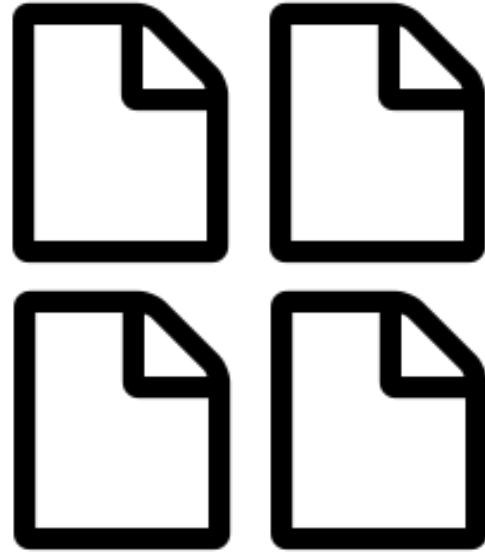
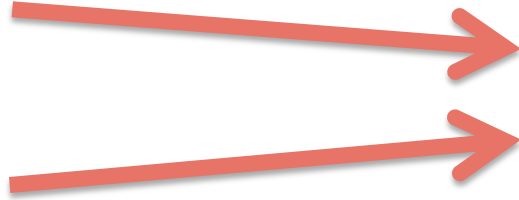




?



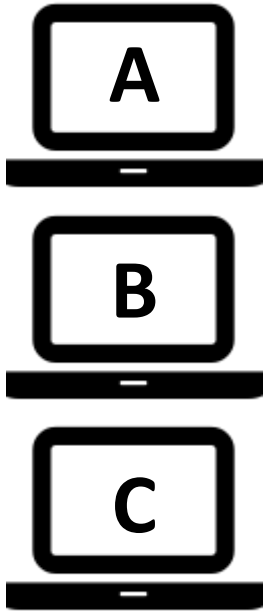


Versionsverwaltung mit **git**

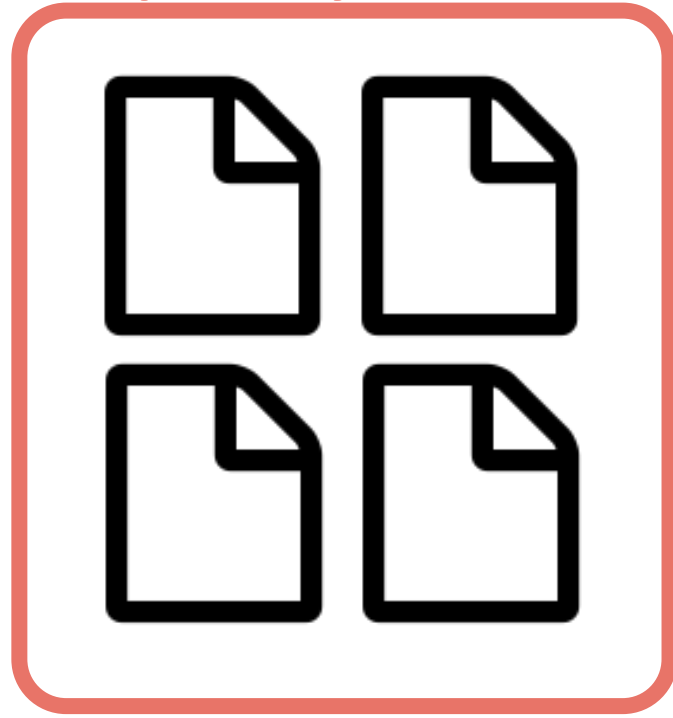
- Grundprinzip
- Git als Versionsverwaltungssoftware
- Allgemeine Funktionsweise
- Branches
- Zusammenarbeit mehrerer Repositorys
- GitHub als Projekthoster
- Praxisübung



