Version Control: Merging

PU Tools, Ressourcen, Infrastruktur

Nils Reiter,
nils.reiter@uni-koeln.de

November 19, 2020
(Winter term 2020/21)

Section 1

Introduction

Version Control

So far

- Collect changes for committing: Staging area
- Mark a set of changes as one ‘commit’
- Continue development in a secondary ‘branch’
- Handling remotes: Clone, pull, push

Version Control

So far

- Collect changes for committing: Staging area
- Mark a set of changes as one ‘commit’
- Continue development in a secondary ‘branch’
- Handling remotes: Clone, pull, push

Addendum

Git provides mechanisms for *file handling*

- You still have to do the actual programming
- You may use any programming environment (Notepad++, PyCharm, IDLE)
- What matters for git is the files on the disk

Branches

The result of last weeks demo:

```
$ git log --oneline --graph --all
* 1fe6381 (HEAD -> master) Welt -> Universum
| * 6ccaa54 (french-version) french version
|/
* 1c211da german localisation
* 7f4f4cf new file
```

Branches

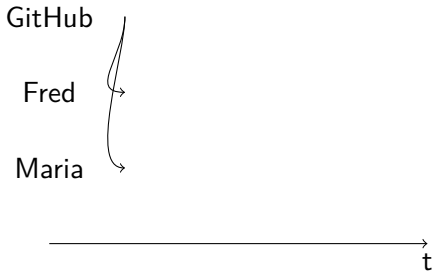
The result of last weeks demo:

```
$ git log --oneline --graph --all
* 1fe6381 (HEAD -> master) Welt -> Universum
| * 6ccaa54 (french-version) french version
|/
* 1c211da german localisation
* 7f4f4cf new file
```

What now?

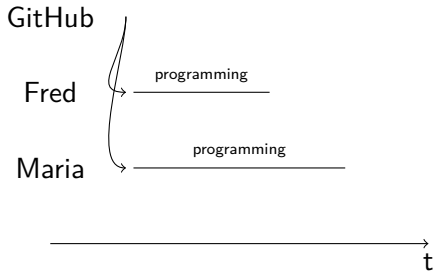
Merging

Situations



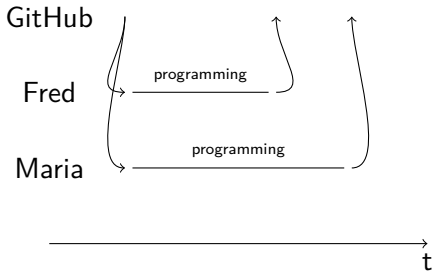
Merging

Situations



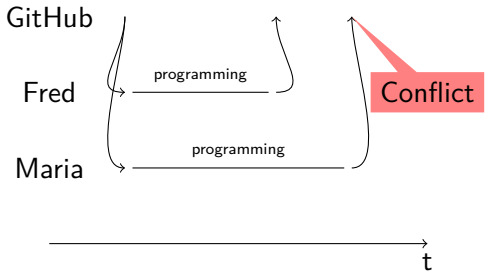
Merging

Situations



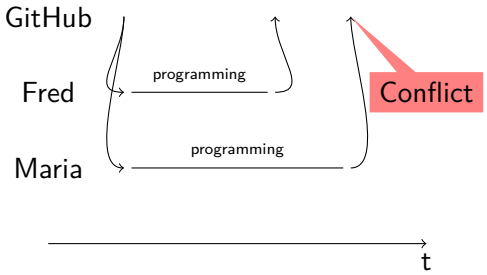
Merging

Situations



Merging

Situations



Conflict resolution options

- Ignore, let Maria overwrite Freds code (this is bad!)
- Create a second copy (this is what Dropbox does)
- Force Maria to *explicitly* merge the code (this is what git does)

Merging

with local branches

Setup:

```
$ git log --oneline --graph --all
* 1fe6381 (HEAD -> master) Welt -> Universum
| * 6ccaa54 (french-version) french version
|/
* 1c211da german localisation
* 7f4f4cf new file
```

- Our branch: `main` (this is what `HEAD ->` tells us)
- We want to incorporate what has been done in branch `french-version`

```
$ git merge french-version
Auto-merging program.py
CONFLICT (content): Merge conflict in program.py
Automatic merge failed; fix conflicts and then commit the result.
```

Merging

with local branches

original

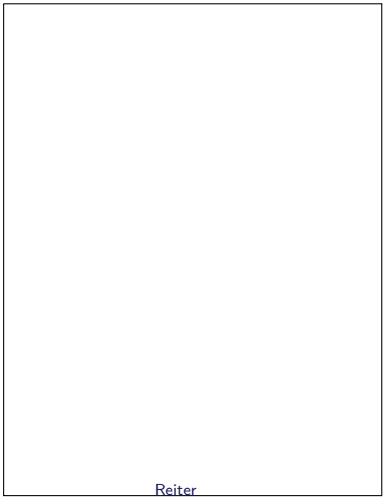
```
def add(x,y):
    return x+y

for i in range(0,10):
    d = add(i,i*2)
    print(d)
```

