Zürcher Hochschule
für Angewandte Wissenschaften

School of
Management and Law

AACSB ACCREDITED
Seit 2015

DevOps
Versionskontrolle

Building Competence. Crossing Borders.

Adrian Moser
mosa@zhaw.ch, FS2024

Agile Software Development

DevOps benötigt zwingend ein agiles Software Development

Agile Software Development Cycle

Meet & Plan

Was soll in der nächsten Phase gemacht werden?

Design

Wie soll es gemacht werden?

Develop

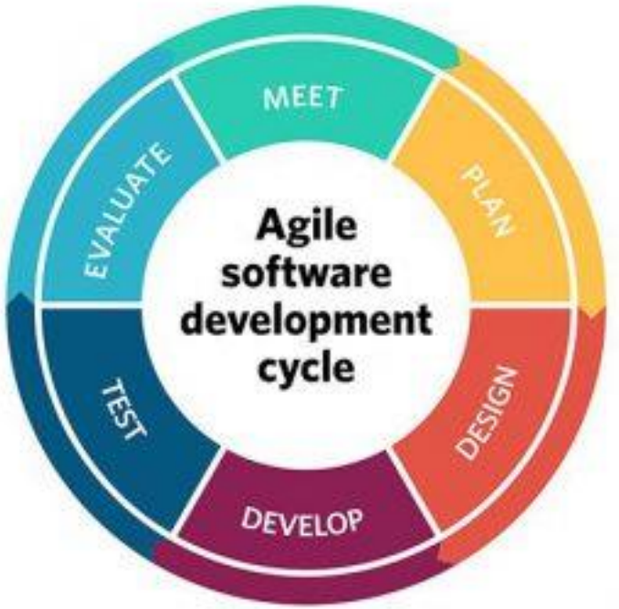
Umsetzung

Test

Mit Endbenutzern

Evaluate

Ergebnisse auswerten und in neue Planung einfliessen lassen



Quelle: https://project-management.com/10-key-principles-of-agile-software-development/

Umgang mit Änderungen

Testgetriebene Entwicklung (TDD, Test-Driven Development)
- Viele Änderungen werden erwartet
- Software muss dennoch stabil sein

Refactoring
- Änderungen sauber durchziehen, ohne Angst vor Fehlern

Code Reviews
- führt zu besserem Code

Verteilte Versionskontrolle des Source Codes
- Nachvollziehbarkeit
- Parallele Entwicklung verschiedener Versionen

Versionskontrolle

Themen:

- Aufgabe von Versionskontrollsystemen
- Zentrale und verteilte Versionskontrollsysteme
- Worflow und Begriffe bei git
- GitHub verwenden

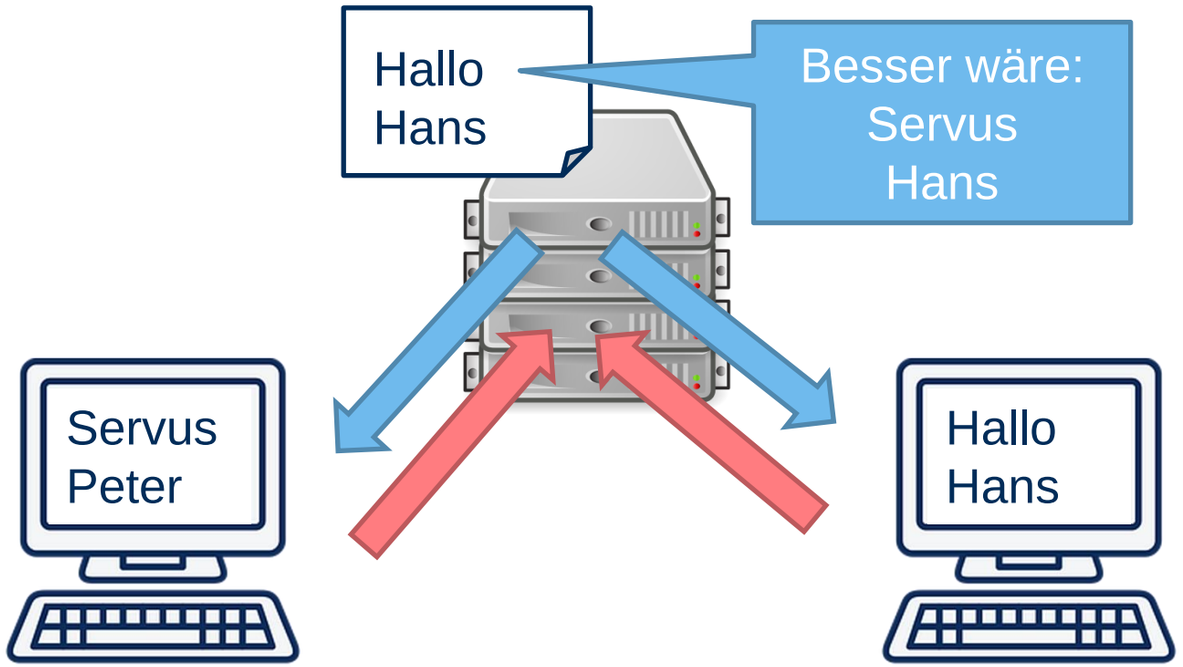
Versionskontrollsysteme

Versionskontrollsysteme (VCS – Version Control System) beantworten die Fragen:

