# Unterschied zwischen Compiler- und Laufzeitfehlern

**Grob vorweg:**

Wohl offensichtlich, sind Compilerfehler jene, welche bereits zur Zeit der Kompilierung auffallen und bemängelt werden. Dies sind syntaktische Fehler im Code, die mit grammatikalischen in natürlicher Sprache vergleichbar sind.

Laufzeitfehler hingegen sind diese, welche erst während der Ausführung, also Laufzeit, eintreten; wobei diese entweder erwartbar waren oder unvorhergesehen eintreten. Diese könnte man mit semantischen oder kontextuellen Fehlern natürlicher Sprache vergleichen. Das hieße, der Satz ist formell richtig, macht inhaltlich oder auch logisch einfach keinen Sinn.

**Mein Problem:**

Obgleich mir das bekannt ist, fällt es mir manchmal noch schwer das Eintreten eines dieser Fehlerszenarien vorherzusagen. Dies liegt unter anderem auch danach, dass man insbesondere bei konstruierten oder offensichtlichen Fehlerquellen vom menschlichen Verstand her denkt, das müsse der Compiler doch bereits erkennen, oft aber erst während der Laufzeit der Fehler eintritt, weil logische Fehler den Compiler nicht stören.

Eins was mich irritiert hatte, war zum Beispiel, dass ein formatierter String (also String.format(), mit seinen Ablegern System.out.printf() oder System.out.format(), einen Laufzeitfehler wirft, obwohl falsche Typbelegungen ja erkennbar sein sollten.  
Da Java statisch typisiert ist, denke ich mir, müsste dem Compiler ja auffallen, wenn man versucht einen String in eine Zahl zu drücken. *printf(%.2f von %d, strVar1, strVar2)*Es hat sich nun aber herausgestellt, dass die Parameterliste alles als Objekte entgegennimmt und dann (vermutlich) zu seinen Zieltypen downcastet und dabei der Fehler kommt erst eintritt.

**Beispiele für Compilerfehler (indirekt KI-zitiert):**

Syntaxfehler:

* Semikolon oder Klammer fehlt
* falsches Schlüsselwort benutzt

Typfehler:

* falsche Wertzuweisung (int i = „String“)
* Aufruf einer Funktion mit falsch Rückgabetyp (int i = object.getName)

Unerreichbarer Code:

* Code nach einem return in einer Methode
* Code in einem if-Block, der nicht erreicht ist (always false/true)
* falsch sortierte Catch-Blöcke
  + Spezifische Exception nie erreichbar, weil eine allgemeinere sie schon fängt

**Beispiele für Laufzeitfehler (direkt KI-zitiert und ff):**

 **NullPointerException:**

* **Beispiel:** Aufruf einer Methode auf einem null-Referenz.
* **Warum ein Laufzeitfehler:** Der Compiler kann nicht im Voraus wissen, ob eine Referenz während der Ausführung null sein wird.

 **IndexOutOfBoundsException:**

* **Beispiel:** Zugriff auf ein Array-Element mit einem ungültigen Index.
* **Warum ein Laufzeitfehler:** Der Index wird zur Laufzeit berechnet und kann außerhalb der gültigen Grenzen liegen.

 **ArithmeticException:**

* **Beispiel:** Division durch Null.
* **Warum ein Laufzeitfehler:** Diese Art von Fehler kann nur während der Ausführung auftreten, wenn der Divisor tatsächlich Null ist.

 **ClassCastException:**

* **Beispiel:** Versuch, ein Objekt einer Klasse in ein Objekt einer anderen, nicht kompatiblen Klasse zu casten.
* **Warum ein Laufzeitfehler:** Der tatsächliche Typ eines Objekts kann zur Laufzeit erst bestimmt werden.

**Wo können Compiler- und Laufzeitfehler verwechselt werden?**

* **Logische Fehler:**
  + **Beispiel:** Eine falsche Bedingung in einer if-Anweisung führt zu unerwartetem Verhalten, das schwer zu finden sein kann.
  + **Warum Verwechslung:** Logische Fehler können zu Laufzeitfehlern führen, aber sie werden vom Compiler nicht erkannt, da sie syntaktisch korrekt sind.
* **Unbehandelte Exceptions:**
  + **Beispiel:** Eine Exception wird geworfen, aber nicht abgefangen.
  + **Warum Verwechslung:** Obwohl der Code syntaktisch korrekt ist, führt das Fehlen eines catch-Blocks zu einem Laufzeitfehler, wenn die Exception tatsächlich auftritt.

**Zusammenfassung:**

* **Compilerfehler:** Werden früh erkannt, verhindern die Ausführung des Programms.
* **Laufzeitfehler:** Treten während der Ausführung auf und können zu Programm abstürzen führen.
* **Verwechslungsgefahr:** Logische Fehler und unbehandelte Exceptions können zu Laufzeitfehlern führen, obwohl der Code syntaktisch korrekt ist.