**Fachwoche Git/GitHub**

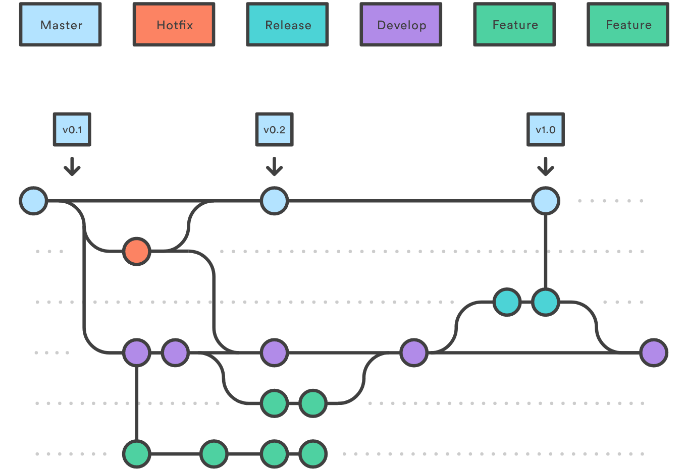
**1. Einführungsaufgaben**

**1a Informieren**

**Wichtige Befehle**

|  |  |
| --- | --- |
| **Befehl** | **Beschreibung** |
| git clone *URL* | Setzt das Ziel der vorhandenem Repository fest. |
| cd *Name des Repo* | Wechselt in das gewünschte Repository |
| git add\* | **Alle** Aktualisierung wird in den nächsten Commit aufgenommen |
| git comit –m «*commit Beschreibung*» | Speicherung mit zusätzlicher Beschreibung |
| git push | Lädt die Datei im Repository hoch |
| git remote add origin *URL* | Vor dem Ersten Push, teil mit in welchem Repository auf GitHub die Datei gespeichert werden soll und benennt die lokale Datei als remote. |
| git push –set-upstream origin *branch\_name* | Push den ganzen Branch |

**Workflow**

****

Main

Hotfix

Feature

Develop

Release

1. Der Develop-Branch wird auf der Basis des Main-Branch erstellt.

2. Der Release-Branch wird auf der Basis vom Develop-Branch erstellt.

-Sobald der Release-Branch fertiggestellt ist, wird er mit dem Develop-Branch und dem

Main-Branch gemergt.

3. Der Feature-Branch wird ebenfalls auf der Basis vom Develop-Branch erstellt.

-Sobald das Feature fertig ist wird es mit dem Develop-Branch gemergt.

4. Taucht ein Problem im Main-Branch auf, wird ein Hotfix-Branch basierend auf dem Main-Branch erstellt.

-Sobald der Hotfix abgeschlossen ist wird dieser mit dem Develop-Branch und dem

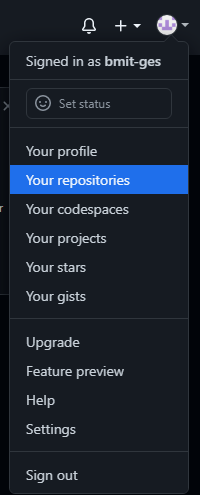
Main-Branch gemergt.

