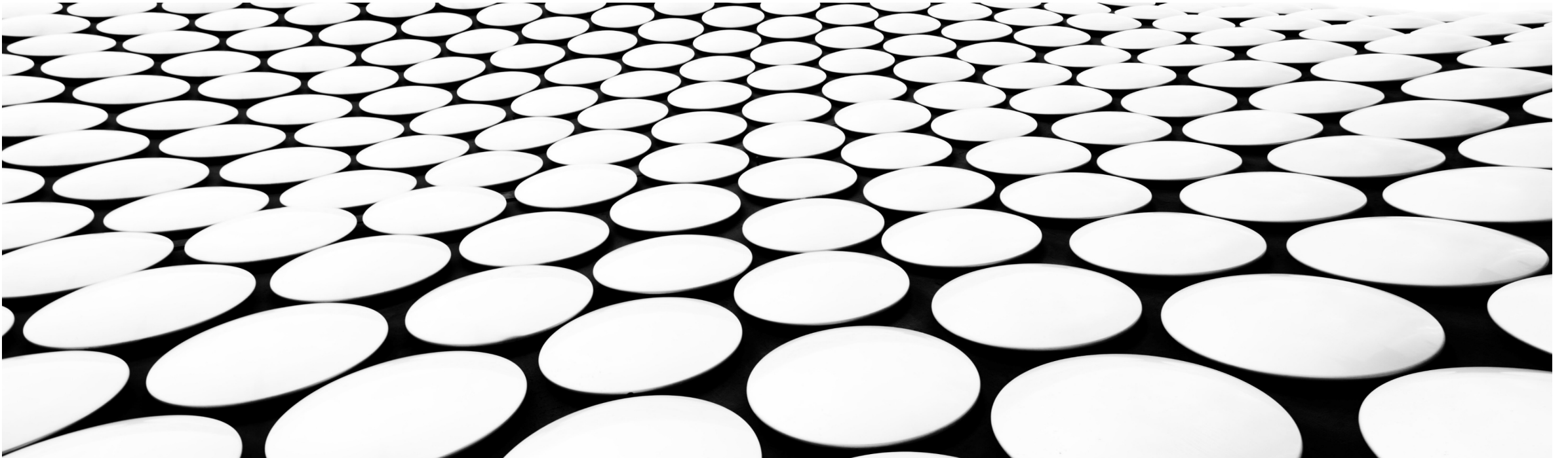
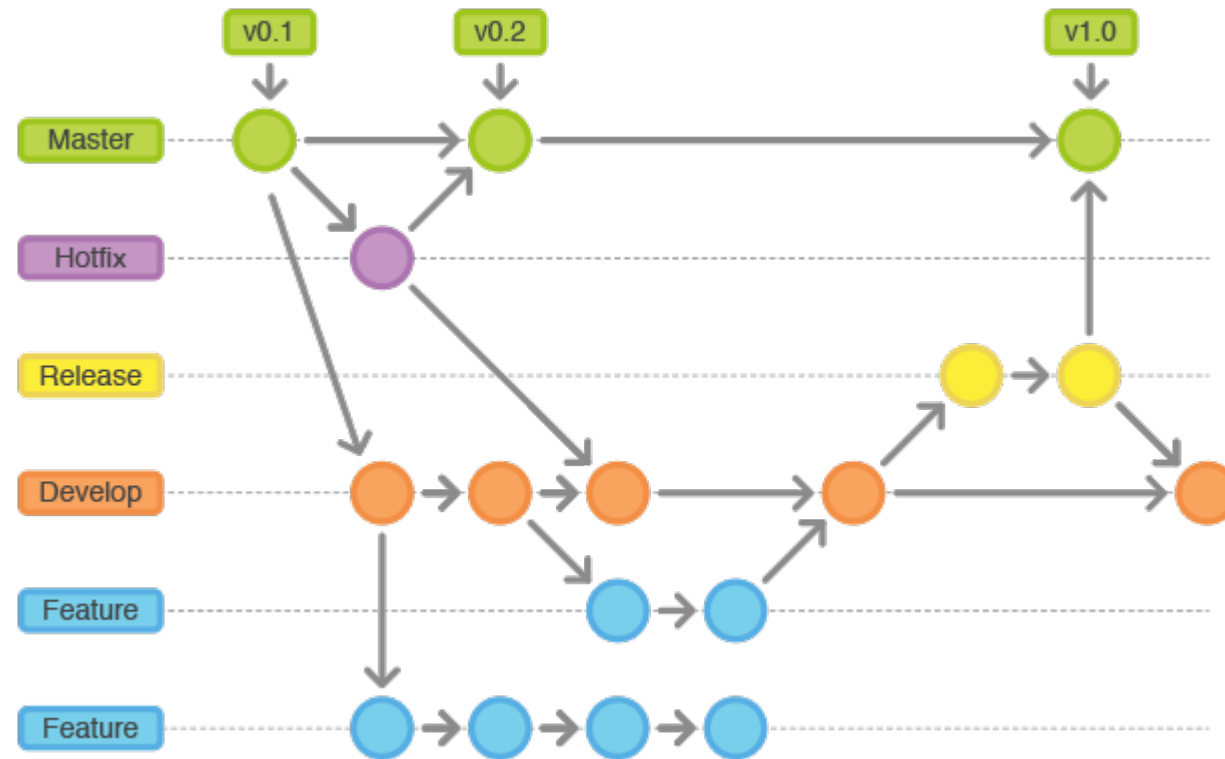

