Git – gewusst wie

Informatica Feminale
12.–13. August 2017

Nimm dir Post-Its und schreibe zu git jeweils 3 Dinge auf die du schon weißt und die du noch lernen möchtest

Versionsverwaltung - Was und Wozu

Welches Problem löst eine Versionsverwaltung?

Mehrere Personen arbeiten gleichzeitig an Textdateien und möchten ihre Arbeitsstände regelmäßig austauschen, bei Bearbeitungskonflikten darauf hingewiesen werden und gelegentlich einen Blick auf alte Versionen werfen.

Was ist eine Versionsverwaltung?

- Speichert Dokumente in einem zentralen Topf
- Speichert zu jeder Änderung einen Eintrag mit Autor, Datum, Zusammenfassung und Inhalt der Änderung
- Falls ein Fehler passiert, kann auf eine alte Version zurückgesprungen werden
- Falls mehrere Personen die gleiche Datei ändern, weist die Versionsverwaltung darauf hin
- Ermöglicht die Pflege mehrerer Versionen einer Software (1.x, 2.x, etc)

Alternativen

- Mercurial
  - Ähnliche Konzepte wie git
  - schlechtere Toolunterstützung, wenig verbreitet
- SVN
  - früher sehr populär
  - einfacheres Konzept, weniger Möglichkeiten (z.B. kein offline arbeiten, paralleles Pflegen von Versionen schwierig)
- Exotisch: CVS, bazaar

Warum hat sich git durchgesetzt?

- Stark erweiterte und vereinfachte Funktionen gegenüber älteren Systemen
- Hohe Sichtbarkeit und Reputation: Initiiert von Linus Torvald, eingesetzt für den Linux Kernel
- Umfangreiches Tooling, Dokumentation und Webangebote verfügbar

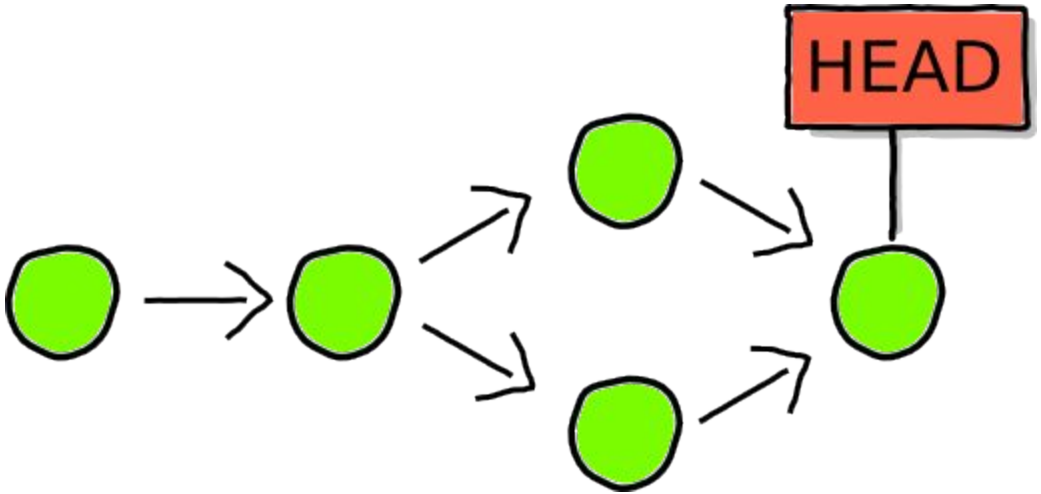
Aufgabe

Fasse die letzten Folien in einem Satz zusammen und notiere ihn in der Datei notizen.md.