1. GRUNDPRINZIP



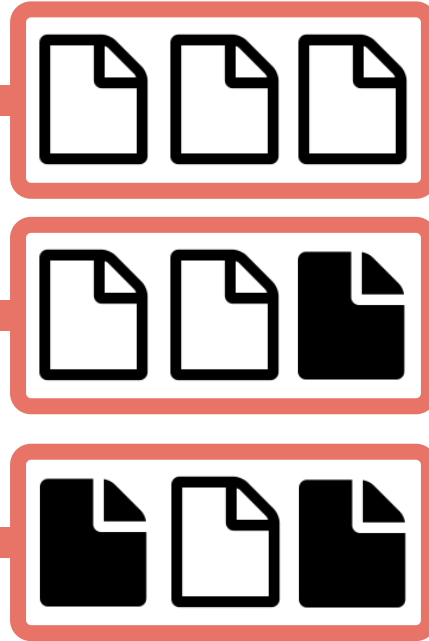
Repository



Initial Commit

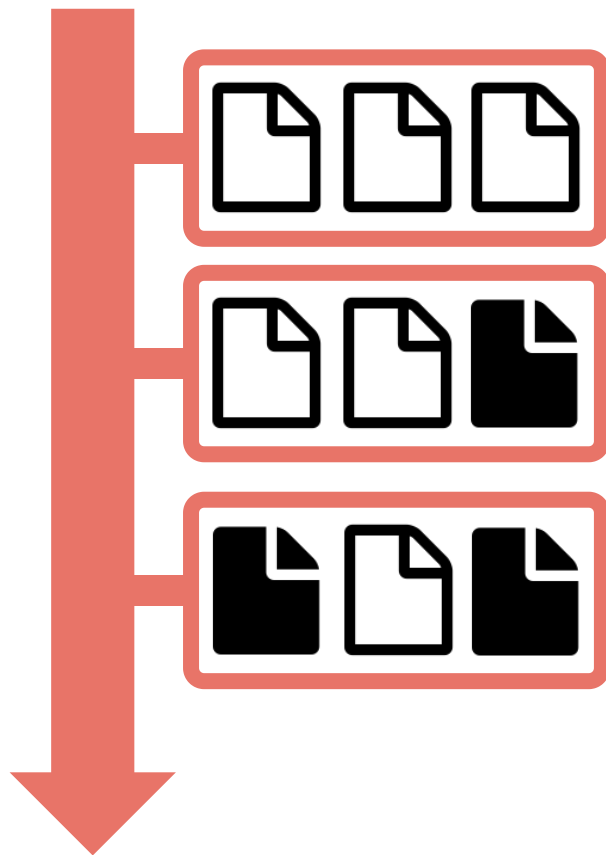
Neuer Header in
index.htm von Benutzer A

Bugfix in style.css von
Benutzer C

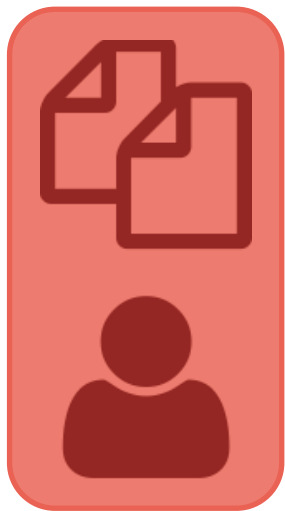


VORTEILE

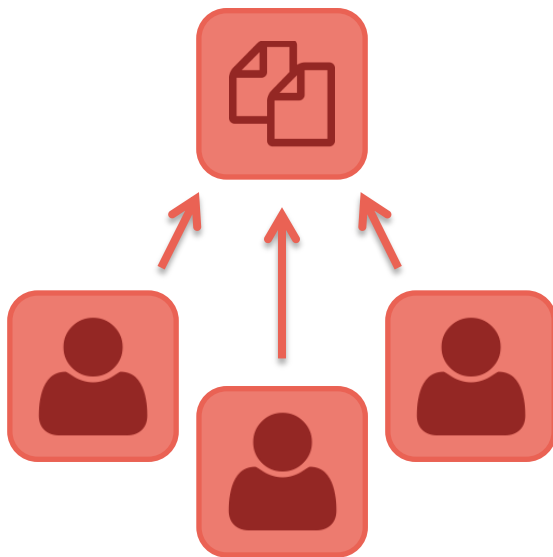
- Protokollierung
- Datensicherung
- Bessere Kollaboration