GIT EINFÜHRUNG

HANDS-ON MIT GIT ARBEITEN



WARUM GIT?

- Leistung
- Sicherheit
- Flexibilität



<https://blog.seibert-media.net/wp-content/uploads/2014/03/Gitflow-Workflow-4.png>

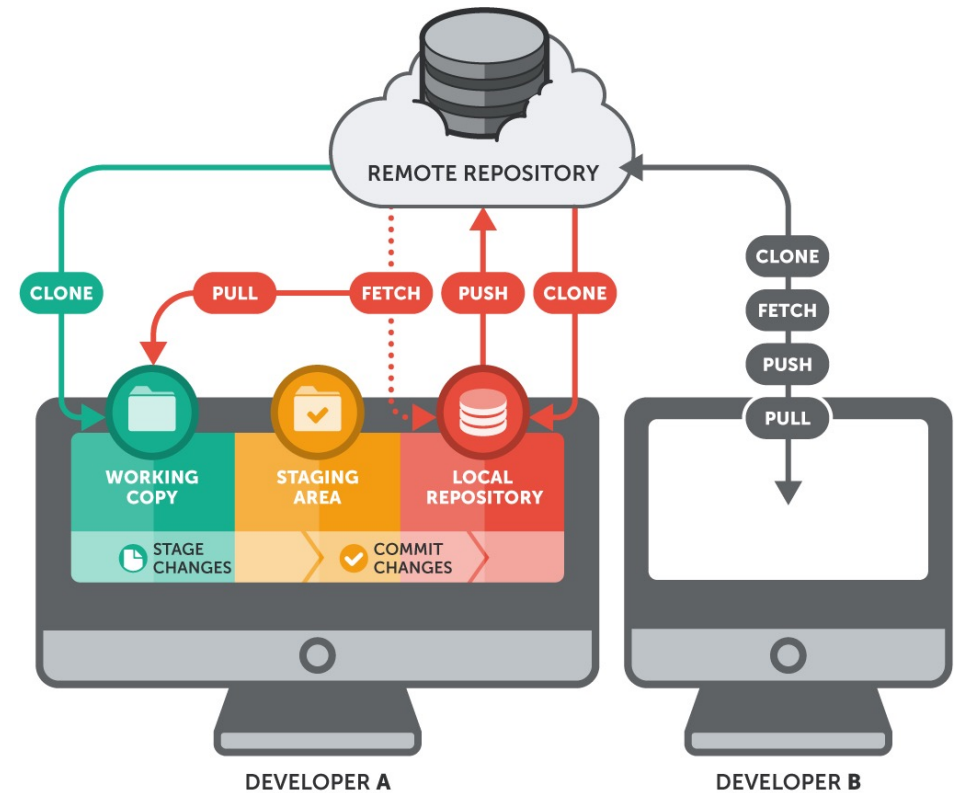
ZIEL



<https://blog.seibert-media.net/wp-content/uploads/2014/03/Gitflow-Workflow-2.png>

