**Git**

Ziele der Versionskontrolle

1. Eindeutigkeit bzgl. der aktuellen Version

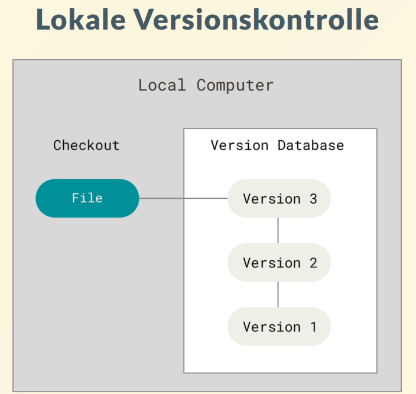
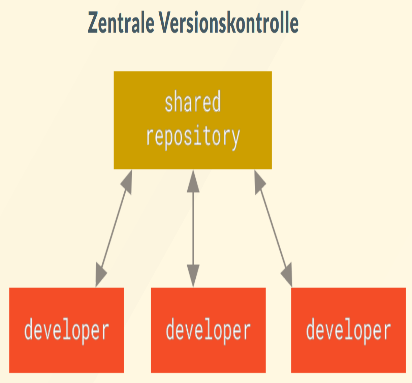
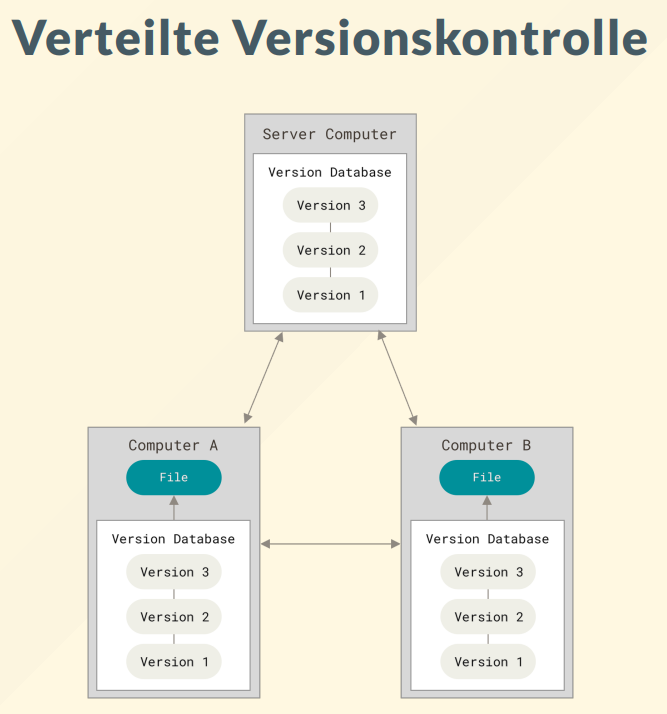
2. Austausch von Daten

