

Programmieren II

TUTORIUM 1



Organisation

- Termin
 - Freitags 10:00 – 12:00 Uhr
- Es gibt keine Anwesenheitspflicht im Tutorium
- **Achtung:** Für die Termine zum Vorstellen der Übungsblätter herrscht Anwesenheitspflicht. Wer unentschuldigt fehlt ist automatisch durch das Übungsblatt durchgefallen.
- Ankündigungen, Termine, Folien, etc. im **StudIP**
- Sprechstunden nach Vereinbarung

Was erwartet mich im Tutorium

- Praktische Ergänzungen zur Vorlesung
- Vorbereitung auf das Übungsblatt
- Möglichkeit Fragen zu klären
- Möglichkeit weitere Programmiererfahrungen zu sammeln
- Betreutes Programmieren
- Feedback

Programmieren II

GIT BASICS



Warum ein Versionsverwaltungssystem?

- Protokollierungen der Änderungen
 - Es kann jederzeit nachvollzogen werden, wer wann was geändert hat.
- Wiederherstellung von alten Ständen einzelner Dateien
 - Somit können versehentliche Änderungen jederzeit wieder rückgängig gemacht werden.
- Archivierung der einzelnen Stände eines Projektes
 - Dadurch ist es jederzeit möglich, auf alle Versionen zuzugreifen.
- Koordinierung des gemeinsamen Zugriffs von mehreren Entwicklern auf die Dateien.
- Gleichzeitige Entwicklung mehrerer Entwicklungszweige (engl. Branches) eines Projektes.

Git

- 2005 von Linus Torvalds für die Linux-Kernel-Entwicklung entwickelt
- Jeder Benutzer hat lokal ein eigenes vollständiges Repository mit Historie
- Verteilt heißt, dass man kein zentrales Repository braucht
- Kein Netzwerkzugriff notwendig
- git ist sehr schnell und speicherzeffizient, alle Operationen sind lokal
- Viele Open-Source Projekte verwenden git:
 - Android, Debian, Eclipse, Fedora, GIMP, Gnome, openSUSE, Perl, Ruby on Rails, Samba, VLC, Wine, X.org

Git Installation

- Windows
 - <http://msysgit.github.io>
- Mac OS
 - git-osx-installer
 - <http://sourceforge.net/projects/git-osx-installer/>
 - Macports
 - Homebrew
- Linux
 - Installation über Paketmanager
 - Ubuntu: `sudo apt-get install git`

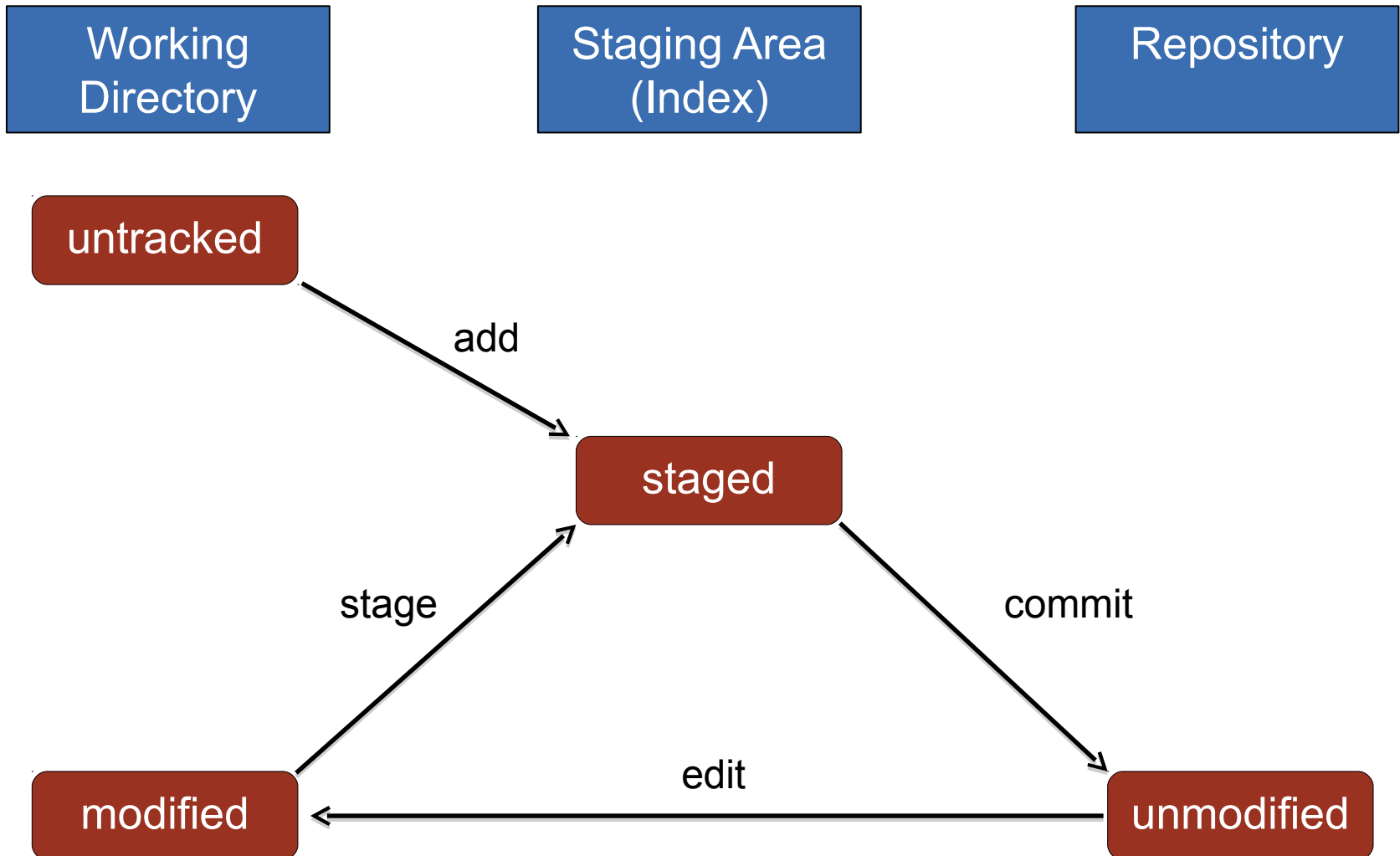
Git konfigurieren

- Man kann git konfigurieren
 - global für alle git-Projekte (mit --global)
 - lokal je git-Projekt (ohne --global)
- Benutzer einrichten (unter diesem Namen erscheinen die Commits)

```
$ git config --global user.name "Max Musteruser"  
$ git config --global user.email user@domain.de
```

- Weiter Konfigurationsmöglichkeiten
 - <http://git-scm.com/docs/git-config>

Git im Überblick



Git im Überblick: Working Directory

Working
Directory

- Beinhaltet Sourcecode & weitere Dateien
- Aktuelle Version der Dateien

Git im Überblick: Staging Area (Index)

Staging Area
(Index)

- Geänderte Dateien, die im nächsten Commit in das Repository übernommen werden sollen.
- Warum Staging-Area als Zwischenstufe?
 - Ein Commit sollte nur die Änderung einer Einheit (Feature, Bug, ...) enthalten, da es sonst unübersichtlich wird.
 - In der Realität arbeitet man aber oft parallel
 - Durch die Staging Area können Dateien getrennt voneinander commitet werden

Git im Überblick: Repository

Repository

- Historie über verschiedene Staging Areas
- Wird ein commit gemacht, so landet das aktuelle Staging Area als Snapshot der Dateien im Repository

Ein Repository anlegen

```
$ git init [/path/to/project/]
```

- Ein lokales Git Repository (.git Ordner) wird im Verzeichnis `/path/to/project/` bzw. im aktuellen Ordner erstellt

Ein Remote-Repository hinzufügen

```
$ git remote add Name URL
```

- **Name**
 - Bezeichnung für das entfernte Repository
- **URL**
 - Adresse des entfernten Git-Repository, z.B. über HTTP, SSH, ...
- Ein Remote-Repository wird mit dem angegebenen Namen im lokalen Repository hinterlegt.
- Es können mehrere Remote-Repositorys hinterlegt werden

Ein Repository klonen

```
$ git clone URL [/path/to/project]
```

- **URL**
 - Adresse des entfernten Git-Repository, z.B. über HTTP, SSH, ...
 - Kann auch ein anderes Lokales Repository sein
- Ein lokales Git Repository (.git Ordner) wird im Verzeichnis `/path/to/project/` als Kopie des Geklonten erstellt
- Geklontes Repository ist als Remote im lokalen Repository hinterlegt