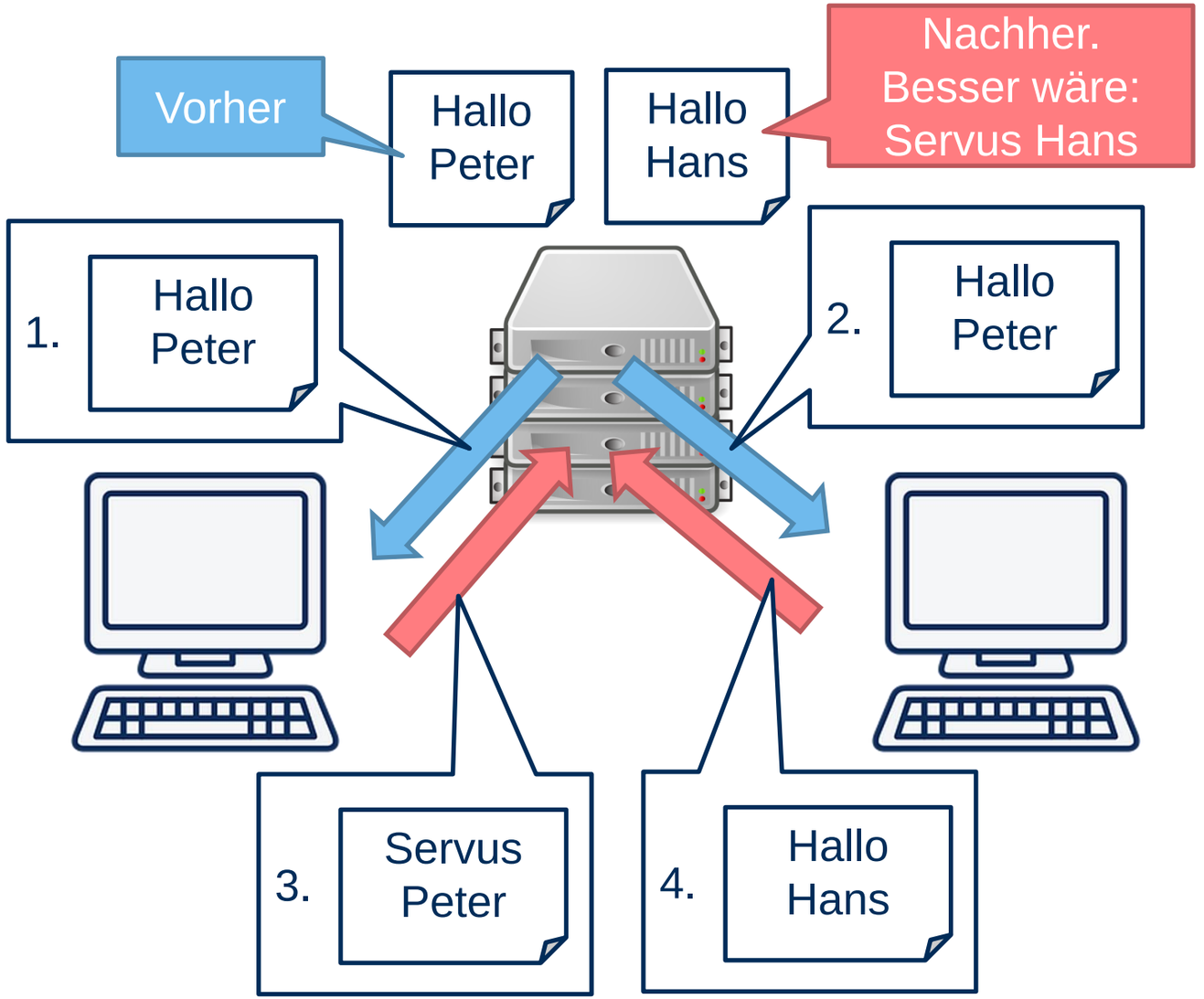
Was wurde **wann** von **wem** geändert?

Dabei speichert das VCS nicht nur die letzte Version einer Datei, sondern deren **gesamte Geschichte**. Dies erlaubt es Änderungen rückgängig zu machen und vereinfacht es Änderungen, die von mehreren Personen parallel gemacht wurden, zusammenzuführen.

