

Web-Entwicklung

Überblick

THEMA	BESCHREIBUNG
Intro	Grundlagen
Git	Grundlagen Git, Lokale Git Installation, ausgewählte/gängige Befehle
GitHub	Grundlagen GitHub, UI und Bedienung über Browser, Integration und Nutzung in VSCode
Shell Befehle	Grundlagen, ausgewählte/gängige Befehle

Git Grundlagen



GIT

≠

GITHUB



Was ist Git?

Git ist ein verteiltes Versionskontrollsystem zur Verwaltung von Quellcode.

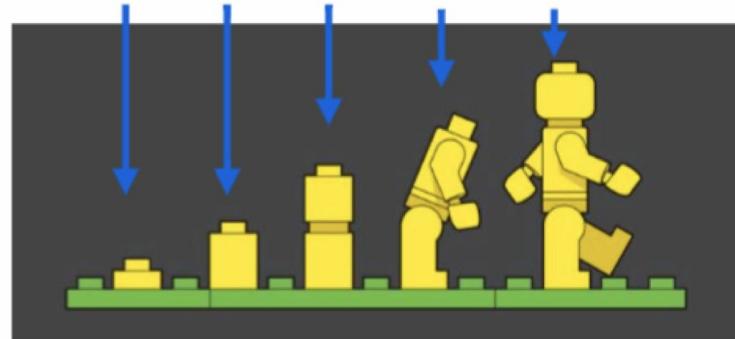
Ok, was ist also Versionskontrolle? Einfach ausgedrückt, ist die Versionskontrolle ein System zur Verfolgung von Änderungen an Dateien. Wenn wir Dateien ändern, zeichnet das Versionskontrollsystem jede Änderung auf und speichert sie. Auf diese Weise können wir jederzeit eine frühere Version unseres Codes wiederherstellen.

Quelle: <https://backlog.com/git-tutorial/what-is-git/>

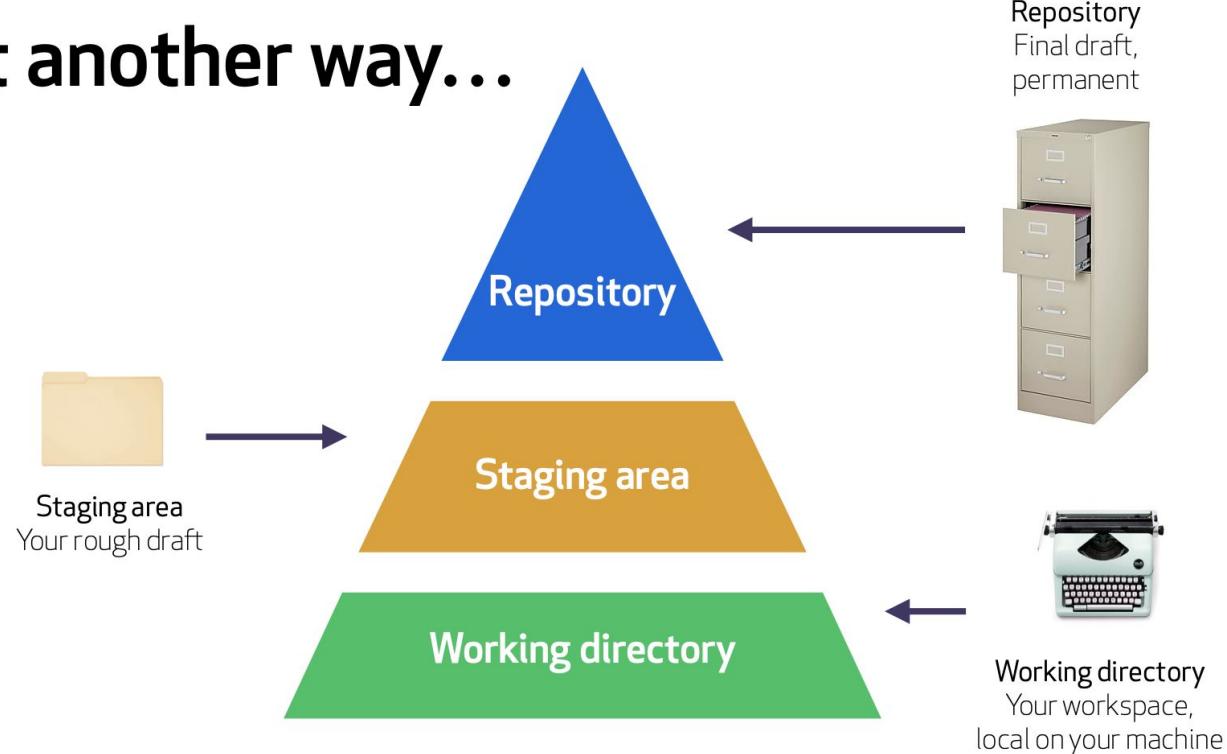
Git is a *version control system*

A tool that lets you track your progress over time.

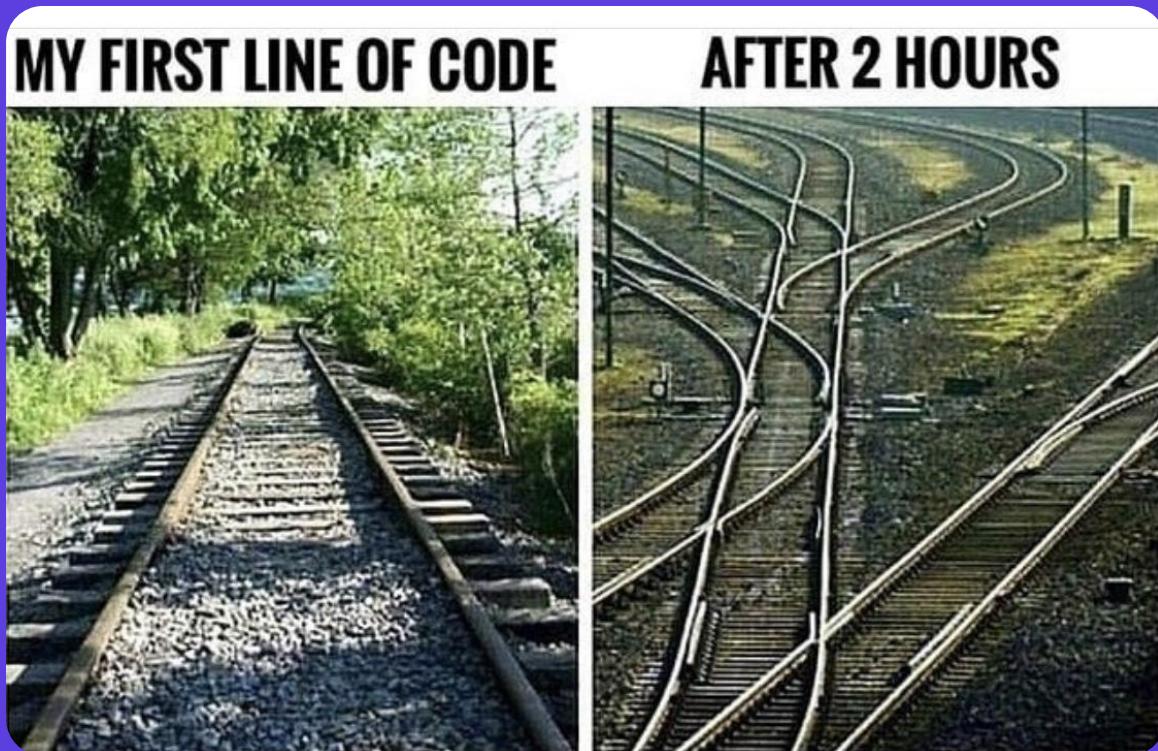
v.01 v0.2 v0.3 V0.4 V1.0



Put another way...

