01 Git und GitLab

Einführung in die moderne (Full-Stack) Webentwicklung mit JavaScript, Node.js und React.js

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Universität Regensburg

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Was ist Git?

- Git ist ein Versionskontrollsystem, das es ermöglicht, Änderungen an Dateien und Codeprojekten zu verfolgen, zu verwalten und zu teilen.
- Git bietet eine Möglichkeit für mehrere Personen, an einem Projekt zusammenzuarbeiten und Änderungen auf einfache und effiziente Weise zu verwalten.
- Git ist plattformunabhängig und kann auf verschiedenen Betriebssystemen wie Windows, macOS und Linux verwendet werden.
- Git wurde im Jahr 2005 von Linus Torvalds entwickelt, dem Schöpfer des Linux-Kernels.

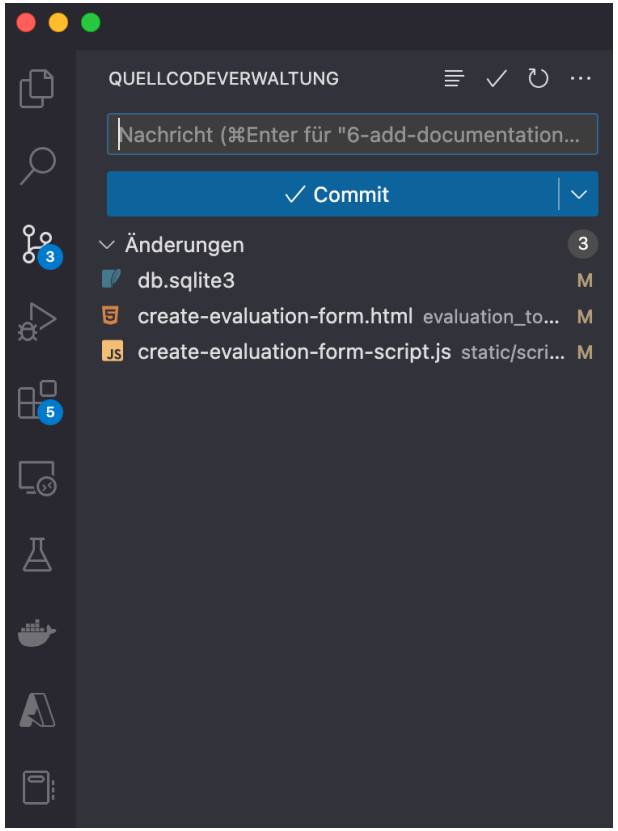
Universität Regensburg

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Was ist Git?

- Git kann sowohl über die Kommandozeile als auch über eine grafische Benutzeroberfläche (GUI) verwendet werden
- Einige der populären Git-GUI-Tools sind GitHub Desktop, Sourcetree, GitKraken, etc.
- Auch in VSC gibt es eine integrierte Git-Verwaltung mit GUI

```
nils_hellwig@MacBook-Pro-von-Nils edu-quality-eval % git status
On branch 6-add-documentation-to-start-page
Your branch is up to date with 'origin/6-add-documentation-to-start-page'.

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:   db.sqlite3
        modified:   evaluation_tool/templates/evaluation_tool/components/create-evaluation-form.html
        modified:   static/scripts/create-evaluation-form-script.js
```



Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Git mit der Kommandozeile verwenden

- Um Git zu verwenden, öffnen wir zunächst die Kommandozeile
- Als Erstes können wir prüfen, ob Git richtig installiert wurde:

```
git --version
>> git version 2.30.2.windows.1
```

Universität Regensburg

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Erstellen eines Ordners mit Git-Versionskontrolle

- Ist man mit der Kommandozeile in dem Ordner, in dem man Git initialisieren möchte, so kann man dies mit `git init`
- Damit wird ein Verzeichnis mit Git-Versionskontrolle initialisiert, sodass man die Versionsgeschichte des Codes verfolgen, verschiedene Versionen wiederherstellen oder Änderungen vergleichen kann
- Das Verzeichnis ist dadurch aber noch nicht für andere Programmierer*innen zugänglich, sondern nur auf dem eigenen Gerät
- Um an dem Verzeichnis kollaborativ zu arbeiten, benötigt man ein **Remote-Repository**

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Remote Repository am Beispiel GitLab

