1. Fehlerbehandlung

In den Kapiteln dieser Learning Unit geht es um die Behandlung von Fehlern sowie dem Schreiben von automatisierten Tests. Aber zunächst befassen wir uns damit, was ein Fehler ist und welche Arten davon wichtig sind fürs Programmieren.

Syntaktische Fehler

Bei der Programmerstellung wird die Syntax der Programmiersprache nicht eingehalten. So können

- Befehle falsch geschrieben sein (`vor` statt `for`),
- Klammern fehlen oder zu viel stehen (`def a_method(x))`)
- oder Attribut bzw. Methodennamen falsch sein.

Der Compiler bzw. Interpreter meldet diese Fehler VOR der Ausfühung des Programms.

Funktionale Fehler

Bei einem funktionalen Fehler wird ein Teil des Codes nicht resp. falsch ausgeführt. In diesem Fall sprechen wir von einer *Ausnahme* resp. *Exception*. Solche Fehler können im Programm abgefangen und behandelt werden. Ein Beispiel dafür ist die Division durch 0. Ein weiteres Beispiel ist der Zugriff auf eine nicht verfügbare Ressource (z.B. eine Datei). Der Umgang mit funktionalen Fehlern behandeln wir in dieser **Learning-Unit Exception**.

Logische Fehler

Bei einem logischen Fehler läuft der Code wohl korrekt ab, aber die Datenverarbeitung führt nicht zum erwarteten Ergebnis. Dies muss in der Programmlogik sichergestellt und nötigenfalls korrigiert werden. Das Vermeiden von logischen Fehlern kann durch das Testen dieser Logik erreicht werden. Als Beispiel sei hier die Klasse `BankAccount` genannt, die Sie aus der LU01 kennen und deren Saldo nicht negativ sein darf (d.h. kein Kredit resp. Überbezug). Wie man dies mittels automatisierten *Unit Tests* sicherstellt, behandeln wir im **Modul 450**.

© René Probst

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

Permanent link:
https://wiki.bzz.ch/modul/archiv/m320/learningunits/lu03/theorie/lu4-kapitel_1

Last update: **2026/08/30 14:06**