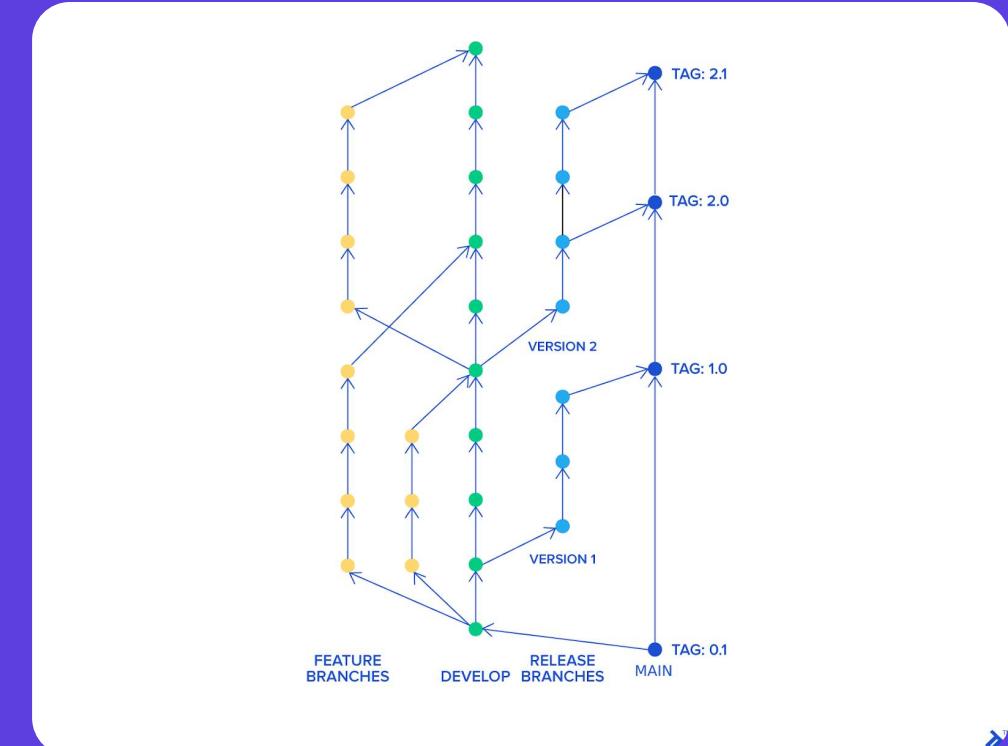


Warum brauchen wir Git?

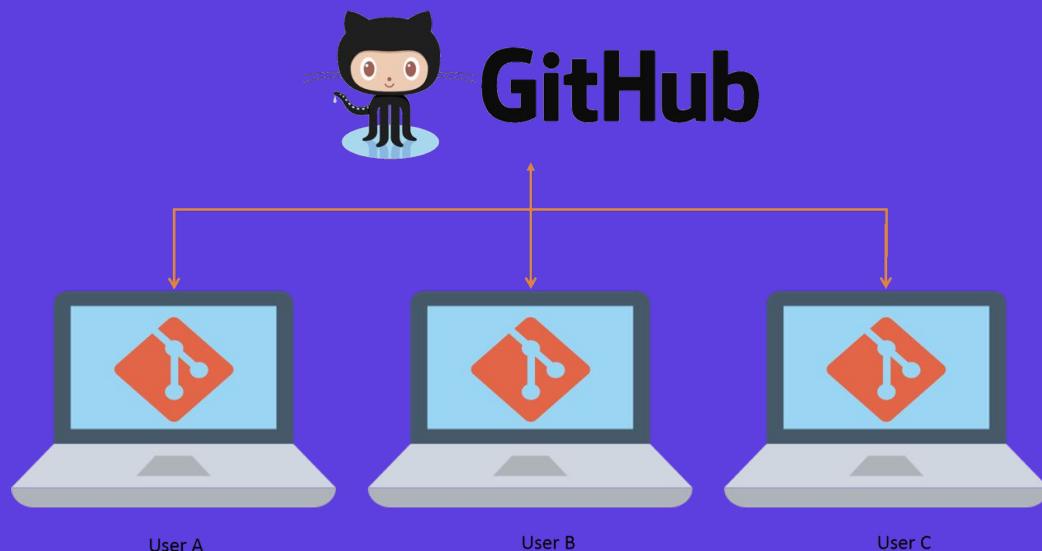


Git Flussdiagramm

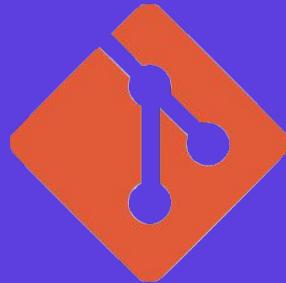
Allgemeiner Git Workflow als Git-Flussdiagramm dargestellt.



Local & remote



Installation & Anmeldung



GIT



GITHUB

Die Installation von Git ist recht simple, wir geben im Terminal den Befehl:

```
steffenklompges@MBP-von-Steffen git_installation % git --version
```

Nun sollte sich ein Installations-Menü öffnen, sobald dieses ausgeführt ist, ist Git auch schon auf deinem PC installiert. Falls sich das Menü nicht öffnet, kannst du Git auch einfach [herunterladen](#) und so installieren.

Nachdem Git nun auf deinem PC installiert ist, kannst du jetzt Git-Befehle nutzen und damit arbeiten. Doch bevor du einfach deine Projekte hochladen kannst, müssen wir deine Userdaten von GitHub in Git auf deinem PC hinterlegen.



```
git config --global user.name "your_username"  
git config --global user.email "your_email_address"  
git config --global --list
```

Bitte setzt deine Daten für user.name, user.email und das ganze ohne die Anführungsstriche. Der letzte Befehl dient nur zur Kontrolle für dich, ob alles richtig eingetragen ist. Wenn alles passt, dann beginnt der ganz normale Git/Github Workflow zum Initialisieren und hochladen von einem Projekt.

Es gibt mehrere Möglichkeiten, eine Verbindung von deinem lokalen Git zu GitHub herzustellen. Die beiden Standard Möglichkeiten sind über **https** oder **ssh**, welche der beiden du nutzt, bleibt dir überlassen. Da sich hier ständig etwas ändert, musst du dir hier immer die aktuellen Gegebenheiten anschauen.

git init

Du kannst einen neuen Repo von Grund auf neu erstellen, indem Du den Befehl “**git init**” verwendest. Es kann verwendet werden, um Git in ein bestehendes, nicht versioniertes Projekt einzuführen, um Änderungen zu verfolgen.

neues repository erstellen

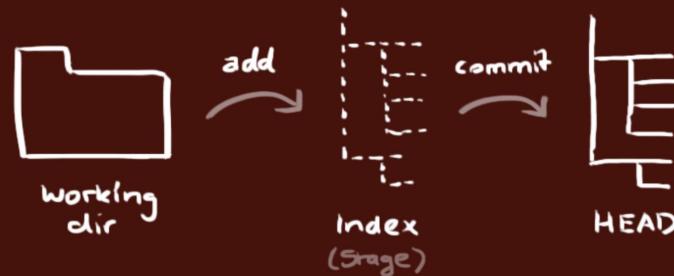
erstelle ein neues Verzeichnis, öffne es und führe

git init

aus, um ein neues git-Repository anzulegen.

workflow

Dein lokales Repository besteht aus drei "Instanzen", die von git verwaltet werden. Die erste ist deine **Arbeitskopie**, welche die echten Dateien enthält. Die zweite ist der **Index**, welcher als Zwischenstufe agiert und zu guter Letzt noch der **HEAD**, der auf deinen letzten Commit zeigt.



