# Lern- und Arbeitsauftrag

|  |  |
| --- | --- |
| Titel | *GitHub Flow* |
| Modul | 324 |
| Autor | [nicola.colic@bbbaden.ch](mailto:nicola.colic@bbbaden.ch) |
| Hilfsmittel | * [Tutorial](https://www.freecodecamp.org/news/how-to-use-git-and-github-in-a-team-like-a-pro/)[[1]](#footnote-1) |
| Sozialform | Einzelarbeit |
| Leistungsziele | *siehe Lektionenplan* |

## Ausgangslage

Sie haben den *GitHub Flow* kennengelernt. In diesem Auftrag arbeiten Sie hierzu ein Tutorial durch, bei welchem Sie diesen Arbeitsablauf anhand eines Beispiels *hands on* wiederholen. Das Tutorial ist hervorragend, leicht verständlich, ausführlich beschrieben, dass es schwierig wäre, bessere Unterlagen zu diesem Thema herzustellen. Es ist allerdings auf Englisch verfasst; jedoch in einem einfachen Englisch, welches sich auch gut, falls nötig, eine maschinelle Übersetzung eignen würde.

## Aufgabenstellung

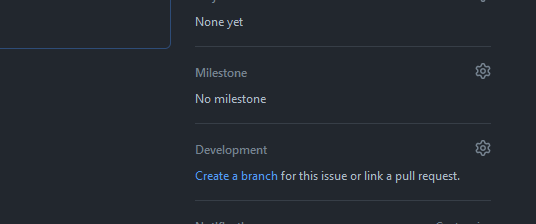
Arbeiten Sie also dieses Tutorial durch. Sie brauchen dafür einen zweiten GitHub-Account, welchen Sie beispielsweise mit einer Einweg-E-Mail-Adresse oder Ihrer persönlichen E-Mail-Adresse einrichten können. Daraufhin spielen Sie mit Ihren beiden Accounts den *GitHub Flow* durch: einmal ohne *merge conflict*, und einmal mit.

### Teilaufgabe 1: Einrichtung

Erstellen Sie, wie im Tutorial beschrieben, eine neue Ablage, und fügen Sie Ihren Zweit-Account als Kollaborator hinzu (Step 1.1); dann befüllen Sie die Ablage mit einer ersten Beispiel-Datei (Step 1.2).

### Teilaufgabe 2: *flow*

Dann spielen Sie die 6 Phasen des *GitHub Flow* durch: Erstellen Sie ein *issue* (Step 1.3). Im Tutorial erstellen Sie daraufhin einen *branch* manuell und verwenden die ID Ihrer *issue* in dessen Namen (Step 2.1). Sie können aber auch direkt von der *issue* aus einen neuen Ast auf dem Server erstellen:



Daraufhin führen Sie git pull aus, um das ganze Projekt mit seinen verschiedenen Ästen auf Ihre Maschine zu laden und dann mit git checkout Ast-Name auf den auf dem Server erstellten Ast zu wechseln (Step 2.2).

👽 Natürlich könnten Sie auch den neuen Ast wie in den Folien beschrieben erst lokal erstellen, und auf den Server hochladen.

|  |
| --- |
| git checkout -b Ast-Name  git push origin Ast-Name |

Buchen Sie die Änderungen ein, laden Sie diese auf den Server hoch und erstellen Sie einen Änderungsantrag.

### Teilaufgabe 3: PR

Besehen Sie sich den PR (Step 3.1) und nörgeln Sie an den vorgeschlagenen Änderungen herum. Daraufhin können Sie darauf reagieren (Step 3.2 und 3.3) und schliesslich die Änderungen übernehmen und den PR schliessen (Step 3.4).

### Teilaufgabe 4: *merge conflict*

🌴 Bisher haben wir unseren *main*-Ast als Ziel für fertige Änderungen genommen. In der Praxis allerdings wird oft ein weiterer Ast verwendet, welcher in diesem Tutorial *develop* heisst. Die Idee dabei ist, dass auf dem *main*-Ast nur Code ist, der für die Auslieferung geeignet ist, und dass Sie im *develop*-Ast zwar eine stabile, aber vielleicht noch nicht durch alle abschliessenden Überprüfungen Version Ihres Codes laufen lassen.

A diagram of a process

Description automatically generated

Spielen Sie also den Ablauf nochmal mit einer neuen Änderung durch; das Mal aber werden Sie, bevor Sie Ihren PR abschliessen können, über den Browser einige Änderungen am *develop*-Ast (oder *main*) vornehmen, die zu einem Konflikt führen. Lösen Sie diesen im Browser, und schliessen Sie den PR ab.

## Gütekriterien

Der Lern- und Arbeitsauftrag ist erfüllt, wenn …

* Sie das Tutorial durchgespielt haben und verstanden haben, wie der *GitHub Flow* mit Konflikt funktioniert.

1. https://www.freecodecamp.org/news/how-to-use-git-and-github-in-a-team-like-a-pro/ [↑](#footnote-ref-1)