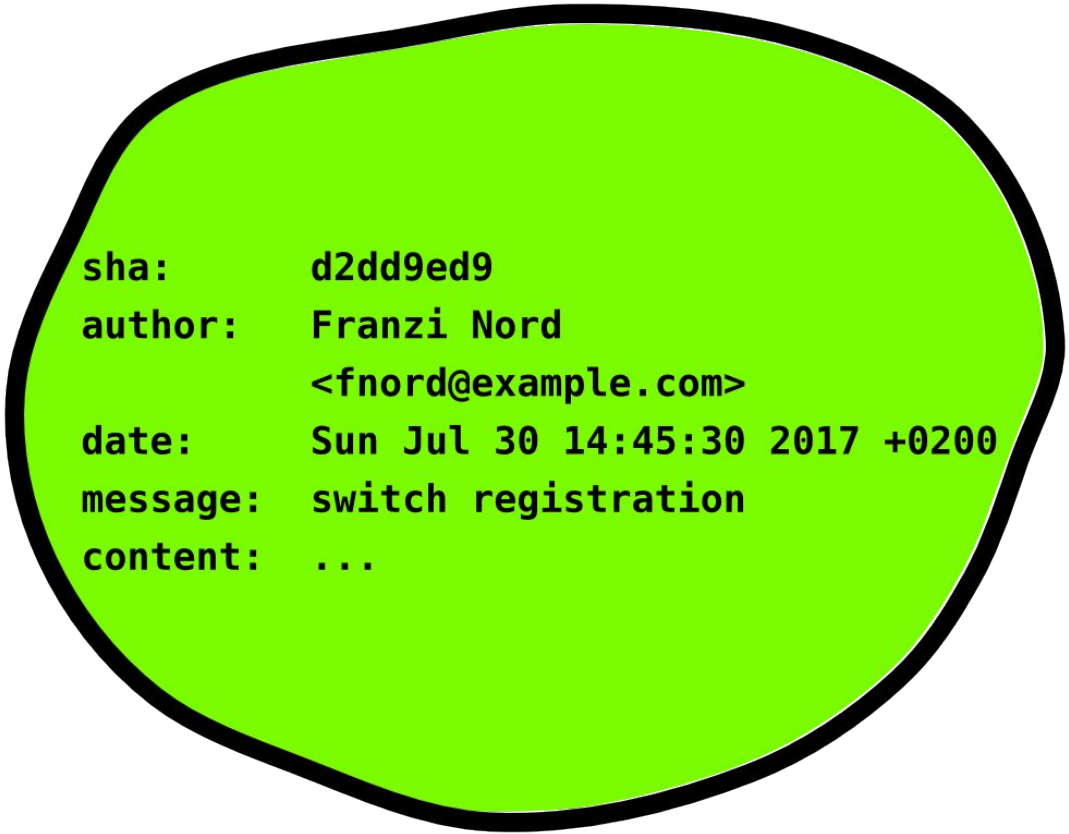
Git Grundbegriffe

Repository

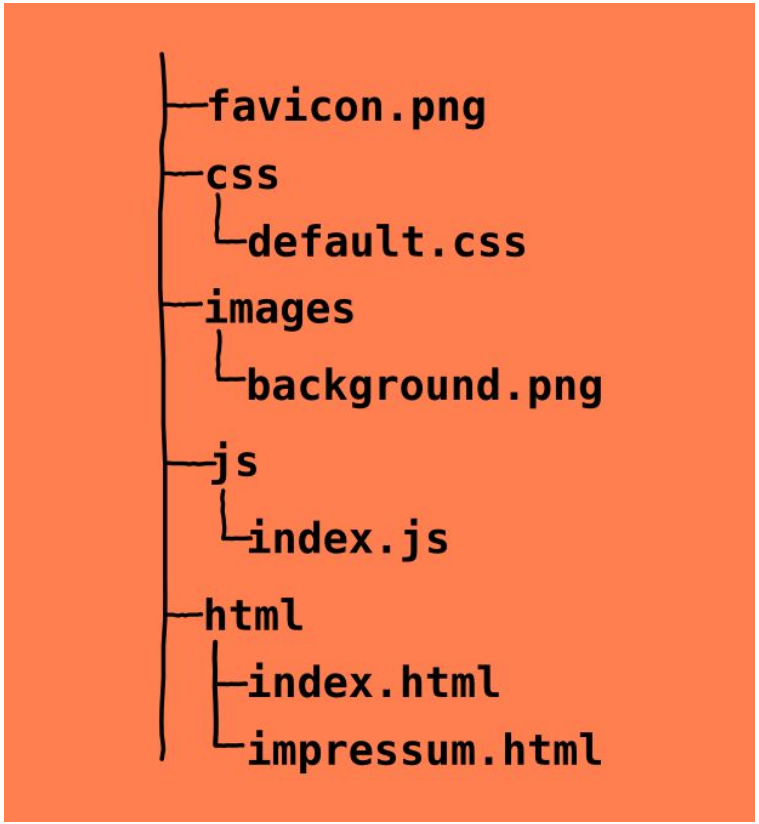
master



Commit


sha: d2dd9ed9
author: Franzi Nord
<fnord@example.com>
date: Sun Jul 30 14:45:30 2017 +0200