Versionskontrollsysteme: Motivation

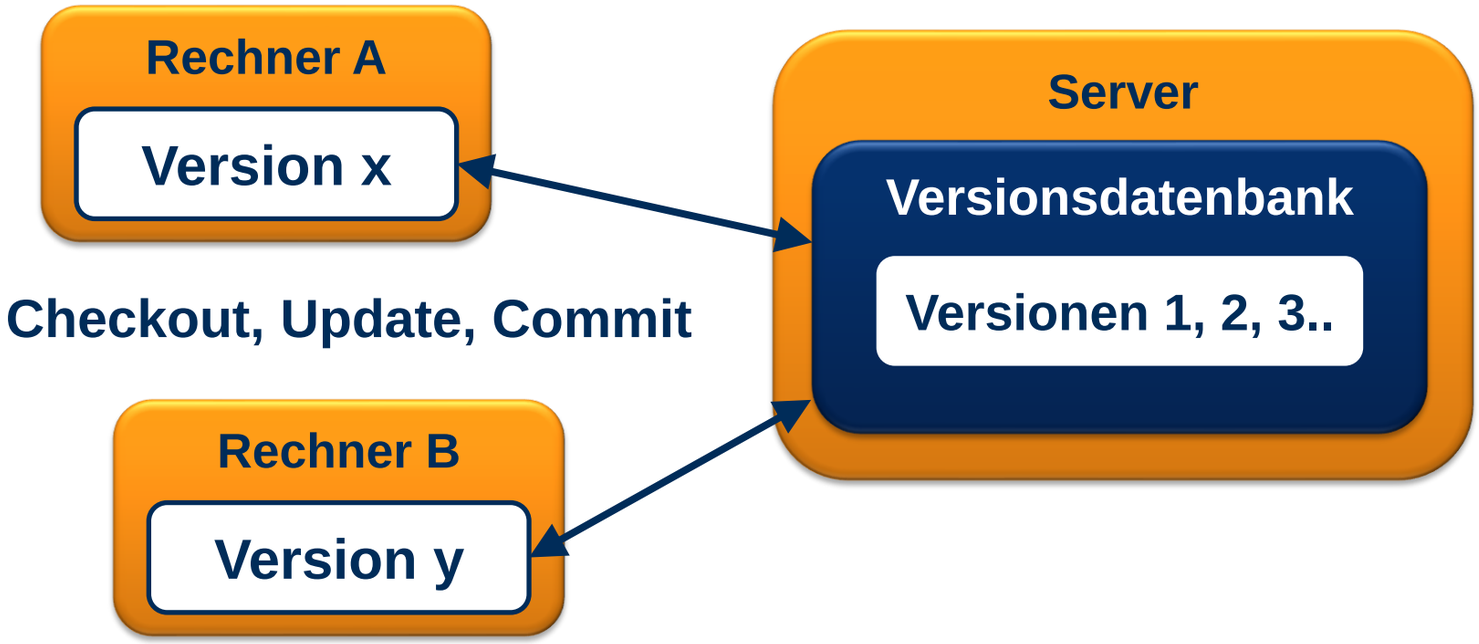


Versionskontrollsysteme: Motivation



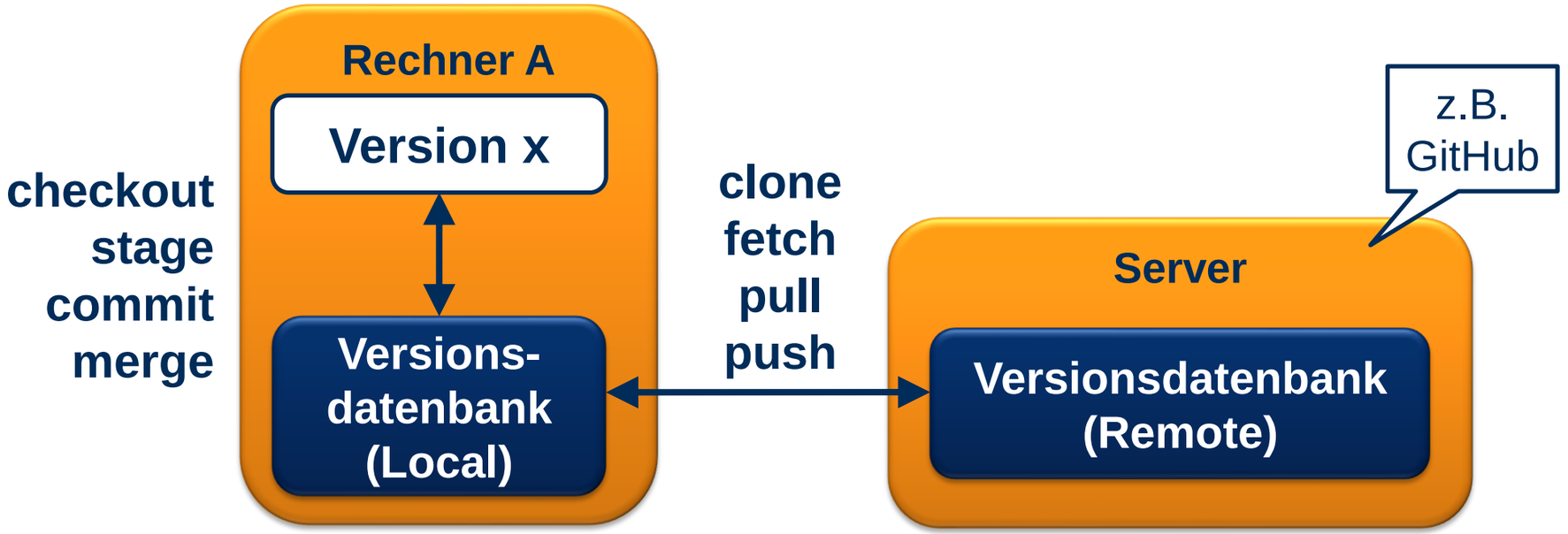
Zentrale Versionskontrollsysteme

Zentrale Versionskontrollsysteme haben einen zentralen Server mit der Versions-datenbank (Repository).
Beispiele: CVS (uralt) und Subversion (kurz SVN).

