

Git auf der Konsole

Author: Jan Franz Kammellander

Login: jkammellander

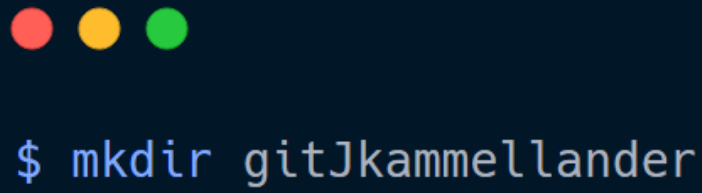
Jahrgang: 3BHIT

Erstellt am: 12.01.2021

Git auf der Konsole

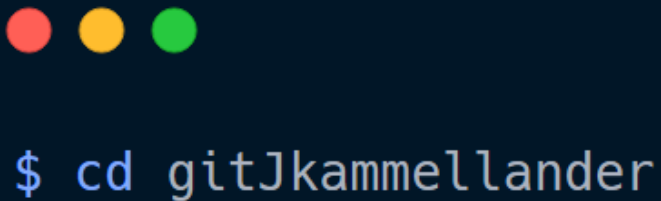
- Ordner des Repositorys erstellen
- Git Repository erstellen
- Text erstellen
- Status Überprüfen
- Datei zum Staging Bereich hinzufügen
- Ziffer der Datei ändern
- Erneut zum Staging Bereich hinzufügen
- Log-Eintrag
- Jahrgang.info
- Ausschließen der Versionierung
- Änderungen dem Repository Hinzufügen
- Unterordner erstellen
 - Unterordner dem Repository hinzufügen
- Dem Unterordner Dateien hinzufügen
 - Datei verschieben
- Dateien im Unterordner in Git stagen
- Neue Version
- Das lokale repository als remote-repository hinzufügen

Ordner des Repositorys erstellen



```
$ mkdir gitJkammellander
```

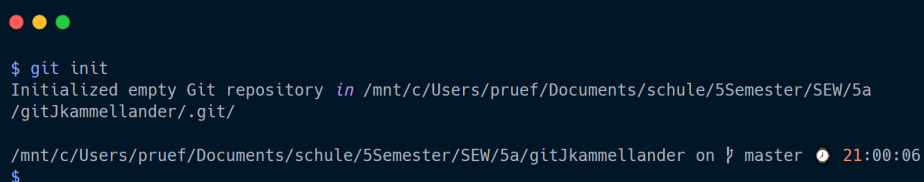
Mit dem Befehl `mkdir` wird ein Ordner namens `gitJkammellander` erstellt, welcher der Ordner für mein Git Repository wird.



```
$ cd gitJkammellander
```

Mit dem Befehl `cd` navigiere ich in den Ordner.

Git Repository erstellen

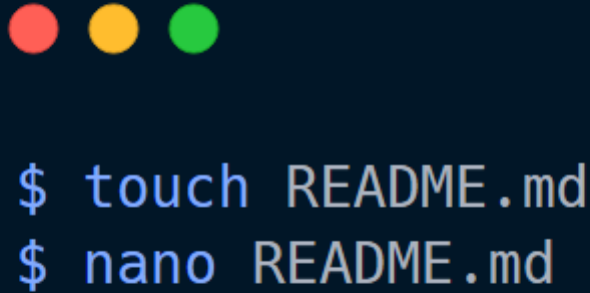


```
$ git init
Initialized empty Git repository in /mnt/c/Users/pruef/Documents/schule/5Semester/SEW/5a/gitJkammellander/.git/

/mnt/c/Users/pruef/Documents/schule/5Semester/SEW/5a/gitJkammellander on ? master ● 21:00:06
$
```

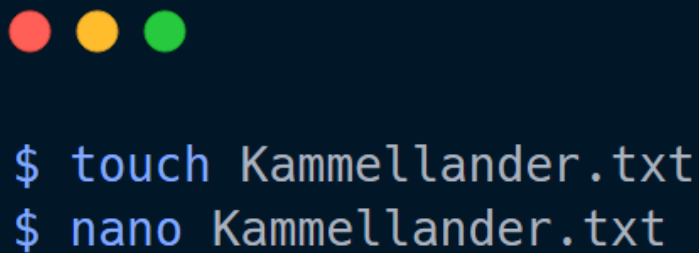
Mit dem Befehl `git init` wird ein lokales Git Repository zur Versionierung erstellt.

