

Lernzettel

Einführung in Akka: Actors, Message-Passing,
Supervision, Lebenszyklus.

Universität: Technische Universität Berlin
Kurs/Modul: Programmieren II für Wirtschaftsinformatiker
Erstellungsdatum: September 6, 2025



Zielorientierte Lerninhalte, kostenlos!
Entdecke zugeschnittene Materialien für deine Kurse:

<https://study.AllWeCanLearn.com>

Programmieren II für Wirtschaftsinformatiker

Lernzettel: Einführung in Akka: Actors, Message-Passing, Supervision, Lebenszyklus.

(1) Grundidee des Actor-Modells.

Das Actor-Modell dient der nebenläufigen Programmierung. Die zentralen Bausteine sind:

- **Actors:** unabhängige, zustandsbehaftete Einheiten.
- **Nachrichten:** unveränderliche Daten, die über Postfächer ausgetauscht werden.
- **Verarbeitung:** ein Actor beendet die Bearbeitung einer Nachricht, bevor er die nächste annimmt.

(2) Architektur von Akka.

Kernkonzepte in Akka:

- *ActorSystem*: Container, der Actors verwaltet.
- *Actor*: eigenständiger Ausführungskontext, der Nachrichten empfängt und verarbeitet.
- *Messages*: unveränderliche Objekte, die zwischen Actors ausgetauscht werden.

(3) Message-Passing.

Nachrichten werden asynchron versendet und verarbeitet.

- **Senden:** `actorRef ! message` (fire-and-forget).
- **Antworten:** über `sender()` bzw. `context.sender()` möglich.
- **Optional:** `ask`-Muster mit Future-Wert, z. B. `actorRef ? message` liefert einen Future.

(4) Supervision.

Eltern-Actoren überwachen ihre Kind-Actors. Wichtige Punkte:

- Supervisor wählt beim Fehler eines Kind-Actors eine Strategie.
- Typische Strategien: Restart, Resume, Stop, Escalate.
- Ziel: Wiederherstellung der Teilanwendung und Fehler-Isolation.

(5) Lebenszyklus.

Der Lebenszyklus eines Actors umfasst Zustände und Lifecycle-Hooks:

- **preStart:** Vor dem ersten Verarbeiten einer Nachricht.
- **onReceive:** Bearbeitung einer eingehenden Nachricht.
- **postStop:** Ressourcenfreigabe beim Stop.
- **preRestart / postRestart:** Neustart-Verarbeitung bei Fehlern.

(6) Typische Anwendungsbeispiele in der Wirtschaftsinformatik.

- Ereignisgesteuerte Verarbeitung von Geschäftsereignissen (BI-, ETL-Pipelines).
- Skalierbare, fehlertolerante Systeme für Datenströme.
- Integration von asynchronen Prozessen in Unternehmensanwendungen.

Hinweis zur Praxis.

In Scala/Akka arbeiten Actors oft mit unveränderlichen Messages und dem Kontext, um Nebenläufigkeit sicher zu handhaben.