Dateien adden / stagen

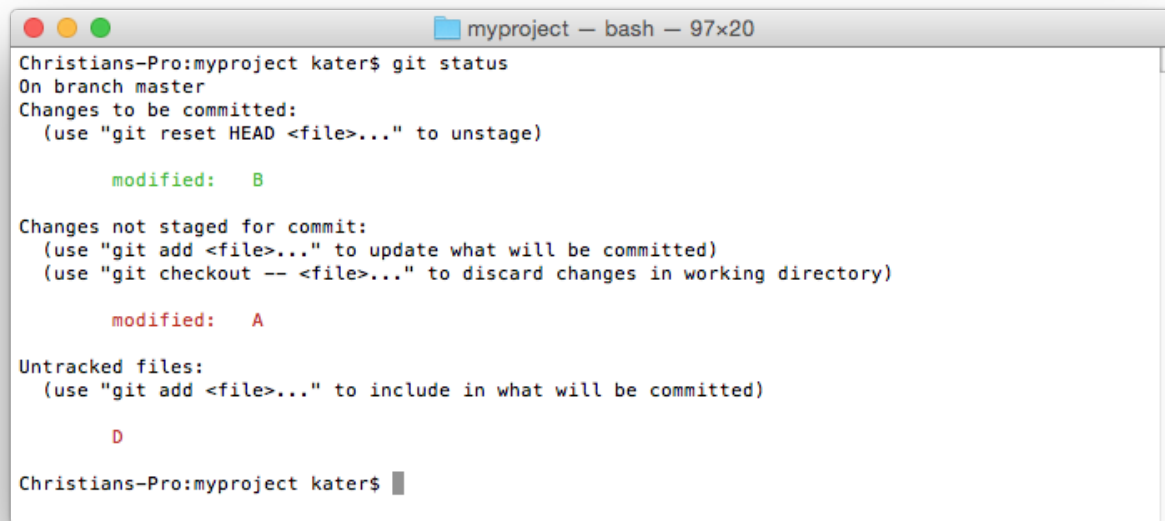
```
$ git add [file/directory...]
```

- Es können mehrere Ordner und Dateien durch Leerzeichen getrennt angegeben werden.
- Dateien in Ordnern werden rekursiv hinzugefügt.
- Noch nicht getrackte Dateien und Ordner sind nach dem add getrackted
- Dateien und Ordner sind nach dem add in der Staging Area

Status des Git-Repository anzeigen

```
$ git status
```

- Übersicht über modifizierte Dateien
- Getrennt nach gestaged/ungestage bzw. getracked/ungetracked



```
myproject — bash — 97x20
Christians-Pro:myproject kater$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:   B

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   A

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

    D

Christians-Pro:myproject kater$
```

Einen commit anlegen

```
$ git commit -m "Beschreibung".
```

- Die Beschreibung sollte nach Möglichkeiten kurz und knapp die Änderungen beschreiben.

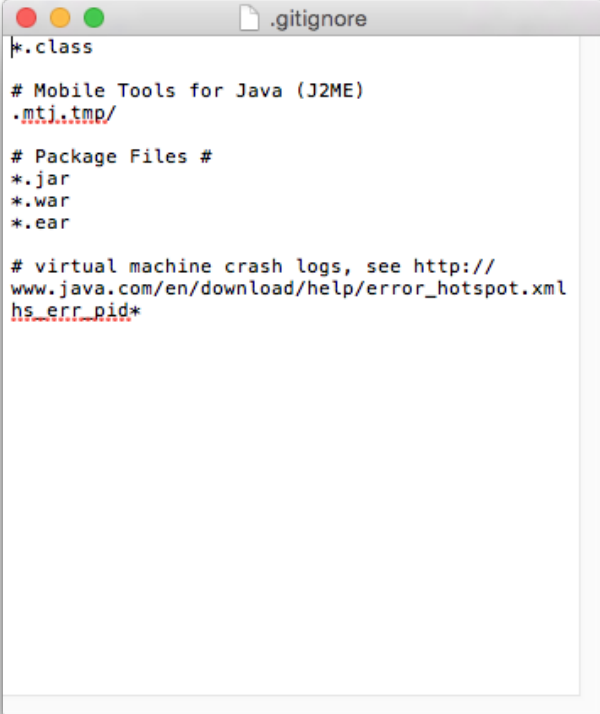
Commits zum Remote Repository übertragen

```
$ git push
```

- Die am Server noch nicht vorhandenen Commits werden übertragen.
- Konflikte möglich:
 - Z.B. ein anderer Entwickler hat an der gleichen Stelle etwas geändert. Welche Fassung ist gültig?
 - Mehr dazu in den nächsten Tutorien

.gitignore

- In der .gitignore werden zeilenweise die Dateien angegeben, die vom Repository ignoriert werden sollen
- Halt das Repository frei von kompilierten Dateien und Rechnerspezifischen Konfigurationsdateien
- Hilfreiche Links
 - <https://www.gitignore.io>
 - <http://git-scm.com/docs/gitignore>

A screenshot of a code editor window titled ".gitignore". The window contains the following text:

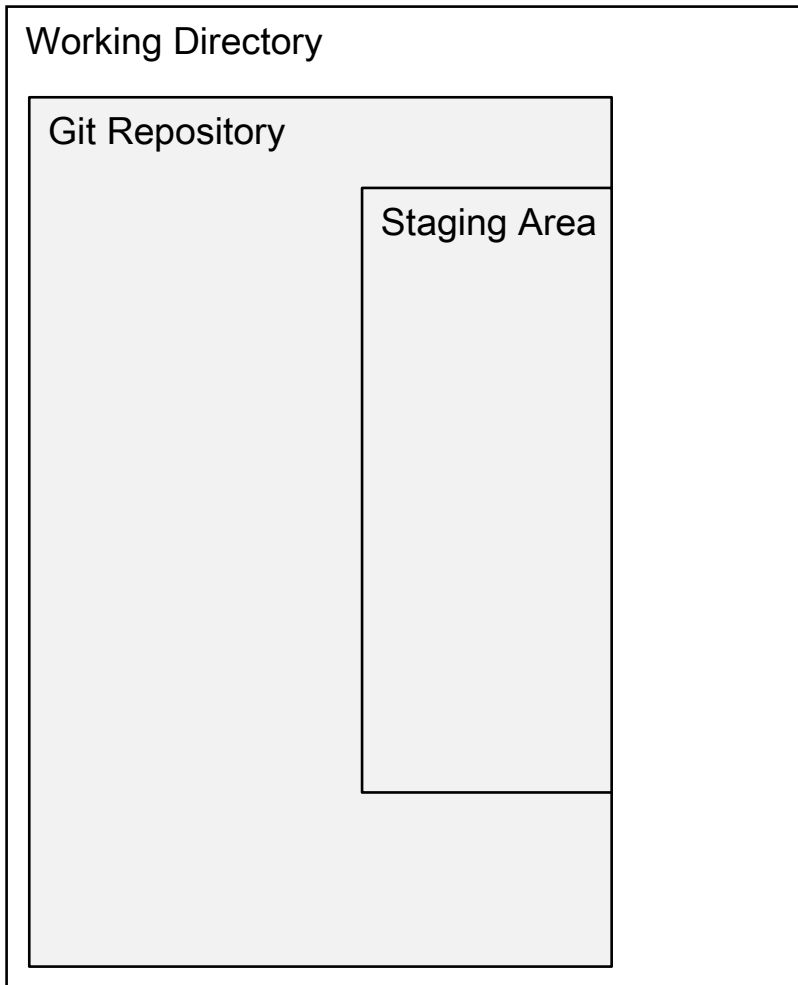
```
*.class  
  
# Mobile Tools for Java (J2ME)  
*.mtj.tmp/  
  
# Package Files #  
*.jar  
*.war  
*.ear  
  
# virtual machine crash logs, see http://  
www.java.com/en/download/help/error_hotspot.xml  
hs_err_pid*
```

Beispiel: Ein Repository anlegen



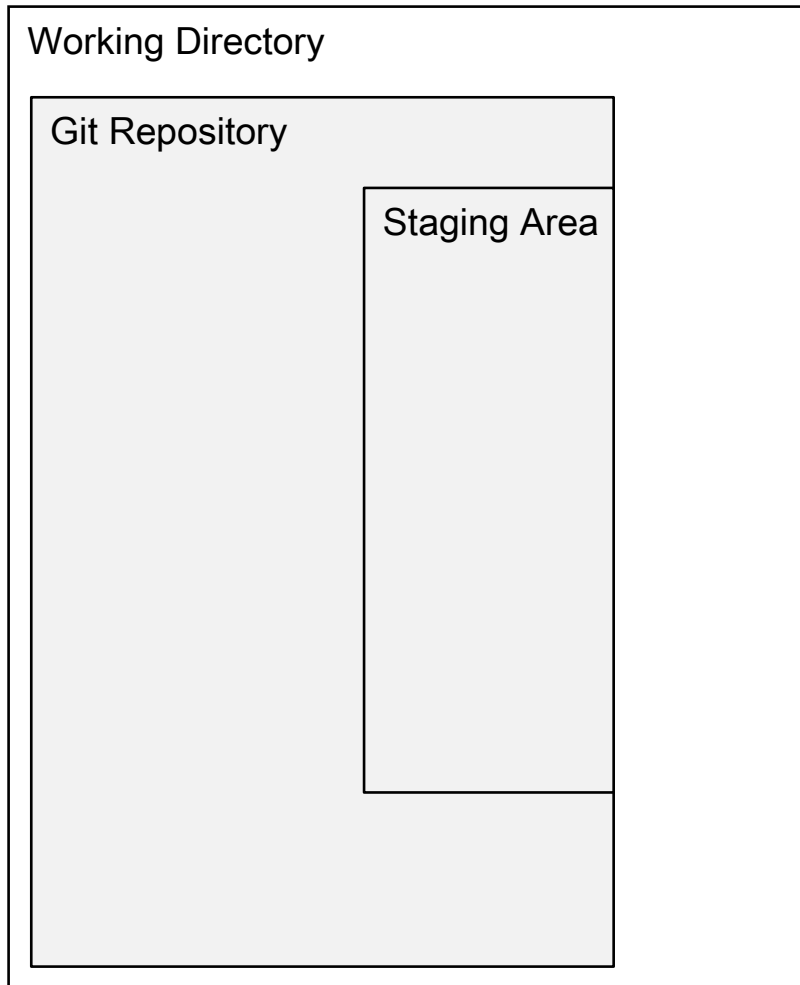
```
$ git init ~/myproject/ && cd ~/myproject/
```