Text erstellen



```
$ touch README.md  
$ nano README.md
```

Mit dem Befehl `touch` erstelle ich eine README.md datei und bearbeite es mit dem Texteditor **Nano**.



```
$ touch Kammellander.txt  
$ nano Kammellander.txt
```

Danach erstellte ich eine Textdatei namens **Kammellander.txt** und sfügte dieser den Text "*Kammellander1*" hinzu.

Status Überprüfen

```

$ git status
On branch master

No commits yet

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

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

Mit dem Befehl `git status` kann ich den Status meines Staging Bereiches überprüfen und es ist zu erkennen, dass noch keine Version oder Elemente im Staging-Bereich vorhanden ist.

Datei zum Staging Bereich hinzufügen

```

$ git add README.md
$ git add Kammellander.txt
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   Kammellander.txt
    new file:   README.md
```

Mit dem Befehl `git add` wird eine Datei dem Staging-Bereich hinzugefügt. In meinem Fall habe ich die Datei **README.md** und **Kammellander.txt** diesem hinzugefügt. Anschließend habe ich mit `git status` überprüft, ob die Dateien im Staging-Bereich vorhanden sind; sie sind vorhanden.

```
$ git diff --staged
diff --git a/Kammellander.txt b/Kammellander.txt
new file mode 100644
index 0000000..8787227
--- /dev/null
+++ b/Kammellander.txt
@@ -0,0 +1 @@
+Kammellander1
diff --git a/README.md b/README.md
new file mode 100644
index 0000000..febabdf
--- /dev/null
+++ b/README.md
@@ -0,0 +1 @@
+First Commit.

$ git commit -m "NEW: First Commit"
[master (root-commit) 9e4c161] NEW: First Commit
 2 files changed, 2 insertions(+)
 create mode 100644 Kammellander.txt
 create mode 100644 README.md
```

Danach habe ich mit dem Befehl `git diff --staged` überprüft, ob bereits vorhandene Elemente überschrieben wurden und welche Zeile verändert wurde; es wurde nichts verändert, nur etwas Neues hinzugefügt.

Anschließend habe ich mit dem Befehl `git commit` meine erste Version bereitgestellt. Die Flag `-m` benötige, sodass ich danach mit Anführungszeichen `"NEW: First Commit"` meine Commit-Nachricht hinschreiben konnte, ohne dass ich diese in einem Texteditor vorher schreiben und speichern muss.

Ziffer der Datei ändern

```
$ nano Kammellander.txt

$ git status
On branch master
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:   Kammellander.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

Nun habe ich die Datei **Kammellander.txt** geändert indem ich eine **1** mit einer **2** ausgetauscht habe. Danach habe ich mit `git status` überprüft, ob die Datei modifiziert wurde; sie wurde.

Erneut zum Stagin Bereich hinzufügen

```
$ git add Kammellander.txt
warning: LF will be replaced by CRLF in Kammellander.txt.
The file will have its original line endings in your working directory
```

Nun füge ich meine veränderte Datei **Kammellander.txt** dem Staging-Bereich hinzu.

```
$ git diff --staged
diff --git a/Kammellander.txt b/Kammellander.txt
index 8787227..b17936e 100644
--- a/Kammellander.txt
+++ b/Kammellander.txt
@@ -1,1 @@
-Kammellander1
+Kammellander2

$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   Kammellander.txt
```

Als nächstes überprüfe ich mit dem Befehl `diff`, ob die Ziffer richtig modifiziert wurde; sie wurde. Anschließend habe ich noch den Staging-Bereich überprüft, ob die Datei modifiziert wurde; sie wurde verändert.