git add & git commit

Du kannst Änderungen vorschlagen (zum Index hinzufügen) mit

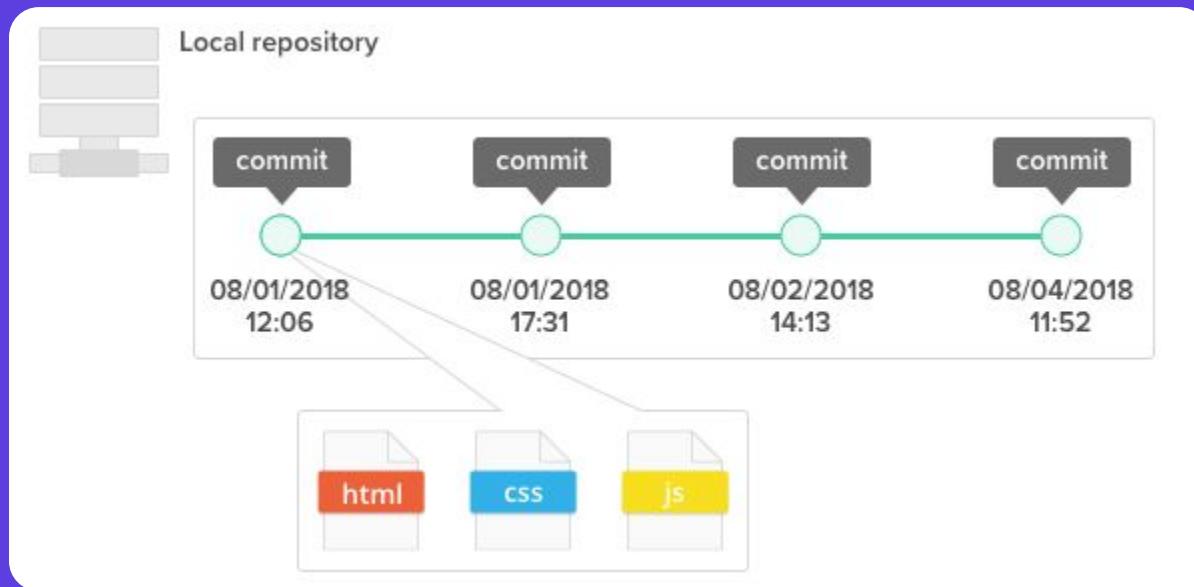
```
git add <dateiname>  
git add .
```

Du bestätigst deine Änderungen mit:

```
git commit -m "Commit-Nachricht"
```

Jetzt befindet sich die Änderung im HEAD, aber noch nicht im entfernten Repository.

git commit



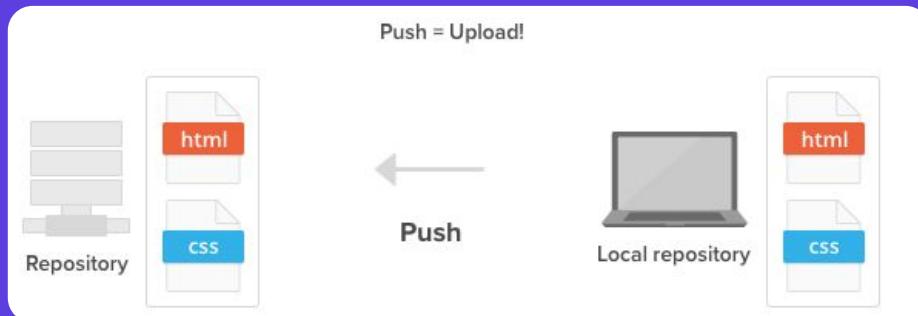
git push

änderungen hochladen

Die Änderungen sind jetzt im **HEAD** deines lokalen Repositories. Um die Änderungen an dein entferntes Repository zu senden, führe:

```
git push origin main
```

aus. Du kannst *main* auch mit einem beliebigen anderen Branch ersetzen, mehr über Branches erfährst du später.



git remote

Wenn du dein lokales Repository nicht von einem entfernten geklont hast und du diese aber mit einem anderen Repository verbinden möchtest, musst du dieses mit `git remote add origin <server>` hinzufügen. Jetzt bist du bereit, deine Änderungen hochzuladen

git tag & git log

Du kannst einen Tag mit
“git tag” erstellen und erhältst die
Liste der Commit-IDs mit **“git log”**.

tagging

Es wird empfohlen, für Software Releasestags zu verwenden. Dies ist ein bekanntes Konzept, das es schon mit SVN gab. Du kannst einen neuen

Tag namens *1.0.0* mit folgendem Befehl erstellen:

```
git tag 1.0.0 1b2e1d63ff
```

1b2e1d63ff steht für die ersten 10 Zeichen der Commit-Id, die du mit
deinem Tag referenzieren möchtest. Du erhältst die Liste der Commit-

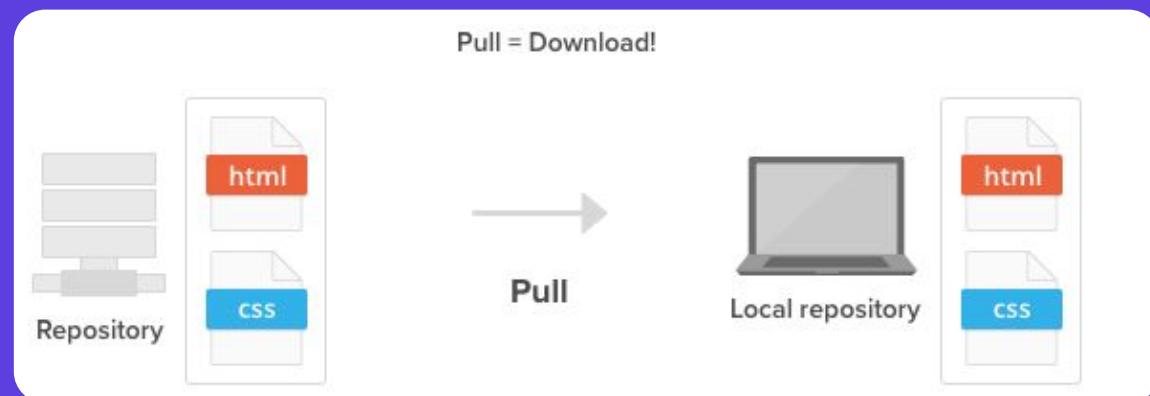
IDs mit:

```
git log
```

Du kannst auch weniger Zeichen verwenden, es muss einfach eindeutig
sein.

git pull

Immer wenn jemand seine Änderungen in das freigegebene Remote-Repository verschiebt, wird Dein lokales Repository veraltet. Um Dein lokales Repository mit dem neu aktualisierten Remote-Repository neu zu synchronisieren, führe einfach die Aktion “**git pull**” aus.



git clone

ein repository auschecken

erstelle eine Arbeitskopie, indem du folgenden Befehl ausführst:

```
git clone /pfad/zum/repository
```

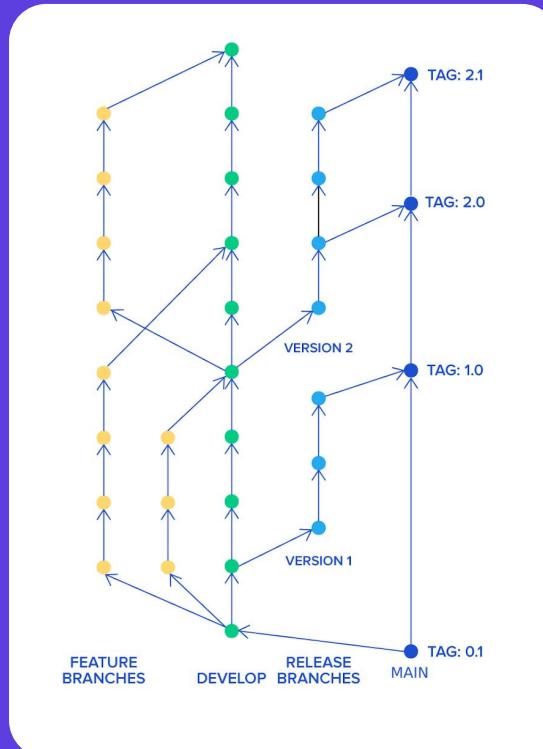
Falls du ein entferntes Repository verwendest, benutze:

```
git clone benutzername@host:/pfad/zum/repository
```

