Migration von GitHub Classroom auf Classroom 50

Diese Seite beschreibt, was beim Umstieg einer bestehenden Klasse zu tun ist. Ein neues Klassenzimmer von Grund auf: Einrichtung.

Solange beide Wege parallel installiert sind, laufen bei **jedem Push zwei Autograder** nebeneinander — der alte scheitert dabei am Moodle-Aufruf. Die Migration ist erst mit Schritt 2 abgeschlossen.

Was sich ändert

	vorher	**nachher**
Bewertung ausgelöst von	`.github/workflows/classroom.yml` im Studi-Repo	zentralem `autograde-runner.yaml`
Bewertungs-Logik	`BZZ-Commons/pygrader`	`BZZ-Commons/pygrader50`, per Tag gepinnt
Moodle-Übertrag	direkt aus dem Studi-Repo	nachts, zentral aus dem Config-Repo
Moodle-Token	in der Org, per `secrets: inherit` überall lesbar	nur im Config-Repo
Zuordnung zu Moodle	GitHub-Name im Moodle-Profil	Moodle-Benutzername = GitHub-Login
Konfigurationsdateien	`.github/autograding/`	Bundle im Config-Repo, `.github/autograding/` bleibt Fallback

Die drei Dateien `unittests.json`, `lint.json` und `pylintrc` behalten ihr Format. Es ist **kein** Umschreiben der Aufgaben nötig.

1. Klassen-Default setzen

Erst wenn `<CLASSROOM>/autograder.py` im Config-Repo liegt, bewertet Classroom 50 überhaupt etwas — vorher liefert es überall `0/0`.

Ablauf und Prüfungen: Einrichtung, Schritt 2.

2. Alten Workflow entfernen

2.1 Was raus muss

`.github/workflows/classroom.yml`. Zwei Jobs stecken darin:

Job	Was damit passiert
`grading`	ruft den alten pygrader-Workflow — ersetzt durch Classroom 50
`copy-issues`	läuft nur unter `if: contains(github.actor, ''classroom)''`, also nur beim GitHub-Classroom-Bot; unter Classroom 50 feuert er nie

Ein separates `copyissues.yml` mit `workflow_dispatch` macht dasselbe manuell. Beim Löschen von `classroom.yml` geht also **keine Funktion verloren**.

2.2 Was bleiben muss

`.github/autograding/` mit `unittests.json`, `lint.json` und `pylintrc` — das ist der Fallback, solange nicht jede Aufgabe ein Bundle im Config-Repo hat.

`requirements.txt` und Hilfsskripte wie `_run_pylint.py` stören nicht, werden aber nicht mehr benutzt: pygrader50 installiert die Studi-Abhängigkeiten bewusst nicht.

2.3 Die Zielliste richtig bilden

Nicht die Repos der Template-Organisation auflisten. Dort liegen auch Templates von Modulen, die noch auf dem alten Pfad laufen — ihnen den Workflow zu nehmen, stoppt dort **still** die Bewertung.

Massgeblich ist der `template`-Block in `<CLASSROOM>/assignments.json`:

```
gh api repos/<ORG>/classroom50/contents/<CLASSROOM>/assignments.json \
   -H 'Accept: application/vnd.github.raw' \
  | jq -r '.assignments[] | "\(.template.owner)/\(.template.repo)"' | sort -
u
```

2.4 Ausführen

```
scripts/remove-legacy-classroom-yml.sh <ORG> <CLASSROOM>            #
Trockenlauf
scripts/remove-legacy-classroom-yml.sh <ORG> <CLASSROOM> --apply    #
löschen
```

Das Skript nimmt die Liste aus 2.3 und ergänzt die bereits angenommenen Studi-Repos. Es ist wiederholbar; fehlende Dateien meldet es als `absent`.

Template-Änderungen erreichen bestehende Repos nicht. Ein Studi-Repo trägt seine Kopie aus dem Moment der Annahme — deshalb behandelt das Skript beide Seiten. Umgekehrt sind Repos früherer Klassen, die dasselbe Template benutzt haben, von der Template-Änderung nicht betroffen.

3. Alten Moodle-Token entwerten

Sicherheitsrelevant. Der alte Workflow reichte den Moodle-Token per `secrets: inherit` an den Bewertungs-Workflow weiter. Jede Person mit Schreibrecht auf ein Studi-Repo konnte ihn mit drei Zeilen auslesen — und der Token kann Noten für beliebige Personen setzen.

1. Secret in **allen** betroffenen Organisationen löschen — typischerweise die Studi-Org *und* die Template-Org. Namen prüfen: es gibt `MOODLE_TOKEN` und `MOODLE_TOKEN2`.
2. Den Token **in Moodle neu erzeugen und den alten invalidieren.** Löschen allein beendet nur die künftige Exposition, nicht die vergangene.

```
gh secret list --org <ORG>
```

4. Moodle-Seite umstellen

Unter GitHub Classroom trugen die Lernenden ihren GitHub-Namen im Moodle-Profil ein. Classroom 50 vergleicht stattdessen direkt **Moodle-Benutzername und GitHub-Login**. Vor dem ersten echten Übertrag prüfen, ob das für alle Teilnehmenden zutrifft.

Der Rest — Aktivität *External Assignment*, Slug als Name, Token im Config-Repo — steht unter Noten nach Moodle übertragen.

5. Prüfen, dass wirklich umgestellt ist

- In einem Studi-Repo pushen: Es startet **genau ein** Bewertungslauf.
- Das Release zeigt eine echte Punktzahl, nicht `0/0`.
- `gh secret list –org <ORG>` listet keinen Moodle-Token mehr.
- Ein `moodle-sync`-Trockenlauf zeigt die erwarteten Personen und Punkte.