Beispiel: Ein Repository anlegen



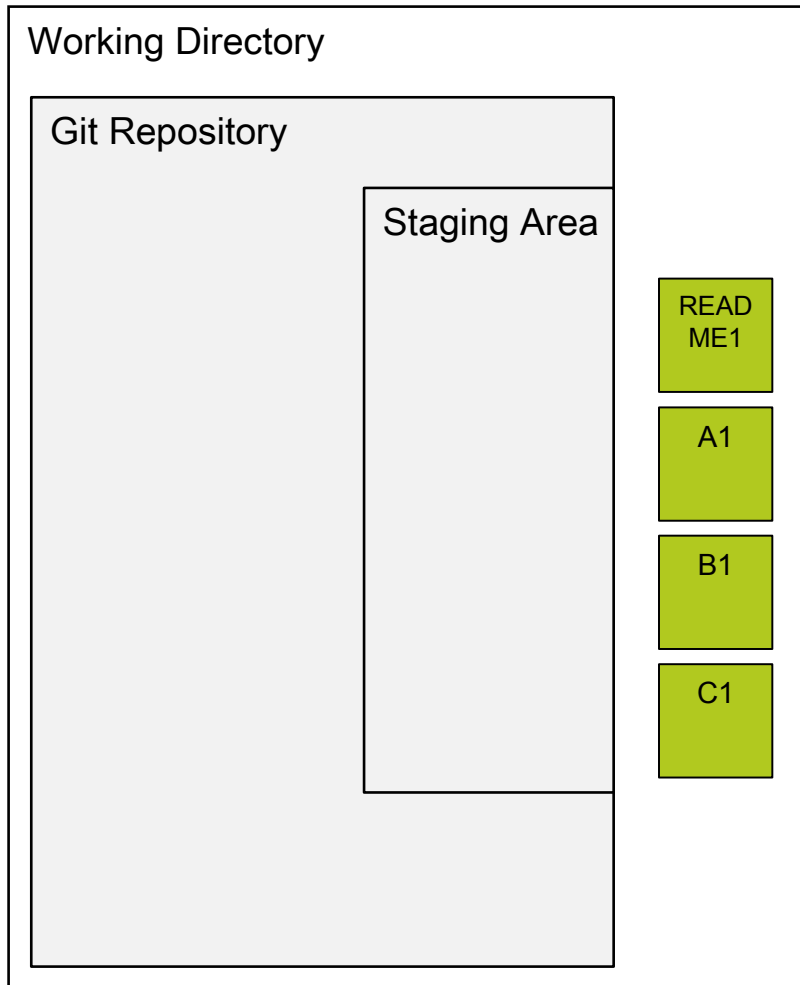
```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$
```

Beispiel: Beispieldateien erstellen



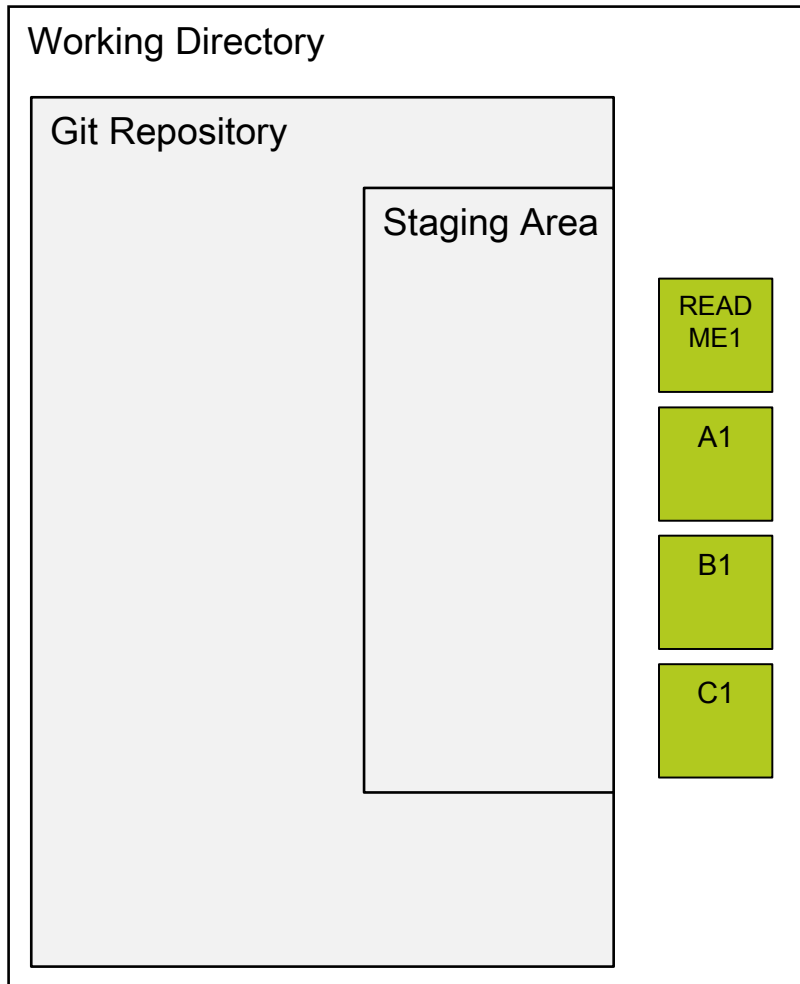
```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
```

Beispiel: Beispieldateien erstellen



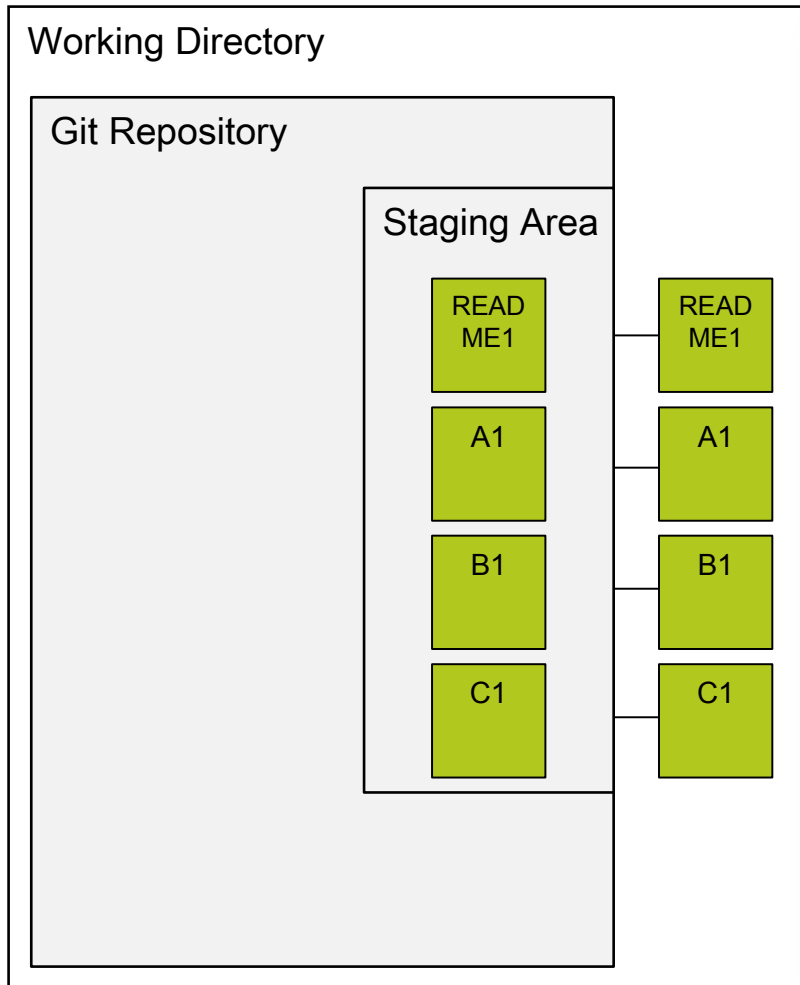
```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$
```


Beispiel: Beispieldateien stagen & tracken



```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$ git add .
```