Branching

branching

Branches werden benutzt, um verschiedene Funktionen isoliert voneinander zu entwickeln. Der *main*-Branch ist der "Standard"-Branch, wenn du ein neues Repository erstellst. Du solltest aber für die Entwicklung andere Branches verwenden und diese dann in den Main-Branch zusammenführen (mergen). Auch das lernst du später.



git branch & git checkout

Erstelle einen neuen Branch mit dem Namen "feature_x" und wechsle zu diesem:

```
git checkout -b feature_x
```

Um zum Main zurück zu wechseln:

```
git checkout main
```

Und um den eben erstellten Branch wieder zu löschen:

```
git branch -d feature_x
```

Ein Branch ist *nicht für andere verfügbar*, bis du diesen in dein entferntes Repository hochlädst:

```
git push origin <branch>
```



Keine Sorge!
Wir wenden an
und lernen dabei :)

Terminal Grundlagen

Was ist Terminal?

Ein Terminal ist textbasiert und dient als Befehlszeilenschnittstelle (Command Line Interface = CLI), in die Ihr Eure Befehle eingeben könnt. Eine Shell nimmt diese Befehle entgegen und weist das Betriebssystem an, sie auszuführen.

Was ist Shell?

In der Informatik wird als Shell die Software bezeichnet, mittels derer ein Benutzer mit einem Betriebssystem interagiert – eine Mensch-Maschine-Schnittstelle.

Während „Kernel“ den innersten Kern eines Betriebssystems bezeichnet, bezeichnet „Shell“ (englisch für „Schale“, „Hülle“ oder „Außenhaut“) dessen äußerste Schicht und damit jene Schnittstelle (englisch Interface) auf die ein Benutzer trifft.

Damit ist es möglich, Kernelservices zu nutzen und sich über Systemkomponenten zu informieren oder sie und andere Programme zu benutzen.

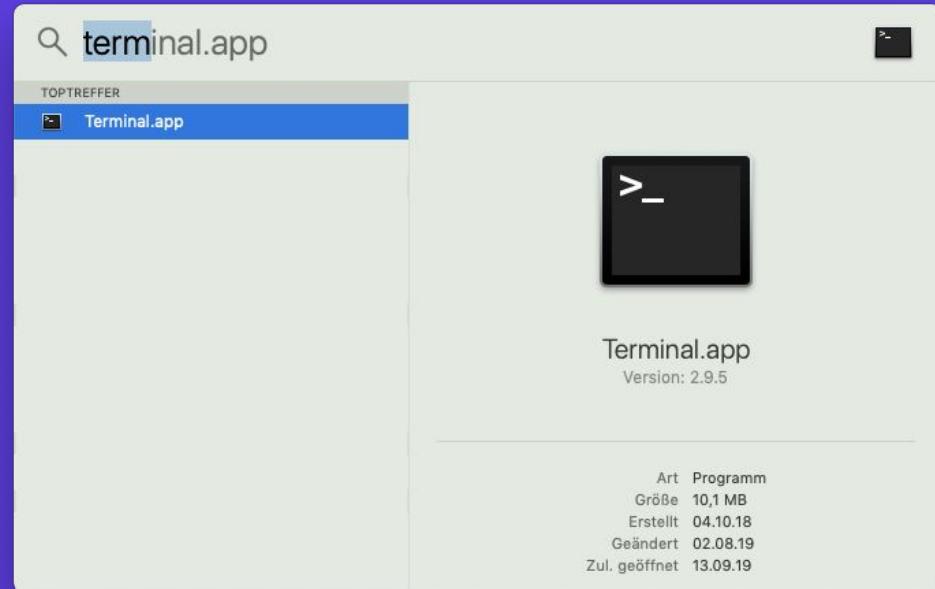
Terminal app via “Launchpad” öffnen

Auf Launchpad unten im Dock klicken und in dem ordner “Andere” das App-Icon für “Terminal” klicken.

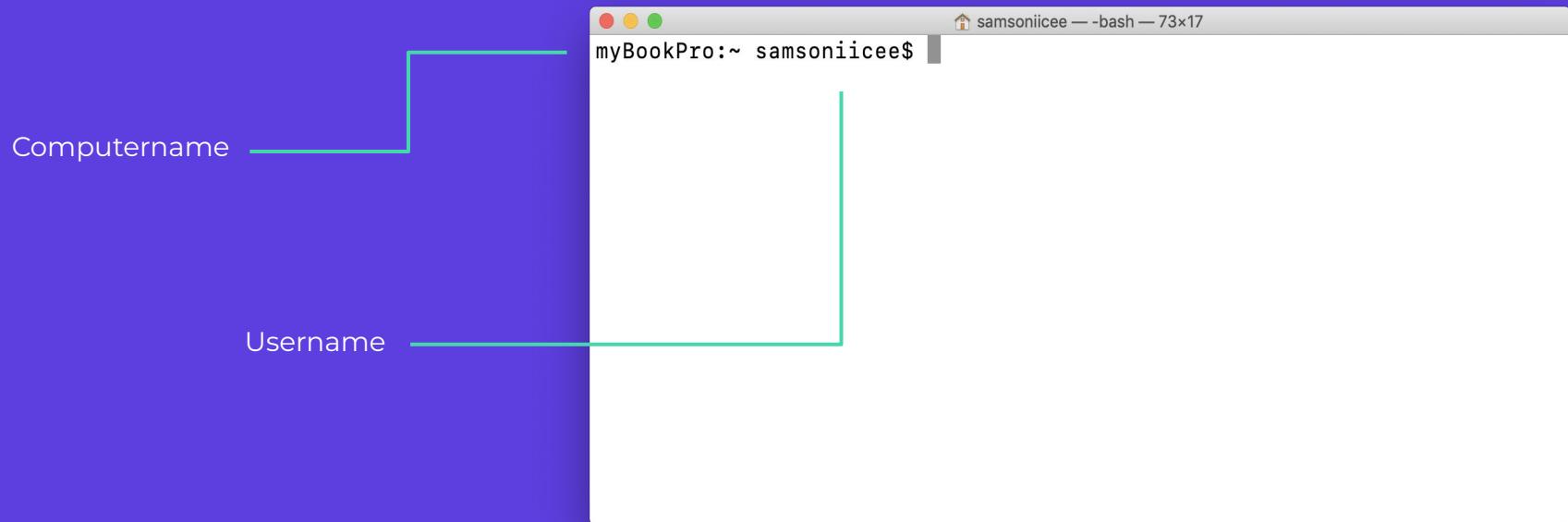


Terminal app via “Spotlight” öffnen

Unter macOS könnt Ihr Spotlight (⌘ + Space) verwenden um "Terminal" (oder jede andere App) einzugeben, um es zu finden und mit "Enter" zu öffnen.



