



FH Salzburg

# Grundlagen der Informatik

Labor/Übung 02: Versionsverwaltung

Technik  
Gesundheit  
Medien

# Was ist Versionsverwaltung (VCS)?



- Zur Verwaltung von Änderungen
- Ausschließlich für nicht-binär Files! (→ Textfiles jeder Art, z.B. Quellcode, ...)
- Wichtig in Softwareentwicklung, wissenschaftlichen Arbeiten und Projektmanagement

# Wozu wird Versionsverwaltung verwendet?



- Nachvollziehbarkeit: alle Änderungen und durchführender User über Zeit
- Wiederherstellbarkeit: beliebige, alte Versionen können wiederhergestellt werden
- Zusammenarbeit: gleichzeitige Bearbeitung durch mehrere Personen

# Begriffsdefinitionen



- Repository: zentrale Ablage für Dateien und Historie
- Commit: gespeicherte Momentaufnahme von Dateien zu bestimmtem Zeitpunkt
- Merge: Zusammenführen von verschiedenen Änderungen
- Branches: parallele Entwicklungszweige für verschiedene Features/Versionen

# Arten von VCS



- Lokale VCS
  - Speichert Versionen einer Datei auf lokalem System
  - Beispiel: einfaches Backup-Management mit Datei-Kopien
- Zentralisierte VCS (CVCS)
  - Einzelnes, zentrales Repository, auf das alle zugreifen
  - Beispiele: CVS, Subversion (SVN)
  - Vor-/Nachteile: einfach zu verwalten, Single Point of Failure

# Arten von VCS



- Verteilte VCS (DVCS)
  - Jeder User hat Kopie von Repositories
  - Beispiele: git, Mercurial
  - Vorteile: keine Abhängigkeit von zentralen Server, schnelle Operationen, bessere Zusammenarbeit

# git für Department IT



- <https://its-git.fh-salzburg.ac.at/>
- Zugriff mit fhsXXXXXX@fh-salzburg.ac.at und Passwort

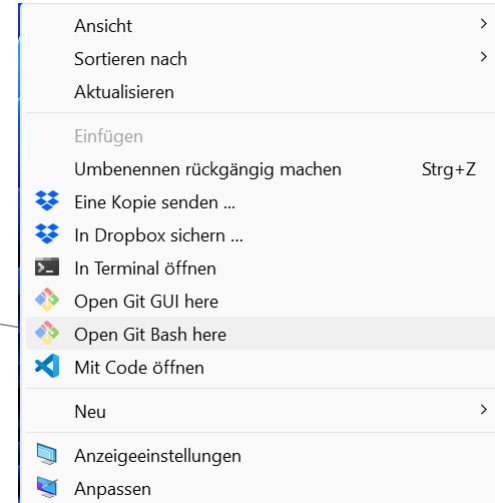
# Versionsverwaltung – git



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

Öffnen der git Bash

- ssh-keygen in Bash eingeben
- Bei Frage nach Pfad “Enter” drücken
- Bei Frage nach Passwort “enter” drücken
- cat ~/.ssh/id\_rsa.pub eingeben und Ausgabe kopieren



```
thsch@Notebook MINGW64 ~/Desktop/git test
$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/c/Users/thsch/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
```



# Versionsverwaltung – git



1.

2.

3.

**User settings**

- Profile
- Account
- Applications
- Chat
- Access Tokens
- Emails
- Notifications
- SSH Keys**
- SSH Keys

## SSH Keys

SSH keys allow you to establish a secure connection between your computer and GitLab. SSH fingerprints verify that the client is connecting to the correct host. Check the [current instance configuration](#).

4.

Your SSH keys 8

Title	Key	Usage type	Created	Last used	Expires
-------	-----	------------	---------	-----------	---------

**SSH Keys**

Your SSH keys 8

**Add an SSH key**

Add an SSH key for secure access to GitLab. [Learn more.](#)

Key

Begin with 'ssh-rsa', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', 'ecdsa-sha2-nistp521' or 'ssh-ed25519@openssh.com'.

**Title**

Key titles are publicly visible.

**Usage type**

**Expiration date**

Optional but recommended. If set, key becomes invalid on the specified date.

5.

Kopierte Ausgabe von letzter Folie einfügen

# Versionsverwaltung – git



Your work > Projects

## Projects

Yours 224 Starred 0

Filter by name

Language

Name

Expl

New project



### Create blank project

Create a blank project to store your files, plan your work, and collaborate on code, among

Namen für  
Repository vergeben

Project name

GINF2024

Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, d

Project URL

https://its-git.fh-salzburg.ac.at/ thomas.schmuck

Project slug

ginf2024

Want to organize several dependent projects under the same namespace? [Create a group.](#)

Visibility Level ?