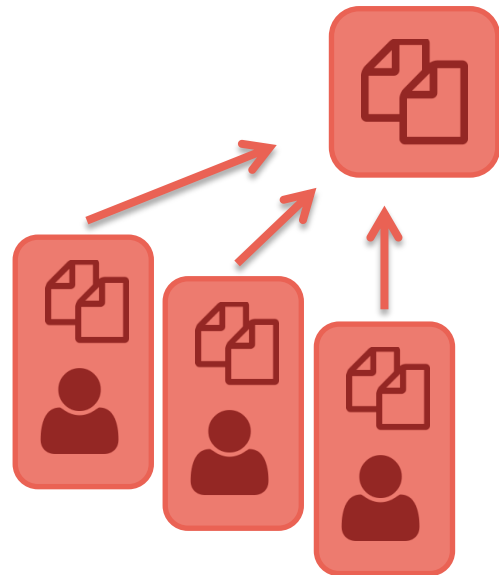
Lokale Versionsverwaltung



Zentrale Versionsverwaltung



Verteilte Versionsverwaltung





2. GIT ALS VERSIONSVERWALTUNG

I'm an egoistical bastard, and I name all my projects after myself. First ,Linux', now ,Git'.

- Linus Torvalds



[http://upload.wikimedia.org/wikipedia/commons/5/5c/Linus_Torvalds_\(cropped\).jpg](http://upload.wikimedia.org/wikipedia/commons/5/5c/Linus_Torvalds_(cropped).jpg)

Über git

- Verteilte Versionsverwaltung
- Ursprünglicher Zweck: Verwaltung von Linux
- Schneller als Alternativen
- Befehlszeilenprogramm
- Optionale Verwendung von GUIs



3. ALLGEMEINE FUNKTIONSWEISE

Repositorys in git

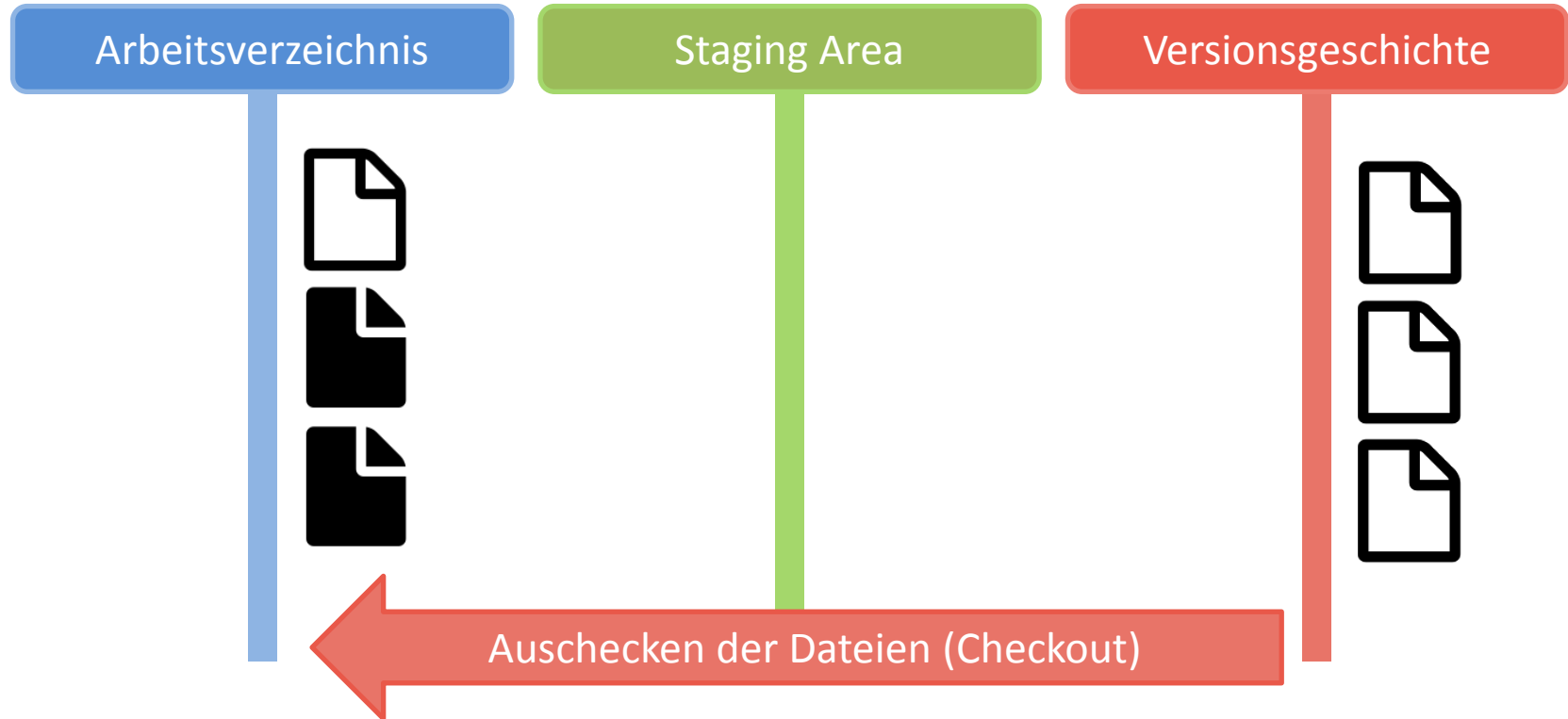
```
$ mkdir speicherort
```

```
$ cd speicherort
```

