CheatSheet

Raveendran Shajiran

Inhaltsverzeichnis

Einleitung

Dieses CheatSheet dient als eine Dokumentation für das **Modul 300 *Plattformübergreifende Dienste im Netzwerk integrieren.*** Wir werden GitBash verwenden und dazu findet man hier wichtige Befehle und passende Anleitungen. Dieses Dokument ist für Github gedacht.

GIT Repository

Konfigurations Dateien

Damit GIT weiss welcher User sich authentifiziert hat, geben wir dem GIT unseren Username und die E-Mail Bescheid. Diese Daten müssen in der globalen Konfiguration gespeichert werden. Es dient uns dazu, wer z.B. das letzte File bearbeitet hat, nachdem ein User ein Commit ausgeführt hat.

Folgende Befehle müssen erfolgen, damit die Konfiguration zu speichern:

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

git config --global user.email shajiran.raveendran@edu.tbz.ch
```

GIT Repository erstellen

Lokales GIT Repository erstellen

Wir erstellen als erstes ein Lokales GIT Repository:

1. Man sollte zuerst ein Ordner erstellen, auf der die Repository gespeichert werden soll: `mkdir Repository`
2. Auf dem Ordner wechseln: `cd Repository/`
3. Das Repository initialisieren: `git init`

Remote GIT Repository erstellen

Wenn aber schon ein Repository existiert, aber diese nicht lokal befindet, kann man einen Remote zu diesem Repository erstellen:

1. Man sollte zuerst ein Ordner erstellen, auf der die Repository gespeichert werden soll: `mkdir Repository`
2. Auf dem Ordner wechseln: `cd Repository/`
3. Man benötigt von einem existierenden Repository ein SSH-PublicKey.
4. Wenn man diesen SSH-PublicKey hat gibt man folgenden Befehl ein: `git clone "SSH-PublicKey LINK"`

Repository aktualisieren & herunterladen

Repository aktualisieren (push)

Wenn man nun eine Datei geändert, hinzugefügt oder gelöscht hat, muss man die Änderungen am Repository übertragen. Dabei muss man folgend vorgehen:

1. Zuerst geht man auf das entsprechende Verzeichnis des Repository: `cd Repository/`
2. Man sollte überprüfen, ob Dateien auch geändert sind: `git status`
3. Die Dateien dann dem Upload hinzufügen: `git add -A`
4. Diesen Upload mit einem Message commiten: `git commit -m "Mein Kommentar"`
5. Und schliesslich den Upload pushen: `git push`

Repository herunterladen (pull)

Wenn nun Dateien im Repository geändert, hinzugefügt oder gelöscht wurden, sollte man diese zuerst auf dem Lokalen Rechner holen, bevor man weiterarbeitet. Dabei muss man folgend vorgehen:

1. Zuerst geht man auf das entsprechende Verzeichnis des Repository: `cd Repository/`
2. Die Dateien herunterladen: `git pull`

Branch

Ein Branch ist in prinzip ein Arbeitsverzeichnis, in der wir unsere Dateien abspeichern. Neue Commits werden als Historie im Branch festgehalten. Nachdem man eine Repository erstellt hat, wird automatisch ein **main** Branch erstellt. Man kann weitere Branches erstellen, die z.B. für Testzwecken benutzt werden.

Branch erstellen & löschen

Folgende Befehle helfen das Bedienen der Branches.

1. Listet alle Branches im Repository auf: `git branch`
2. Erzeugt einen neuen Branch: `git branch <Branch-Name>`

3. Wechselt in ein anderen Branch: `git checkout <Branch-Name>`
4. Benennt den aktuellen Branch um: `git branch -m <Branch-Name-New>`
5. Löscht einen Branch: `git branch -D <Branch-Name>`

Branch mergen

Wie schon vorhin erwähnt kann man einen Branch für Testzwecken nutzen. Wenn nun eine Testdatei in der **Test-Branch** erfolgreich war, sollte man diesen in den **Main-Branch** mergen.

1. Zuerst wechselt man auf den Main-Branch: `git checkout main`
2. Den Test-Branch mit dem Main-Branch mergen: `git merge <Test-Branch>`

Dies ist sehr nützlich, wenn man z.B. an einer Testdatei arbeitet und man ausversehen einen Fehler eingebaut hat, kann man auf den letzt erfolgreichten Test zurückgreifen, welcher sicher auf der **Main-Branch** liegt. Verwenden oft Programmierer.

Quellenangaben

Ressourcen

Um **detailierte** Informationen zu holen, stehen folgende Links zur Verfügung:

- GIT Repository einrichten
- Branch Nutzung
- Markdown Formating

Sonstige

Um **weitere** Informationen zu holen, stehen folgende Links zur Verfügung:

- Unterlagen zu Modul 300
- 10-Toolumgebung
- 20-Infrastruktur
- 25-Sicherheit