message: switch registration
content: ...

Working copy



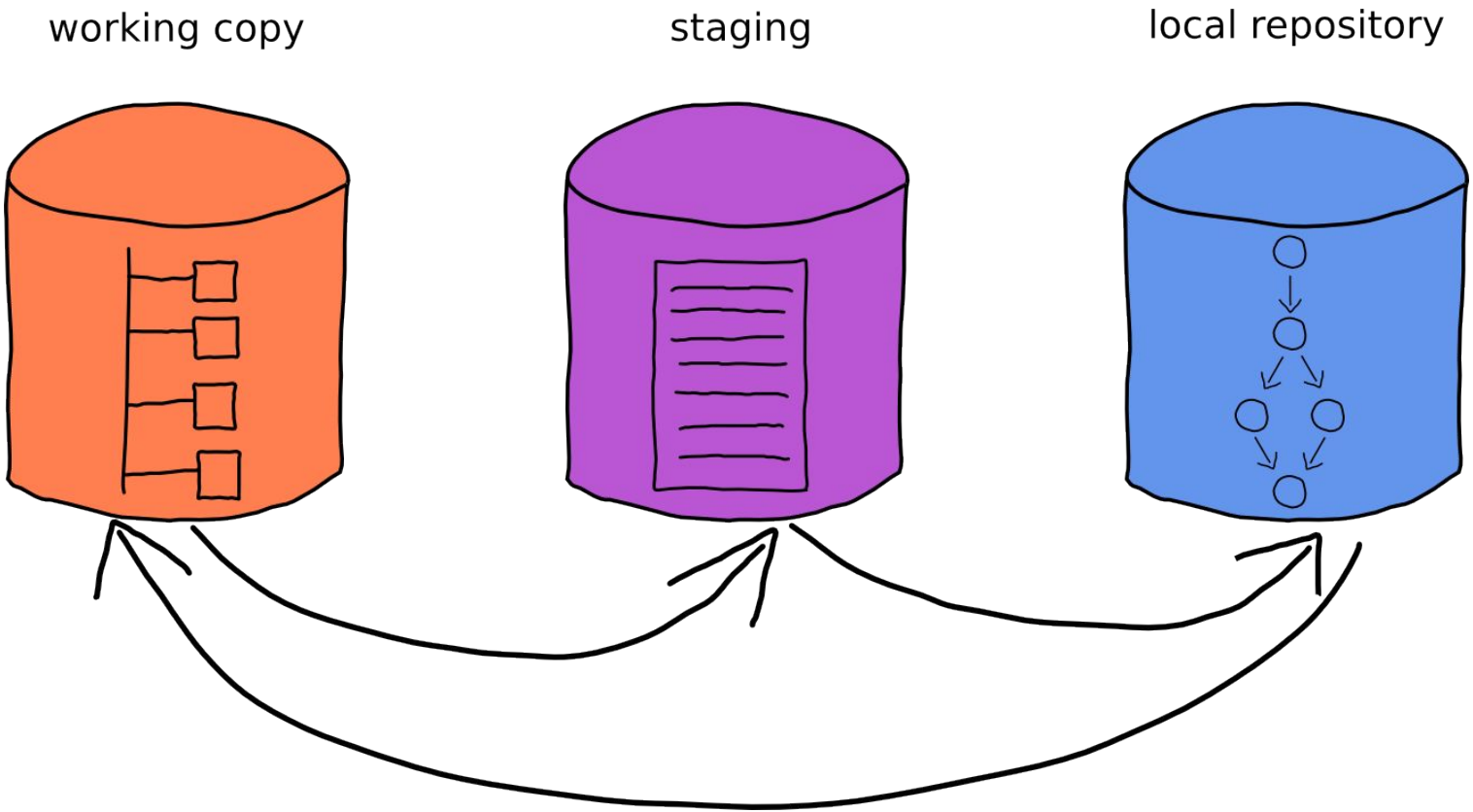
Staging Area

```
js/index.js geändert

html/news.html hinzugefügt

images/background2.jpg gelöscht
```