- Ein Remote-Repository kann bei Diensten wie GitHub, GitLab oder BitBucket gehostet werden
- Um ein Remote-Repository in GitLab zu erstellen, muss man sich zunächst bei einem GitLab anmelden und ein neues Projekt erstellen
- Nachdem das Remote-Repository erstellt wurde, kann man es mithilfe von `git clone` auf den eigenen Computer herunterladen bzw. "clonen"
- Wenn mehrere Entwickler*innen an einem Projekt arbeiten, können alle eine Kopie des Remote-Repositories auf ihren eigenen Computer herunterladen und lokal Änderungen vornehmen
- Um ihre Änderungen mit anderen Entwickler*innen zu teilen, müssen sie ihre Änderungen auf das Remote-Repository auf dem GitLab Server pushen

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Dateien zu einem Git-Repository hinzufügen

- Erstellen wir nun eine Datei (z.B. test.html) in unserem Arbeitsverzeichnis
- Mithilfe von `git status` können wir prüfen, ob die Datei im Staging Environment ist
- Wenn eine Datei im Arbeitsverzeichnis ist, aber nicht zum Staging-Environment hinzugefügt wurde, wird sie als *untracked* bezeichnet
- Wenn eine Datei zum Staging Environment hinzugefügt wurde, gilt diese als *tracked*
- Dateien die *untracked* sind und in das Repository aufgenommen werden sollen, müssen *gestaged* werden (in das Staging Environment aufgenommen werden)

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Git Staging Environment

- Das Staging-Environment, auch Index oder Zwischenablage genannt, ist ein Bereich in Git, in dem Dateien vor dem Committing vorbereitet werden
- Immer, wenn man einen Meilenstein erreichen oder einen Teil der Arbeit abschließen, sollten die Dateien zum Staging Environment hinzugefügt werden
- Dateien können zum Staging Environment hinzugefügt werden mithilfe von
  `git add <Dateiname/Verzeichnisname>`
- Es können auch mehrere Dateien auf einmal committet werden, beispielsweise mit `git add .` können alle Dateien hinzugefügt werden im Verzeichnis, in dem man sich gerade in der Kommandozeile befindet
- Wenn man mit den Änderungen zufrieden ist, kann man sie schließlich committen

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Git Staging Environment

- Um eine Datei aus dem Staging Environment zu entfernen, kann "`git reset HEAD -- <Dateiname/Verzeichnisname>`" verwendet werden
- Führt man "`git reset HEAD -- .`", dann werden alle Dateien aus dem Staging Environment entfernt, die sich in dem Verzeichnis befinden, in dem man gerade in der Kommandozeile ist

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Git Commit

- Nachdem wir unsere Arbeit abgeschlossen haben, können wir die Dateien im Staging Environment in unser Repository committen
- Zu jedem Commit sollte immer eine Nachricht hinzugefügt werden, der die darin enthaltene Änderung beschreibt
- Nachricht: Kurze und prägnante Beschreibung, wozu der Commit dient
- Durch das Hinzufügen einer Beschreibung zu jedem Commit ist es für Sie (und andere Entwickler*innen) einfach zu sehen, was wann geändert wurde
- Aktiver statt passiver Sprachstil, z.B. „add device list“ oder „fix authentication bug“