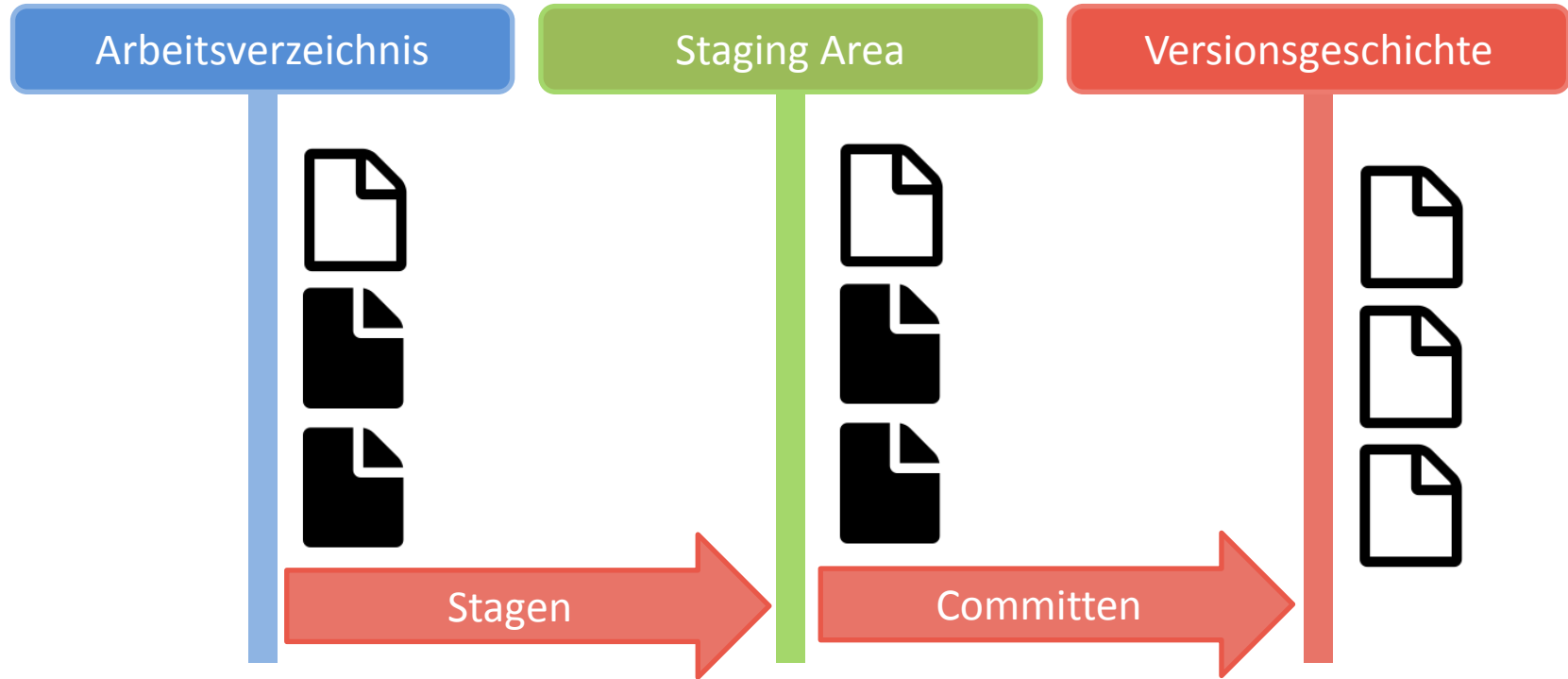
```
$ git init
```

```
Initialized empty Git repository in  
/Users/JonDoe/speicherort/.git
```