Log-Eintrag

```
$ git log
commit 9e4c161200272e2e3bc7412558a2cd7b9a5118c7 (HEAD -> master)
Author: Jan Kammellander <janschwiderik@gmail.com>
Date:   Mon Jan 11 21:23:57 2021 +0100

    NEW: First Commit
```

Mit dem Befehl `git log` schaue ich nach, ob meine erste Version vorhanden ist. Wie es im Bild zu sehen ist, wurde die Version von mir bereitgestellt und zwar am Montag dem 11. Jänner.

Jahrgang.info

```
$ touch 3BHIT.info
$ nano 3BHIT.info

$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   Kammellander.txt

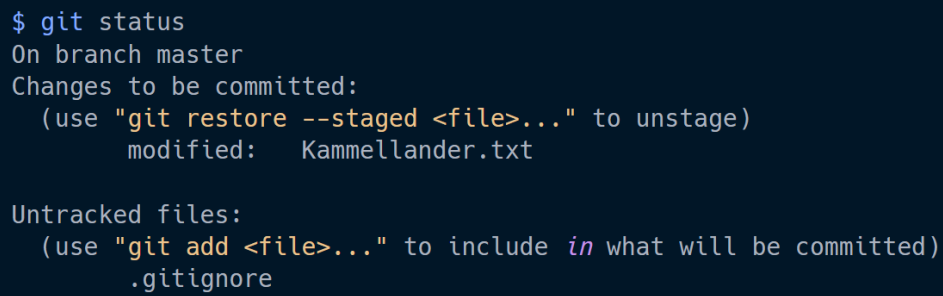
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    3BHIT.info
```

Nun erstelle ich eine Datei Namens **3BHIT.info**, welche meine Klasse als Text ("3BHIT") enthält. Anschließend habe ich mit `status` überprüft, ob die Datei nicht von git getrackt wird und aus der Version ausgeschlossen ist.

Ausschließen der Versionierung

```
$ touch .gitignore
$ nano .gitignore
```

Dass zukünftig und im hier und jetzt alle Dateien welche auf ".info" enden von meinen und den Versionen anderer Contributor ist, erstelle ich ein sogenanntes **.gitignore-file**. Dieses beinhaltet Dateien, Ordner und Dateiendungen, welche aus den Versionen vorenthalten beliben soll und von git ignoriert werden sollen. Deswegen habe ich der Datei folgendes hinzugefügt: `*.info`.

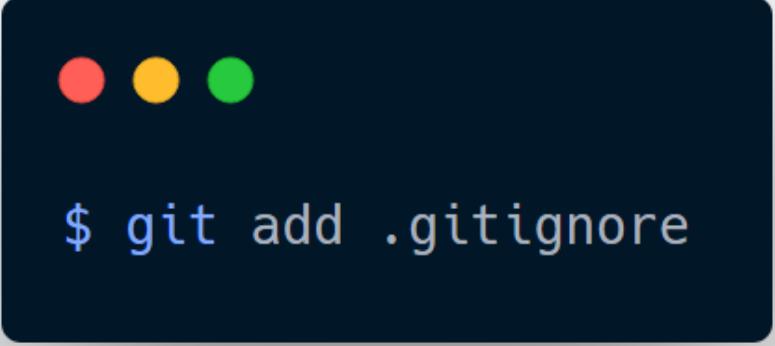
A terminal window with a dark blue background and three colored window control buttons (red, yellow, green) in the top left corner. The terminal displays the output of the 'git status' command.

```
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   Kammellander.txt

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

Zuguterletzt habe ich mit `status` überprüft, ob das **.gitignore-file** von git erkannt wurde und alle Dateien mit der Endung `.info` von Git ignoriert werden.

Änderungen dem Repository Hinzufügen

A terminal window with a dark blue background and three colored window control buttons (red, yellow, green) in the top left corner. The terminal displays the command to add the .gitignore file to the staging area.

```
$ git add .gitignore
```

Nun wird das **.gitignore-file** dem Staging-Bereich hinzugefügt.