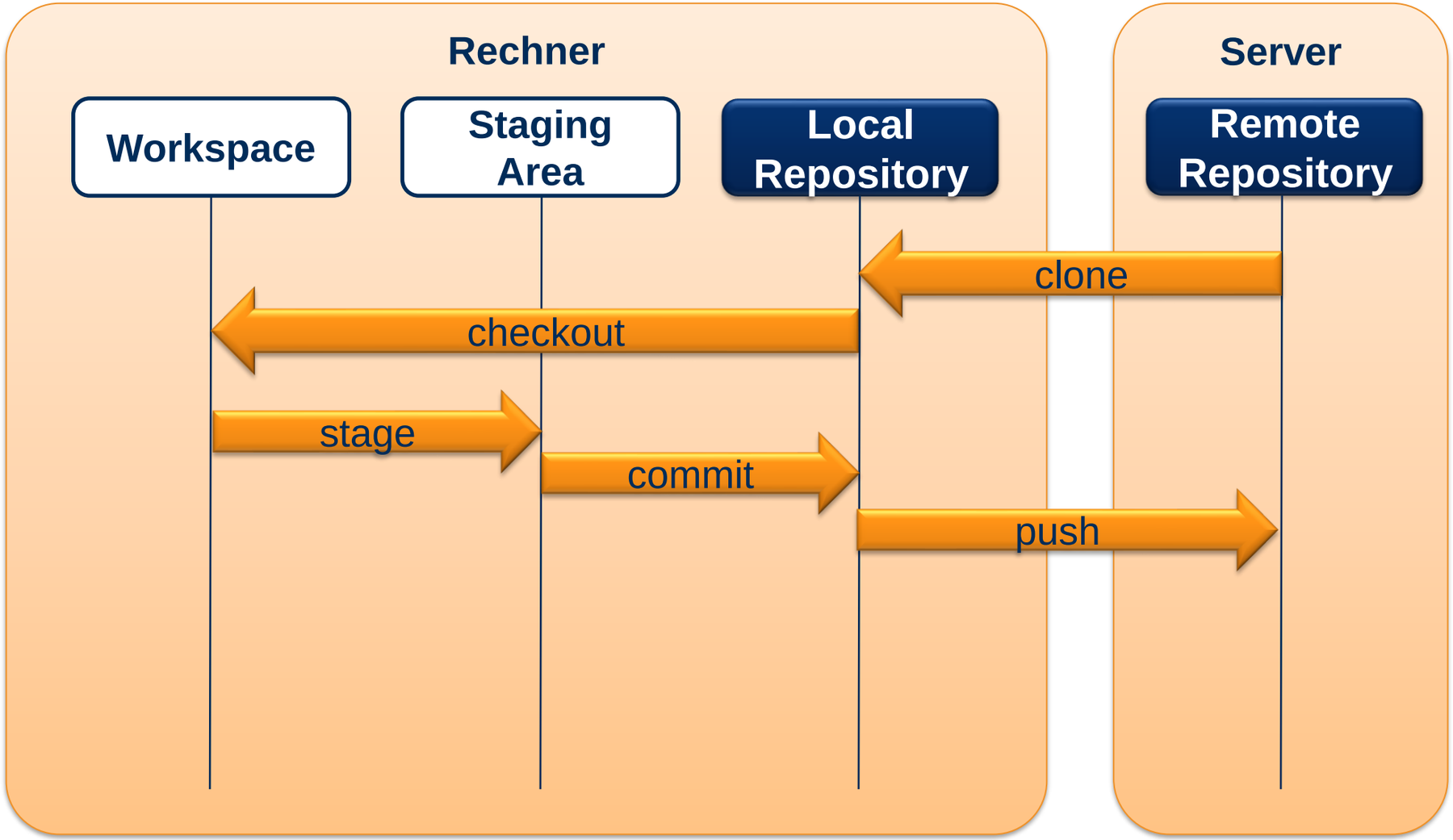


Verteilte Versionskontrollsysteme

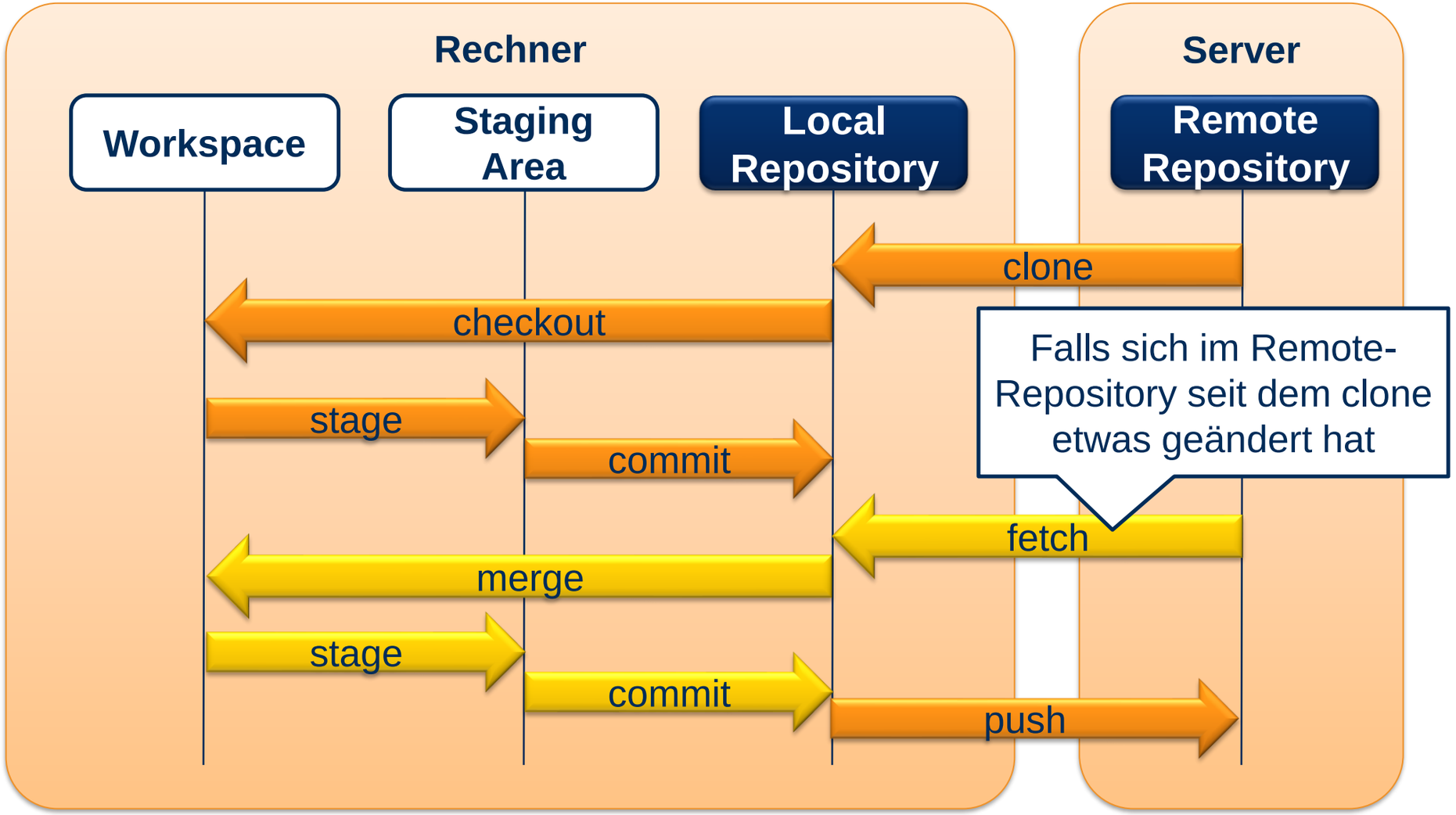
Verteilte Versionskontrollsysteme haben in der Regel ebenfalls einen zentralen Server, allerdings wird nicht nur eine einzelne Version, sondern das ganze Repository lokal gespeichert. Damit kann auch ohne Serveranbindung mit der Versionskontrolle gearbeitet werden. Bei Bedarf können die Änderungen aus der lokalen Datenbank auf den Server übertragen resp. Änderungen vom Server in die lokale DB geholt werden. Beispiele: Mercurial (Bedeutung abnehmend) und Git (der Standard).



Git-Workflow (kein Konflikt)



Git-Workflow (Konflikt)

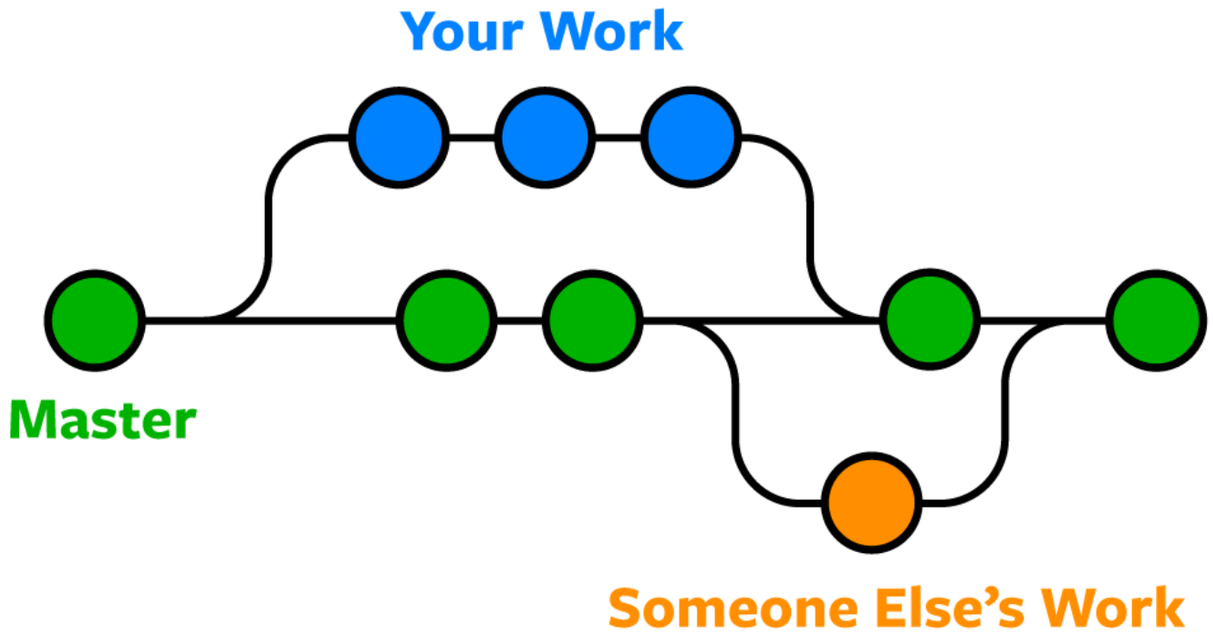


Git: Begriffe (Auswahl)

- **Clone**: Macht eine lokale Kopie eines Repositories. In der Regel wird das Repository von einem Server wie z.B. von GitHub geholt.
- **Checkout**: Im Repository kann man nicht direkt Programmcode ändern oder ausführen. Mit checkout extrahiert man eine bestimmte Version in einen separaten Ordner (Workspace) um damit zu arbeiten.
- **Stage**: Beim Stagen werden im Workspace geänderte Dateien oder Teile davon für den nächsten Commit vorgemerkt (markiert).
- **Commit**: Beim Commit werden alle gestageten Daten ins lokale Repository übertragen. Es entsteht eine neue Version, die all diese Änderungen enthält.
- **Fetch**: Aktualisiert das lokale Repository.
- **Merge**: Kombiniert die Version im Workspace mit einer (in der Regel neueren) Version aus dem lokalen Repository. →Eventuell manuelle Konfliktauflösung.
- **Pull**: Kombiniert Fetch und Merge (in der Abbildung nicht dargestellt).
- **Push**: Überträgt Änderungen von lokalen ins Remote-Repository.

