

Classroom 50: Noten nach Moodle übertragen

Kurz gesagt: Die Lernenden geben ihre Aufgabe auf GitHub ab. In der Nacht holt ein Job die Punkte ab und trägt sie in Moodle ein. Du musst dafür nichts von Hand starten — aber einmal pro Kurs etwas einrichten.

Die drei Dinge, die stimmen müssen

1. In Moodle gibt es eine Aktivität, die **genau gleich heisst wie der Slug der Aufgabe**.
2. Die lernende Person hat ihren **GitHub-Login im Moodle-Profil** eingetragen.
3. Die Person ist **im Kurs eingeschrieben**.

Stimmt eines davon nicht, kommt die Note nicht an.

Schritt 1: Den Workflow ins Repo kopieren

`moodle-sync.yaml` ist **BZZ-eigen** und wird *nicht* von `gh teacher classroom add` angelegt. Die anderen drei Workflows im Config-Repo (`autograde-runner`, `publish-pages`, `collect-scores`) kommen mit, dieser nicht — du kopierst ihn einmal von Hand hinein.

Die Vorlage liegt im Repo [BZZ-Commons/pygrader50](#) im Ordner `classroom50/`:

```
cp classroom50/moodle-sync.yaml <config-repo>/.github/workflows/moodle-sync.yaml
```

Danach committen und pushen. Ab jetzt erscheint der Workflow im Config-Repo unter **Actions** → **Moodle Sync**.

Eigener Dateiname, keine bestehende Datei anfassen. `gh teacher` überschreibt seine mitgelieferten Workflows bei einem Refresh — was du hineinschreibst, wäre weg.

Die Datei gilt für **alle** Classrooms im Repo. Ein Config-Repo mit mehreren Klassen braucht also keine zweite Kopie. Anzupassen ist höchstens der gepinnte `pygrader50`-Tag, falls er von dem in `bootstrap/autograder.py` abweicht.

Ausführlich: [Einen Classroom einrichten](#), Schritt 5.

Schritt 2: Aktivität in Moodle anlegen

Pro Aufgabe **eine** Aktivität vom Typ **External Assignment** (`mod_externalassignment`).

Der Name der Aktivität muss **exakt** der Slug der Aufgabe sein, zum Beispiel:

```
m323-lu01-a02-imperativer-ggt
```

Kein zusätzliches Wort, kein Leerzeichen davor oder dahinter. Am besten den Slug kopieren und einfügen.

Schritt 3: GitHub-Login im Moodle-Profil

Jede lernende Person trägt ihren GitHub-Login im eigenen Moodle-Profil ein, im Feld *GitHub Classroom*. Anleitung dazu: [GitHub Benutzernamen in Moodle eintragen](#).

Wichtig: Der **Moodle-Benutzername spielt keine Rolle**. Gesucht wird nur in diesem einen Profelfeld. Welches Feld das ist, legt die Plugin-Einstellung `mod_externalassignment/external_username` fest (Vorgabe: `github_user`).

Schritt 4: Zugangsdaten hinterlegen

Das machst du **einmal** — und zwar im Repo `classroom50`, *nicht* in der Organisation der Lernenden.

Art	Name	Wert
Secret	MOODLE_TOKEN	Webservice-Token aus Moodle
Variable	MOODLE_URL	z.B. https://moodle.bzz.ch
Variable	MOODLE_FUNCTION	optional, Vorgabe <code>mod_externalassignment_update_grade</code>

```
gh secret set MOODLE_TOKEN --repo <ORG>/classroom50
gh variable set MOODLE_URL --repo <ORG>/classroom50 --body
"https://moodle.bzz.ch"
```

Für den Token in Moodle einen **eigenen Webservice-Benutzer** anlegen, der nur diese eine Funktion ausführen darf. Je weniger der Token kann, desto kleiner der Schaden, falls er einmal ausserhalb landet.

Schritt 5: Testen

Teste mit einem **echten Login einer lernenden Person**. Dein eigener Lehrer-Account ist im Kurs meistens kein Teilnehmer — damit schlägt der Test fehl, obwohl alles richtig eingerichtet ist.