EINFÜHRUNG

- Git muss installiert sein
- Git Befehle beginnen immer mit git ...
 - init – initialisieren, erstellt ein .git Ordner
 - remote – verknüpfen mit anderen Repositories
 - clone – «klonen»
 - fetch – Inhalte von «remote» herunterladen, aber «local» nicht überschreiben
 - pull – Inhalte von «remote» herunterladen und «local» überschreiben
 - add – Inhalte hinzufügen bzw. «tracken»
 - commit – «getrackte» Inhalte speichern
 - push – Lokale «commits» auf den definierten remote speichern
 - merge – Inhalte zusammenführen
 - status – Status abfragen
 - ... und viele mehr
- Praktisch: [cheat sheet](#)



<https://www.git-tower.com/learn/git/ebook/en/command-line/remote-repositories/introduction/>

HANDS-ON BEISPIEL (1)

- Bitte in Gruppen von 3-4 Personen zusammensetzen
 - Wichtig Personen kurz nummerieren und nicht verändern
- Person 1 ist der Ersteller / die Erstellerin, Person 2-4 sind Kollaborateure
- Vorbereitung:
 - Alle: Git Installiert
 - Alle: IDE (VSCode, Pycharm, ...) oder ein Terminal (Kommandozeile) geöffnet
 - Nur Person 1: einloggen bei Github

HANDS-ON BEISPIEL (2)


- Person 1:

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 *

 dpinezich ▾


 /


collab ✓

Great repository names are short and memorable. Need inspiration? How about [miniature-octo-adventure?](#)

Description (optional)

Hier könnte eine Beschreibung stehen

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

☐  **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.

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

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

.gitignore template: Python ▾

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

This will set  **main** as the default branch. Change the default name in your [settings](#).

Create repository

HANDS-ON BEISPIEL (3)


main ▾


1 branch


0 tags


Go to file


Add file ▾



 Code ▾


 dpinezich Initial commit

cb65ed2 now  1 commit

 .gitignore

Initial commit


now

 README.md

Initial commit

now

README.md



collab


Hier könnte eine Beschreibung stehen


Clone

HTTPS SSH GitHub CLI

`https://github.com/dpinezich/collab.git`

Use Git or checkout with SVN using the web URL.

 Open with GitHub Desktop

 Download ZIP

Releases

No releases published

[Create a new release](#)

