Allgemeine Hinweise: Fügen Sie Kommentare hinzu, sobald Sie es für sinnvoll halten. Der Code auf dem `main`-Branch wird als Abgabe gewertet. Abgaben, bei denen die Kompilierung fehlschlägt, werden mit 0 Punkten bewertet. Wenn Sie bei einer Aufgabe nicht weiterkommen, fragen Sie gerne in den Tutorien oder in Teams nach. Die Designeentscheidungen in den Übungsaufgaben entsprechen aus Gründen der Vereinfachung nicht immer denen, die man in der Realität antreffen würde.

1 Abgabegruppe finden

Bilden Sie Abgabekleingruppen mit 2-3 Personen. Alle Personen müssen im gleichen Individualtutorium sein. Tragen Sie Ihre Gruppe eigenständig bis Montag, dem 25.04., um 18 Uhr in LMS an der vorgesehenen Stelle ein.

2 Entwicklungstools einrichten

Richten Sie sich eine Entwicklungsumgebung ein, mit der Sie Scala-Projekte kompilieren und ausführen können. Wir empfehlen die Verwendung von IntelliJ IDEA, das Sie in der Community-Edition kostenlos nutzen können. Dafür müssen Sie außerdem die Scala-Erweiterung installieren. Andere Code-Editoren sind natürlich erlaubt, aber wir können eventuell keine Hilfestellung leisten, wenn etwas nicht funktioniert.

Richten Sie im Anschluss Git auf Ihrem Rechner so ein, dass Sie Projekte von gitlab.rlp.net klonen, pushen, und pullen können. Für die Kompilierung von Scala-Projekten brauchen Sie außerdem eine Version des Java-Development-Kits in der Version 8 oder 11 sowie `sbt` ab Version 1.5. Alle drei Tools können Sie über IntelliJ IDEA installieren: Ihnen wird eine entsprechende Benachrichtigung angezeigt, sobald das jeweilige Tool benötigt wird, zusammen mit einer Schaltfläche zur automatischen Installation.

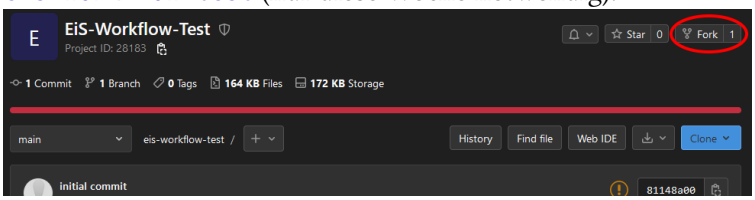
Wenden Sie sich bei Problemen an Ihren Übungsleiter oder fragen Sie im Teams-Kanal nach.

3 Abgabe-Workflow testen

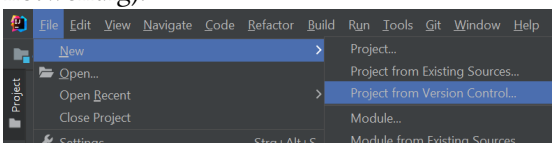
Diese Aufgabe stellt den Abgabeworkflow für alle kommenden Blätter vor. In dieser Woche sollen Sie den Workflow in Einzelarbeit durchspielen. Ab nächster Woche bekommt jede Abgabegruppe ein eigenes Repository zur Entwicklung und Abgabe bereitgestellt.

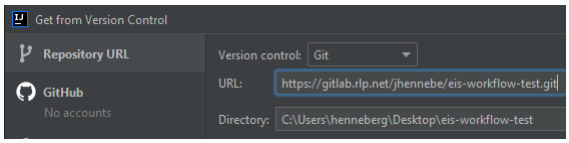
Auch hier gilt: Wenden Sie sich bei Problemen an Ihren Übungsleiter oder fragen Sie im Teams-Kanal nach.

1. Forken Sie das zentrale Repository für diese Aufgabe unter `https://gitlab.rlp.net/juhenneb/eis-workflow-test` (nur diese Woche notwendig).