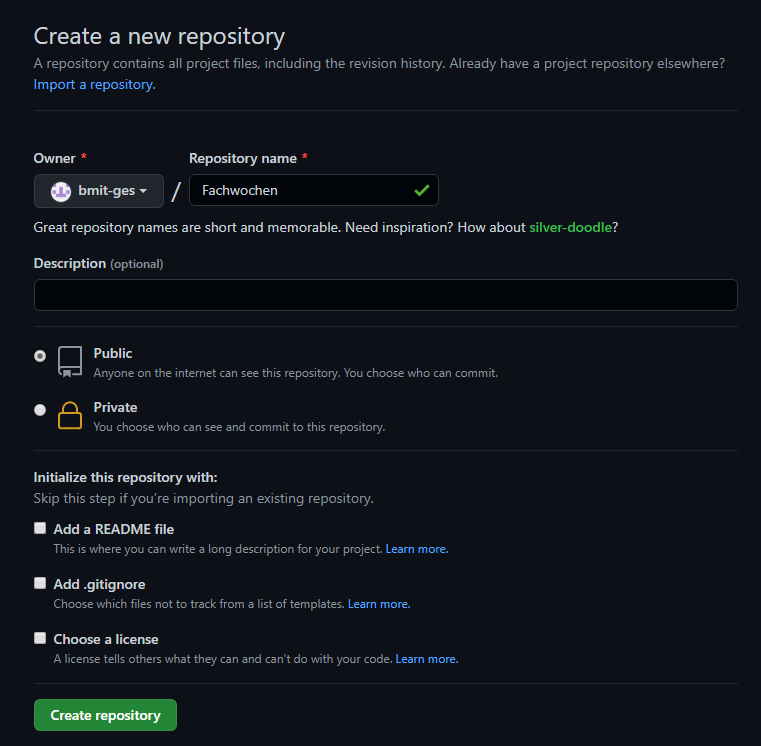
**Merge**

Mit git merge werden mehrere Commit-Abfolgen zu einem Einheitlichen Verlauf zusammengefügt. Vor allem wird git merge für das Zusammenfügen von zwei Branches verwendet.

**1b Ein Repository erstellen**

**Repository erstellen**

****1. 2.

 3.

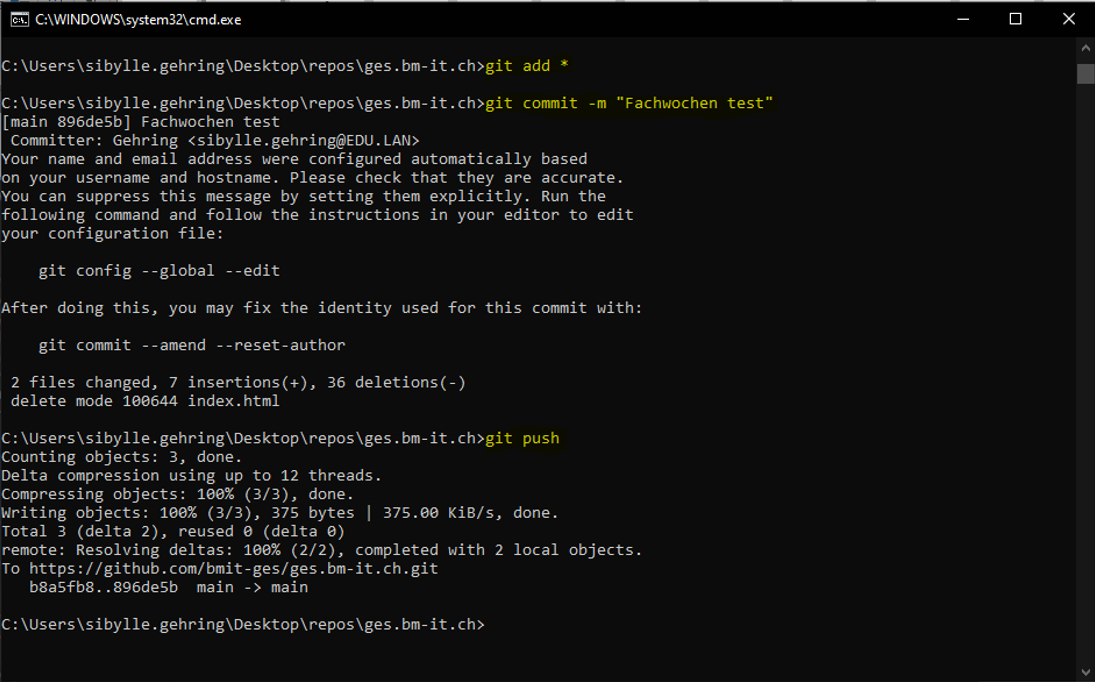
1. Auf Profilbild Klicken um das Dropdown Menü zu öffnen, «Your repositories» auswählen

2. New Auswählen

3. Name des Repository angeben und mit Create repository erstellen.

**2 Intermediate Aufgaben**

**2a clone, commit und push**

****

1. git clone URL Erstellt eine Kopie des Repository von GitHub.