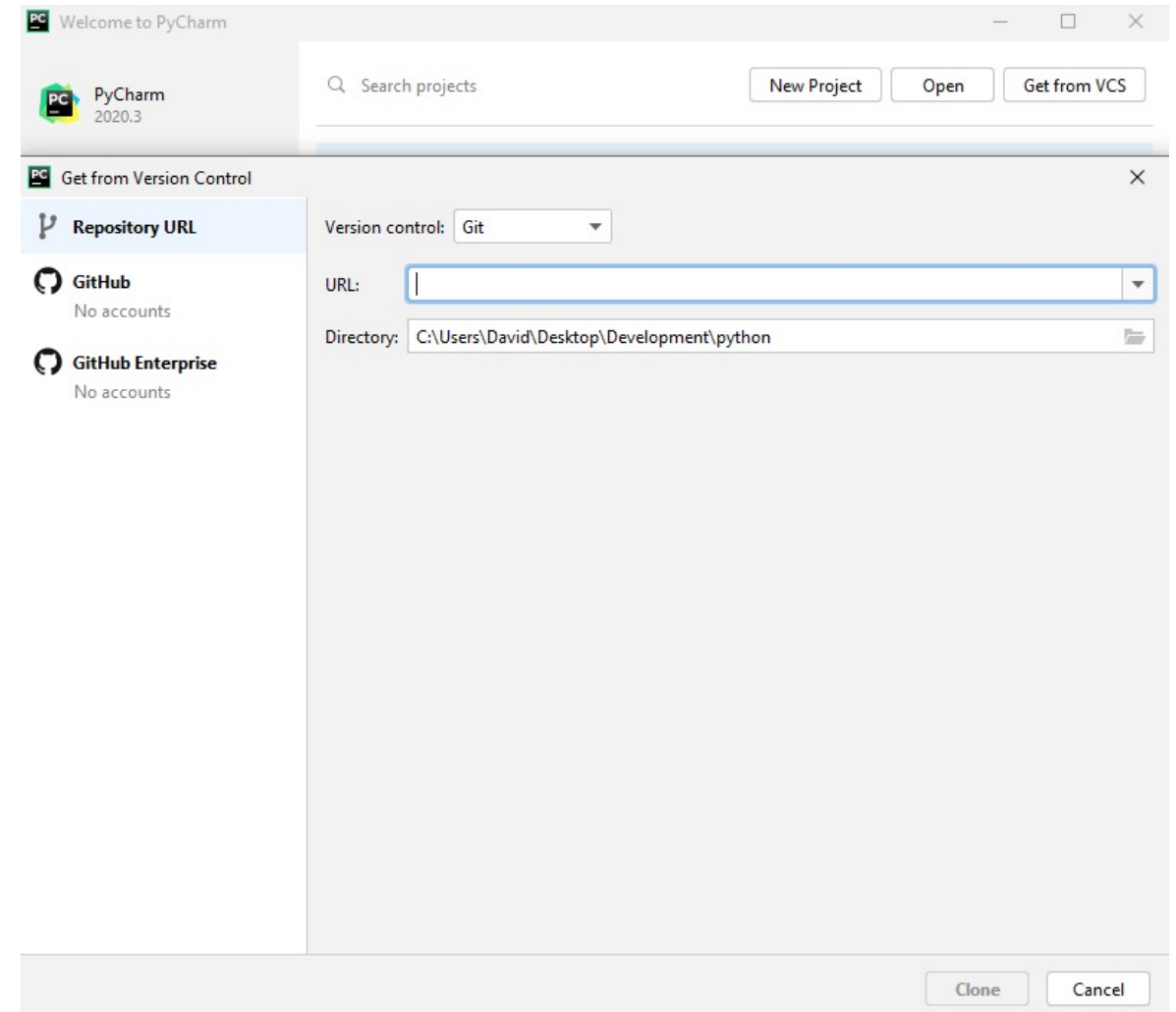
Packages

No packages published

[Publish your first package](#)

HANDS-ON BEISPIEL (4)

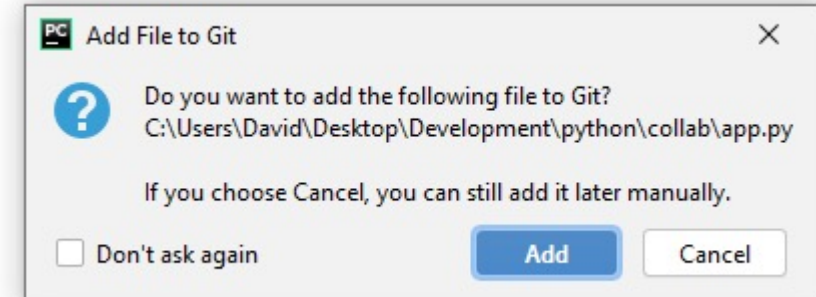
- Alle: IDE öffnen und «git clone <Repository>»
 - Beim Start (empfohlen, kein «command» nötig)
 - Oder via Terminal wenn schon ein Projekt erstellt wurde



HANDS-ON BEISPIEL (5)

- Person 2:
 - Erstelle Bitte eine Datei app.py
 - Programmiere eine kleine Funktion
 - Die zwei Zahlen berechnet
 - Und auf der Kommandozeile «The number is: nummer» ausgibt
 - app.py sollte nun farbig dargestellt sein und im status «added»
 - «git status» eingeben
 - «.idea» in .gitignore hinzufügen
 - «git add .» eingeben um alle «not staged» hinzuzufügen

```
Changes to be committed:
(use "git restore --staged <file>..." to unstage)
    modified:   .gitignore
    modified:   README.md
    new file:   app.py
```



«git add»

```
Changes to be committed:
(use "git restore --staged <file>..." to unstage)
    new file:   app.py

Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git restore <file>..." to discard changes in working directory)
    modified:   .gitignore
    modified:   README.md
    modified:   app.py

Untracked files:
(use "git add <file>..." to include in what will be committed)
    .idea/
```

HANDS-ON BEISPIEL (6)

