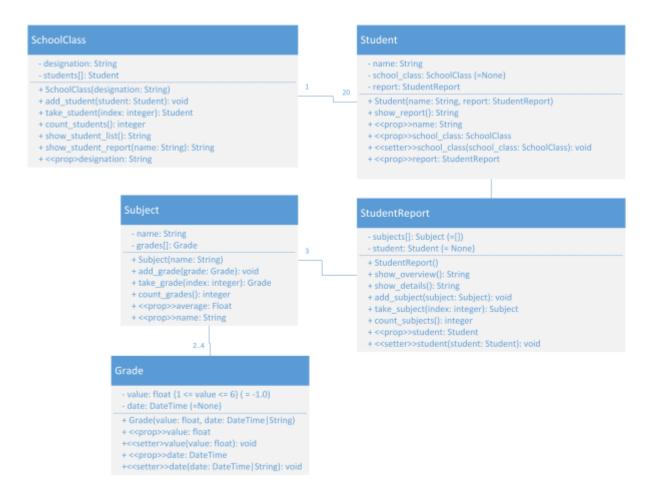
2025/11/05 06:17 1/3 LU11.A01 - Schulverwaltung

LU11.A01 - Schulverwaltung



Entwickeln Sie die Applikation als Gruppe nach dem Prinzip des Test Driven Developments.

UML Klassendiagramm



Hinweise

- Die Klasse Grade wird als @dataclass realisiert.
- Der Unittests für den Konstruktor einer Klasse muss jeweils als erstes realisiert werden. Erst danach können weitere Methoden der Klasse ausgewählt werden.
- Falls eine Methode ein Objekt bzw. Funktion einer anderen Klasse benötigt, verwendest du eine Mockup-Klasse bzw. -Funktion.
- Die Klasse StudentReport enthält zwei Methoden zur Anzeige der Noten:
 - Die Methode show_overview() liefert ein Zeugnis mit allen Fächern und dem entsprechenden Notenschnitt. Eine mögliche Ausgabe kann wie folgt aussehen:

Zeugnis für: Mathe: 4.25 Deutsch: 5.0 Turnen: 5.0

• Die Methode show details() liefert alle Fächern mit den einzelnen Noten. Eine mögliche Ausgabe kann wie folgt aussehen:

Mathe

Fach: Mathe mit 2 Noten 1: 4.0 1.1.11 2: 4.5 2.2.22 Schnitt: 4.25

Fach: Deutsch mit 3 Noten 1: 4.0 3.3.33

> 2: 6.0 4.4.44 3: 5.0 5.5.55

Schnitt: 5.0

Fach: Turnen mit 4 Noten 1: 4.5 6.6.66 2: 5.0 7.7.77 3: 5.0 8.8.88 4: 5.5 9.9.99

Schnitt: 5.0

Vorgehen

Vorbereitung

- 1. Akzeptiere das GitHub Classroom Assignment.
- 2. Wähle eine offene Gruppe aus oder eröffne eine neue Gruppe.
- 3. Klone das gemeinsame Repository in deine Entwicklungsumgebung.

Realisierung

Zu Beginn müssen alle Gruppenmitglieder zu mindestens einer Funktion die Unit Tests schreiben. Im weiteren Verlauf des Projekts kannst du entweder

- eine Funktion realisieren, zu der die Unit Tests schon vorhanden sind oder
- die Unit Tests zu einer weiteren Funktion schreiben.

Unit Tests schreiben

Vergleiche die Issues (offen und geschlossen) in GitHub mit dem Klassendiagramm.

https://wiki.bzz.ch/ Printed on 2025/11/05 06:17 2025/11/05 06:17 3/3 LU11.A01 - Schulverwaltung

1. Wähle eine Klasse (Konstruktor) oder Funktion aus, die noch von keinem Gruppenmitglied bearbeitet wird.

- 2. Erstelle ein Issue auf GitHub mit dem Titel "Unittests *Funktionsname*", z.B. "Unittests add student"
- 3. Trage dich als Verantwortlichen ein.
- 4. Schreibe die Unittests für die gewählte Funktion.
- 5. Wenn deine Testfunktionen fertig sind,
 - 1. führst du einen Commit durch, in der Commit Message schreibst du unter anderem resolves #n (n steht für die Nummer des entsprechenden Issues),
 - 2. aktualisierst du das gemeinsame Repository auf GitHub: **Pull** ⇒ **Push** und
 - du erstellst ein Issue auf GitHub.
 Der Titel des Issues ist die Funktion zu welcher du die Tests geschrieben hast, z.B. "add_student".

Funktion realisieren

Anhand der offenen Issues siehst du, welche Funktionen bereit für die Realisierung sind.

- 1. Wähle das Issues aus und trage dich als Verantwortlichen ein.
- 2. Führe einen **Pull** des Repositories in deine Entwicklungsumgebung durch.
- 3. Realisiere und teste die Funktion.
- 4. Wenn deine Funktion korrekt und fertig ist,
 - 1. führst du einen Commit durch, in der Commit Message schreibst du unter anderem resolves #n (n steht für die Nummer des entsprechenden Issues),
 - 2. aktualisierst du das gemeinsame Repository auf GitHub: Pull ⇒ Push

Vielleicht stellst du fest, dass die Unittests zu deiner Funktion unvollständig oder fehlerhaft sind. Kontaktiere die Person, welche die Unittests geschrieben hat und besprich mit ihr die fehlenden/fehlerhaften Unittests.

M450-LU01



From

https://wiki.bzz.ch/ - BZZ - Modulwiki

Permanent link:

https://wiki.bzz.ch/modul/m450/learningunits/lu11/aufgaben/school?rev=1731662772

Last update: **2024/11/15 10:26**