2. git add \* Alle Aktualisierungen werden zu dem nächsten commit hinzugefügt.

3. git commit –m “Beschreibung” Vorübergehende Speicherung mit Beschreibung.

4. git push Pusht den ganzen Branch.

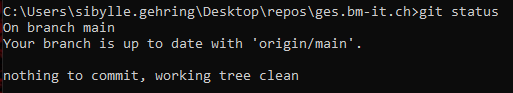
**Antwort auf Frage** *Was ist ein "Working-Tree"?*

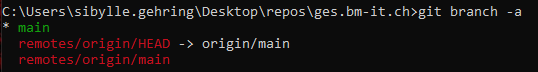
Das Arbeiten mit verschieden Branches und das navigieren innerhalb diesen.

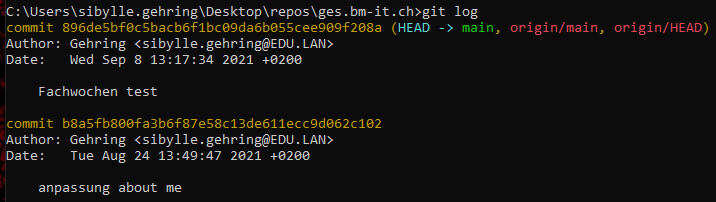
**Antwort auf Frage** *Was ist der Unterschied zwischen einem Commit und Push?*

Der Commit ist nur ein Zwischenspeicher der Push ist die effektive Speicherung.

**2b Git Befehle**

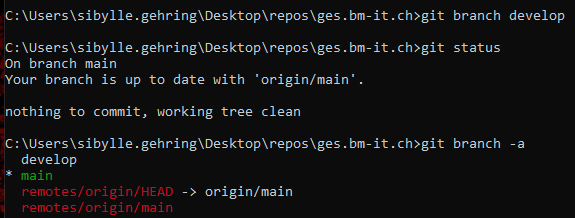






|  |  |
| --- | --- |
| **Befehl** | **Beschreibung** |
| git status | Zeigt den Status an |
| git branch -a | Zeigt die Branchhierarchie an |
| git log | Zeigt alle commits an die gemacht wurden |

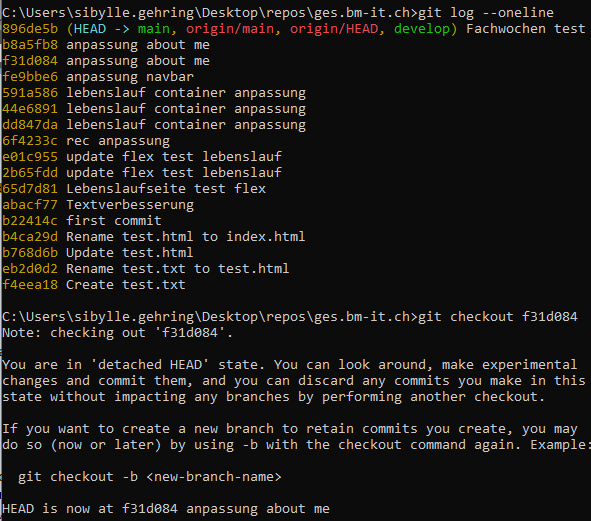
**2c Branches und Checkout**

****

|  |  |
| --- | --- |
| **Befehl** | **Beschreibung** |
| git branch «*Branchname*» | Erstellt einen neuen Branch mit dem gewählten Namen. |

**Antwort auf Frage** *Wie kommen Sie auf den Develop-Branch, den Sie erstellt haben?*

Der neue Branch wird bei der Auflistung der Branches aufgeführt.