- Person 2:
 - «git commit -m "changes to files"»
 - «git push»

```
[main 878fab3] changes to files
3 files changed, 140 insertions(+), 131 deletions(-)
create mode 100644 app.py
```

```
Username for 'https://github.com': dpinezich
Password for 'https://dpinezich@github.com':
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 16 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 1.43 KiB | 122.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0)
To https://github.com/dpinzich/collab.git
cb65ed2..878fab3 main -> main
```

HANDS-ON BEISPIEL (7)

- Alle ausser Person 2:
 - «git pull» status sollte nun bei allen gleich sein
- Person 3
 - Bitte app.py verändern
 - «git add app.py»
 - «git commit -m "more changes to files"»
 - «git push»
- Alle ausser Person 3
 - «git pull» status sollte nun wieder bei allen gleich sein

```
[main 878fab3] changes to files
3 files changed, 140 insertions(+), 131 deletions(-)
create mode 100644 app.py
```

```
Username for 'https://github.com': dpinezich
Password for 'https://dpinezich@github.com':
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 16 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 1.43 KiB | 122.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0)
To https://github.com/dpinezich/collab.git
cb65ed2..878fab3 main -> main
```

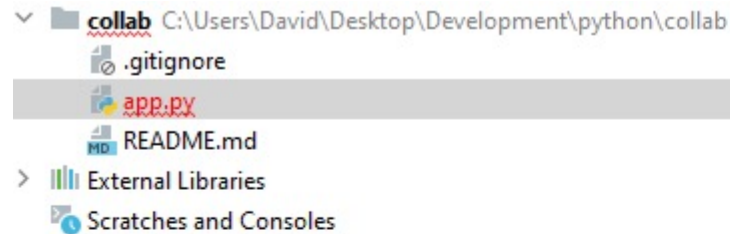
HANDS-ON BEISPIEL (8)

- Zwei beliebige Personen:
 - Bitte app.py nach belieben anpassen (Kommentare, Ausgabe, Anzahl Zahlen)
 - Eine Person macht «git commit» + «git push»
 - Die zweite Person macht ebenfalls «git commit» und «git push»
 - Die zweite Person macht «git pull»

```
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 658 bytes | 24.00 KiB/s, done.
From https://github.com/dpinezich/collab
  878fab3..0b89761  main      -> origin/main
Auto-merging app.py
CONFLICT (content): Merge conflict in app.py
Automatic merge failed; fix conflicts and then commit the result.
```

```
! [rejected]        main -> main (fetch first)
error: failed to push some refs to 'https://github.com/dpinezich/collab.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

HANDS-ON BEISPIEL (9)

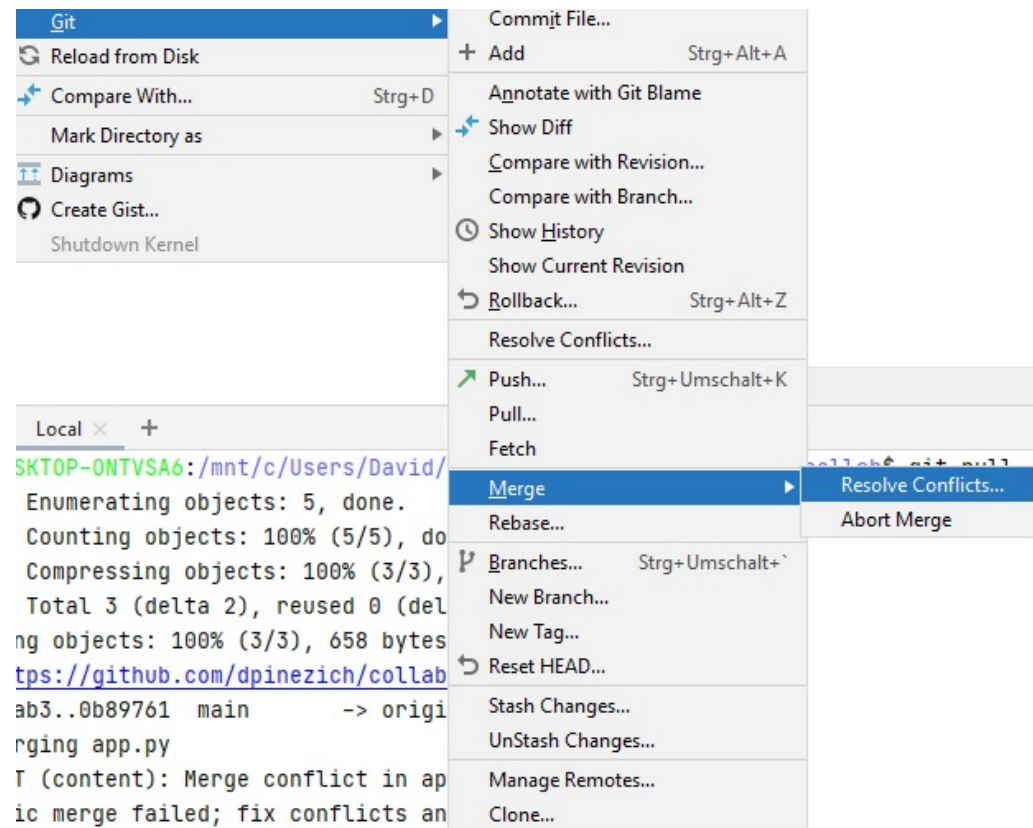