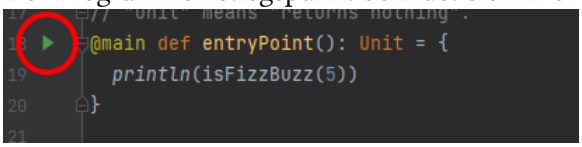


2. Klonen Sie Ihre geforkte Version des Repositorys auf Ihren Rechner (nur diese und nächste Woche notwendig).

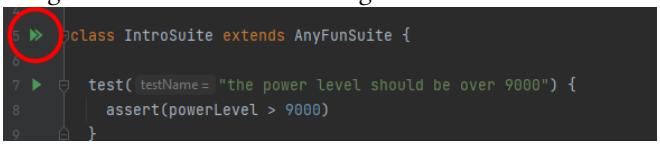




3. Kompilieren Sie das Projekt und führen Sie es aus.
Der Programmeinstiegspunkt befindet sich in `src/main/scala/week01/introduction.scala`.

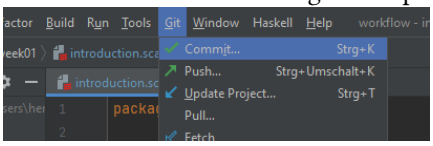


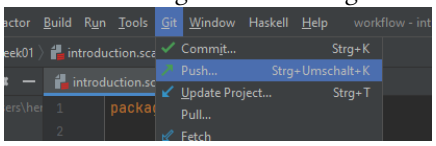
4. Führen Sie die Tests aus. Die Tests befinden sich in `src/test/scala/week01/IntroSuite.scala`.
Einige der Tests sollten fehlschlagen.



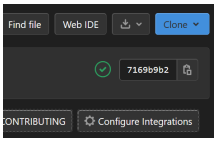
5. Passen Sie das Programm so an, dass es alle Tests besteht. Die Tests dürfen natürlich nicht verändert werden!

6. Committen Sie die Änderungen und pushen Sie das Programm in Ihr eigenes Repository auf GitLab.



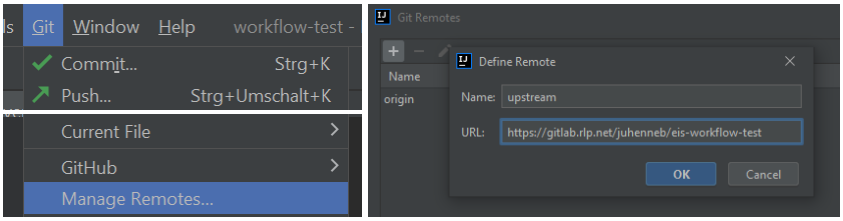


7. Verifizieren Sie, dass die automatischen Tests in GitLab ebenfalls erfolgreich durchlaufen.

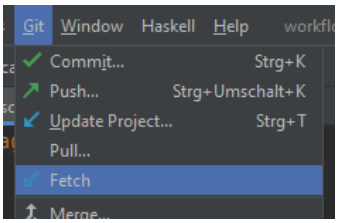


Ab nächster Woche werden alle Codevorgaben zu den Übungsblättern in ein zentrales Repository geladen. Wenn sich im zentralen Repository etwas ändert, wird Ihr (Gruppen-)Repository nicht automatisch aktualisiert. Stattdessen sollen Sie die Änderungen manuell pullen und mergen:

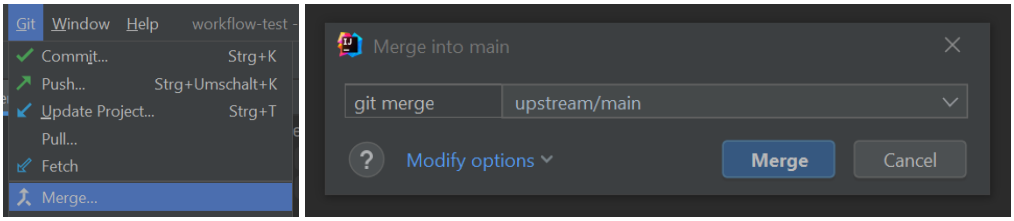
1. Richten Sie das zentrale Repository als zusätzliches Remote-Repository ein. Wählen Sie `upstream` als Namen.



2. Fetchen Sie die Änderungen aus dem `upstream`-Repository.



3. Mergen Sie die Änderungen aus dem `upstream`-Repository in Ihren `main`-Branch.



Kommandozeile

Wenn Sie Git über die Kommandozeile nutzen wollen, sind hier noch einmal alle Befehle aufgeführt:

```
git clone https://gitlab.rlp.net/GITLAB_USER_NAME/eis-workflow-test
git add .
git commit -am "COMMIT_MESSAGE"
git push
git pull
git remote add upstream https://gitlab.rlp.net/juhenneb/eis-workflow-test
git fetch upstream
git merge upstream/main
```

Wir verweisen hier auch auf https://learngitbranching.js.org/.