Command line prompt



Git & Terminal Grundlagen

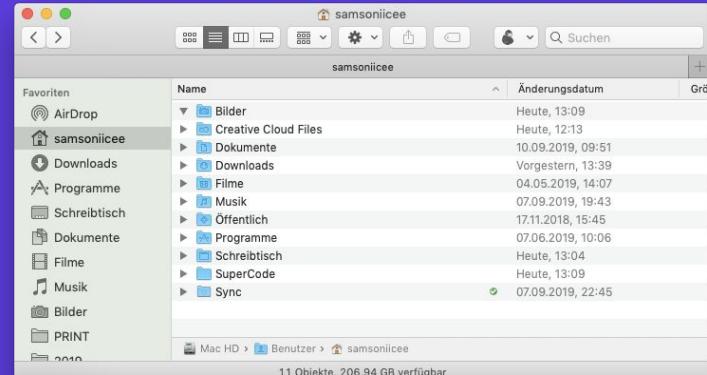
Terminal

pwd

Print Working Directory



```
samsoniicee ~ -bash 57x17
myBookPro:~ samsoniicee$ pwd
/Users/samsoniicee
myBookPro:~ samsoniicee$
```



Git & Terminal Grundlagen

Terminal

ls

List

```
myBookPro:~ samsoniicee$ ls
Applications          Movies
Creative Cloud Files  Music
Desktop               Pictures
Documents              Public
Downloads             SuperCode
Library                Sync
myBookPro:~ samsoniicee$
```



Favoriten	Name	Änderungsdatum
AirDrop	Bilder	Heute, 13:09
samsoniicee	Creative Cloud Files	Heute, 12:13
Downloads	Dokumente	10.09.2019, 09:51
Programme	Downloads	Vorgestern, 13:39
Schreibtisch	Filme	04.05.2019, 14:07
	Musik	07.09.2019, 19:43
	Öffentlich	17.11.2018, 15:45
	Programme	07.06.2019, 10:06
	Schreibtisch	Heute, 13:04
	SuperCode	Heute, 13:09
	Sync	07.09.2019, 22:45

Git & Terminal Grundlagen

Terminal

ls -1

List in einer Spalte

```
myBookPro:~ samsoniicee$ ls -1
Applications
Creative Cloud Files
Desktop
Documents
Downloads
Library
Movies
Music
Pictures
Public
SuperCode
Sync
myBookPro:~ samsoniicee$
```

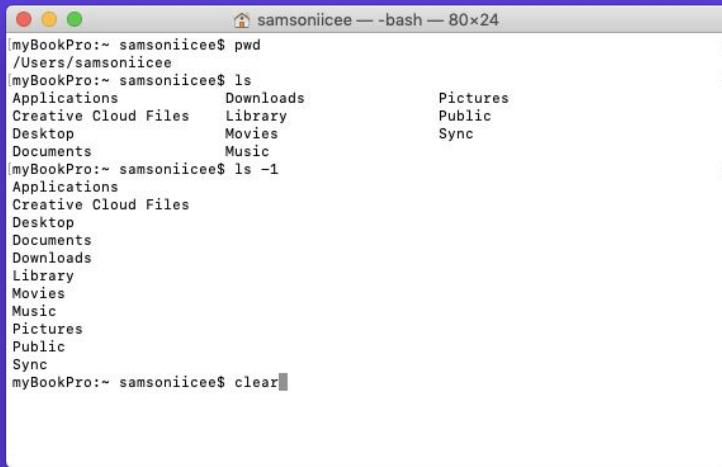


Favoriten	Name	Änderungsdatum
AirDrop	Bilder	Heute, 13:09
samsoniicee	Creative Cloud Files	Heute, 12:13
Downloads	Dokumente	10.09.2019, 09:51
Programme	Downloads	Vorgestern, 13:39
Schreibtisch	Filme	04.05.2019, 14:07
Dokumente	Musik	07.09.2019, 19:43
Filme	Öffentlich	17.11.2018, 15:45
Musik	Programme	07.06.2019, 10:06
Öffentlich	Schreibtisch	Heute, 13:04
Programme	SuperCode	Heute, 13:09
Schreibtisch	Sync	07.09.2019, 22:45
SuperCode		
Sync		

Git & Terminal Grundlagen

Terminal

clear



```
samsoniicee — -bash — 80x24
myBookPro:~ samsoniicee$ pwd
/Users/samsoniicee
myBookPro:~ samsoniicee$ ls
Applications          Downloads          Pictures
Creative Cloud Files  Library           Public
Desktop               Movies            Sync
Documents             Music
myBookPro:~ samsoniicee$ ls -l
Applications
Creative Cloud Files
Desktop
Documents
Downloads
Library
Movies
Music
Pictures
Public
Sync
myBookPro:~ samsoniicee$ clear
```



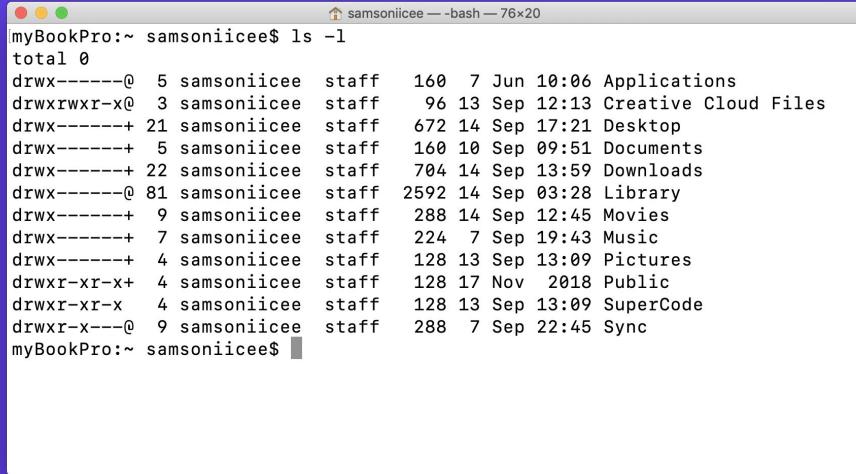
```
samsoniicee — -bash — 80x24
myBookPro:~ samsoniicee$
```

Git & Terminal Grundlagen

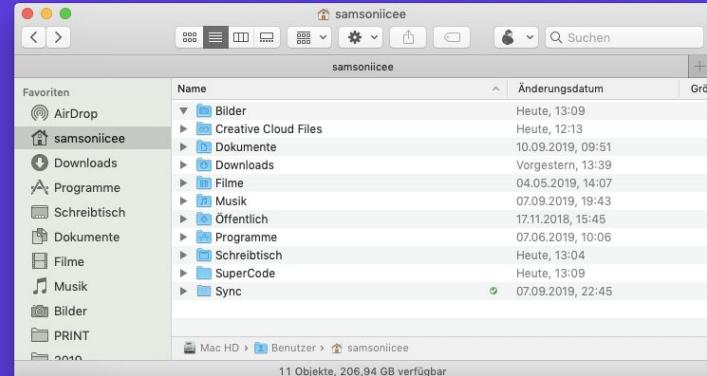
Terminal

ls -l

List - long format



```
myBookPro:~ samsoniicee$ ls -l
total 0
drwx-----@ 5 samsoniicee  staff  160  7 Jun 10:06 Applications
drwxrwxr-x@ 3 samsoniicee  staff   96 13 Sep 12:13 Creative Cloud Files
drwx-----+ 21 samsoniicee  staff  672 14 Sep 17:21 Desktop
drwx-----+ 5 samsoniicee  staff  160 10 Sep 09:51 Documents
drwx-----+ 22 samsoniicee  staff  704 14 Sep 13:59 Downloads
drwx-----@ 81 samsoniicee  staff 2592 14 Sep 03:28 Library
drwx-----+ 9 samsoniicee  staff  288 14 Sep 12:45 Movies
drwx-----+ 7 samsoniicee  staff  224  7 Sep 19:43 Music
drwx-----+ 4 samsoniicee  staff  128 13 Sep 13:09 Pictures
drwxr-xr-x+ 4 samsoniicee  staff  128 17 Nov 2018 Public
drwxr-xr-x+ 4 samsoniicee  staff  128 13 Sep 13:09 SuperCode
drwxr-x---@ 9 samsoniicee  staff  288  7 Sep 22:45 Sync
myBookPro:~ samsoniicee$
```