☒ Private

Project access must be granted explicitly to each user. If this project is part of a group, access is grante

☐ Internal

The project can be accessed by any logged in user except external users.

☐ Public

The project can be accessed without any authentication.

Project Configuration

☒ Initialize repository with a README

Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing rep

☐ Enable Static Application Security Testing (SAST)

Analyze your source code for known security vulnerabilities. [Learn more.](#)

Eigenen Usernamen  
wählen

Mit README  
initialisieren

Create project

Cancel

# Versionsverwaltung – git



**GINF2024**  
Project ID: 6519

Star 0 Forks 0

1 Commit 1 Branch 0 Tags 3 KiB Project Storage

**Initial commit**  
Thomas Schmuck authored just now

8e6c853f

main

ginf2024 /

History

Find file

Edit

Clone

README Add LICENSE Add CHANGELOG Add CONTRIBUTING Add Kubernetes cluster Set up C

Configure Integrations

README.md

GINF2024

Clone with SSH

git@its-git.fh-salzburg.ac.at:tl

Clone with HTTPS

https://its-git.fh-salzburg.ac.

Open in your IDE

Visual Studio Code (SSH)

Visual Studio Code (HTTPS)

IntelliJ IDEA (SSH)

Repository Inhalt

Repository Pfad

# git Kommandos



- Konfiguration von Usernamen und E-Mail

```
git config --global user.name
```

```
"[Username]"
```

```
git config --global user.email "[E-Mail]"
```

```
thsch@Notebook MINGW64 ~/Desktop/git test/ginf2024 (main)
$ git config --global user.email "thomas.schmuck@fh-salzburg.ac.at"

thsch@Notebook MINGW64 ~/Desktop/git test/ginf2024 (main)
$ git config --global user.name "Harry"
```

# git Kommandos



- Kopieren des remote Repositories auf Rechner

`git clone [Pfad zu Repository]`

```
thsch@Notebook MINGW64 ~/Desktop/git test
$ git clone git@its-git.fh-salzburg.ac.at:thomas.schmuck/ginf2024.git
Cloning into 'ginf2024'...
The authenticity of host 'its-git.fh-salzburg.ac.at (193.170.193.10)' can't be e
stablished.
RSA key fingerprint is SHA256:iBD/wlV7jswGtugHQ3DS8jvOAJn+wkkFOQdMEUHAtnY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'its-git.fh-salzburg.ac.at' (RSA) to the list of know
n hosts.
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

# git Kommandos



- Zum Wechseln in das Repository

`cd [Repository Name]`

```
thsch@Notebook MINGW64 ~/Desktop/git test
$ cd ginf2024/

thsch@Notebook MINGW64 ~/Desktop/git test/gin2024 (main)
$ |
```

# git Kommandos



- Geänderte Dateien hinzufügen

`git add [Datei Namen] (* für alle)`

```
thsch@Notebook MINGW64 ~/Desktop/git test/gin2024 (main)
$ git add *
```

# git Kommandos



- Zum Hinzufügen einer zuvor hinzugefügten Änderung als Meilenstein

```
git commit -m "[Commit Message]"
```

```
thsch@Notebook MINGW64 ~/Desktop/git test/ginf2024 (main)
$ git commit -m "First Test Commit"
[main 5d6ed21] First Test Commit
1 file changed, 7 insertions(+)
create mode 100644 test.c
```



# git Kommandos



- Hochladen der Änderungen zu remote Repository

git push

```
thsch@Notebook MINGW64 ~/Desktop/git test/ginff2024 (main)
$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 394 bytes | 394.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To its-git.fh-salzburg.ac.at:thomas.schmuck/ginff2024.git
   8e6c853..5d6ed21  main -> main
```

# git Kommandos



- Herunterladen von Änderungen vom remote Repository

`git pull`

```
thsch@Notebook MINGW64 ~/Desktop/git test/gin2024 (main)
$ git pull
Already up to date.
```

# Best Practices



- Commit Messages: kurze und aussagekräftige Beschreibungen von Änderungen
- Regelmäßige Commits: kleine, häufige Commits für leichtere Nachvollziehbarkeit → eine abgeschlossene, selbstständige Änderung ist ein Commit
- Konfliktmanagement: frühzeitig lösen für einfacheren Integrationsprozess