Überblick



Aufgabe

Stell dir vor du erstellst ein Quiz über git. Schreibe zu den Grundbegriffen die du gerade gehört hast eine Testfrage auf eine Karte. Tausche die Karte mit einer Nachbarin und prüfe ob sie deine Frage beantworten kann.

Notiere Frage und Antwort in notizen.md.

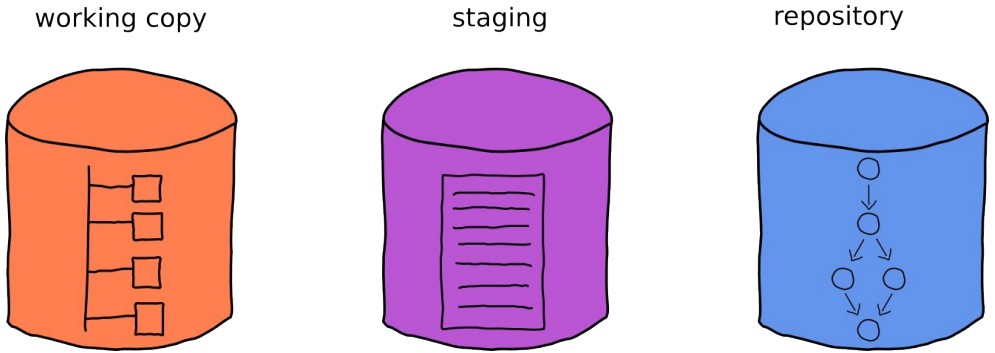
Grundlegende Kommandos

Neues Repository anlegen

- git init
  - erzeugt Verzeichnisstruktur und Metadaten für Repository und Staging Area
  - Umgebendes Verzeichnis ist Ablage für Arbeitskopie

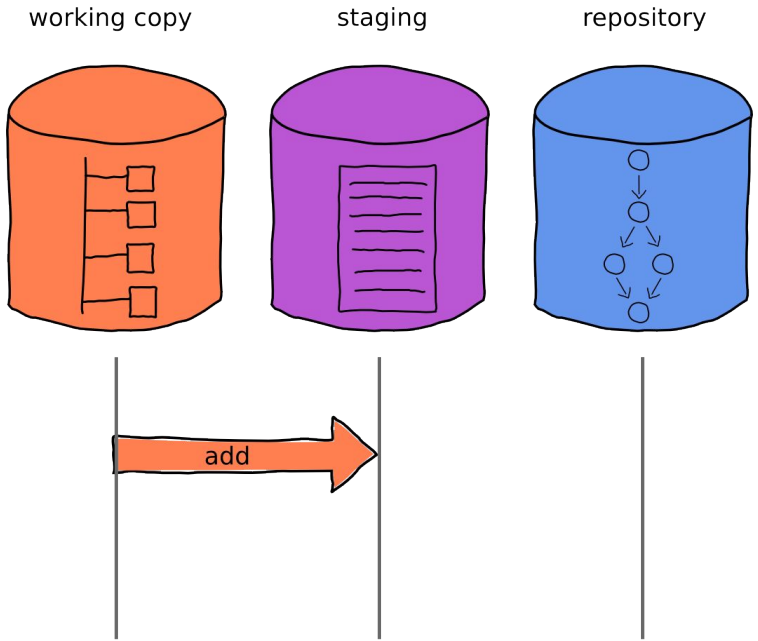
Beispiel:

```
$ cd git-kurs
$ git init
```