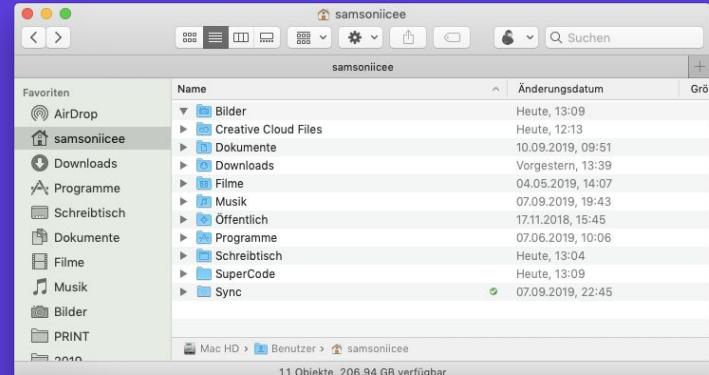
Git & Terminal Grundlagen

Terminal

ls -a

List mit versteckten Dateien

```
myBookPro:~ samsoniicee$ ls -a
.                 .gitconfig        Desktop
..                .localized         Documents
.509830.padl      .openjfx          Downloads
.CFUserTextEncoding .rstudio-desktop Library
.DS_Store          .sip_v1           Movies
.RData              .ssh              Music
.Rhistory          .subversion       Pictures
.Trash              .viminfo          Public
.bash_history      .vscode           SuperCode
.bash_sessions     Applications     Sync
.cups               Creative Cloud Files
myBookPro:~ samsoniicee$
```



Git & Terminal Grundlagen

Terminal

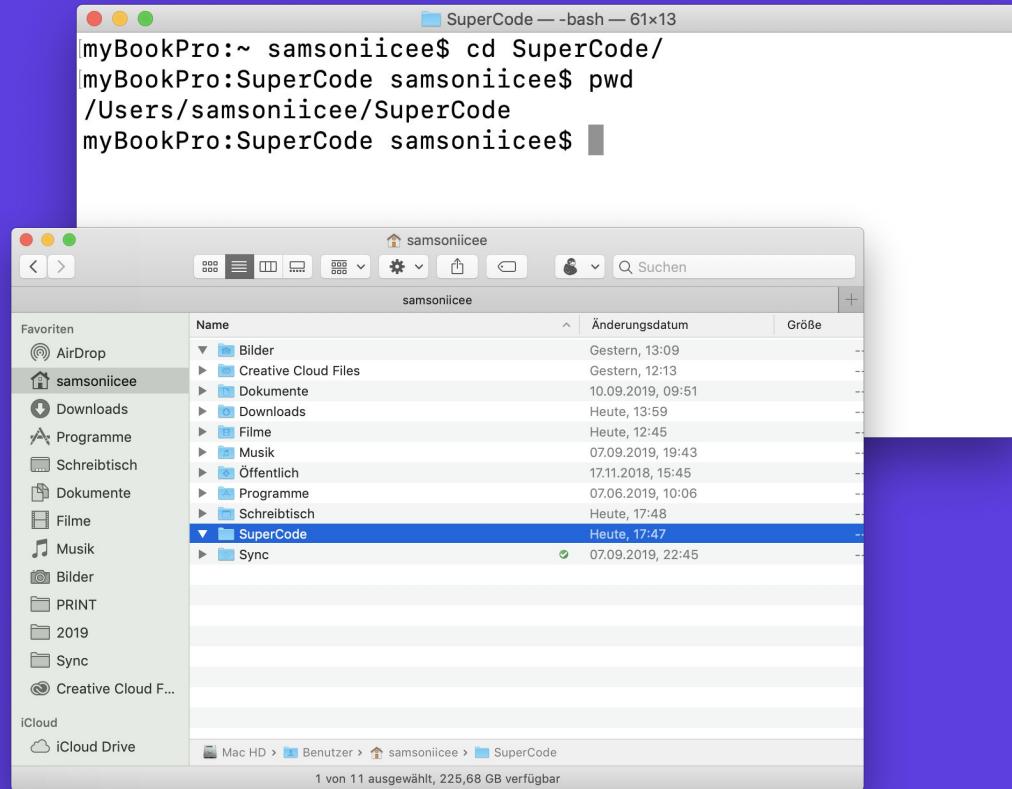
cd

Change directory

Hier bewegen wir uns beispielhaft in den "SuperCode" Ordner.

Shortcut:
Verwendung der Tab-Taste

Wo bin ich?



Git & Terminal Grundlagen

Terminal

mkdir

Make directory

```
SuperCode — --bash — 61x13
myBookPro:SuperCode samsonicee$ mkdir beispielProjekt
myBookPro:SuperCode samsonicee$
```

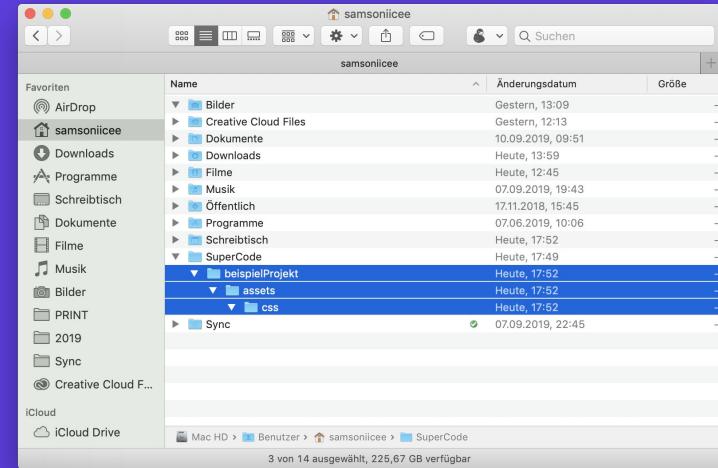


samsonicee		
Favoriten	Name	Änderungsdatum
AirDrop	Bilder	Gestern, 13:09
samsonicee	Creative Cloud Files	Gestern, 12:13
Downloads	Dokumente	10.09.2019, 09:51
Programme	Downloads	Heute, 13:59
Schreibtisch	Filme	Heute, 12:45
Dokumente	Musik	07.09.2019, 19:43
Filme	Öffentlich	17.11.2018, 15:45
Musik	Programme	07.06.2019, 10:06
Bilder	Schreibtisch	Heute, 17:48
PRINT	SuperCode	Heute, 17:49
2019	beispielProjekt	Heute, 17:49
Sync	Sync	07.09.2019, 22:45
Creative Cloud F...		
iCloud		
iCloud Drive		

mkdir

Übung: Erstelle einen “assets” und “css” ordner, wie folgt:

```
myBookPro:SuperCode samsoniicee$ mkdir beispielProjekt
myBookPro:SuperCode samsoniicee$ cd beispielProjekt/
myBookPro:beispielProjekt samsoniicee$ mkdir assets
myBookPro:beispielProjekt samsoniicee$ cd assets/
myBookPro:assets samsoniicee$ mkdir css
myBookPro:assets samsoniicee$
```