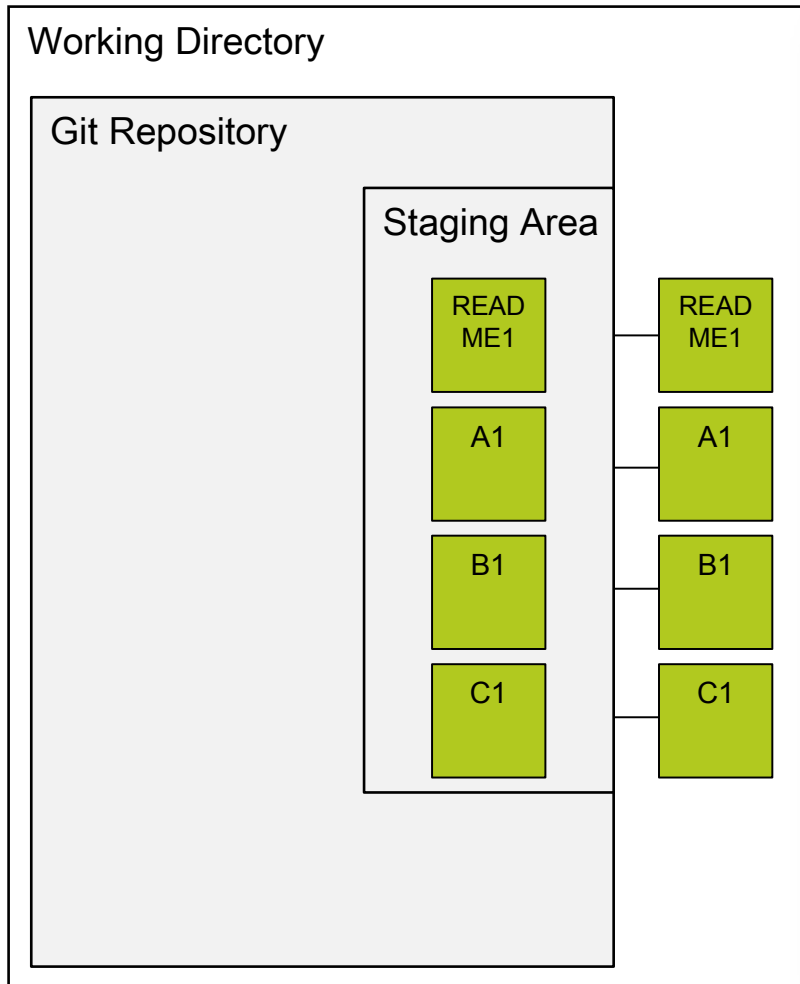
Beispiel: Beispieldateien stagen & tracken



```

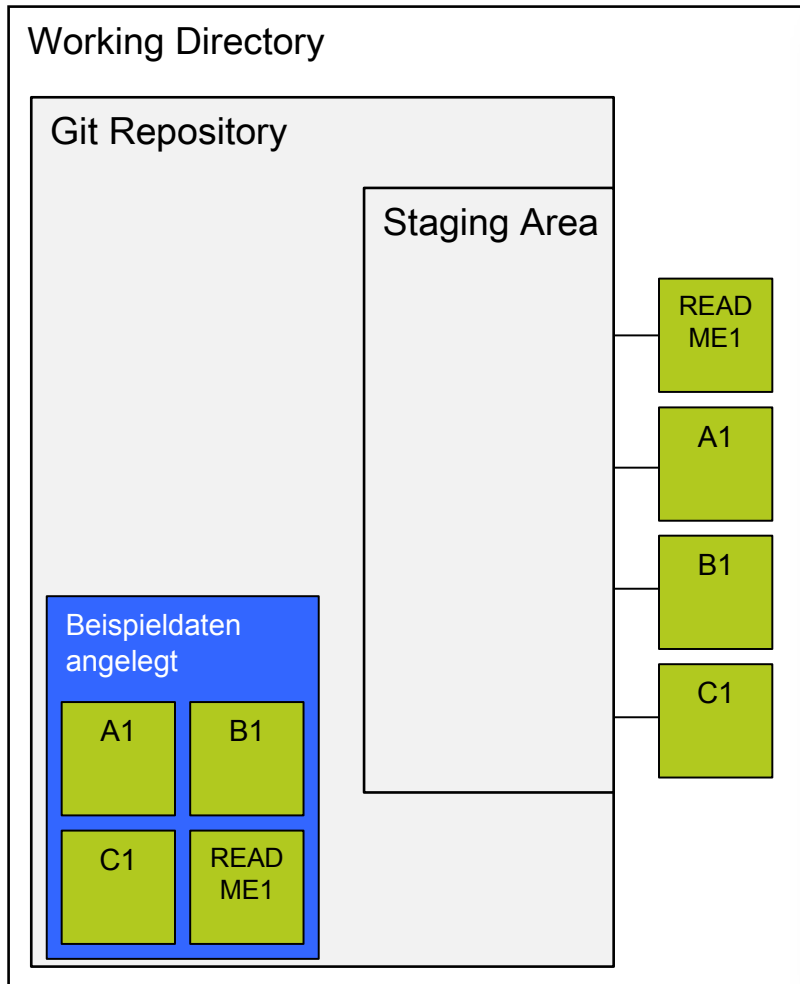
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$ git add .
$
  
```

Beispiel: Beispieldateien commiten



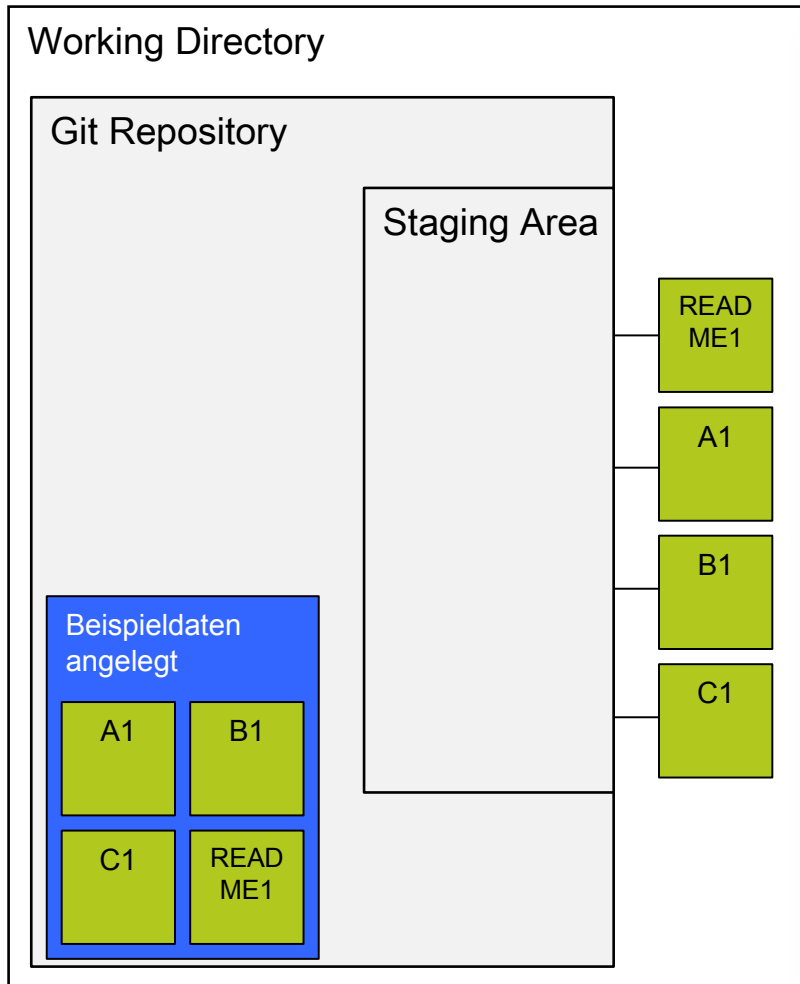
```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$ git add .
$ git commit -m "Beispieldaten angelegt"
```

Beispiel: Beispieldateien commiten



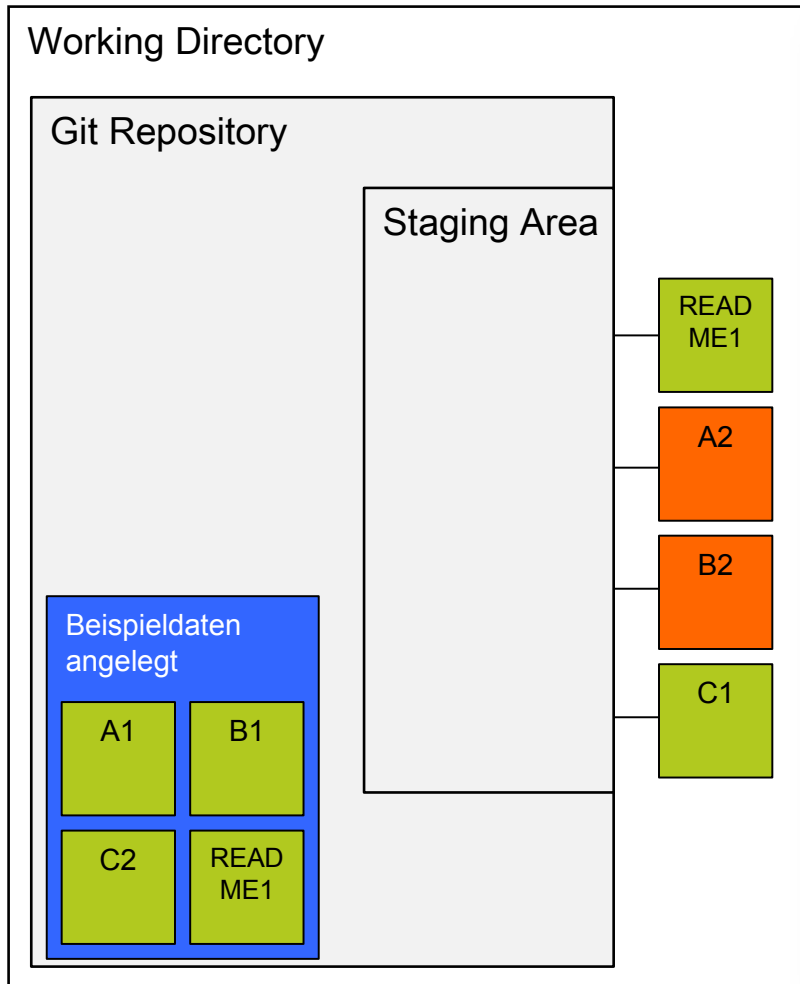
```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$ git add .
$ git commit -m "Beispieldaten angelegt"
[master (root-commit) b6db29e] Beispieldaten
angelegt
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 A
create mode 100644 B
create mode 100644 C
create mode 100644 README,
$
```