Git Konzepte: Branch

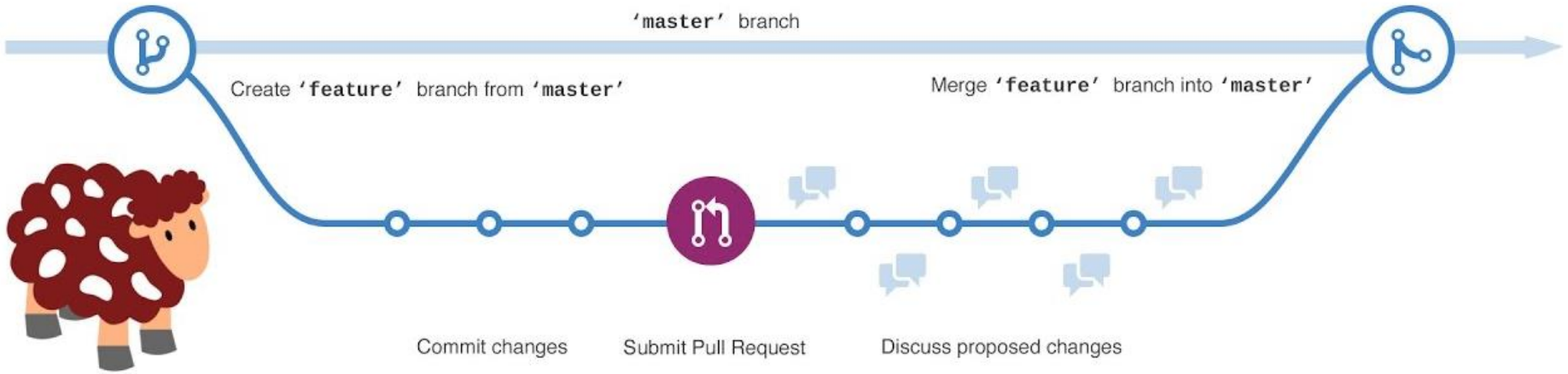
Mit einem Git **Branch** kann an einem neuen Feature, einem Bug-Fix oder auch nur an einer neuen Idee gearbeitet werden. Der Haupt-Branch heisst **Master** oder **Main**.



Quelle: https://www.nobledesktop.com/learn/git/git-branches

Git Pull Request (Merge Request)

Mit einem **Pull Request** wird eine Code-Änderung (neues Feature oder Bugfix) in den Haupt-Branch zurückgespielt. Dabei können Tests gemacht werden und/oder die vorgeschlagene Änderung auch diskutiert werden.



Quelle: https://www.youtube.com/watch?v=hSbJaIdqwKg

Lernjournal

Lernjournal «Versionskontrolle»

Dieses Lernjournal muss **in Gruppen von zwei Studierenden** gemacht werden.

Ziele

- Git als Beispiel für Versionskontrolle: Verstehen und anwenden können
- GitHub.com und Visual Studio Code (VS Code) kennenlernen
- Lokale Änderungen auf GitHub übertragen (direkt und mittels Branch/Pull Request)
- Jeder Studierende holt sich wieder aktuellen Stand (fetch)
- Thema «GitHub Pull Requests» / «Branches» eigenständig in Gruppe vertiefen

Checkliste

- ✓ Es existieren zwei Repository, mit gegenseitigem Zugriff (jeweils Owner und Collaborator)
- ✓ Abwechslungsweise Commits durch beide Studierende mit Visual Studio Code vorhanden
- ✓ mehrere Dateien und Zeilen, welche abwechslungsweise geändert/hinzugefügt wurden, vorhanden
- ✓ Branches und (abgeschlossene, diskutierte) Pull Requests von jedem Studierenden
- ✓ Beschreibung Zusammenarbeit in der Gruppe

GitHub: Zwei Repositories erstellen

Journal

GitHub.com

- GitHub Konto erstellt und bestätigt
- Neues Repository auf GitHub erstellen
  (1 Repository pro Gruppe)
- Teammitglieder hinzufügen

Visual Studio Code

- Visual Studio Code installiert
- Repository klonen
- Anpassungen **lokal** vornehmen
  commit & push
- Anpassungen fetchen