Arbeitsverzeichnis und Staging Area



Arbeitsverzeichnis und Staging Area



Arbeitsverzeichnis und Staging Area

Status überprüfen:

```
$ git status
```

```
On branch master
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will be  
committed)
```

```
README.txt
```

```
nothing added to commit but untracked files present (use  
"git add" to track)
```


Arbeitsverzeichnis und Staging Area

Dateien zur Staging Area hinzufügen:

```
$ git add datei.h  
$ git add pfad/zu/verzeichnis
```

Arbeitsverzeichnis und Staging Area

Status überprüfen:

```
$ git status
```

```
On branch master
```

```
Changes to be committed:
```

```
(use "git reset HEAD <file>..." to unstage)
```

```
new file:   datei.h
```

```
Untracked files:
```

```
[...]
```

Arbeitsverzeichnis und Staging Area

Änderungen comitten:

```
$ git commit
```

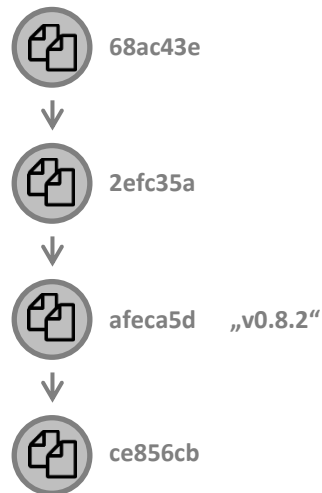
```
$ git commit -m „Änderungskommentar“
```

Commit Bezeichnungen

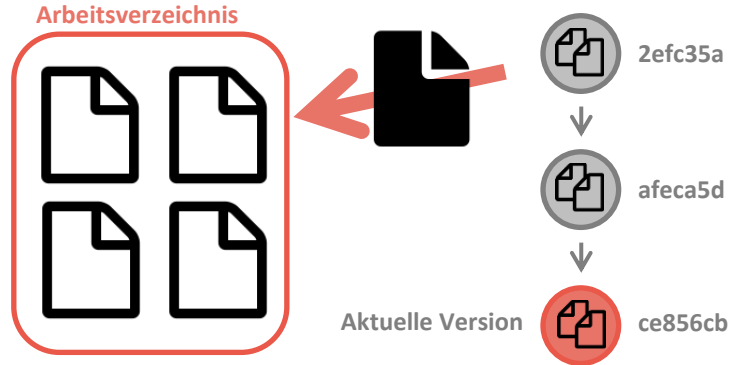
- 40-stellige SHA-1 Checksumme
- Berechnet aus Repository Dateien
- „Taggen“ von Versionen möglich

```
$ git tag <vname> <commit>
```

```
$ git tag v0.8.2 afeca5d
```



Auschecken



Auschecken

Dateien auschecken:

```
$ git checkout <commit> <file>  
$ git checkout 47efd8c datei.h
```