```
$ git commit -m "Added a .gitignore-File"
[master bbcce3a] Added a .gitignore-File
 2 files changed, 2 insertions(+), 1 deletion(-)
 create mode 100644 .gitignore

$ git log
commit bbcce3aab14350b51116e5e93db81291bc3d54ab (HEAD -> master)
Author: Jan Kammellander <janschwiderik@gmail.com>
Date:   Mon Jan 11 23:32:25 2021 +0100

    Added a .gitignore-File

commit 9e4c161200272e2e3bc7412558a2cd7b9a5118c7
Author: Jan Kammellander <janschwiderik@gmail.com>
Date:   Mon Jan 11 21:23:57 2021 +0100

    NEW: First Commit
```

Anschließend wird eine neue Version von mir erstellt und mit `log` überprüfe, ob meine neue Version vorhanden ist.

Unterordner erstellen

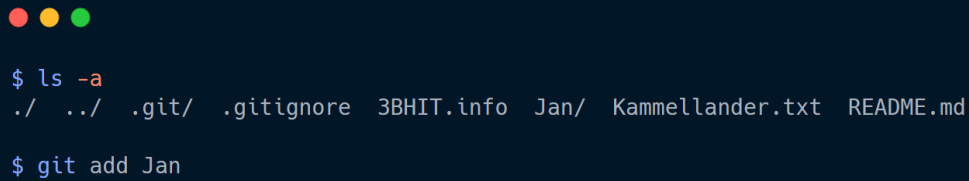
```
$ mkdir Jan

$ git status
On branch master
nothing to commit, working tree clean
```

Ich habe einen Unterordner mit dem Befehl `mkdir` namens Jan erstellt und überprüft, ob dieser von git erkannt wurde.

Unterordner dem Repository hinzufügen

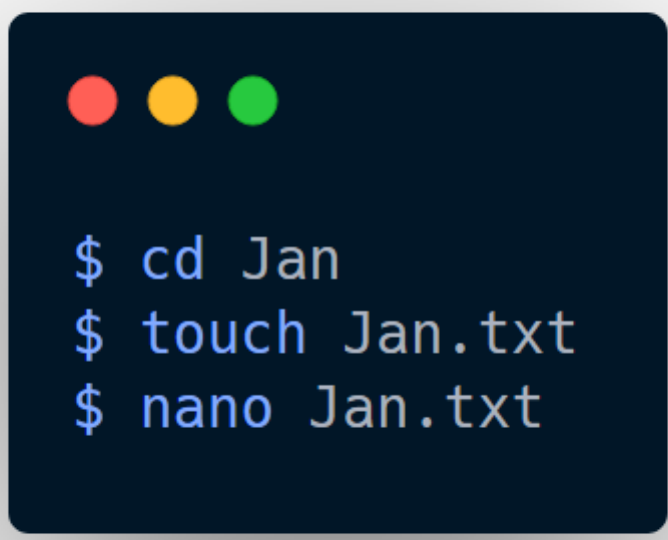
Der Ordner kann dem Repository übergeben werden. Er wird allerdings von Git nicht erkannt und muss manuell dem Staging-Bereich hinzugefügt werden. Allerdings macht es nur Sinn, wenn der Ordner ein **Subfolder** ist und keine großen irrelevanten Dateien enthält

A terminal window with a dark blue background and three colored window control buttons (red, yellow, green) in the top left corner. The terminal shows the output of a 'ls -a' command and the execution of a 'git add' command.

```
$ ls -a
./  ../  .git/  .gitignore  3BHIT.info  Jan/  Kammellander.txt  README.md

$ git add Jan
```

Dem Unterordner Dateien hinzufügen

A terminal window with a dark blue background and three colored window control buttons (red, yellow, green) in the top left corner. The terminal shows the execution of three commands: 'cd Jan', 'touch Jan.txt', and 'nano Jan.txt'.