```
1
2
3 <<<<<<< HEAD
4 def calc_two_sums(a: int, b: int, c: int) -> None:
5     print("The number is:", str(a + b))
6
7
8 calc_two_sums(3, 10, 22)
9 =====
10 def calc_two_sums(a: int, b: int) -> None:
11     print("test")
12     print("The number is:", str(a + b))
13
14
15 calc_two_sums(3, 10)
16 >>>>>> 0b89761496b7967b1ec6116a5e574732e6717cc5
17
```

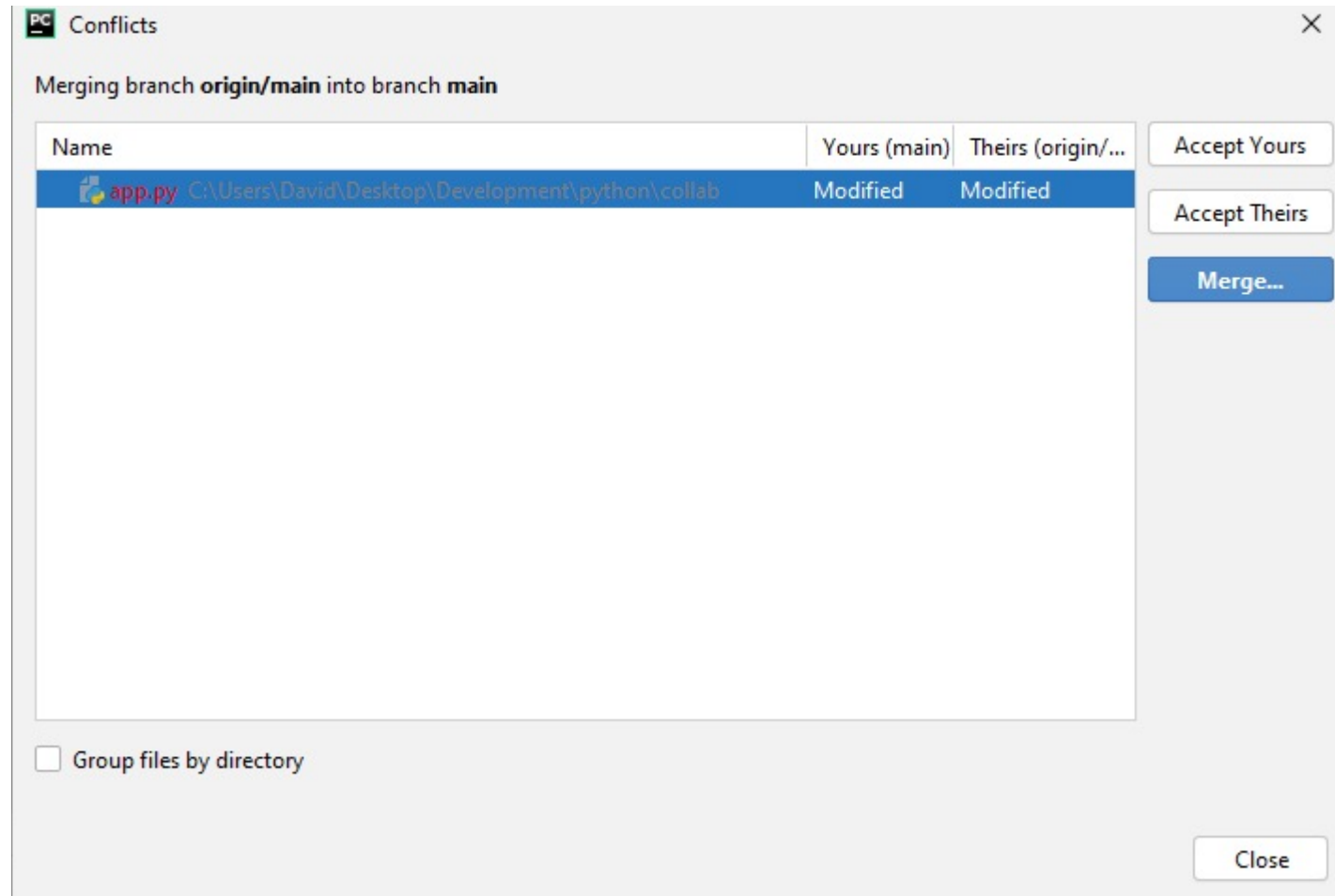
Person 2

Person 1

HANDS-ON BEISPIEL (10)



HANDS-ON BEISPIEL (10)



HANDS-ON BEISPIEL (11)

Merge Revisions for C:\Users\David\Desktop\Development\python\collab\app.py

↑ ↓ ↺ Apply non-conflicting changes: » Left « All « Right ↺ | Do not ignore | Highlight words | ?

Changes from main

	Show Details	Result
1	1	
2	2	
3	3	3
4	4	
5	5	
6	6	
7	7	7