Beispiel: Beispieldateien ändern



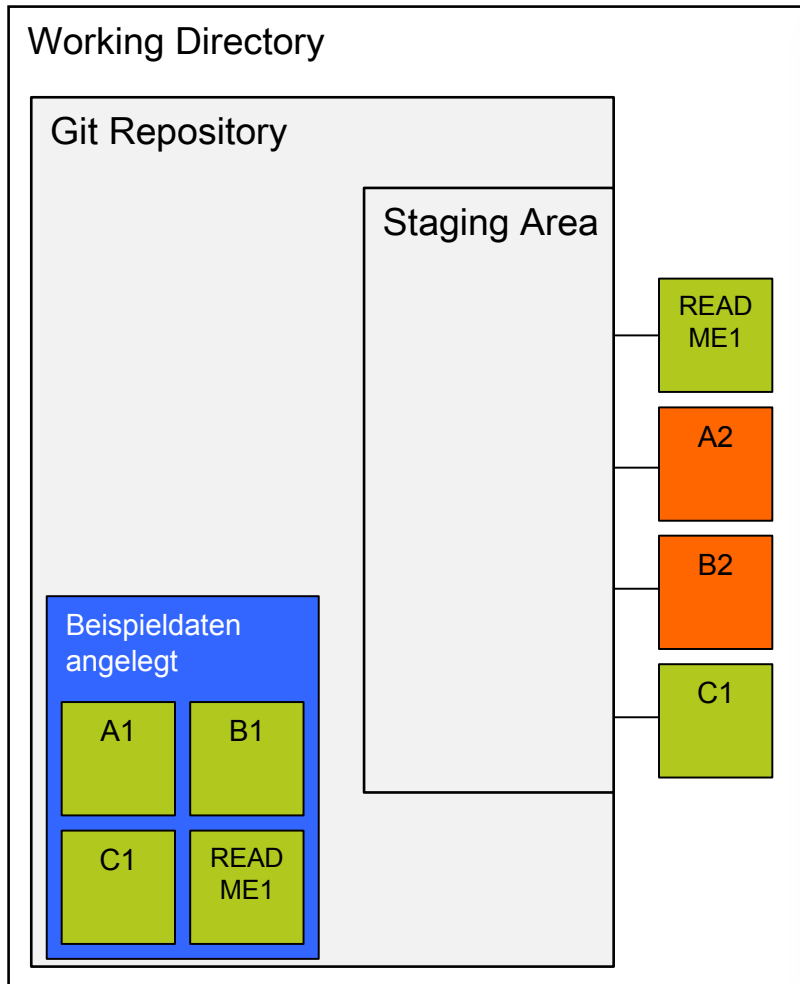
```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$ git add .
$ git commit -m "Beispieldaten angelegt"
[master (root-commit) b6db29e] Beispieldaten
angelegt
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 A
create mode 100644 B
create mode 100644 C
create mode 100644 README,
$ echo "Beispiel A" >> A && echo "Beispiel B" >>
B
```

Beispiel: Beispieldateien ändern



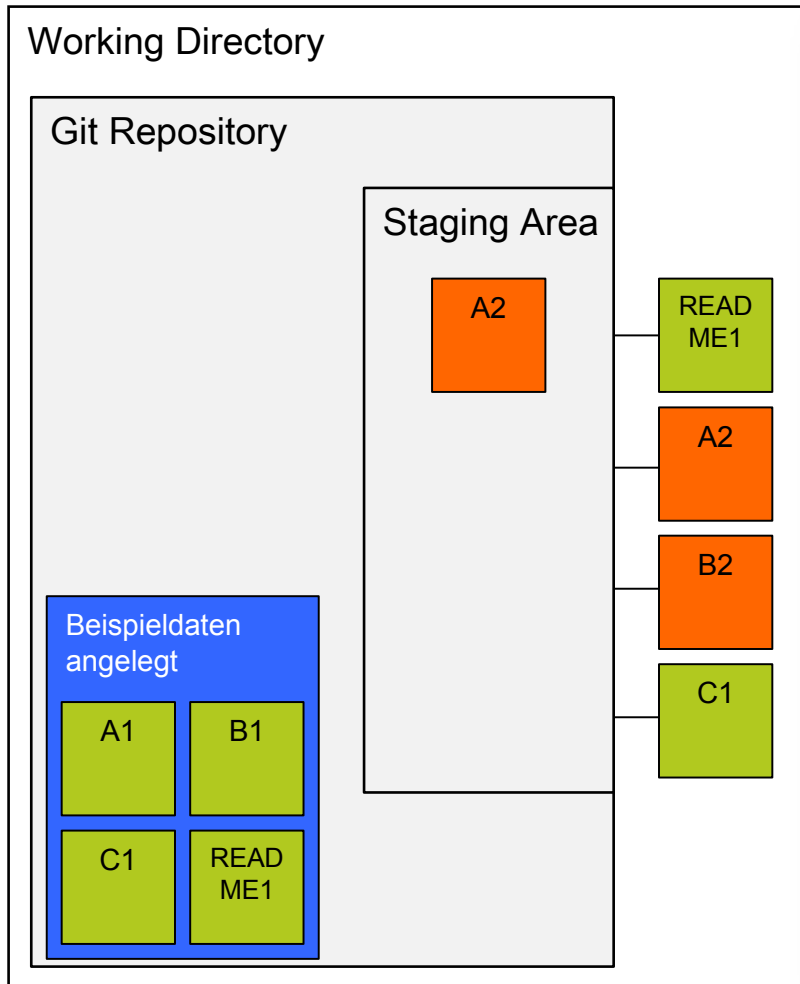
```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$ git add .
$ git commit -m "Beispieldaten angelegt"
[master (root-commit) b6db29e] Beispieldaten
angelegt
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 A
create mode 100644 B
create mode 100644 C
create mode 100644 README,
$ echo "Beispiel A" >> A && echo "Beispiel B" >>
B
$
```

Beispiel: Änderung stagen



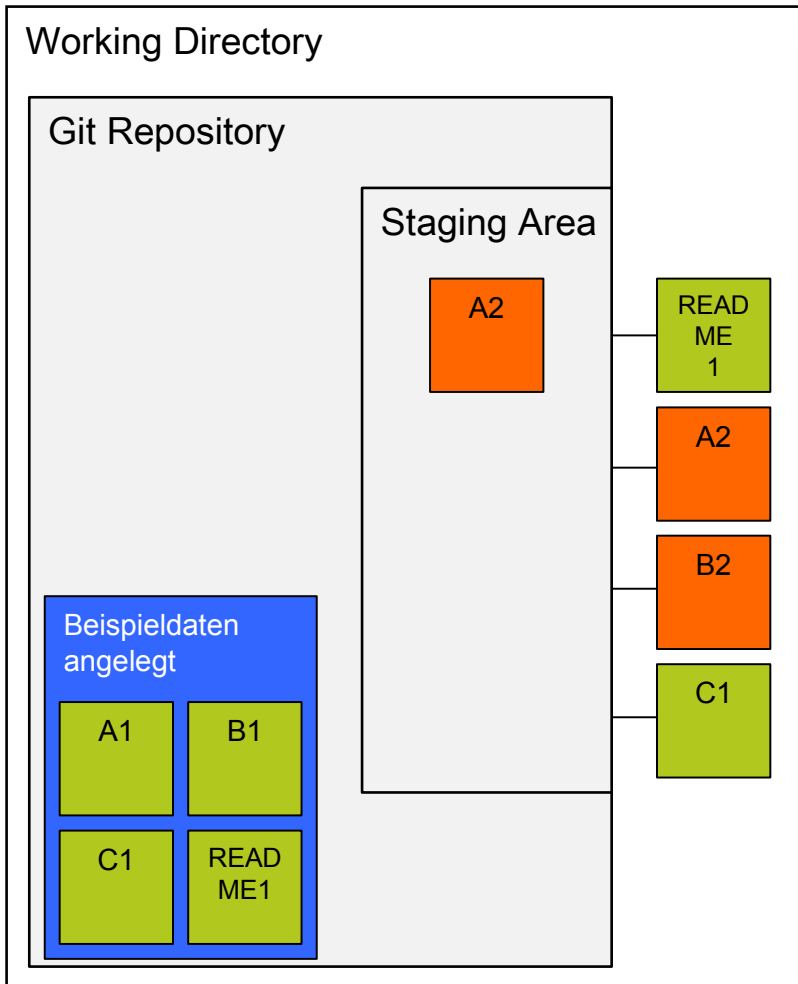
```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$ git add .
$ git commit -m "Beispieldaten angelegt"
[master (root-commit) b6db29e] Beispieldaten
angelegt
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 A
create mode 100644 B
create mode 100644 C
create mode 100644 README,
$ echo "Beispiel A" >> A && echo "Beispiel B" >>
B
$ git add A
```