```
$ cd Jan
$ touch Jan.txt
$ nano Jan.txt
```

Nun habe ich dem Unterordner **Jan** die Datei **Jan.txt**, welche meinen Vornamen ("*Jan*") beinhaltet.

Datei verschieben

```
$ mv ~/Documents/schule/5Semester/SEW/5a/gitKonsole/Kammellander.txt ~/Documents/schule/5Semester/SEW/5a/gitKonsole/Jan
$ nano Kammellander.txt
```

Mit dem Befehl `mv` kann ich eine Datei eines Pfades in einen anderen Pfad verschieben. Anschließend wurde die Datei **Kammellander.txt** erneut modifiziert und zwar wurde die Ziffer **2** nun mit der Zahl **3** ersetzt wurde.

```
$ git status
On branch master
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    deleted:    ../Kammellander.txt

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

no changes added to commit (use "git add" and/or "git commit -a")
```

Zuletzt habe ich mit `status` überprüft, ob alle Änderungen übernommen und erkannt wurden.

Dateien im Unterordner in Git stagen

```
$ git add .

$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   Jan/Jan.txt
    new file:   Jan/Kammellander.txt
    deleted:    Kammellander.txt
```

Mit dem Befehl `git add .` wurden alle Dateien, die nicht ignoriert werden, im Ordner in welchem ich mich befinde, dem Staging-Bereich hinzugefügt. Zuletzt überprüfe ich mit `status` meine Veränderungen.

Neue Version



```
$ git commit -m "NEW: Added a new Subfolder"
[master e7ee9b2] NEW: Added a new Subfolder
3 files changed, 2 insertions(+), 1 deletion(-)
create mode 100644 Jan/Jan.txt
create mode 100644 Jan/Kammellander.txt
delete mode 100644 Kammellander.txt
```

Nun wird eine dritte Version erstellt, in welche ein Unterordner, mit dessen Dateien, hinzugefügt wurde.



```
$ git log
commit e7ee9b295c4f7d26ecd6d4d355da124fbbb4e735 (HEAD -> master)
Author: Jan Kammellander <janschwiderik@gmail.com>
Date: Mon Jan 11 23:48:32 2021 +0100

    NEW: Added a new Subfolder

commit bbcce3aab14350b51116e5e93db81291bc3d54ab
Author: Jan Kammellander <janschwiderik@gmail.com>
Date: Mon Jan 11 23:32:25 2021 +0100

    Added a .gitignore-File

commit 9e4c161200272e2e3bc7412558a2cd7b9a5118c7
Author: Jan Kammellander <janschwiderik@gmail.com>
Date: Mon Jan 11 21:23:57 2021 +0100

    NEW: First Commit
```

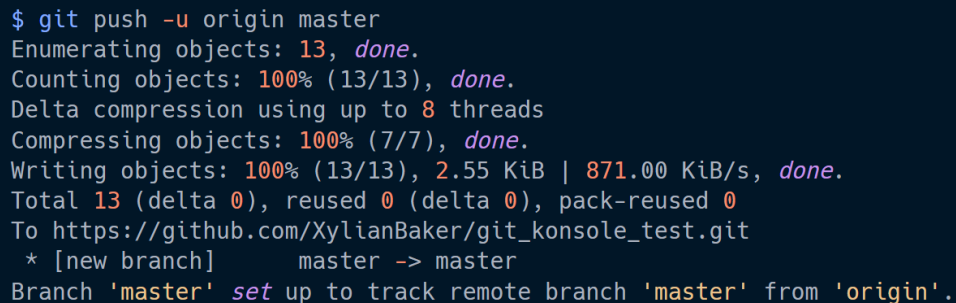
Zuletzt habe ich mit `log` alle meine Versionen überprüft.

Das lokale repository als remote-repository hinzufügen



```
$ git remote add origin https://github.com/XylianBaker/git_konsole_test.git
$ git branch -M master
$ git push -u origin master
```

Mit dem Befehl `git remote add origin` wird das lokale Repository mit einem Remote Repository verbunden. Der Befehl `branch -M master` erstellt ein branch für das Remote Repository.



```
$ git push -u origin master
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 8 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (13/13), 2.55 KiB | 871.00 KiB/s, done.
Total 13 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/XylianBaker/git_konsole_test.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

Zuletzt wird mit `git push -u origin master` alle Commits auf das remote Repository auf den Branch Master gepushed.

Github repository

-> [My Github repo](#)