivan oreo
	Fixed the initial migration and added a new one	6c23671	Ivan Oreshnikov vivan ores
	'publish' command finally works.	095d3ed	Ivan Oreshnikov kivan.ores
	New models, cellery setup + slow migration towards DRF technologies.	6b2f75c	Ivan Oreshnikov «Ivan ores
	Didentity Dissiphitievelay Merge branch issues/808-473-whole-dataset-upload into	6389044	Ivan Oreshnikov kivan.or
	If origin/asses/BCG-473-whole-dataset-upload. One can now upload the whole dataset by drop	d349c3d	Ivan Oreshnikov civan ores
	Merge branch "ssues/805-468-deploy-djange-mt" into develop	Cfc5919	Jean-Claude Passy «jean»
	Updated the docker-compose file for the deployment, Additional: * updated MeshAnnotator ve	9109cff	Jean-Claude Passy «jean-
	Updated MeshAnnotator version	bf73b2b	Jean-Claude Passy «jean-
	Merge branch 'issues,'805-466-commands' into develop	606ceb4	Jean-Claude Passy «jean»
		606ceb4	
	Merge branch 'ssues/900-466-commands' into develop	606ceb4	Jean-Claude Passy <jean-< td=""></jean-<>
	. Merge branch "soues@65-466-commands" into develop Added preview link command. Additional: "added expiration time to the HT model "fixed pag	606ce64 7852643	Jean-Claude Passy «jean- Jean-Claude Passy «jean-
	Merge branch "sques@69-486"-commands' into develop  Added preview link command. Additional: * added expiration time to the HET model * fixed pag  Added login and logicut pages.	606ceb4 7852643 3686684	Jean-Claude Passy -(ean- Jean-Claude Passy -(ean- Jean-Claude Passy -(ean-
	Merge branch Tossec®03-466-comments' into develop  Added preview link: commend. Additional: "added expiration sine to the HT model "fixed pag.  Added login and logout pages.  Added Tint commends and tests	606ce64 7852643 3686684 e65683 7522196	Jean-Claude Passy «jean- Jean-Claude Passy «jean- Jean-Claude Passy «jean- Jean-Claude Passy «jean-
	Marga branch "issues 1905-466-cennands" hits develop  Added preview linit command. Additional: "added expiration time to the HT model "fixed pag.  Added login and logout open.  Added login and logout open.  Marga branch "issues 1905-400-run-ANTS" into develop  Marga branch "issues 1905-400-run-ANTS" into develop	606ce64 7852643 3686684 e65683 7522196	Jean-Claude Passy -(ean- Jean-Claude Passy -(ean- Jean-Claude Passy -(ean- Jean-Claude Passy -(ean- Jean-Claude Passy -(ean-
	Verge branch Touvel-1004-446- cemenate' into develop  Added private linc command. Additional: added expiration time to the NET model "fixed page.  Added tolgrin end boughtes.  Added first commands and these  Added first commands and these  Configuration, commands and these  Configuration, commands to convect to the burk, including "evilicide" submedular from hours GIFM-	606ce64 7852643 3686684 e65683 7522196 1911298	Jean-Claude Passy «jean- Jean-Claude Passy «jean- Jean-Claude Passy «jean- Jean-Claude Passy «jean- Jean-Claude Passy «jean- Jean-Claude Passy «jean-
	Wags brook Soun-MSG-46-Common Fra develop  Added provise foils command. Additional * added expiration free to the NT model * fixed pag.  Added type and bypot copies.  Added frac commons and lests  Marge brook Sound-Ge-40-co-AMSS into develop  Configuration, update to province to the Sound-Ge-40-co-AMSS into develop  Configuration, update to province to the Sound-Ge-40-co-AMSS into develop  Added from Aurit Office.	606ce64 7852643 3686684 e66583 7522196 1911298 9bd5714	Jean-Claude Passy vjean- Jean-Claude Passy vjean- Jean-Claude Passy vjean- Jean-Claude Passy vjean- Jean-Claude Passy vjean- Jean-Claude Passy vjean- Jean-Claude Passy vjean-
	Ways brown in Supplied 46-command in devices.  Also Sign in Supplied 46-command in Supplied in Supplie	609ceb4 7853843 3686684 eb81083 7852196 1911288 9bd5714 18cbee2	Jean-Claude Passy vjean- Jean-Claude Passy vjean-
	Manys broom foundable-64-Commander for devices Manifest provision foundable-64-Commander for devices Addies layon for devices and devices and segment on the 1st the ATT model of fixed page. Addies layon and relayon pages. Manifest for commander of the commander for devices Manifest for the commander of the	609ceb4 7852643 3686684 e65083 7852196 1911268 Sbd5774 f6tbee2 1962021	Jean-Claude Passy (ean- Jean-Claude Passy (ean-
	Weigh plant of usual/SIG 446 commands in divided Made primaries incommand of sidebases * dealed equipment with to the 16 Mill made * fixed pag- Added begin and regional deaper. Added to commands and the 10 Mills of the commands and the 10 Mills of the command and the 10 Mills of the 1	606ceb4 7852643 3686684 e85168 7522156 191286 9645714 9646274 964021 964021 964021	Jean-Claude Passy (jean- Jean-Claude Passy (jean-
	Weep short in controllo 444 comments if in devices Made province in comment in devices of the size in the 141 model "fined page. After the prince in comment of the size in the 141 model "fined page. After the comment on size in the control of the size in the	609ceb4 7853643 3686694 eb8fo83 7911096 9b45774 f86bec2 9b2021 8b2022 8b24079 felecide	Jean-Claude Passy «Jean- Jean-Claude Passy «Jean-