Dateien im Staging Area vormerken

- git add
  - Merkt Änderungen in Dateien im Staging Area vor



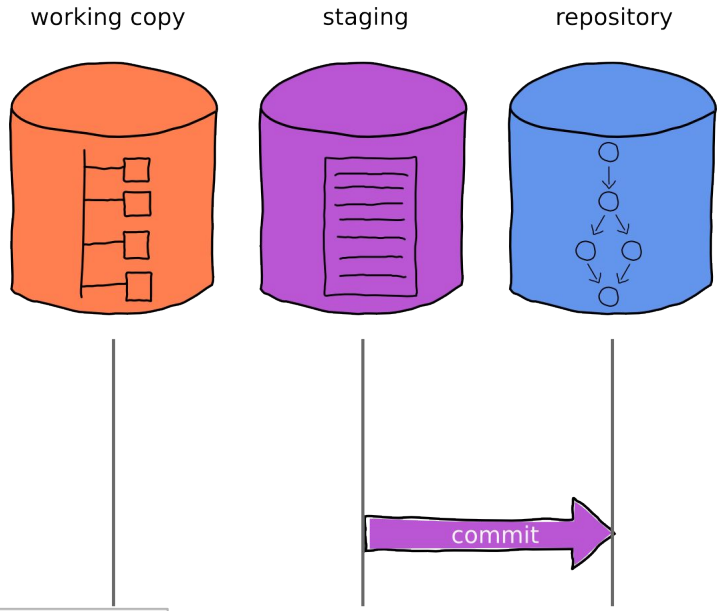
Beispiele:

```
$ git add README.md

$ git add *.md
```

Änderung ins Repository übernehmen

- git commit
  - Commit message als Parameter

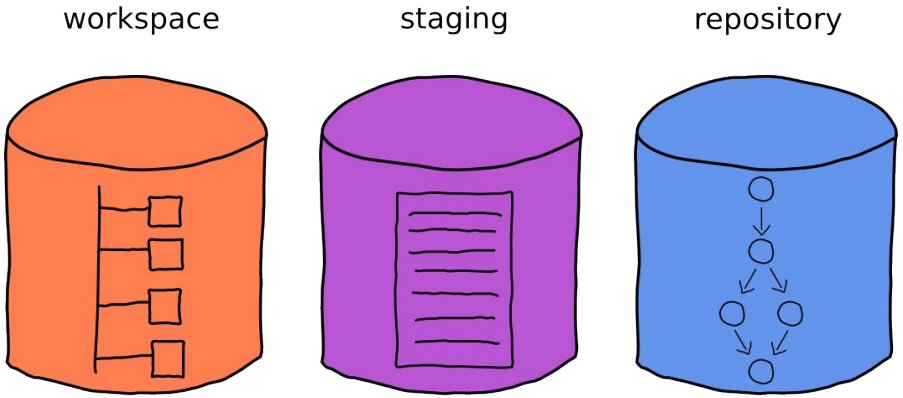


Beispiel:

```
$ git commit -m “README.md hinzugefügt”
```

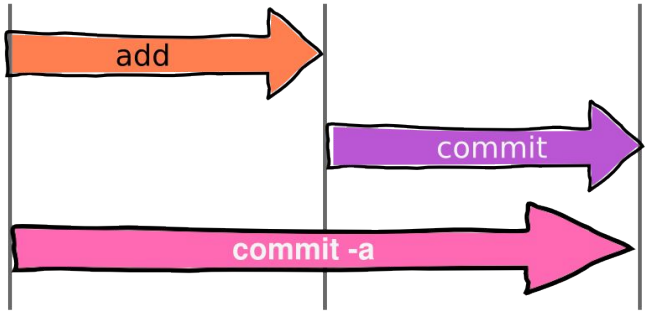
Vereinfachung: add + commit

- git commit -a
  - Alle Änderungen vorhandener Dateien direkt committen
  - Neue Dateien müssen immer mit add hinzugefügt werden



