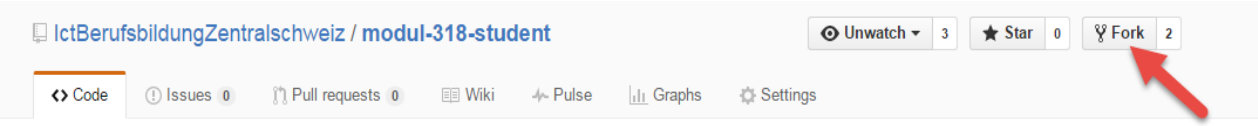
4 Erste Schritte

Wie komme ich zu meiner Visual Studio Solution?

1. Falls Du noch keinen Account bei github habt: erstelle einen.
2. Es gibt ein Repository auf github:
   https://github.com/IctBerufsbildungZentralschweiz/modul-318-student
3. Forke dieses Repository



4. Bereite ein Directory auf deinem PC vor, wo du entwickeln willst.
5. Klone **dein** Repository an diesen Platz:

```
git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY
```

Alles weitere siehst du im folgenden Kapitel «GIT» oder auf den Hilfeseiten von Github.

5 Versionskontrolle mit Git

Den Source Code deiner Software wirst du mit Git verwalten. Hier findest du eine kurze Anleitung wie das grundsätzlich funktioniert. Siehe auch: https://rogerdudler.github.io/git-guide/index.de.html.

5.1 Ein Repository auschecken

Erstelle eine Arbeitskopie, indem du folgenden Befehl ausführst:

```
git clone /pfad/zum/repository
```

Falls du ein entferntes Repository verwendest, benutze:

```
git clone benutzername@host:/pfad/zum/repository
```

5.2 Workflow

Dein lokales Repository besteht aus drei "Instanzen", die von Git verwaltet werden. Die erste Instanz ist deine Arbeitskopie, welche die «echten» Dateien enthält. Die zweite ist der Index (Stage), welcher als Zwischenstufe agiert und zu guter Letzt ist da noch der HEAD, der immer auf deinen letzten Commit zeigt.

Du kannst Änderungen vorschlagen (zum Index hinzufügen) mit

```
git add <dateiname>
git add *
```

Der nächste Schritt im Git-Workflow ist es, deine Änderungen zu bestätigen mit:

```
git commit -m "Commit-Nachricht"
```

Jetzt befindet sich die Änderung im HEAD, jedoch noch nicht im entfernten Repository.

5.3 Änderungen hochladen

Die Änderungen sind jetzt im HEAD deines lokalen Repositories. Um die Änderungen an dein entferntes Repository zu senden, führe folgende Anweisung aus:

```
git push origin master
```

Du kannst *master* auch mit einem beliebigen anderen Branch ersetzen (mehr über Branches erfährst du später).

5.4 fetch & merge

Um dein lokales Repository mit den neuesten Änderungen aus dem entfernten Repo zu aktualisieren, verwende:

```
git pull
```

Mache dies in deiner Arbeitskopie, um die Änderungen erst herunterzuladen (fetch) und dann mit deinem Stand zusammenzuführen (merge).

Wenn du einen anderen Branch mit deinem aktuellen (z.B. master) zusammenführen willst, benutze:

```
git merge <branch>
```

In beiden Fällen versucht Git die Änderungen automatisch zusammenzuführen. Unglücklicherweise ist dies nicht immer möglich und kann in Konflikten enden (merge conflict). Du bist verantwortlich, diese Konflikte durch manuelles Editieren der betroffenen Dateien zu lösen. Bist du damit fertig, musst du das GIT mit folgendem Befehl mitteilen:

```
git add <dateiname>
```

Bevor du Änderungen zusammenführst, kannst du dir die Differenzen auch anschauen:

```
git diff <quell_branch> <ziel_branch>
```

5.5 Tagging

Es wird empfohlen, für Software Release-Tags zu verwenden. Dies ist ein bekanntes Konzept, das es auch schon mit SVN gab. Du kannst einen neuen Tag namens "1.0.0" mit folgendem Befehl erstellen:

```
git tag 1.0.0 1b2e1d63ff
```

1b2e1d63ff steht für die ersten 10 Zeichen der Commit-Id, die du mit deinem Tag referenzieren möchtest. Du erhältst die Liste der Commit-IDs mit:

```
git log
```

Du kannst auch weniger Zeichen verwenden, es muss einfach eindeutig sein.

5.6 Änderungen rückgängig machen

Falls du mal etwas falsch machst - was natürlich nie passiert ;) - kannst du die lokalen Dateien mit:

```
git checkout -- <filename>
```

auf den letzten Stand im HEAD zurücksetzen. Änderungen, die du bereits zum Index hinzugefügt hast, bleiben bestehen.

Wenn du aber deine lokalen Änderungen komplett entfernen möchtest, holst du dir den letzten Stand vom entfernten Repository mit folgenden Befehlen:

```
git fetch origin
git reset --hard origin/master
```

5.7 Git GUI

Falls du lieber mit einer grafischen Benutzeroberfläche arbeitest, als mit der Kommandozeile, kannst du die wichtigsten Arbeitsschritte des Git-Workflows auch mit Git GUI erledigen. Git GUI wird standardmässig zusammen mit Git auf deinem Rechner installiert.