### Protected branches

First thing to do is to create the *basic* branches **master/main** and **develop**, and to protect them against human errors. On GitHub:



If you are using Gitlab or a GitHub repo owned by an organization, you can **restrict** who can push to matching branches (user, role...).

# Configure your identity

It is now time to set up some global settings

### Configuration

```
$ git config --global user.name "John Doe"
$ git config --global user.email "john.doe@tuebingen.mpg.de"
```

On SourceTree, these parameters (and more) can be set under Preferences/General. Additionally, names/emails can be set for each individually for each repository:

- use the same command without the --global option
- on SourceTree, go to Settings/Advanced for the given repository

In case you committed with the wrong information, you can still fix the history afterwards by using the git filter-branch command.

Some files should not be part of the repository: temporary files, by-products...

These files should be specified in a file named .gitignore at the root of the repository.

```
aptipedia > 🚸 .gitignore
     *.pot
     *.pyc
       pycache /
     local settings.py
     db.salite3
     media
     node modules/
     wordpress/
     virtualenvs/
     **/.DS Store
     doc/build/
```

### Warning

.gitignore is shared with other developers. Think carefully before making changes!

- Clone repository
- Request changes from remote
- Work with branches
- Pull changes from the remote
- Show commit log
- Display the difference between commits
- See the status of the repository
- Add files to the staging area
- Place non-committed changes to a shelf
- Merge last commit with the current one
- Push changes onto the remote

git clone url

git fetch --all

git checkout my\_branch

git pull

git log

git diff

git status

git add --update/--all

git stash apply/pop/drop

git commit -m "Your message"

git push

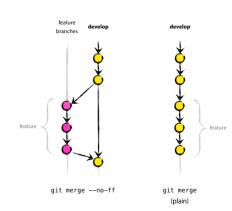
### Demo

Task: add your name to the README on a dedicated branch

- Checkout the base branch on which you want to create you branch (double-click on the commit). The current branch appears in bold.
- 2 Create a new branch by clicking on the Branch button. You are now on this branch.
- Edit the files.
- Uncommitted changes appears in the history, click on it. Details appear in the lower part.
- Stage the changes you want to commit (checkbox or drag-and-drop).
- Olick on the Commit button, write your message, and commit.
- Push onto the remote.

# Takes the union of changes (opposite action to branching).

- Conflict resolution is part of life!
- Only merge when the branch is ready (stable)
- Create new commit to keep topology (no fast-forward)
- Delete branches once they have been merged



Task: let's do some merging!

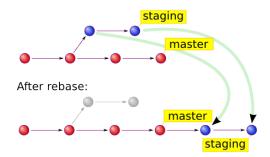
# Demo merging

**Task**: merge your branch onto develop and resolve conflicts if needed.

- Checkout the base branch onto which you want to merge your feature branch.
- Click on the Merge button.
- 3 Select your feature branch and merge.
- If there are conflicts, they will appear in the lower part with a warning sign.
- To resolve conflicts, right-click on the file and choose Resolve Conflicts > Launch External Merge Tool.
- A window with 3 panels should appear: content on the base branch (left), content on the feature branch (right), and content that you want after the merge (bottom). This last panel can be edited directly.
- Save, close the merge tool, and stage the file that has been resolved.
- 3 Do so for all the files containing conflicts, and commit.
- Push to the remote.

# Rebasing

Moves the changes made on a branch onto another commit.



 $\approx$  projects a branch implementation to the space **orthogonal** to other feature branches.

Jean-Claude Passy

ZWE Software Workshop
The basics of git

21 / 35

# Rebasing

### Advantages:

- Keeps branch up-to-date with latest stable
- Keeps branch focused
- Make history shorter and therefore clearer

#### Workflow:

- Rebase your changes locally
- Make a diff between the local and remote branches: should be orthogonal to the feature implemented
- Force push to remote:

### Force push

\$ git push -f

By default, SourceTree does not allow you to force push. The option must be first enabled in Settings > Advanced.