```
def calc_two_sums(a: int, b: int, c: int) -> None:
    print("The number is:", str(a + b))

calc_two_sums(3, 10, 22)
```

Changes from origin/main

No changes, 2 conflicts.

1	
2	
3	3
4	4
5	5
6	6
7	7
8	8
9	9

```
def calc_two_sums(a: int, b: int) -> None:
    print("test")
    print("The number is:", str(a + b))

calc_two_sums(3, 10)
```


HANDS-ON BEISPIEL (12)

- Branching
 - Reduziert Merge-Errors
 - Erleichtert die Übersicht
- Alle (wichtig, alles muss aktuell sein, «git pull»)
 - «git checkout -b feat/mein-feature»
 - «mein-feature» bitte ersetzen, Name spielt keine Rolle
 - Erstellt 2-3 neue Files mit einigen Funktionen
 - Macht mehrmals ein «git commit -m "changes to files"»
 - Macht mehrmals ein «git push»
 - Ggf. «git push --set-upstream origin feat/mein-feature»

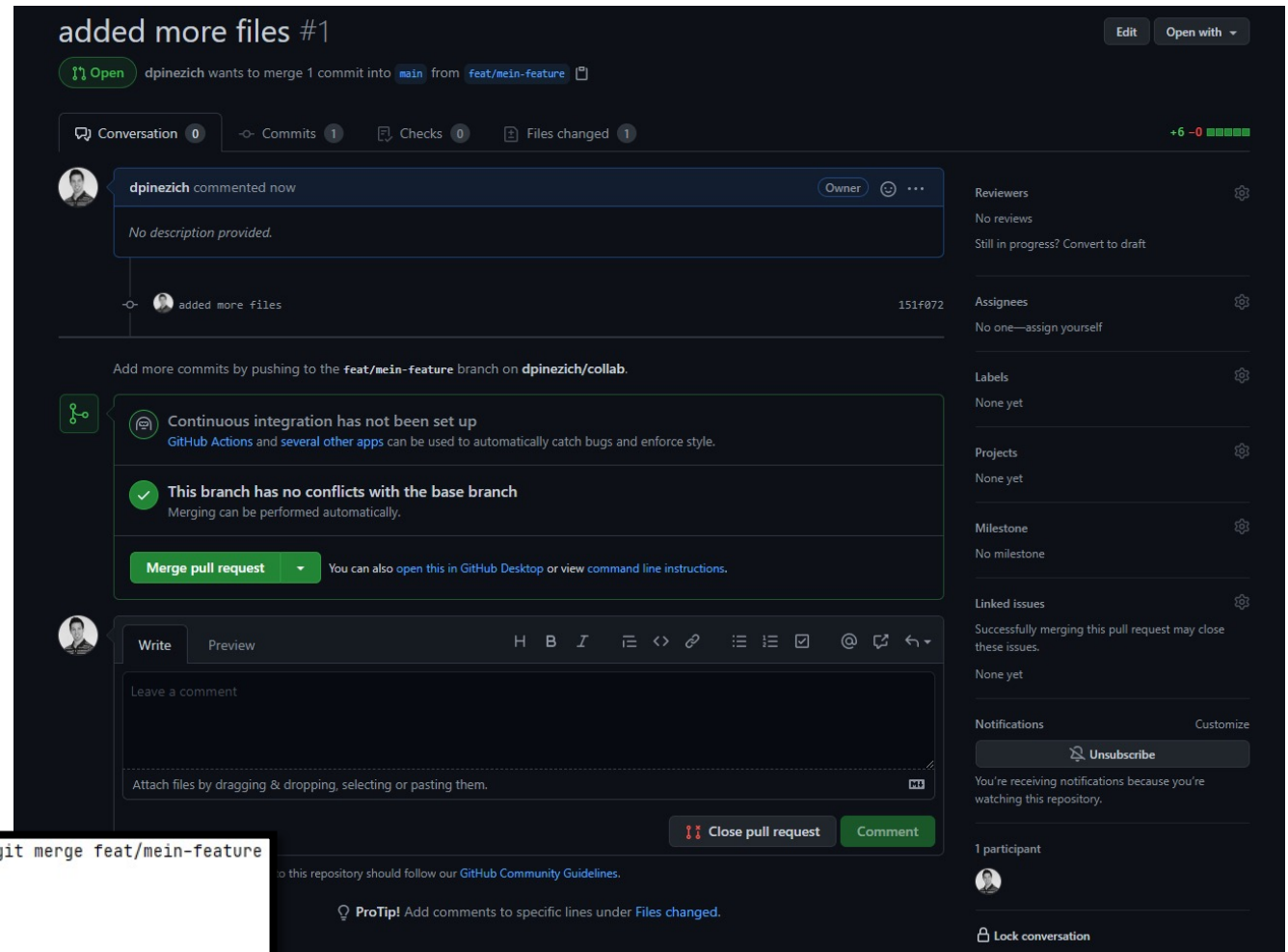


<https://blog.seibert-media.net/wp-content/uploads/2014/03/Gitflow-Workflow-4.png>

HANDS-ON BEISPIEL (13)

- Person 1
 - Bitte wieder auf GitHub einloggen