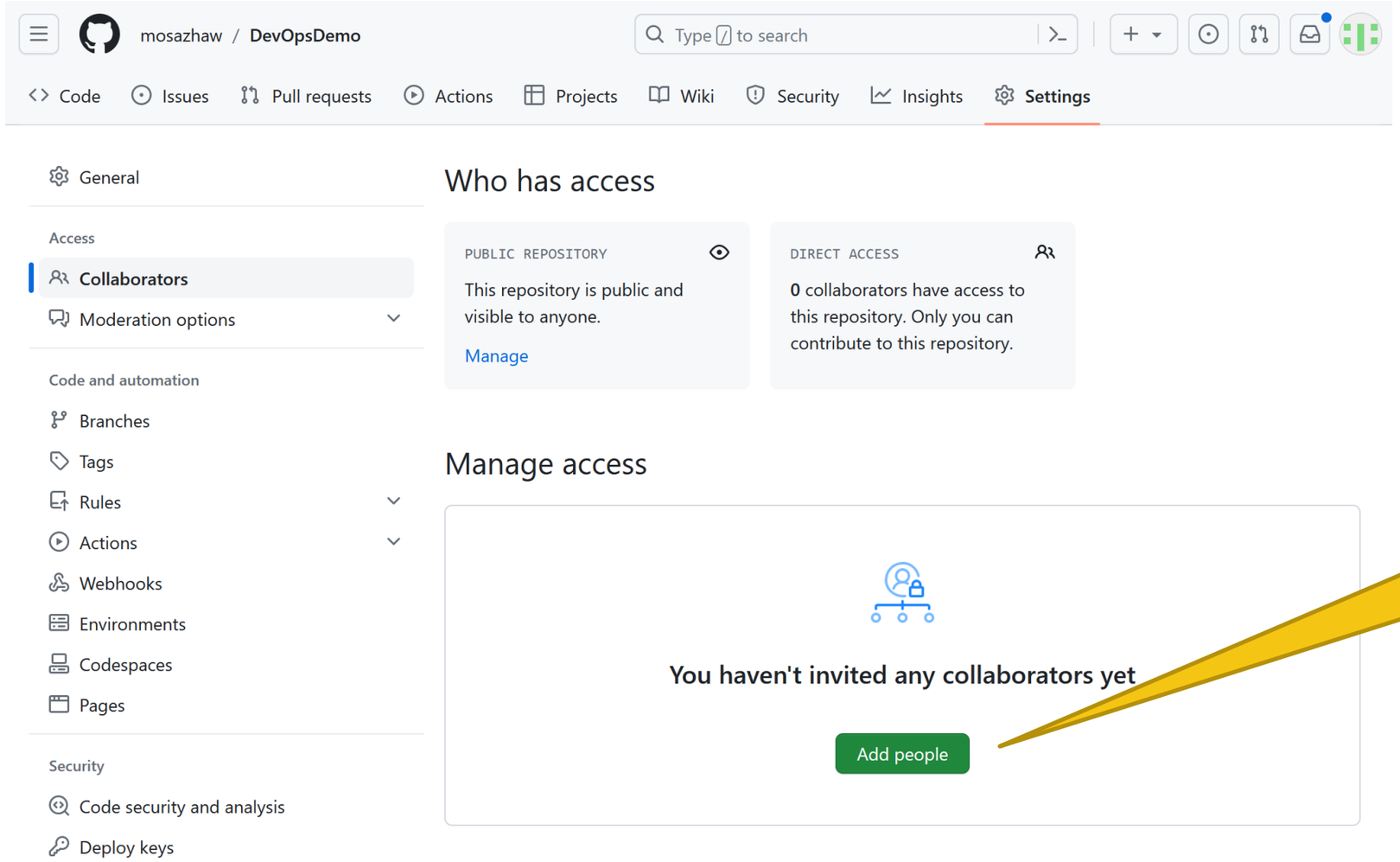
Datei README.md anpassen

- Jeder Studierende trägt z.B. seinen Vornamen selbst ein
- jedes Teammitglied soll mehrere Commits machen (Achtung Konflikte!)

Achtung: Zu Beginn Konflikte vermeiden (immer fetch vor Änderung).