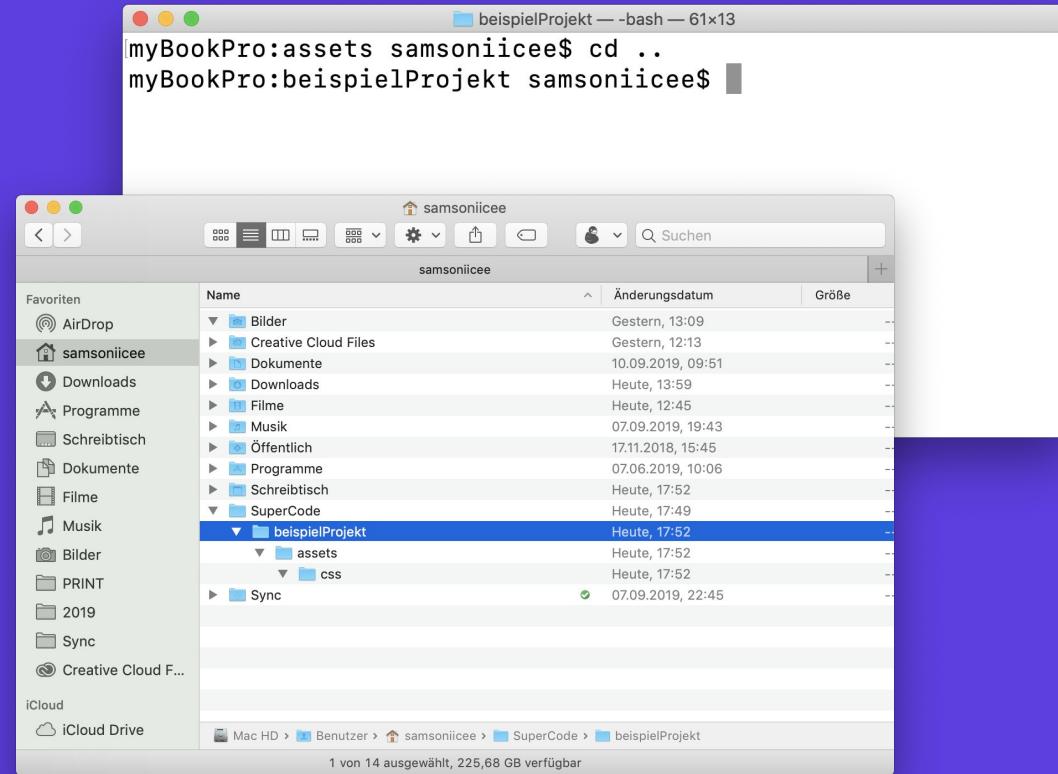


Git & Terminal Grundlagen

Terminal

cd ..

Change directory zurück



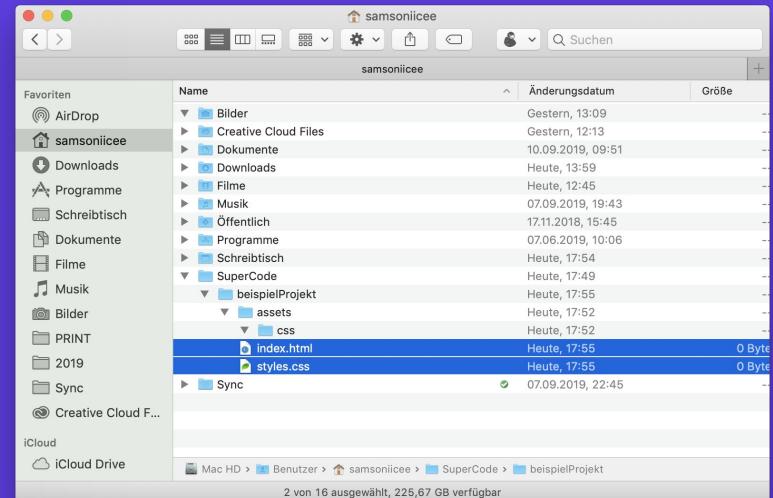
Git & Terminal Grundlagen

Terminal

touch

Eine Datei erstellen

```
myBookPro:beispielProjekt samsoniicee$ touch index.html
myBookPro:beispielProjekt samsoniicee$ touch styles.css
myBookPro:beispielProjekt samsoniicee$
```



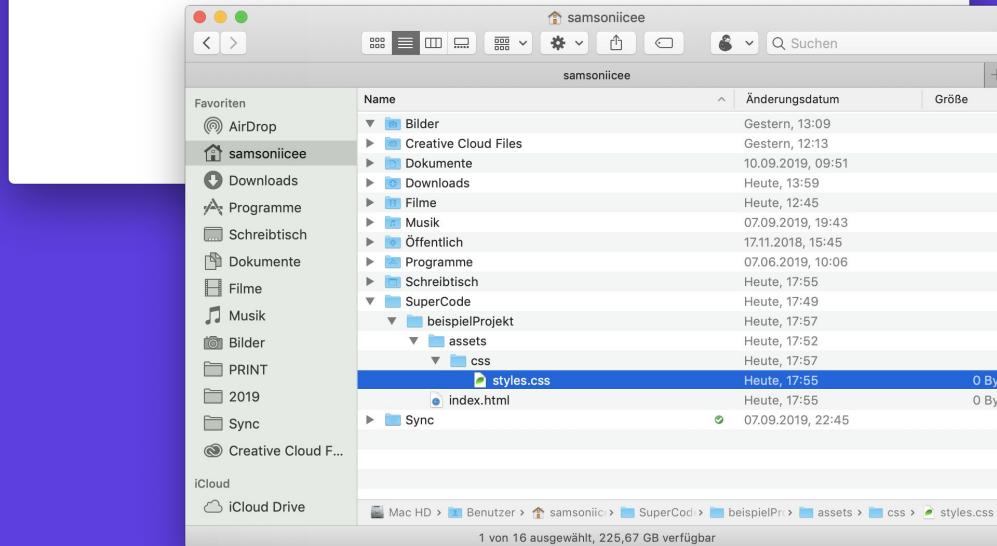
Git & Terminal Grundlagen

Terminal

mv

Datei verschieben

```
beispielProjekt -- bash -- 75x13
myBookPro:beispielProjekt samsonicee$ mv styles.css assets/css/
myBookPro:beispielProjekt samsonicee$
```