Beispiel: Änderung stagen



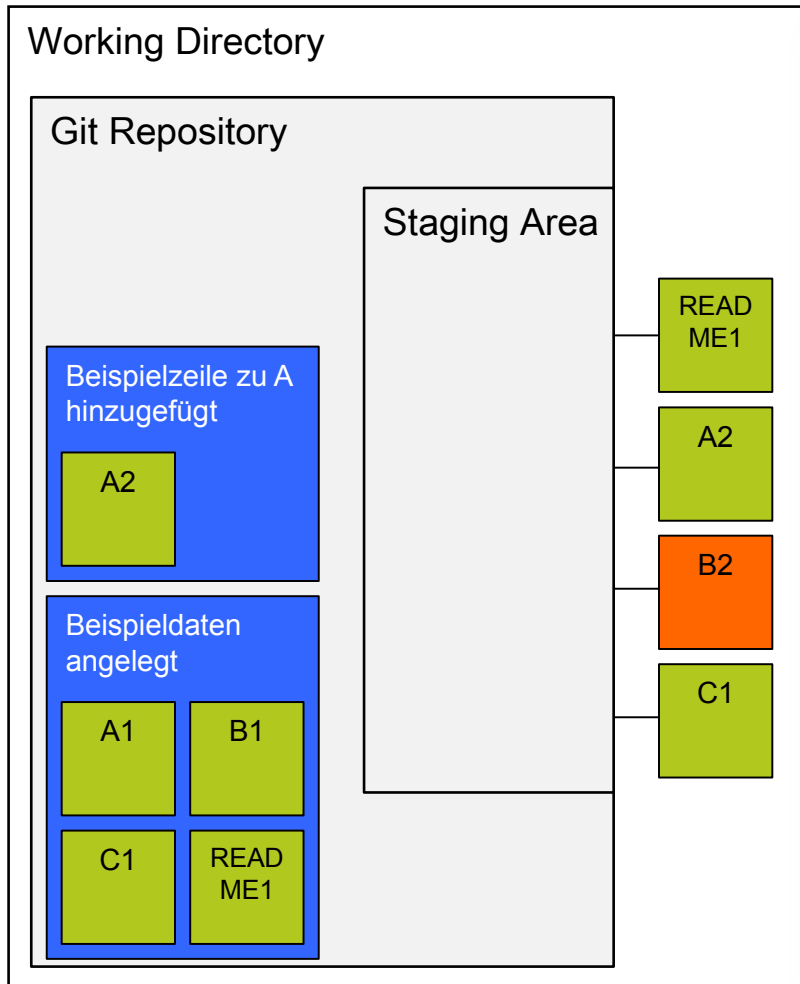
```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$ git add .
$ git commit -m "Beispieldaten angelegt"
[master (root-commit) b6db29e] Beispieldaten
angelegt
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 A
create mode 100644 B
create mode 100644 C
create mode 100644 README,
$ echo "Beispiel A" >> A && echo "Beispiel B" >>
B
$ git add A
$
```


Beispiel: Änderung commiten



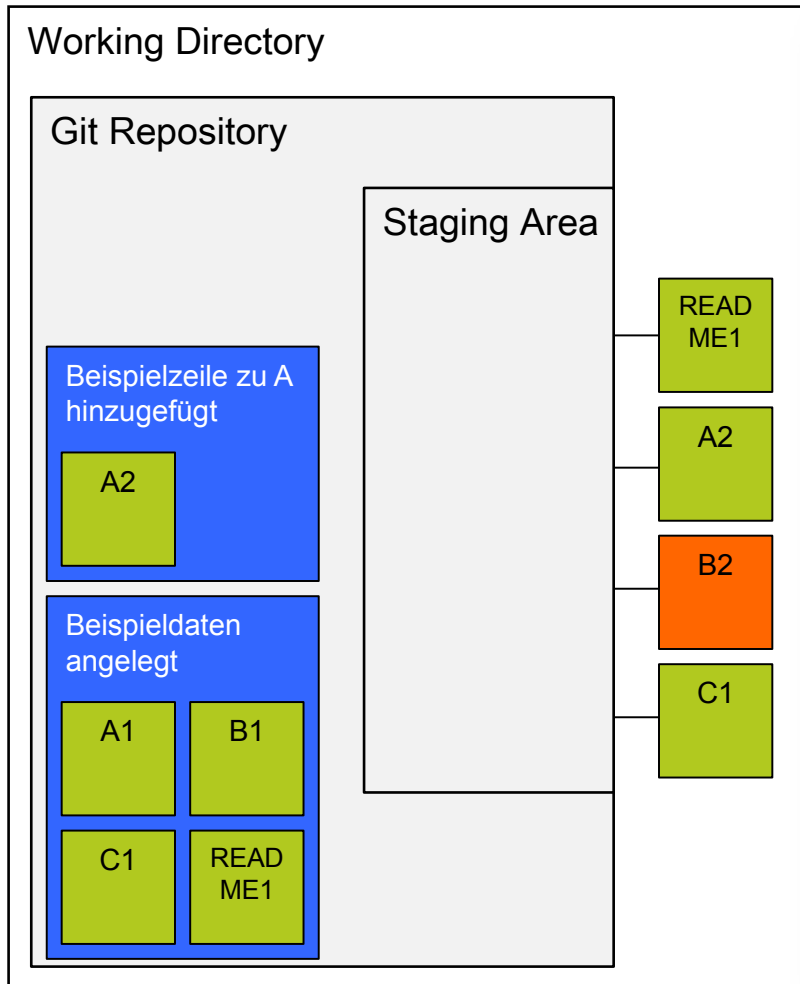
```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$ git add .
$ git commit -m "Beispieldaten angelegt"
[master (root-commit) b6db29e] Beispieldaten
angelegt
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 A
create mode 100644 B
create mode 100644 C
create mode 100644 README,
$ echo "Beispiel A" >> A && echo "Beispiel B" >>
B
$ git add A
$ git commit -m "Beispielzeile zu A hinzugefügt"
```