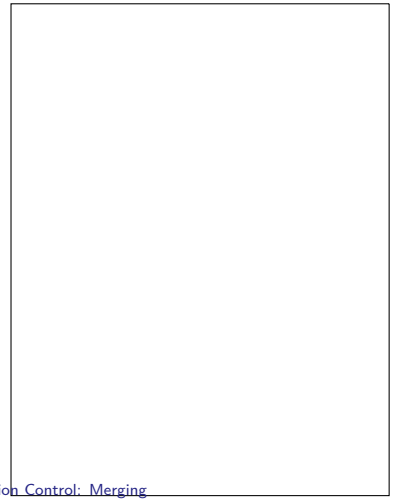
Merging

with local branches

maria



fred



Merging

with local branches

maria

```
def add(x,y):                    10
    return x+y                   11


for i in range(0,10):            20
    d = add(i,i*2)               21
    d = add(i,i*3)               21
    print(d)                     22


print("finished.")               30
```

fred

Merging

with local branches

maria

```
def add(x,y):                    10
    return x+y                   11


for i in range(0,10):            20
    d = add(i,i*2)               21
    d = add(i,i*3)               21
    print(d)                     22


print("finished.")               30
```

fred

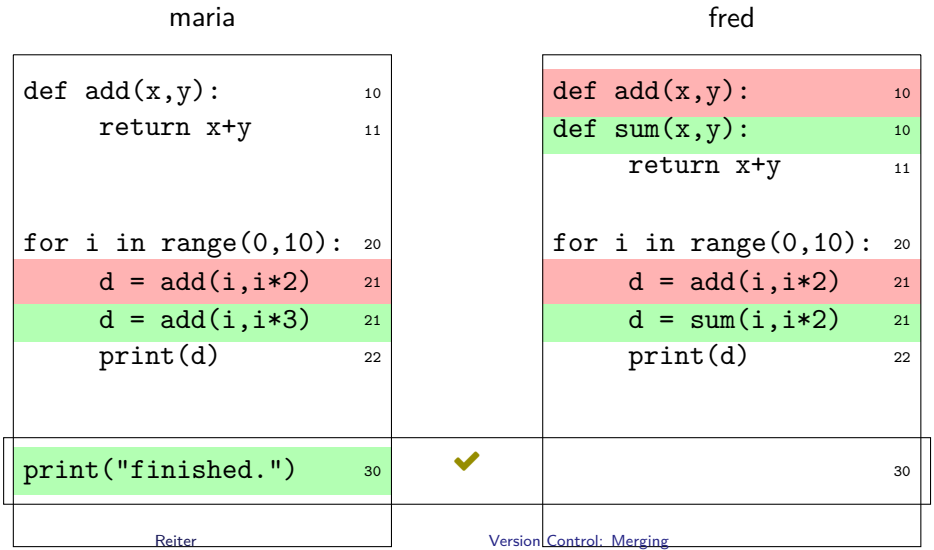
```
def add(x,y):                    10
def sum(x,y):                    10
    return x+y                   11

for i in range(0,10):            20
    d = add(i,i*2)               21
    d = sum(i,i*2)               21
    print(d)                     22


                                 30
```