GitHub: Collaborator

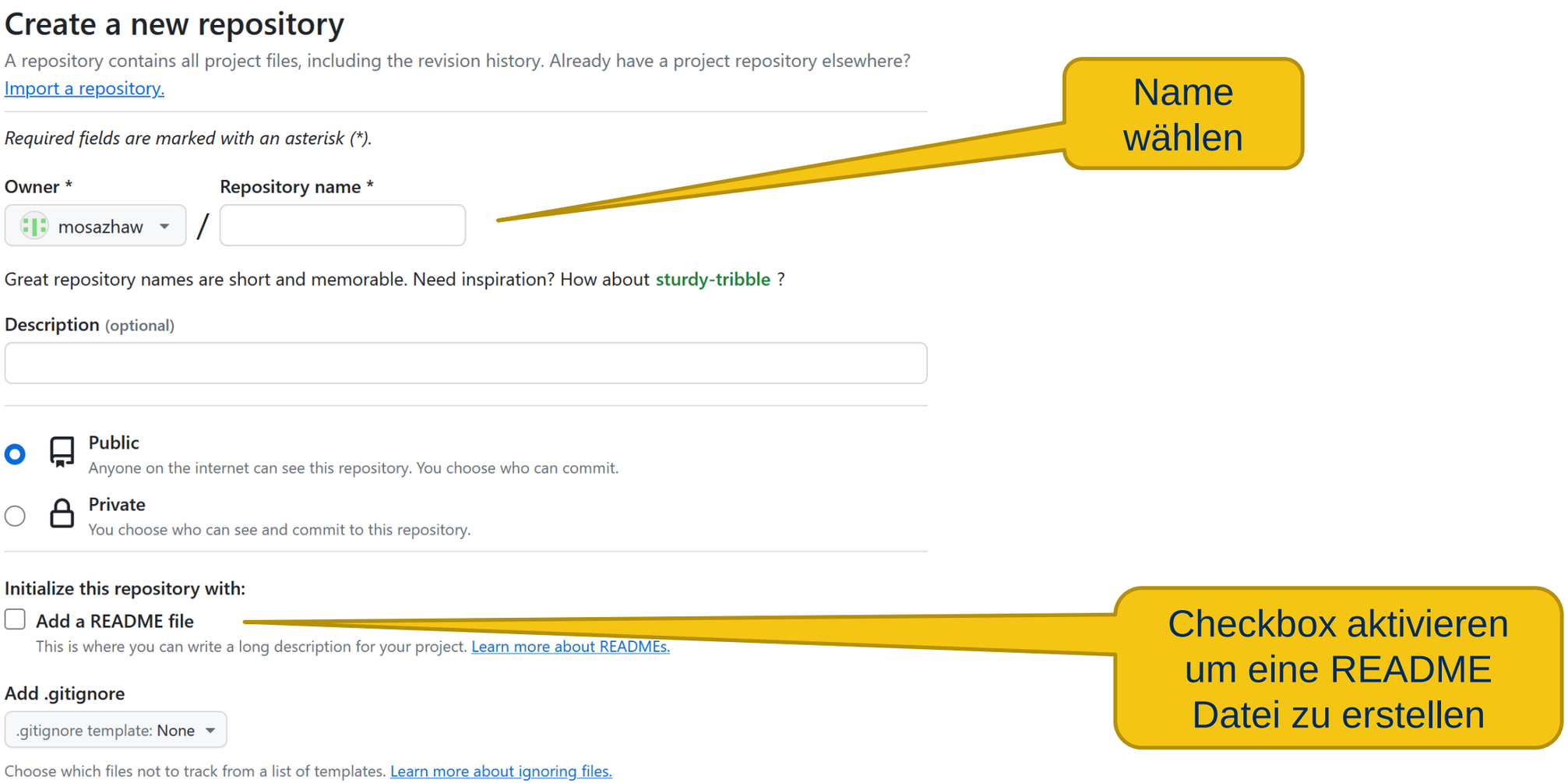
Journal



Mit-Bearbeiter als Collaborator hinzufügen

Hilfe: Neues Repository auf GitHub erstellen

Journal

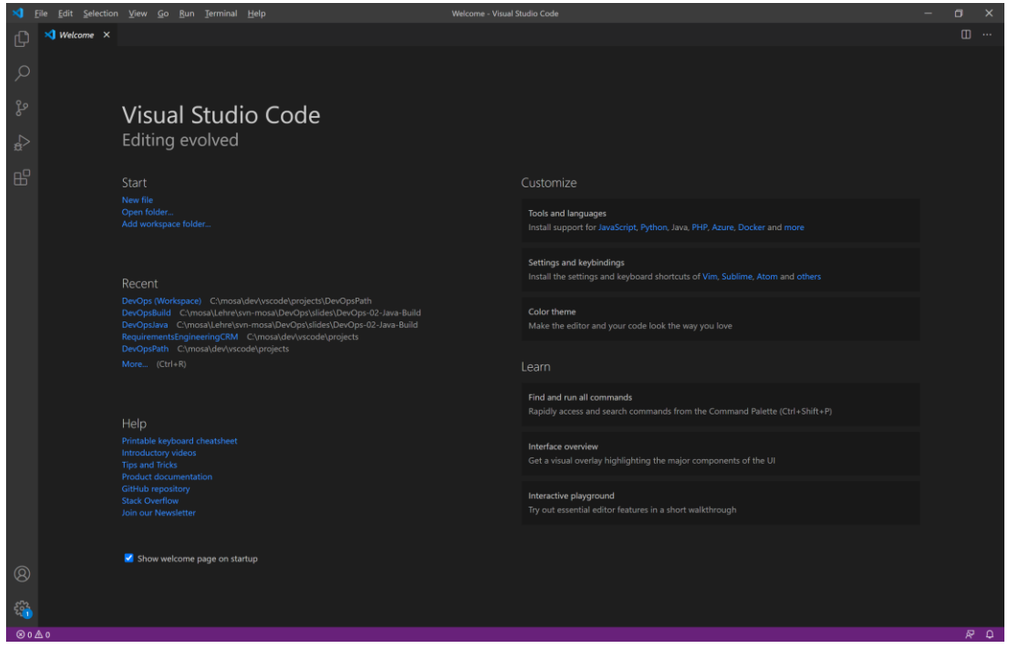


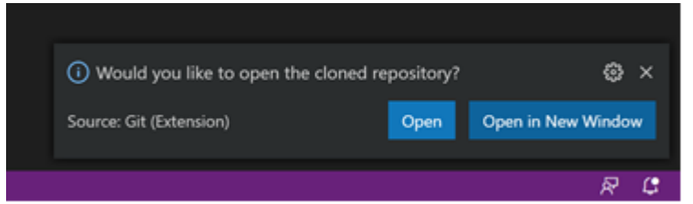
Hilfe: Repository in Visual Studio Code klonen

Journal

Visual Studio Code

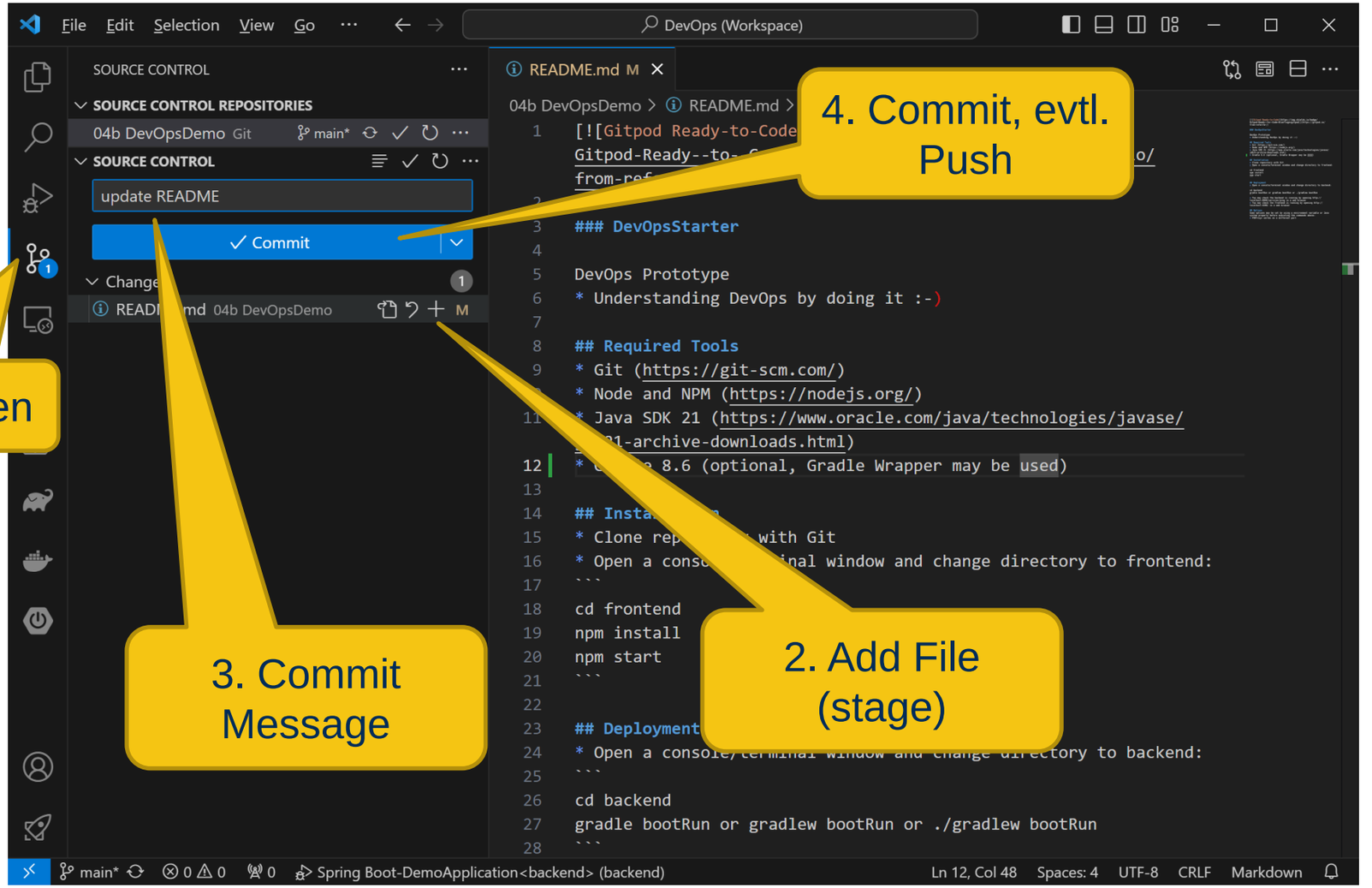
- Ctrl – Shift – P
- Git: Clone
- Clone from GitHub
- Repository auswählen und öffnen