Beispiel: Änderung commiten



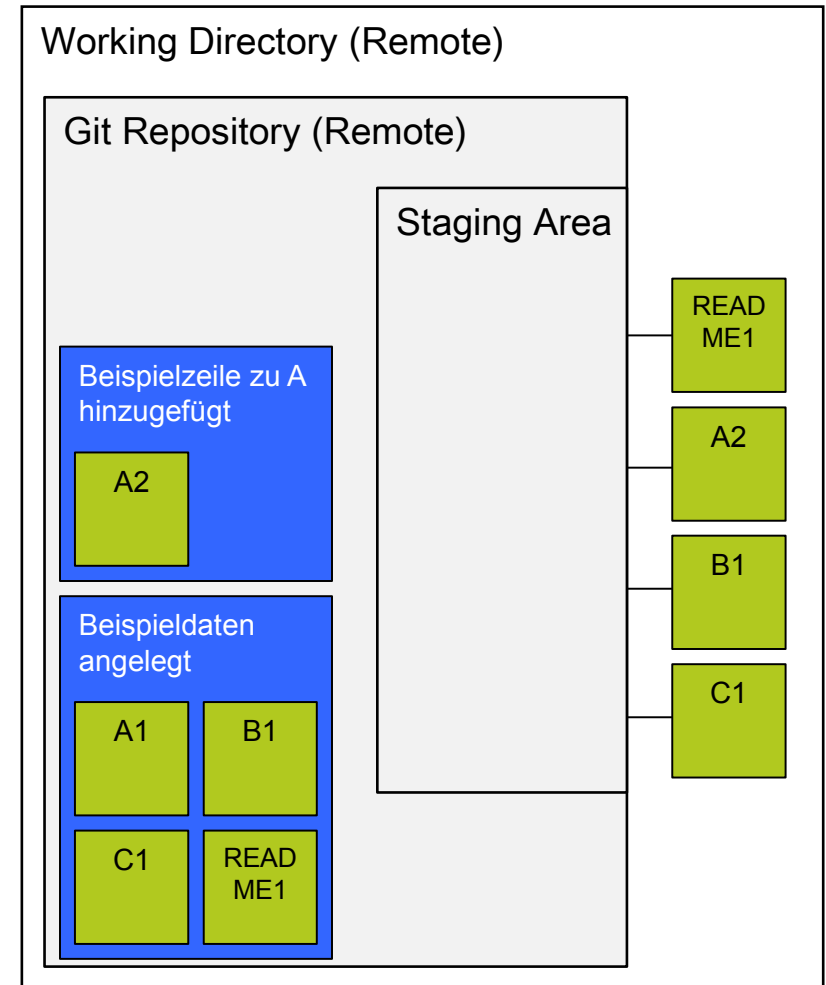
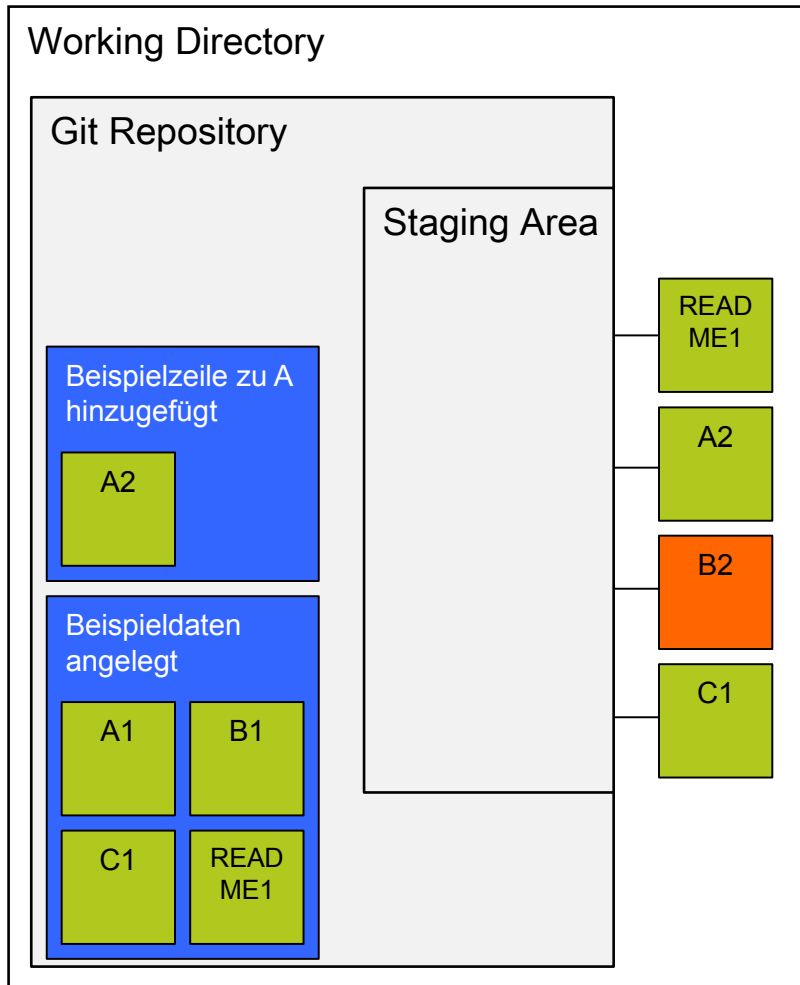
```
$ touch README A B C
$ git add .
$ git commit -m "Beispieldaten angelegt"
[master (root-commit) b6db29e] Beispieldaten
angelegt
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 A
create mode 100644 B
create mode 100644 C
create mode 100644 README,
$ echo "Beispiel A" >> A && echo "Beispiel B" >>
B
$ git add A
$ git commit -m "Beispielzeile zu A hinzugefügt"
[master eb869d] Beispielzeile zu A und B
hinzugefügt
1 file changed, 1 insertion(+)
$
```

Beispiel: Änderung ans Remote übertragen

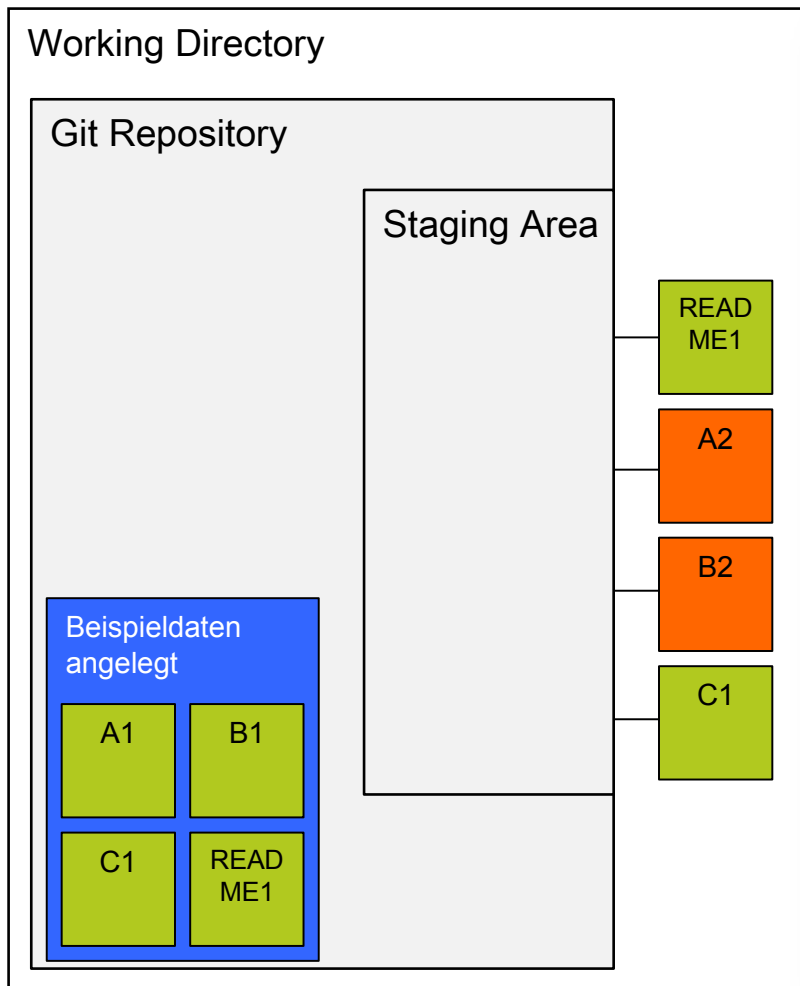


```
$ git commit -m "Beispieldaten angelegt"
[master (root-commit) b6db29e] Beispieldaten
angelegt
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 A
create mode 100644 B
create mode 100644 C
create mode 100644 README,
$ echo "Beispiel A" >> A && echo "Beispiel B" >>
B
$ git add A
$ git commit -m "Beispielzeile zu A hinzugefügt"
[master eb869d] Beispielzeile zu A und B
hinzugefügt
1 file changed, 1 insertion(+)
$ git remote add origin https://pm.stud.sim.uni-
hannover.de/test
$ git push
```