### Warning

- History is changed, so notify your colleagues working on this branch.
- All local copies should be synchronized and the local branch reset:
  - \$ git fetch --all
  - \$ git checkout my\_branch
  - \$ git reset --hard origin/my\_branch

Task: let's do some rebasing!

# Task: rebase your branch on top of develop.

- Checkout the feature branch you want to rebase.
- ② In the left panel, right-click on the base branch you want to rebase onto (develop) and select Rebase current changes onto ...
- 3 If there are conflicts, resolve them as you did when merging.
- At any point, you can stop the rebasing by selecting Actions > Abort Rebasing.
- Once the conflicts have been resolved, select Actions > Continue Rebasing.
- Sanity check: confirm that differences between the rebased local branch and the remote are orthogonal to the feature implemented.
- Force push to the remote.

## Outline

- Standards for Software Development
- 2 What is git?
- 3 Tutorial
- 4 Advanced topics
- Conclusion

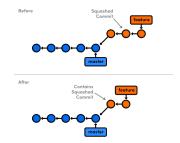
Jean-Claude Passy

ZWE Software Workshop
The basics of git

25 / 35

# Interactive Rebasing

### Squashes the commits and rewrites history



- Takes the union of changes (opposite action to branching)
- Conflict resolution is part of the operation, so don't worry!
- Only merge when the branch is ready (green)
- Create new commit to keep topology (no fast-forward)
- Delete branches after some time once they have been merged

Jean-Claude Passy

ZWE Software Workshop
The basics of git

26 / 35

• Increases S/N by diminishing history

• Makes branch scope easier to understand

• Makes branch easier to handle (rebase, merge, conflicts)

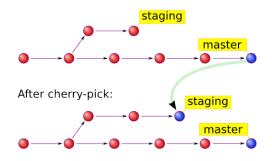
Task: let's do some interactive rebasing!

Jean-Claude Passy ZWE Software Workshop
The basics of git 27 / 35

Task: squash the commits on your branch.

- Checkout the feature branch you want to rebase.
- 2 Select the branch root, i.e. parent commit of the first commit you want to squash.
- 3 Right-click and select Rebase children of ... interactively.
- Squash the commits, edit the commit messages, and confirm.
- **Sanity check:** confirm that there is **no difference** between the rebased local branch and the remote.
- 6 Force push to the remote.

Apply changes introduces by some commit onto the current branch.



Task: let's do some cherry-picking!

# Demo cherry-picking

Task: apply a change from another branch onto your branch.

- Checkout your feature branch.
- 2 Select the commit you want to cherry-pick.
- 3 Right-click and select Cherry Pick.
- If there are conflicts, resolve them as you did when merging.
- Oush to the remote.

# Outline

- Standards for Software Development
- 2 What is git?
- 3 Tutoria
- Advanced topics
- Conclusion

Jean-Claude Passy

ZWE Software Workshop
The basics of git

31/35

# Multiple origins

You can work privately on your git repository (just do not push!). As soon as you push, you work will be visible to anyone having access to repo (no branch restriction).

However, you may sometimes want to restrict some part of your repository (e.g. personal data, dev vs. public repo). The solution is to use **multiple origins**:

- You can compare commits between origins
- Many commits can have the same content



# Last few tips

- Empty folders are not tracked. Good practice is to use a .keep file.
- User git rm to remove a file from the repo (a simple rm is usually not enough).
- Push as often as possible on feature branches
- Release often using a consistent versioning scheme, for instance PEP 440:

- Use tags and protect commits if needed
- $\bullet$  These concepts are sufficient for  $\approx 90\%$  of what you will need to do.

# Copyright and Licensing

Disclaimer: I am not a legal professional!

### Some definitions:

- Copyright: clarifies who owns the code
- License: clarifies who can the use the code and how
- Authors: clarifies who wrote the code

It is a good idea to make these clear in the README.md (see this example).

If your code is public, it should have a copyright and a license. You can choose a permissive (BSD, MIT, Apache) or protective/copyleft license (e.g. GPL).

If a code does not have a license, the owner retains all rights and nobody else can use it.

# Copyright and Licensing: GitHub specificities

Disclaimer: I am still not a legal professional!

GitHub has some specific rules (see here).

By publishing your code, you allow other users to view your work and fork your repository. Changing the visibility of your project does **NOT** delete forks.

There is also the concept of Contributor License Agreement (CLA). On GitHub, as long as the repository has a license, any contribution is licensed under the same terms (no need for a CLA).

However, the PR issuer remains the copyright owner of the contribution, which means:

- the project license cannot be changed without consent (e.g. Linux kernel)
- the contributor can re-used the contributed code