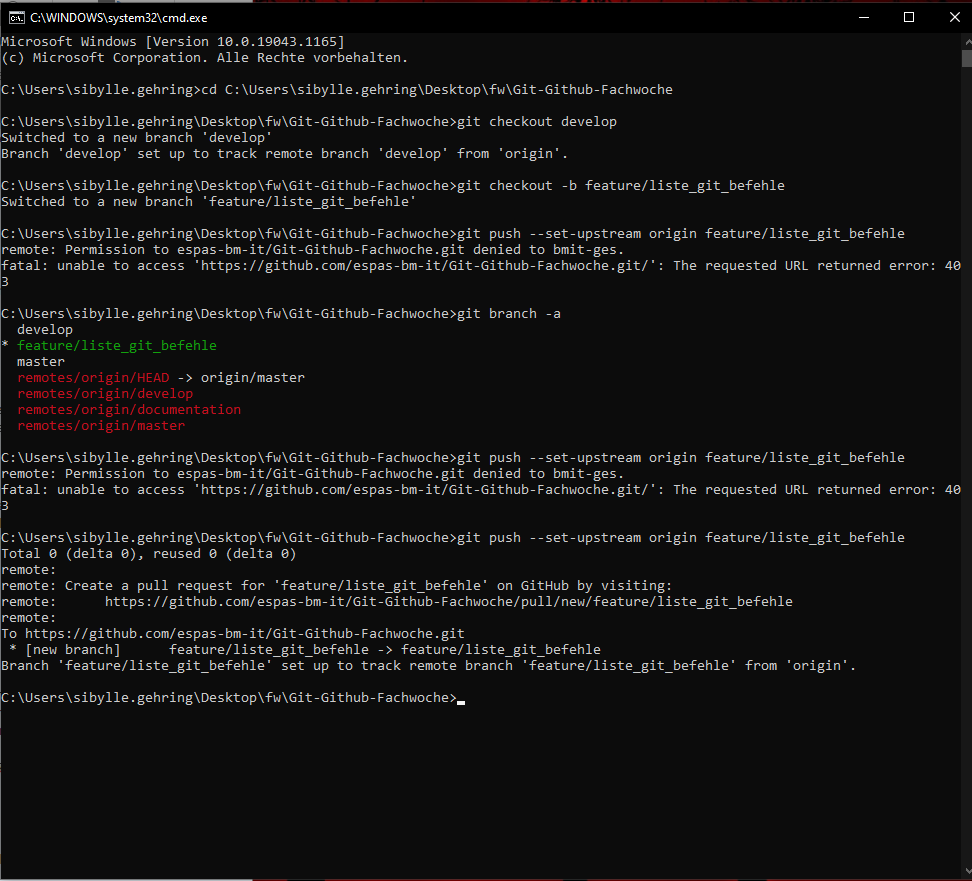
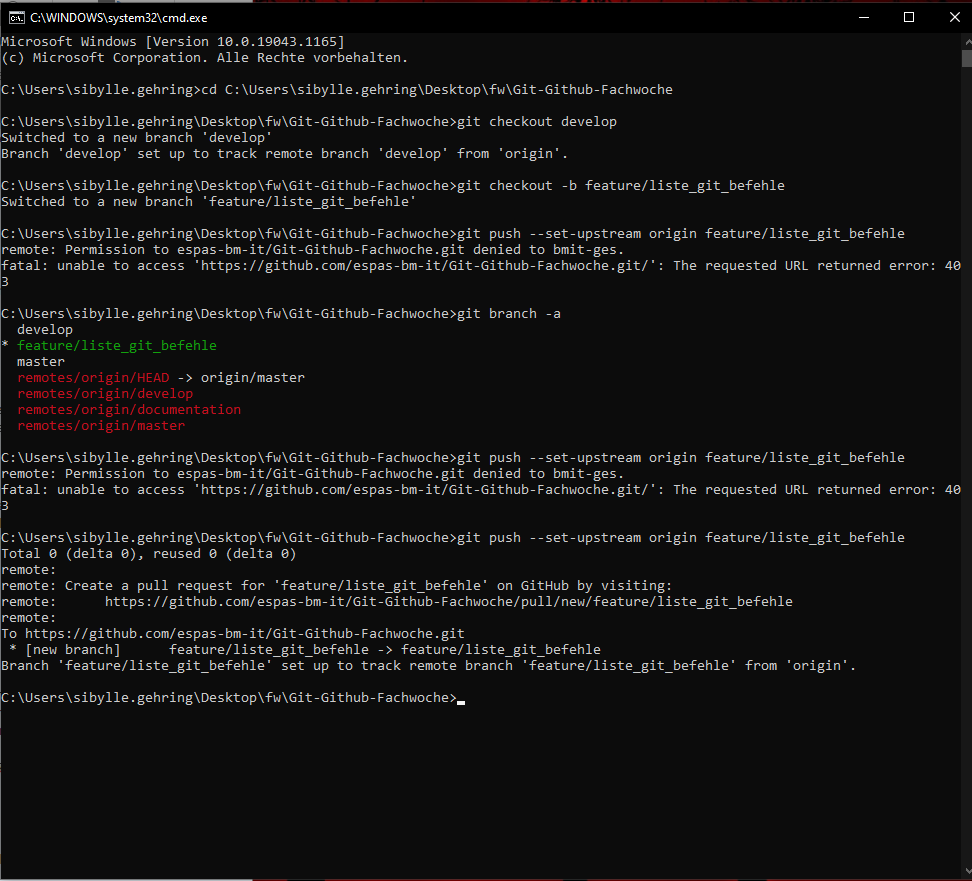
|  |  |
| --- | --- |
| **Befehl** | **Beschreibung** |
| git log --oneline | Zeigt die Hash-Codes der bisherigen Commits an |
| git checkout | Ermöglicht die Navigierung auf Branches oder auch Commits |