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

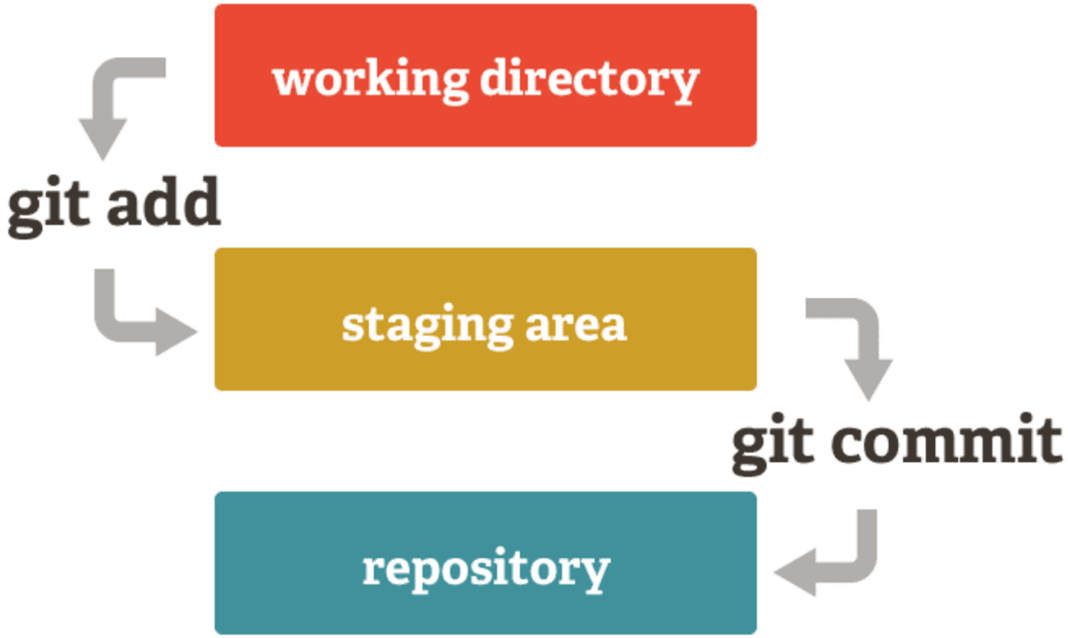
Git Commit

- Ein Commit kann mittels `git commit -m <Nachricht>` erstellt werden
- Um den Verlauf der Commits für ein Repository anzuzeigen, kann `git log` ausgeführt werden

```
nils_hellwig@MacBook-Pro-von-Nils git-test % git log
commit 2e6802bea6070e9fb8f065be78e9906486c94b6c (HEAD -> main)
Author: NilsHellwig <nc.hellwig@gmail.com>
Date:   Sat Mar 4 11:28:29 2023 +0100

    add directory with test files

commit 5b61e4fa346eb4704fb89d9736cc6f8c27b960ea
Author: NilsHellwig <nc.hellwig@gmail.com>
Date:   Sat Mar 4 11:28:08 2023 +0100

    add test.html
```