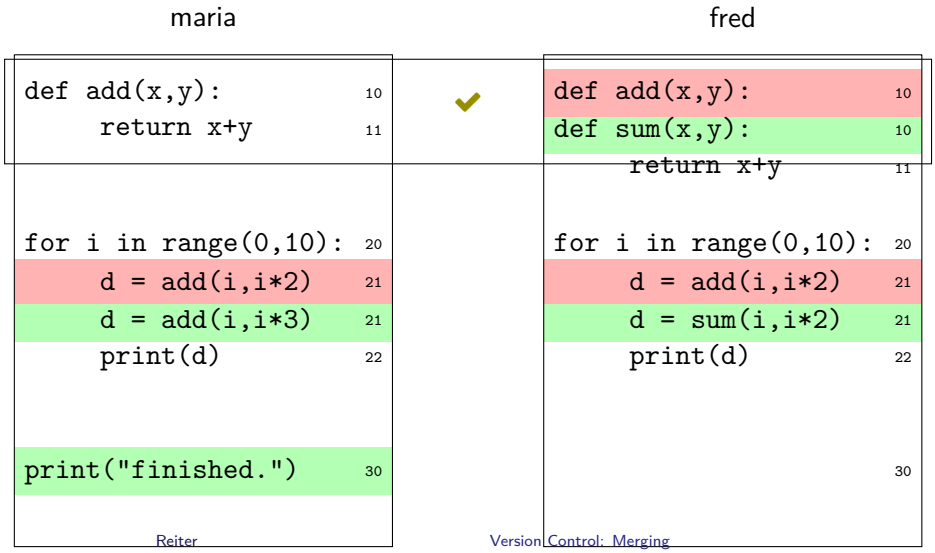
Merging

with local branches



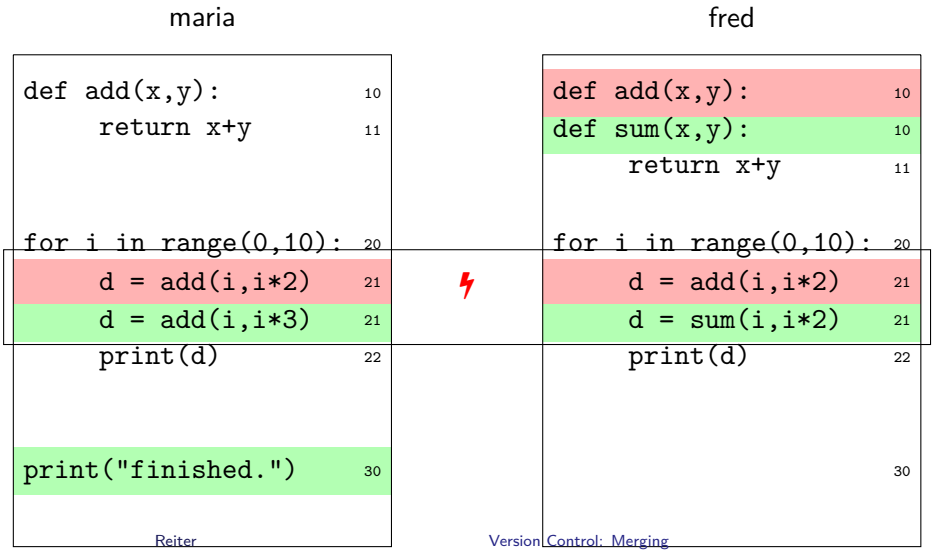
Merging

with local branches



Merging

with local branches



Merging

with local branches

Fred runs: `git merge maria`

```
def sum(x,y):
    return x+y
```

← merged automatically

```
for i in range(0,10):
<<<<<<< HEAD
    d = sum(i,i*2)
=======
    d = add(i,i*3)
>>>>>>> maria
    print(d)
```

} conflict, we need to take care of this manually

```
print("finished.")
```

← merged automatically

Merging

Summary

- If there are independent commits on two branches, they need to be merged *explicitly*

Merging

Summary

- If there are independent commits on two branches, they need to be merged *explicitly*
- Merging works automatically, if different areas of a file have been edited (or entirely different files)

Merging

Summary

- If there are independent commits on two branches, they need to be merged *explicitly*
- Merging works automatically, if different areas of a file have been edited (or entirely different files)
- If the same file area has been edited, we need to manually merge the files
  - After editing, we add and commit as usual

Merging

Summary

- If there are independent commits on two branches, they need to be merged *explicitly*
- Merging works automatically, if different areas of a file have been edited (or entirely different files)
- If the same file area has been edited, we need to manually merge the files
  - After editing, we add and commit as usual
- Merge conflicts are to be expected and *normal*
  - More coordination can avoid merge conflicts to a certain extent

Section 3

Remotes

Decentralized

- "Git is decentralized": What does this mean exactly?

Decentralized

- "Git is decentralized": What does this mean exactly?
- No central server required
- A local git repository stores the entire history, all branches and tags
- Every clone of the repository has the entire history
  - Offline working galore!

Remotes

- Each repository can be associated with multiple ‘remotes’
  - Usually, one remote is called ‘origin’
- `clone` makes a local clone *and* sets on remote to point to the source
- Merging works across remote repositories

Downloading stuff

- A branch can be set to ‘track’ a remote branch
  - Typically, you want the branches to have the same name
- `git fetch` downloads all tracked branches to your local repository, but keeps your working copy as it is
- `git pull` fetches the changes from the server *and* merges them into your working copy
  - Merge conflicts can occur!
- `git push` pushes your local changes to the tracking branch on the server
  - If the remote branch moved on, you’ll be forced to pull and merge first

Section 3

Remotes

Using git

- Git provides technical foundation
- Different ways to use git
- Using git well requires self-discipline
  - Spotting an issue while implementing feature X? Take a note and leave for now!
  - Aim for logical commits

Using git

- Git provides technical foundation
- Different ways to use git
- Using git well requires self-discipline
  - Spotting an issue while implementing feature X? Take a note and leave for now!
  - Aim for logical commits
- If in a group project
  - Discuss how to use git
    - When do we make new branches?
    - Who is responsible for the merging?