Beispiel: Änderung ans Remote übertragen

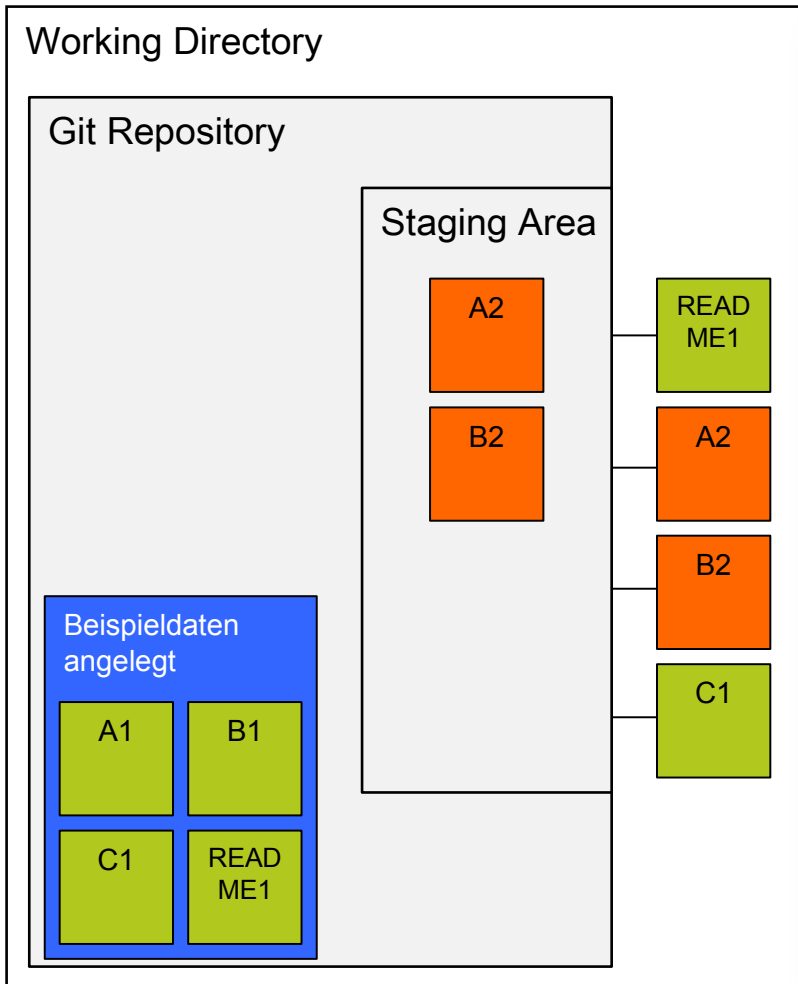


Alternative: Änderung stagen



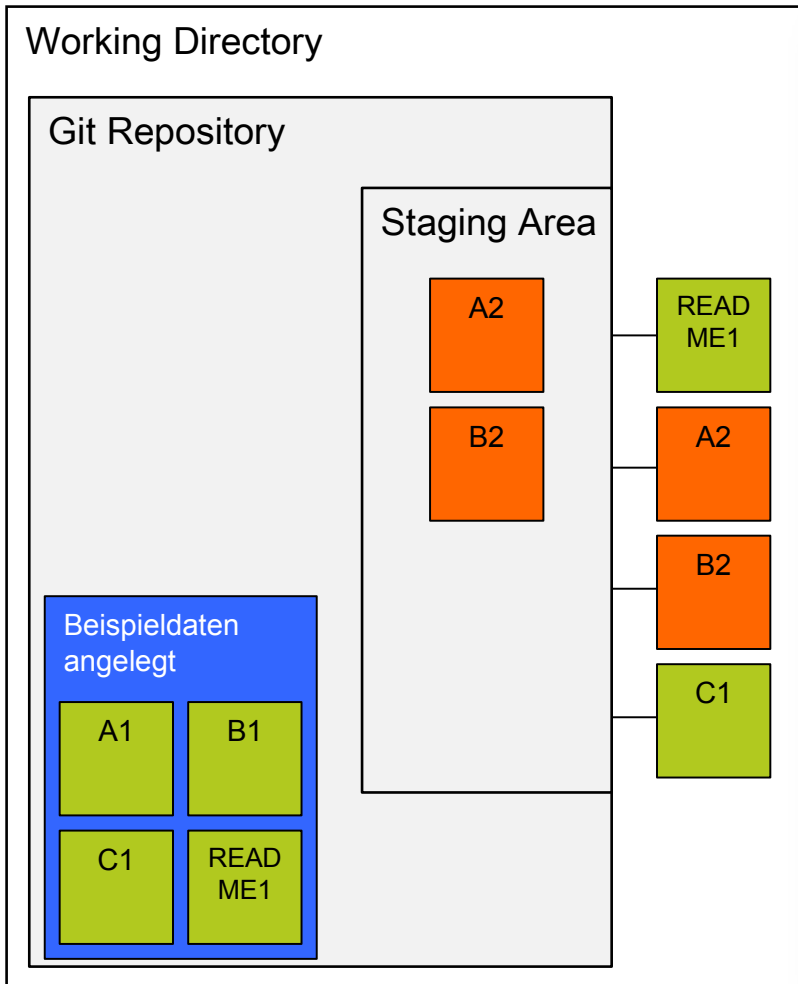
```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$ git add .
$ git commit -m "Beispieldaten angelegt"
[master (root-commit) b6db29e] Beispieldaten
angelegt
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 A
create mode 100644 B
create mode 100644 C
create mode 100644 README,
$ echo "Beispiel A" >> A && echo "Beispiel B" >>
B
$ git add .
```

Alternative: Änderung stagen



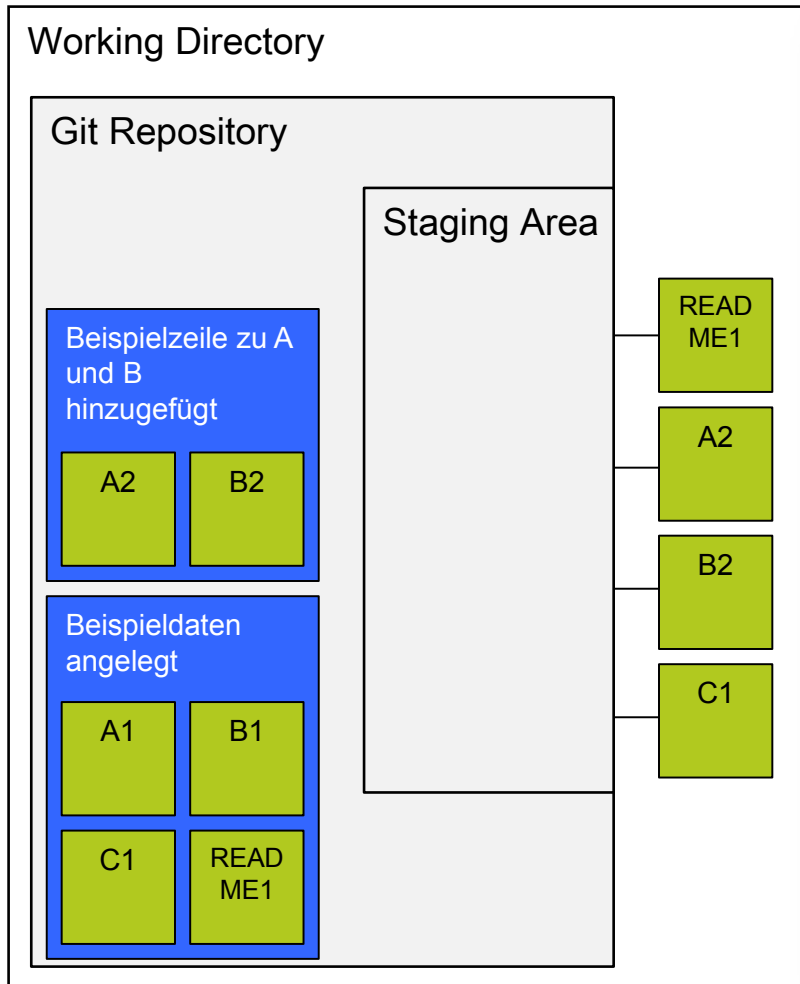
```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$ git add .
$ git commit -m "Beispieldaten angelegt"
[master (root-commit) b6db29e] Beispieldaten
angelegt
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 A
create mode 100644 B
create mode 100644 C
create mode 100644 README,
$ echo "Beispiel A" >> A && echo "Beispiel B" >>
B
$ git add .
$
```

Alternative: Änderung commiten



```
$ git init ~/myproject/ && cd ~/myproject/
Initialized empty Git repository in ~/myproject/.git/
$ touch README A B C
$ git add .
$ git commit -m "Beispieldaten angelegt"
[master (root-commit) b6db29e] Beispieldaten
angelegt
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 A
create mode 100644 B
create mode 100644 C
create mode 100644 README,
$ echo "Beispiel A" >> A && echo "Beispiel B" >>
B
$ git add .
$ git commit -m "Beispielzeile zu A und B
hinzugefügt"
```

Alternative: Änderung commiten



```
$ touch README A B C
$ git add .
$ git commit -m "Beispieldaten angelegt"
[master (root-commit) b6db29e] Beispieldaten
angelegt
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 A
create mode 100644 B
create mode 100644 C
create mode 100644 README,
$ echo "Beispiel A" >> A && echo "Beispiel B" >>
B
$ git add .
$ git commit -m "Beispielzeile zu A und B
hinzugefügt"
[master ccd6853] Beispielzeile zu A und B
hinzugefügt
2 files changed, 2 insertions(+)
$
```


Hinweis zur Abgabe

- Projektmanagementsystem:
- <https://pm.stud.sim.uni-hannover.de>
- Commits erscheinen auch in der Weboberfläche
- Nur wenn eure Commits hier erscheinen, gelten Sie als abgegeben.

pm.stud.sim.uni-hannover.de

Hauptseite Meine Seite Projekte Administration Hilfe

Angemeldet als kater Mein Konto Abmelden

Java 2014 » Gruppe 27

Suche: Gruppe 27

Übersicht Aktivität Tickets Neues Ticket Gantt-Diagramm Kalender News Dokumente Wiki Dateien Projektarchiv

Konfiguration

gruppe-27 @ master

Statistiken | Zweig: master | Revision:

Name	Größe	Revision	Geändert vor	Autor	Kommentar
Libraries		2b0ee56c	9 Monaten		Imported xstream Library. Created IO class for L...
nbproject		d6958204	9 Monaten		Added TSUserUI. Currently working on it
src		4d824b0d	9 Monaten		Final commit. Software now ready for Abgabe.
.gitignore	20 Bytes	2b0ee56c	9 Monaten		Imported xstream Library. Created IO class for L...
build.xml	3,465 KB	c8d32cab	9 Monaten		Adding missing kryonet library files.
description.txt	0 Bytes	837dd724	etwa 2 Jahren		Initial
manifest.mf	82 Bytes	c8d32cab	9 Monaten		Adding missing kryonet library files.

Aktuellste Revisionen

#	Datum	Autor	Kommentar
4d824b0d	13.07.2014 18:49	2b0ee56c	Final commit. Software now ready for Abgabe.
3057835e	13.07.2014 05:47	2b0ee56c	Fixing nearly . Soon ready to be submitted
486a67ee	12.07.2014 00:42	2b0ee56c	BUG IS FIXED
24087488	11.07.2014 15:07	2b0ee56c	Answers tickets in answeredTicketNode are now shown. BUG: ticketmessage shown instead of responsemessage.
5e873aa6	11.07.2014 09:14	2b0ee56c	Kopf kaputt. going to sleep
0a64958d	11.07.2014 08:18	2b0ee56c	working on lalala
2c96b451	11.07.2014 05:59	2b0ee56c	Fixing FCKG bugfix + implementing cool stuff^tm
bb323af3	11.07.2014 02:49	2b0ee56c	bugfixes
d6958204	11.07.2014 01:55	2b0ee56c	Added TSUserUI. Currently working on it
851958ef	10.07.2014 20:02	2b0ee56c	nothing special

Unterschiede anzeigen

Alle Revisionen anzeigen | Revisionen anzeigen

Auch abrufbar als: Atom

Powered by Redmine © 2006-2013 Jean-Philippe Lang