**Antwort auf Frage**

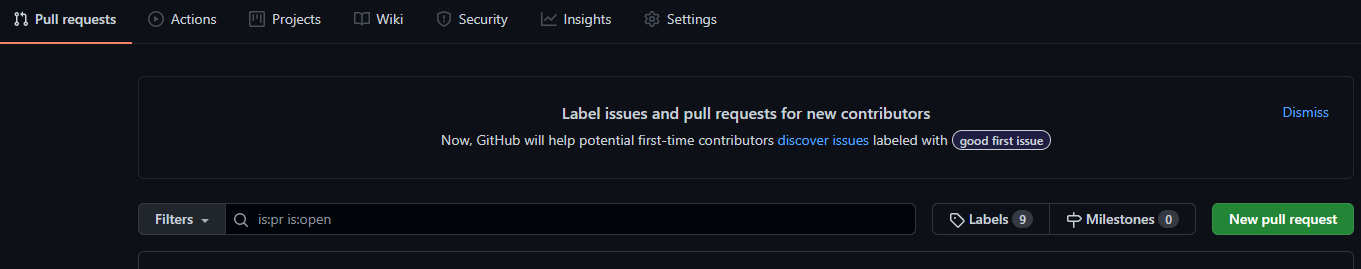
Wechsel auf Branch mit git checkout develop.

**3. Advanced Aufgaben**

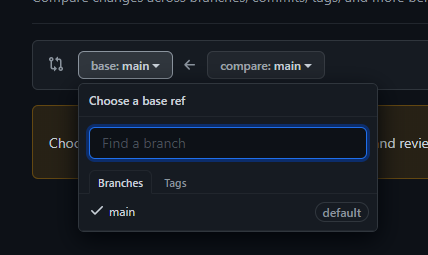
**3a Feature-Branch erstellen**



**3c Erstellen eines Pull-Requests**

****Pullrequest über GitHub/ Pull request/New pull request

Richtiger Branch auswählen in den die Dateien gemerged werden sollen und die Anfrage senden.

****

Ein Pull-Request sollte nur dann gemacht werden, wenn der Code nicht mehr bearbeitet werden muss!