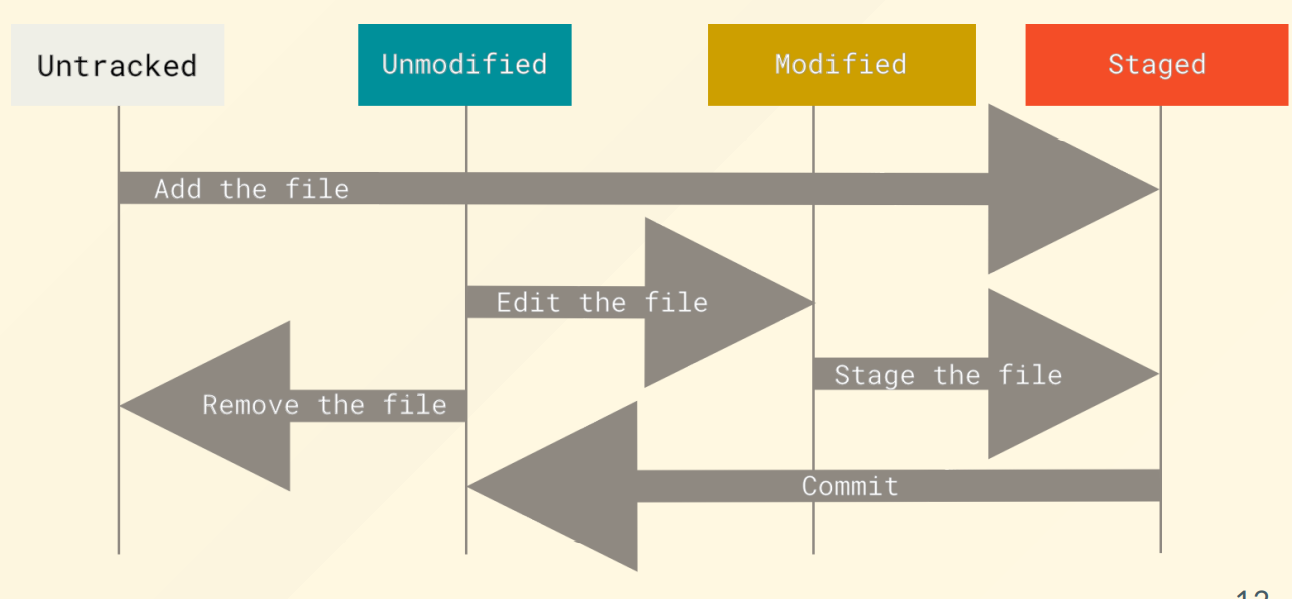
3. Datensicherung

4. Nachvollziehbarkeit

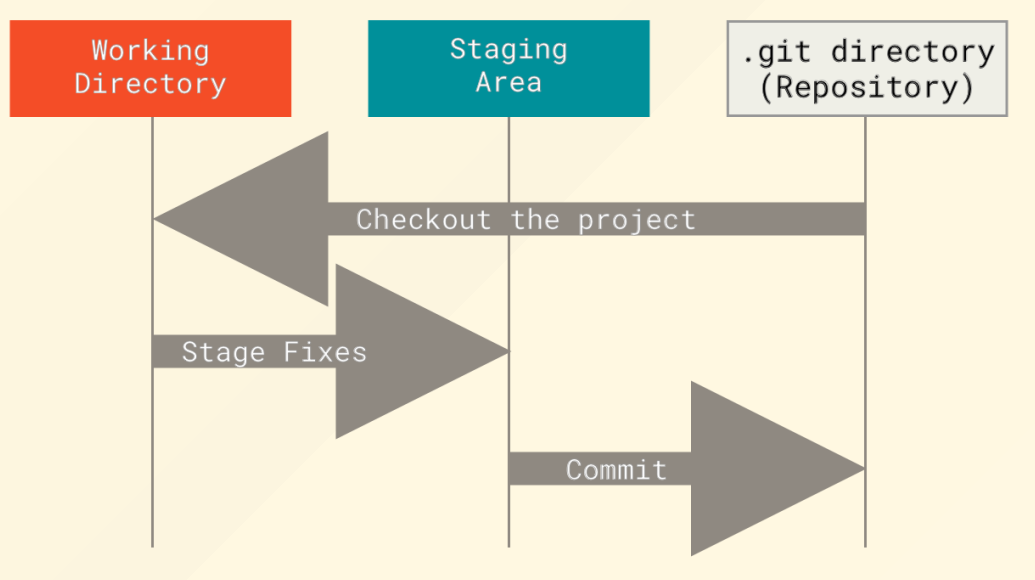
5. Wiederherstellung eines Softwarestande

Arten Versionskontrolle



Lifecycle einer Datei

Arbeitsbereiche in GIT



Branching

In Git gibt es Branches, die auf Commits bzw. Snapshots verweisen

Eine Branch verweist auf einen Commit

Auf einen Commit kann von mehreren Branches gleichzeitig verwießen werden

Die momentan aktive Branch wird durch den HEAD bestimmt

Git-Befehle

git-config ermöglicht Konfiguration (lokal und global)

git init Initialisiert ein neues Repository im angegebenen Ordner

git status gibt den Status des aktuellen Repositories an

git add fügt Datei dem Staging Bereich der Versionskontrolle hinzu

git commit Sichert aktuellen Stand des Staging Bereichs ins Repository (eindeutige ID) Staging Bereich wird geleert

git remote Ermöglicht den Umgang mit Remote Repositories, fügt alias hinzu

git push aktualisier das angegebene Repository

git clone Erstellt eine lokale Kopie eines bestehenden Repositories in einem neuen Verzeichnis

git checkout Ermöglicht es, einen Branch auszuchecken oder neue zu erstellen

git merge Fügt Commit Historien zusammen. So können z.B. Branches zusammengeführt werden

git rebase wendet die Veränderung eines Commits auf einen anderen Softwarestand an

git fetch Empfängt Informationen aus dem Remote Repository. Folgenden Entitäten (refs) werden abgeholt: Branch, Tag und Commit