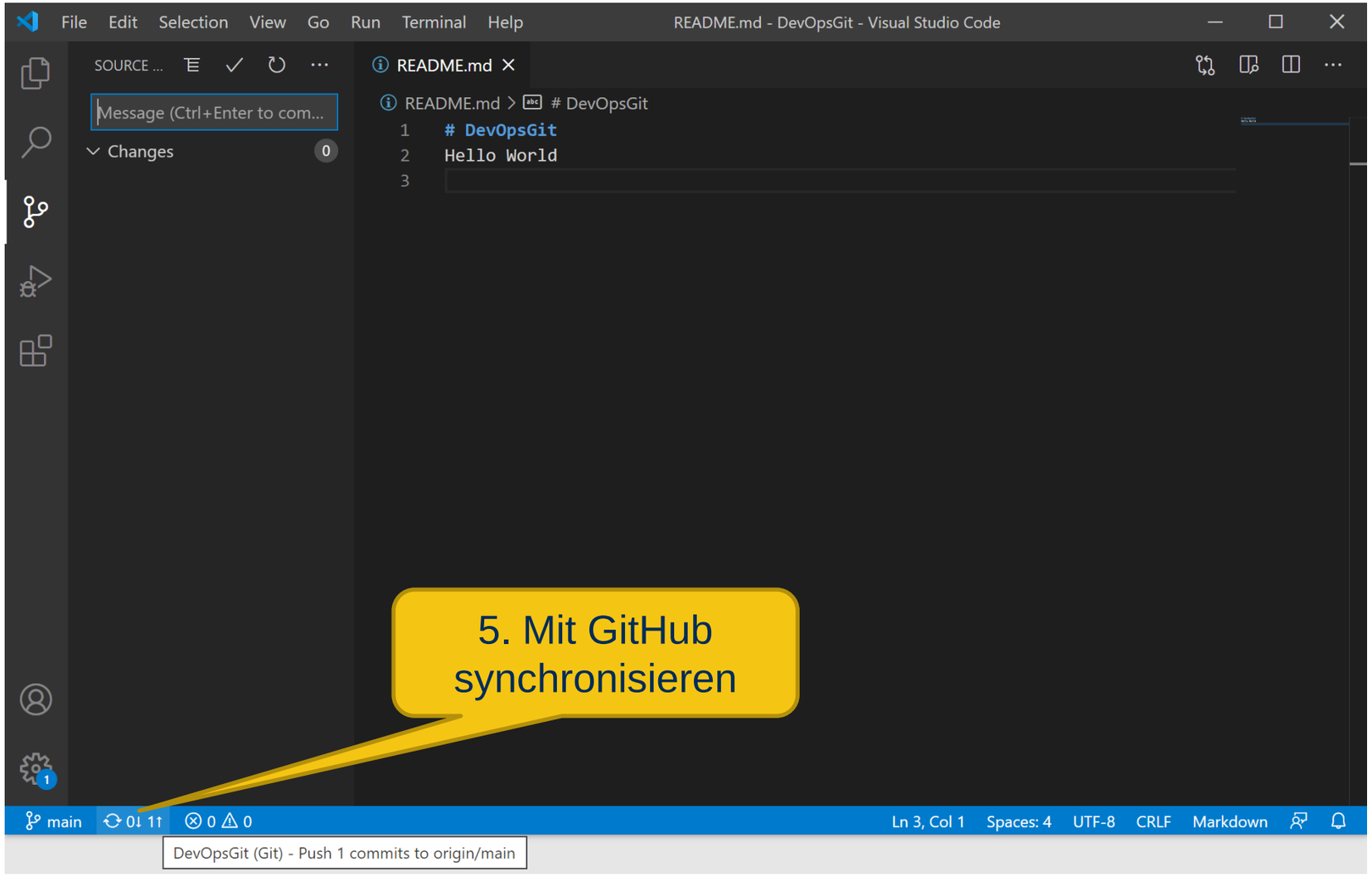
Hilfe: Visual Studio Code-Bedienung «Commit»

Journal



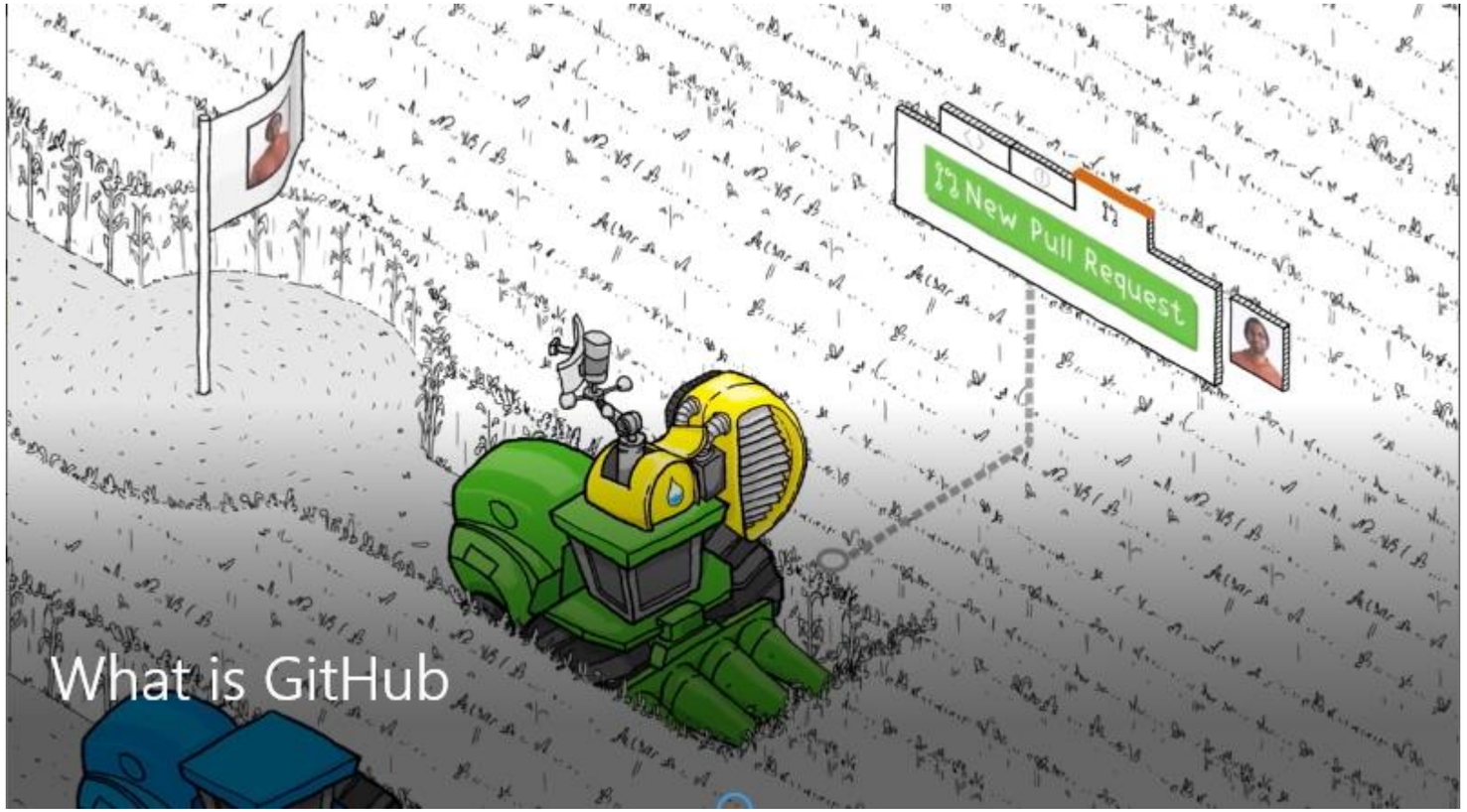
Hilfe: Visual Studio Code-Bedienung «Pull und Push»

Journal



Hilfe: GitHub Pull Requests

Journal



https://www.youtube.com/watch?v=w3jLJU7DT5E