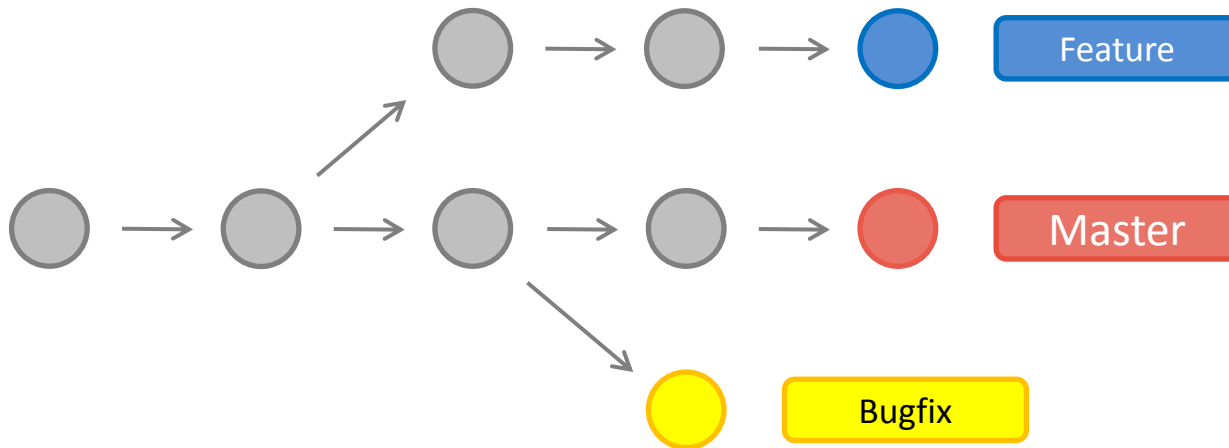
Commit auschecken:

```
$ git checkout <commit>  
$ git checkout 47efd8c
```



4. BRANCHES

Branches



Befehle zu Branching

 `$ git branch`

Listet alle vorhandenen Branches auf

Befehle zu Branching




```
$ git branch
```

```
$ git branch <branchname>
```

Aktueller Branch wird kopiert und unter
<branchname> gespeichert


Befehle zu Branching



```
$ git branch  
$ git branch <branchname>  
$ git checkout <branchname>
```

Wechselt zu <branchname>

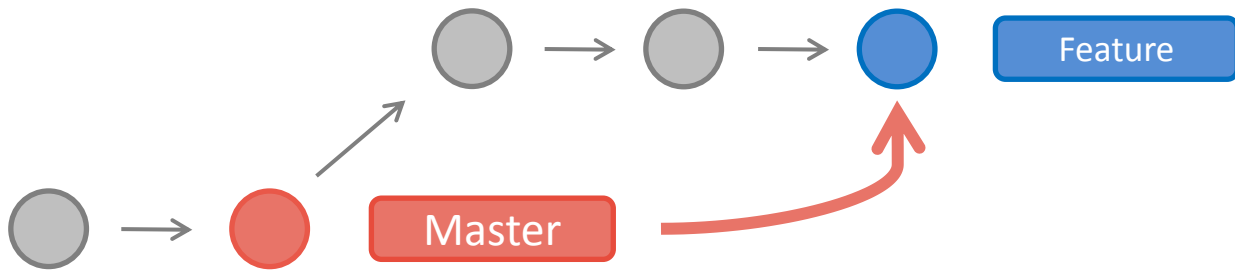
Befehle zu Branching



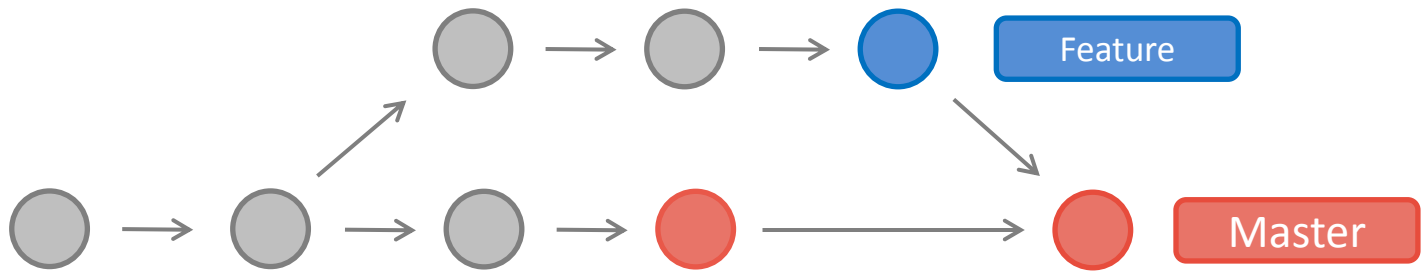
```
$ git branch  
$ git branch <branchname>  
$ git checkout <branchname>  
$ git checkout -b <branchname>
```