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Git Summary



Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

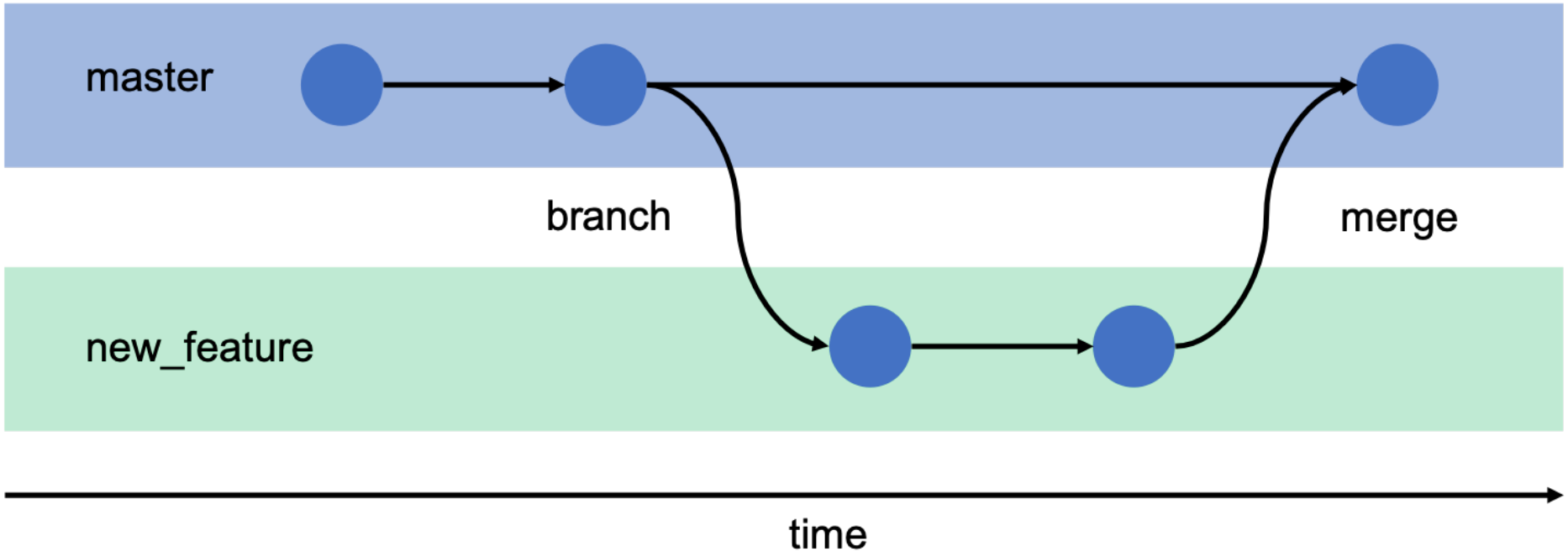
Git Branches

- In Git ist ein Branch eine neue/getrennte Version des Haupt-Repositorys (main-Branch)
- Jeder Branch kann unabhängig voneinander entwickelt und verändert werden, ohne dass dies den ursprünglichen Code beeinflusst
- Durch das Erstellen von Branches können verschiedene Versionen des Codes gleichzeitig existieren, was es Entwickler*innen ermöglicht, an unterschiedlichen Features oder Bugfixes zu arbeiten, ohne dass dies die Arbeit anderer Entwickler*innen behindert
- Wenn ein Branch fertiggestellt ist, kann er in den main-Branch integriert werden
- Dadurch werden die Änderungen in den Hauptcode übernommen

Universität Regensburg

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Git Branches



Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Git Branches

- Einen neuen Branch erstellt man mit `git branch <Name des Branches>`
- Alle verfügbaren Branches sowie der Branch in dem man aktuell arbeitet können mit `git branch` angezeigt werden
- Zu einem anderen Branch kann gewechselt werden mithilfe von `git checkout <Name des Branches>`

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Git Branch Merge

- Nachdem man mit der Implementierung in einem Branch fertig ist, kann dieser in den main-Branch überführt werden (oder ggf. in einen anderen Branch)
- Dafür muss man zunächst in den Branch wechseln (`git checkout`), in den der Branch an dem man gearbeitet hat überführt werden soll, also z.B. der main-Branch
- Das Merging kann dann mittels `git merge` ausgeführt werden
- Optional kann dann der überführte Branch gelöscht werden mithilfe von

  `git delete <Name des Branches>`

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Git: Neue Änderungen seit dem letzten Commit verwerfen

- Wurde eine Änderung noch nicht committet und ist noch nicht im Staging Environment:

  `git clean -f <Dateiname/Verzeichnisname>`

  Achtung!: Änderungen, die man mit diesem Kommando verwirft, sind nicht gespeichert und können nachträglich nicht wiederhergestellt werden!
- Möchte man alle Änderungen verwerfen, kann auch `git clean -f` ausgeführt werden

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Git Push

- `git push` ist ein Befehl, der verwendet wird, um Änderungen, die im lokalen Repository vorgenommen wurden, auf ein Remote-Repository hochzuladen

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Git Pull

- Um Änderungen aus einem Remote-Repository herunterzuladen und in das lokale Repository zu integrieren, kann der Befehl `git pull` in Git verwendet werden
- Wenn eine Person Änderungen im Remote-Repository vornimmt, kann eine andere diese Änderungen herunterladen und mit `git pull` in ihr/sein lokales Repository integrieren, um sicherzustellen, dass sie/er immer mit den neuesten Änderungen arbeitet

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

.gitignore

- Die .gitignore ist eine Datei, die in einem Git-Repository angelegt werden kann
- Sie enthält eine Liste von Dateien oder Verzeichnissen, die von Git beim Tracking und Committing ausgeschlossen werden sollen
- Dadurch können beispielsweise sensible oder temporäre Dateien ausgeschlossen werden, um eine saubere Versionskontrolle zu gewährleisten und Konflikte zu vermeiden
- Häufig werden damit auch Paketinstallationen ignoriert, beispielsweise ein Virtual Environment (Python) oder node_modules (Node.js), also Dateien, die jede*r Entwickler*in eigenständig installieren muss

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Aufgabe bis zum nächsten Termin

- Erstellt ein Repository in unserer GitLab-Gruppe namens "01 To-Do App <euer Vorname/Nachname>“

  (z.B. "01 To-Do App Max Mustermann")
- Erstellt einen neuen Branch namens "01_auffrischung“
- Committet und pusht in diesen Branch eine Datei namens *index.html*

Nils Constantin Hellwig
Wissenschaftliche Hilfskraft
RECHENZENTRUM

Weiterführende Literatur / Quellen

- Offizielles Git Tutorial:

  https://git-scm.com/docs/gittutorial
- Mehr zu Git Branching:

  https://git-scm.com/book/en/v2/Git-Branching-Branches-in-a-Nutshell

  https://gitbookdown.dallasdatascience.com/branching-git-branch.html