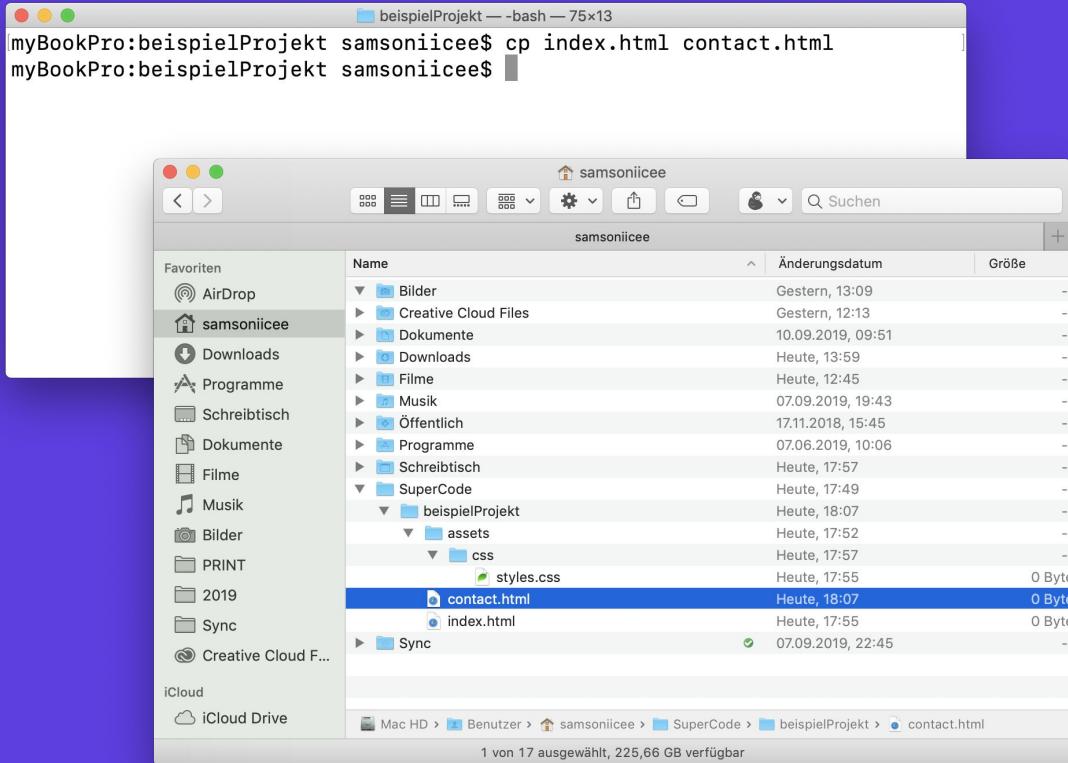


Git & Terminal Grundlagen

Terminal

cp

Datei kopieren

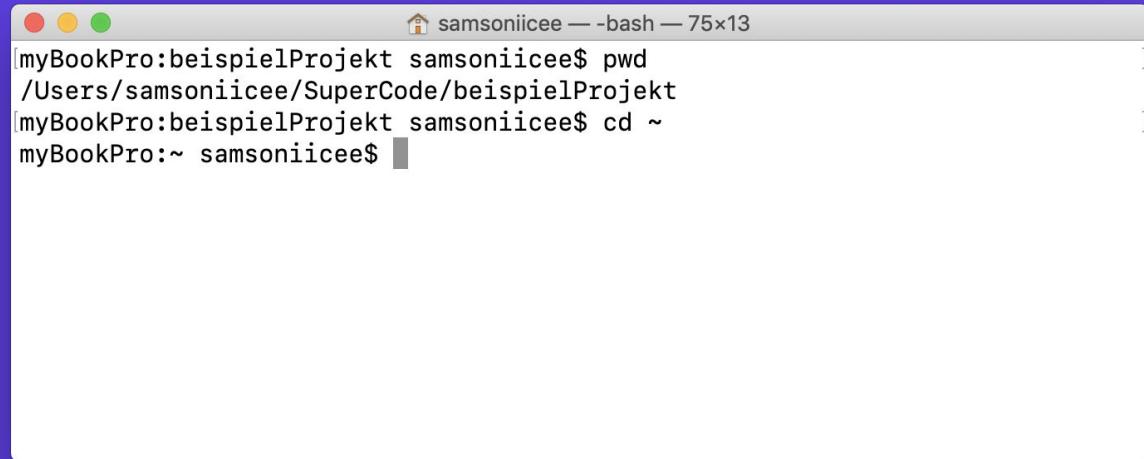


Git & Terminal Grundlagen

Terminal

cd ~

Zurück zum User-Ordner mit
Tilde (Option N).



A screenshot of a macOS terminal window titled "samsoniicee — -bash — 75x13". The window shows the following command history:

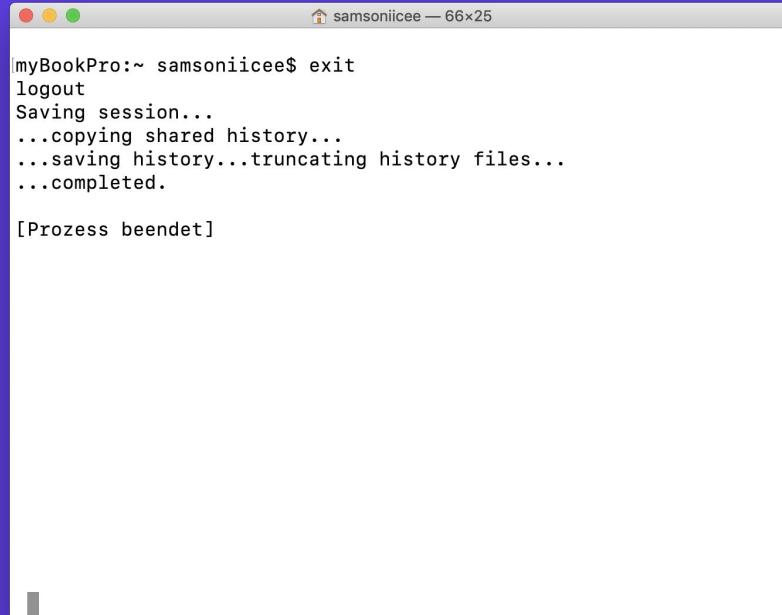
```
[myBookPro:beispielProjekt samsoniicee$ pwd  
/Users/samsoniicee/SuperCode/beispielProjekt  
[myBookPro:beispielProjekt samsoniicee$ cd ~  
myBookPro:~ samsoniicee$ ]
```

Git & Terminal Grundlagen

Terminal

exit

Das aktuelle Terminal beenden/verlassen.



A screenshot of a macOS terminal window titled "samsoniicee — 66x25". The window shows the command "exit" being run, followed by a series of messages indicating the session is being saved and history files are being truncated. The process is completed successfully.

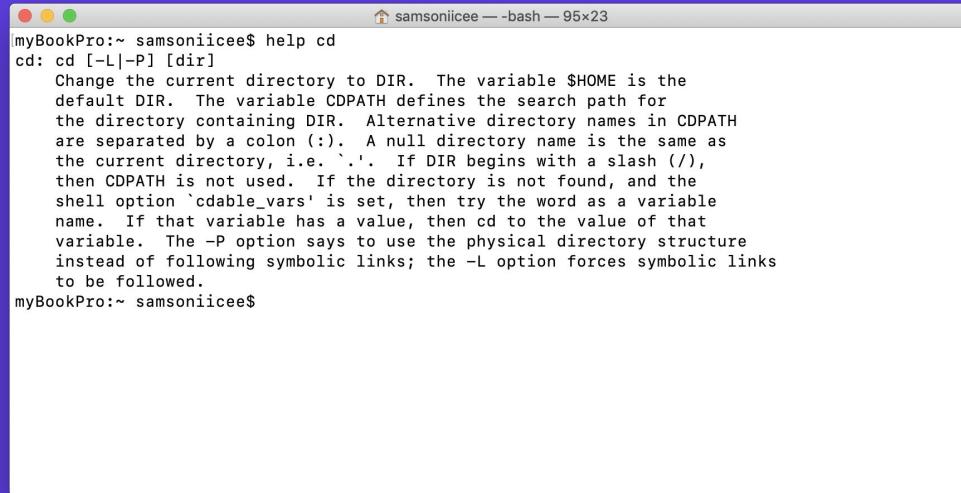
```
myBookPro:~ samsoniicee$ exit
logout
Saving session...
...copying shared history...
...saving history...truncating history files...
...completed.

[Prozess beendet]
```

help

Die Shell Hilfe

Beispiel: help cd



```
myBookPro:~ samsoniicee$ help cd
cd: cd [-L|-P] [dir]
      Change the current directory to DIR.  The variable $HOME is the
      default DIR.  The variable CDPATH defines the search path for
      the directory containing DIR.  Alternative directory names in CDPATH
      are separated by a colon (:).  A null directory name is the same as
      the current directory, i.e. '.'.  If DIR begins with a slash (/),
      then CDPATH is not used.  If the directory is not found, and the
      shell option 'cdable_vars' is set, then try the word as a variable
      name.  If that variable has a value, then cd to the value of that
      variable.  The -P option says to use the physical directory structure
      instead of following symbolic links; the -L option forces symbolic links
      to be followed.
myBookPro:~ samsoniicee$
```

Noch ein paar nützliche Shortcuts

Ctrl + U: Löscht die Linie vom Cursorpunkt zurück zum Anfang.

Ctrl + A: Bewegt den Cursor an den Anfang der Zeile.

Ctrl + E: Bewegt den Cursor an das Ende der Zeile.

Pfeiltaste hoch/runter: Ermöglicht es Ihnen, die vorherigen Befehle zu durchsuchen.



Keine Sorge!
Wir wenden an
und lernen dabei :)

Ressourcen

Git - Der einfache Einstieg

<https://rogerdudler.github.io/git-guide/index.de.html>

Git Dokumentation

<https://git-scm.com/doc>

What is git?

<https://backlog.com/git-tutorial/what-is-git/>

Git Cheat Sheet

<https://github.github.com/training-kit/downloads/de/github-git-cheat-sheet/>

Verwendete Quellen

<https://developer.mozilla.org/de/docs>

<https://www.w3schools.com/>

<https://learn.freecodecamp.org/>