 - Pull-Requests sind eine tolle Möglichkeit um ein 4-Augen Prinzip sicherzustellen
 - Es kann direkt grafisch «gemerged» werden
 - Oder via «Terminal»
 - «git checkout main»
 - Ggf. «git branch --set-upstream-to=origin»
 - «git pull» + «git merge feat/mein-feature»

```
dave@DESKTOP-ONTVSA6:/mnt/c/Users/David/Desktop/Development/python/collab$ git merge feat/mein-feature
Updating af9c8c2..151f072
Fast-forward
 calcsun.py | 6 +++++
 1 file changed, 6 insertions(+)
 create mode 100644 calcsun.py
```



ZUSAMMENFASSUNG

- Zusammenfassung
 - Git ist mächtig und vereinfacht die Zusammenarbeit
 - Git in kombination mit Github (oder Bitbucket, Gitlab, ...) spart auch das nötige Backup
 - Pull-Requests helfen dabei guten Code zu schreiben und durch «Reviews» noch mehr zu lernen
 - «Commits» gerne oft, und wann immer nötig
 - «Pushs» seltener, meist wenn man eine Aufgabe oder ein «Feature» abgeschlossen hat oder man ein «Backup» möchte

FRAGEN



<https://www.w-fragen-tool.com/themes/contrib/wfragentool/images/custom/w-fragen-seo.png>