Trockenlauf (ändert nichts in Moodle):

```
gh workflow run moodle-sync.yaml --repo <ORG>/classroom50 \
-f classroom=<CLASSROOM> -f dry_run=true
```

So sieht es aus, wenn es klappt:

```
Moodle-Übertrag: 2 Abgaben
[dry-run] <SLUG> / anna: 5/7
[dry-run] <SLUG> / bruno: 7/7
übertragen: 2 | unverändert: 0 | fehlgeschlagen: 0
```

Wann läuft der Übertrag?

Job	Zeit	Was er tut
collect-scores	nachts	sammelt alle Abgaben in <CLASSROOM>/scores.json
moodle-sync	04:57 (57 4 * * *)	trägt sie von dort nach Moodle ein

Zwischen den beiden liegen 40 Minuten, damit der erste Job sicher fertig ist.

Übertragen wird pro Person und Aufgabe immer die **neueste** Abgabe. Was schon übertragen wurde, merkt sich die Datei <CLASSROOM>/moodle-state.json. Übersprungen wird nur, wenn Abgabe und Punktzahl gleich geblieben sind — eine Neubewertung derselben Abgabe geht also erneut raus.

Das kommt in Moodle an:

Moodle-Feld	Inhalt
assignment_name	Slug der Aufgabe
user_name	GitHub-Login der Person
points / max	Punkte der neuesten Abgabe
externallink	Link zum GitHub-Release
feedback	Text des Releases, bei verspäteter Abgabe mit Hinweis davor

Von Hand auslösen

Im Repo classroom50: **Actions → Moodle Sync → Run workflow**.

Eingaben: Classroom, optional Aufgabe und Login, dazu die Schalter dry_run (nur anzeigen) und force (auch schon Übertragenes nochmals senden). Lässt du den Classroom leer, laufen **alle** Ordner mit einer scores.json.

Wenn etwas nicht klappt

Meldung / Symptom	Das steckt dahinter	Das hilft
No Moodle user found with username „X“	Profelfeld leer oder falsch geschrieben	Person trägt ihren GitHub-Login im Moodle-Profil nach
No matching assignment found	Aktivität fehlt oder Person ist nicht im Kurs	Beides prüfen — das Plugin sucht Aktivität und Person zusammen
keine XML-Antwort erhalten	falsche MOODLE_URL oder ungültiger Token	Moodle schickt eine Login-Seite statt Daten — URL und Token prüfen
Note kommt nicht an, Log sagt „unverändert,“	Zustandsfile hält sie für erledigt	Workflow mit force erneut auslösen

Meldung / Symptom	Das steckt dahinter	Das hilft
Kein Eintrag Moodle Sync unter Actions	Workflow-Datei fehlt im Config-Repo	Schritt 1 nachholen
Nachtlauf jede Nacht rot, immer dieselbe Person	alter Eintrag in scores.json	siehe Betriebsfallen

Zum Nachschauen

Warum zentral und nicht im Repo der Lernenden?

Mit dem Moodle-Token kann man Noten für **beliebige Personen** setzen. Läge er in einem Studi-Repo, könnte ihn ein selbst hinzugefügter Workflow in drei Zeilen auslesen. Genau so war es unter GitHub Classroom (secrets: inherit). Mit Classroom 50 ist damit Schluss: Der zentrale Workflow autograde-runner.yaml gibt **keine Secrets** an den Bewertungs-Job weiter.

Warum nachts und nicht bei jedem Push?

Der direkte Weg hätte einen Token in jedem Studi-Repo gebraucht, eine eigene Workflow-Datei pro Repo, eine nachgebaute Skip-Logik und rund 33 % mehr Actions-Minuten — und einen Nacht-Fallback hätte es trotzdem gebraucht.

Lokal prüfen, ohne Zugangsdaten

```
gh api repos/<ORG>/classroom50/contents/<CLASSROOM>/scores.json \
  -H 'Accept: application/vnd.github.raw' > scores.json
GH_TOKEN=$(gh auth token) python -m pygrader50.moodle scores.json --dry-run
```

Alle Optionen: [CLI-Referenz](#).

From:
<https://wiki.bzz.ch/> - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/howto/git/classroom50/moodle_simple?rev=1787644286

Last update: **2026/08/25 09:51**