Erstellt Kopie von aktuellem Branch als
<branchname> und wechselt zu diesem

Branch Merging: FF-Merge



Branch Merging: 3-way Merge





Branch Merging

```
$ git merge <branch>
```

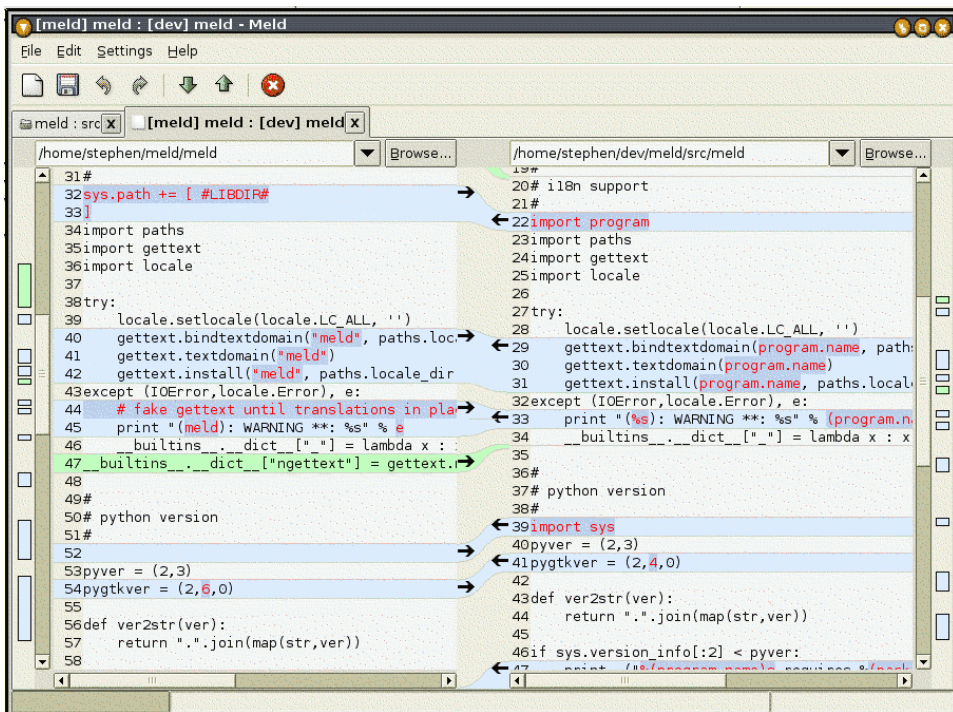
- Führt aktuellen Branch mit <branch> zusammen
- Git bestimmt automatisch Algorithmus

Branch Merging

```
$ git merge --no-ff <branch>
```

- Führt aktuellen Branch mit <branch> zusammen
- Erzwingt Generierung eines neuen Commits

Konflikte mit Meld lösen



```
[meld] meld : [dev] meld - Meld
File Edit Settings Help

meld : src [meld] meld : [dev] meld

/home/stephen/meld/meld /home/stephen/dev/meld/src/meld

31#
32sys.path += [ #LIBDIR#
33]
34import paths
35import gettext
36import locale
37
38try:
39    locale.setlocale(locale.LC_ALL, '')
40    gettext.bindtextdomain("meld", paths.loc
41    gettext.textdomain("meld")
42    gettext.install("meld", paths.locale_dir
43except (IOError,locale.Error), e:
44    # fake gettext until translations in pla
45    print "(meld): WARNING **: %s" % e
46    __builtins__.__dict__["_"] = lambda x :
47    __builtins__.__dict__["ngettext"] = gettext.
48
49#
50# python version
51#
52
53pyver = (2,3)
54pygtkver = (2,6,0)
55
56def ver2str(ver):
57    return ".".join(map(str,ver))
58
```

http://meld.sourceforge.net/meld_file1.png

Rückblick

```
$ git checkout -b new-feature
```

```
$ git add datei.h
```

```
$ git commit -m „Neue Datei“
```

```
[...]
```

```
$ git checkout master
```

```
$ git add andereDatei.h
```

```
$ git commit -m „Neue Datei“
```

```
[...]
```

```
$ git merge new-feature
```

```
$ git branch -d new-feature
```