Best Practice

Driessen (2010)

The following is based on

`https://nvie.com/posts/a-successful-git-branching-model/`

- Two main branches: `main` and `develop`
  - `main` contains stable and released versions
  - `develop` is used to develop the next release version

Best Practice

Driessen (2010)

The following is based on

`https://nvie.com/posts/a-successful-git-branching-model/`

- Two main branches: `main` and `develop`
  - `main` contains stable and released versions
  - `develop` is used to develop the next release version
- Three kinds of additional branches: fixes, features, release
  - `fixes` branch off of the `main` branch
    - Merged back into a `release` branch and `develop`
    - Used for urgent bug fixes
  - `features` branch off of the `develop` branch
    - Merged back into `develop`
    - Used to develop new features of the software
  - `release` branches branch off of the `develop` branch
    - Merged back into `main`
    - Used to clean up everything for a public release
  - Additional branches can be deleted once they are merged into develop/main

Section 5

Summary

Miscellaneous Git Topics

tags A commit can be tagged, which is just a nice name for this commit. Typically used to mark specific (released) versions

help `git help COMMAND` shows information about the COMMAND on the command line

online There are 128 188 questions with answers about git on `stackoverflow.com/questions/tagged/git`

Summary

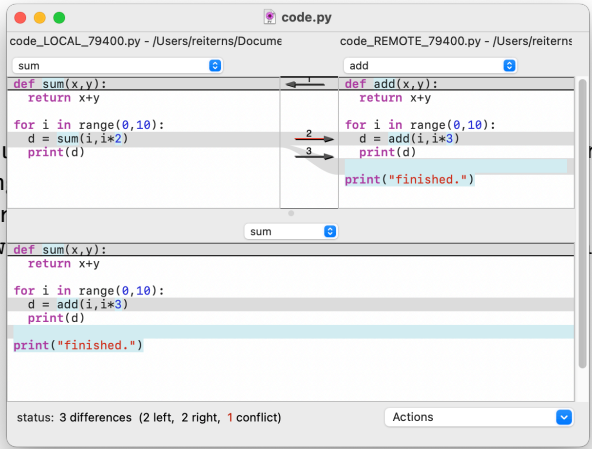
- Merging
  - Git attempts automatic merging of changed lines in different files or file sections
  - Manual merging requires attention and care, but is doable

Summary

- Merging
  - Git attempts automatic merging of changed lines in different files or file sections
  - Manual merging requires attention and care, but is doable
  - GUIs help (search for 'diff tools') or check
    `https://en.wikipedia.org/wiki/Comparison_of_file_comparison_tools`

Summary

- Merging
  - Git attempts au … r file sections
  - Manual mergin…
  - GUIs help (sear… https://en.w… …rison_tools

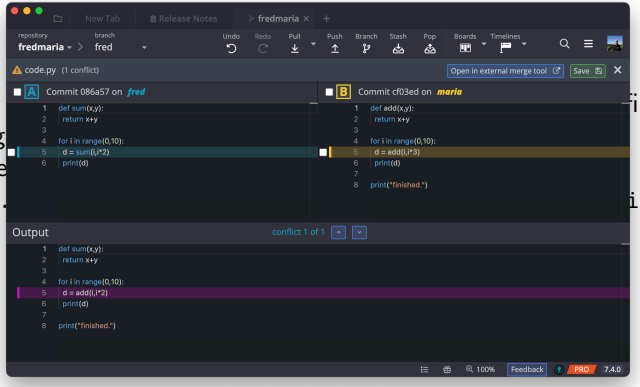


FileMerge on Mac OS (part of XCode)

Summary

- Merging
  - Git attempts
  - Manual merg
  - GUIs help (se

  `https://en.`



GitKraken

Summary

- Merging
  - Git attempts automatic merging of changed lines in different files or file sections
  - Manual merging requires attention and care, but is doable
  - GUIs help (search for 'diff tools') or check `https://en.wikipedia.org/wiki/Comparison_of_file_comparison_tools`
  - Coordination in a group helps to minimize merge effort

Summary

- Merging
  - Git attempts automatic merging of changed lines in different files or file sections
  - Manual merging requires attention and care, but is doable
  - GUIs help (search for 'diff tools') or check `https://en.wikipedia.org/wiki/Comparison_of_file_comparison_tools`
  - Coordination in a group helps to minimize merge effort
- Remotes
  - Entire repository can be synchronized with remote repositories
  - If a branch tracks a remote branch, changes can be pulled directly
    - This may result in a merge conflict, if the remote branch has been altered!

Exercise 3

`https://github.com/idh-cologne-tools-ressourcen-infra/exercise-03`

References I

Driessen, Vincent (2010). *A successful Git branching model*. URL: https://nvie.com/posts/a-successful-git-branching-model/.