Beispiel:

```
$ git commit -a -m “install ergänzt”
```

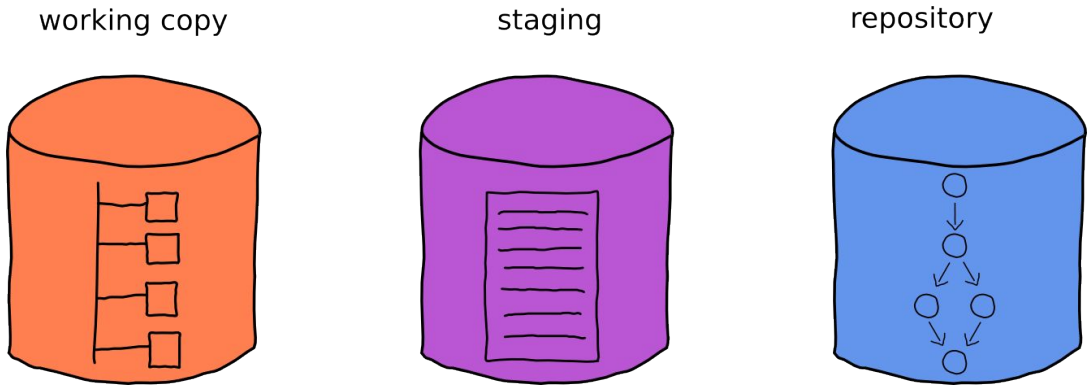


Anzeige geänderter Dateien

- git status
  - Zeigt Dateien und Änderungen, die noch nicht im Staging Area sind
  - Zeigt Dateien und Änderungen, die im Staging Area aber noch nicht im Repository sind

Beispiel:

```
$ git status
```



Anzeige der Commit Historie

- git log
  - Zeigt Inhalt und Metadaten einzelner Commits des Repositories an
  - Viele Formatoptionen

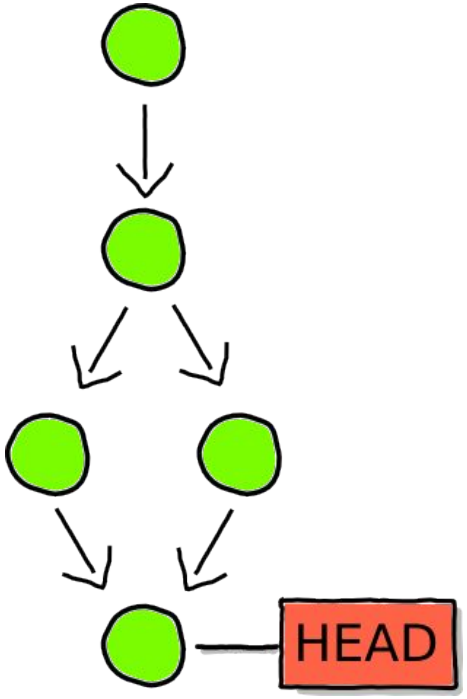
Beispiel:

```
$ git log

$ git log -p -1

$ git log -3

$ git log --oneline
```



Anzeige geänderter Zeilen

- git diff <commit>
  - Änderungen zwischen Working copy und Commits
- git diff <datei>
  - Änderungen zwischen Working Copy und Staging Area
- git diff --staged <datei>
  - Änderungen in der Staging Area

Änderungen rückgängig machen

- Commit: git revert HEAD
  - Macht die Änderungen des letzten Commits im Repository rückgängig in dem es einen neuen, “umgekehrten” Commit erzeugt
- Add: git reset <datei>
  - Entfernt die Änderung aus dem Staging Area, in der Working Copy ist sie aber noch vorhanden
- Arbeitskopie: git checkout <datei>
  - Wann: Änderungen in der Arbeitskopie die noch nicht im Staging Area oder im Repository sind
  - Stellt den Stand nach dem letzten Commit wieder her

Aufgabe

- Schreibe eine Erklärung zu einem der folgenden Befehle: add, commit, revert, reset, checkout, log, status oder diff

Übung

Vorbereitung: Konfiguriere deinen Git Client

```
git config --global user.name "Dein Name"
git config --global user.email deine@emailadresse.de
```

1. Lege ein neues Repository an
2. Lege die Datei `README.md` an und füge sie zum Repository hinzu
3. Nutze `git status`, `git diff` und `git log` um Zwischenstände und Ergebnis zu prüfen
4. Ergänze Zeilen in README.md und experimentiere mit `git revert`, `git reset` und `git checkout`
5. Betrachte dein Repository mit gitk