New-
feature

Master



5. ZUSAMMENARBEIT MEHRER REPOSITORYS

Mehrere Repositorys

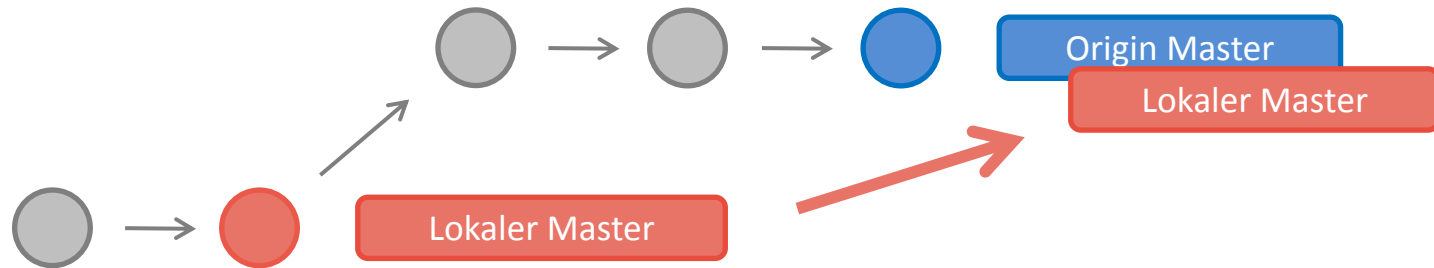
```
$ git init --bare
```

```
$ git clone <url>
```

```
$ git remote add <name> <url>
```

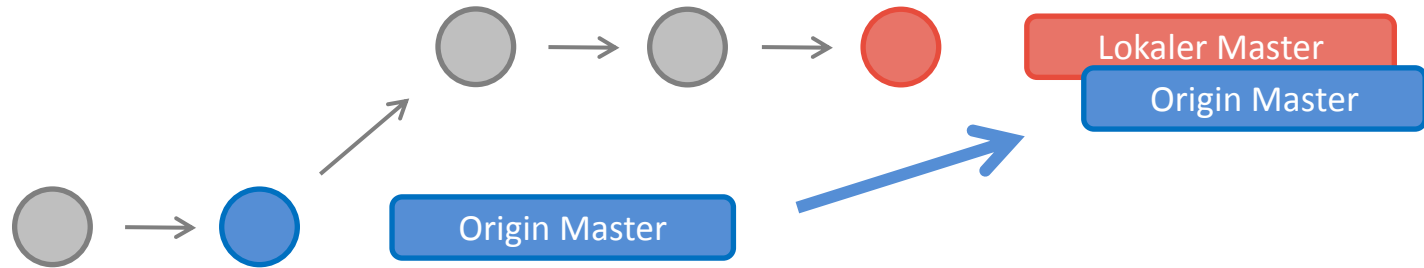


Pulling



```
$ git pull <url>
```

Pushing



```
$ git push <url> [<branch>]
```



6. GITHUB ALS PROJEKTHOST

Arbeitsauftrag

- Download git: git-scm.com/download/win
- Repository klonen:
<https://github.com/lukasbach/Modernizr.git>
- Dateien in der Arbeitskopie ändern, stagen, comitten, pushen

Username: gfsgittest

Passwort: Gfsgittest1

Arbeitsauftrag

```
$ git clone https://github.com/lukasbach/bootstrap.git  
[Änderungen im Arbeitsverzeichnis vornehmen]
```

```
$ git add .
```

```
$ git commit -m „Commit-message“
```

```
$ git push origin master
```

Username: gfsgittest

Passwort: Gfsgittest1

Literatur

- **HERBERT BRAUN:** Unvergessen — Erste Schritte mit dem Versionskontrollsystem Git und mit GitHub. c't 2014-5
- **GIT:** Offizielle Webseite. git-scm.com
- **ATLASSIAN:** Git Anleitungen. www.atlassian.com/de/git
- **SAINTSJD:** What is a bare git repository? www.saintsjd.com/2011/01/what-is-a-bare-git-repository
- **CHRISTIAN JOHNER:** Versionsmanagement. <http://www.johner.org/tech-docs/softwareentwicklung/management/versionsmanagement/>
- **GERRIT VAN AAKEN:** Versionsmanagement für Anfänger. praegnanz.de/weblog/versionsmanagement-fuer-anfaenger
- **KARL FOGEL:** Versionsverwaltung. producingoss.com/de/vc.html
- **NATHAN DE VRIES:** Git Ready. de.gitready.com