

# Nebenläufigkeit



# Gliederung

- Einführung
  - Prozesse & Threads
  - Anwendungsbeispiele
  - Stichwort Performance
- Umsetzung in Java
  - Implementierung
  - Java API
- Praxis
  - Deadlocks
  - Scheduling
  - Threadpooling



# Nebenläufigkeit

- Ereignisse sind nebenläufig, wenn keines eine Ursache des anderen ist
- Aktionen sind parallelisierbar, wenn keine das Resultat des anderen benötigt

# Prozesse & Threads

- Prozess
  - Stellt eine laufende Anwendung dar
  - Erhält eigenen Prozesskontext
- Thread
  - Ausführungsstrang eines Prozesses
  - Teilen sich den Prozesskontext
  - Erhalten eigenen Stack im Adressraum
  - Erleichterte Kommunikation zwischen Threads



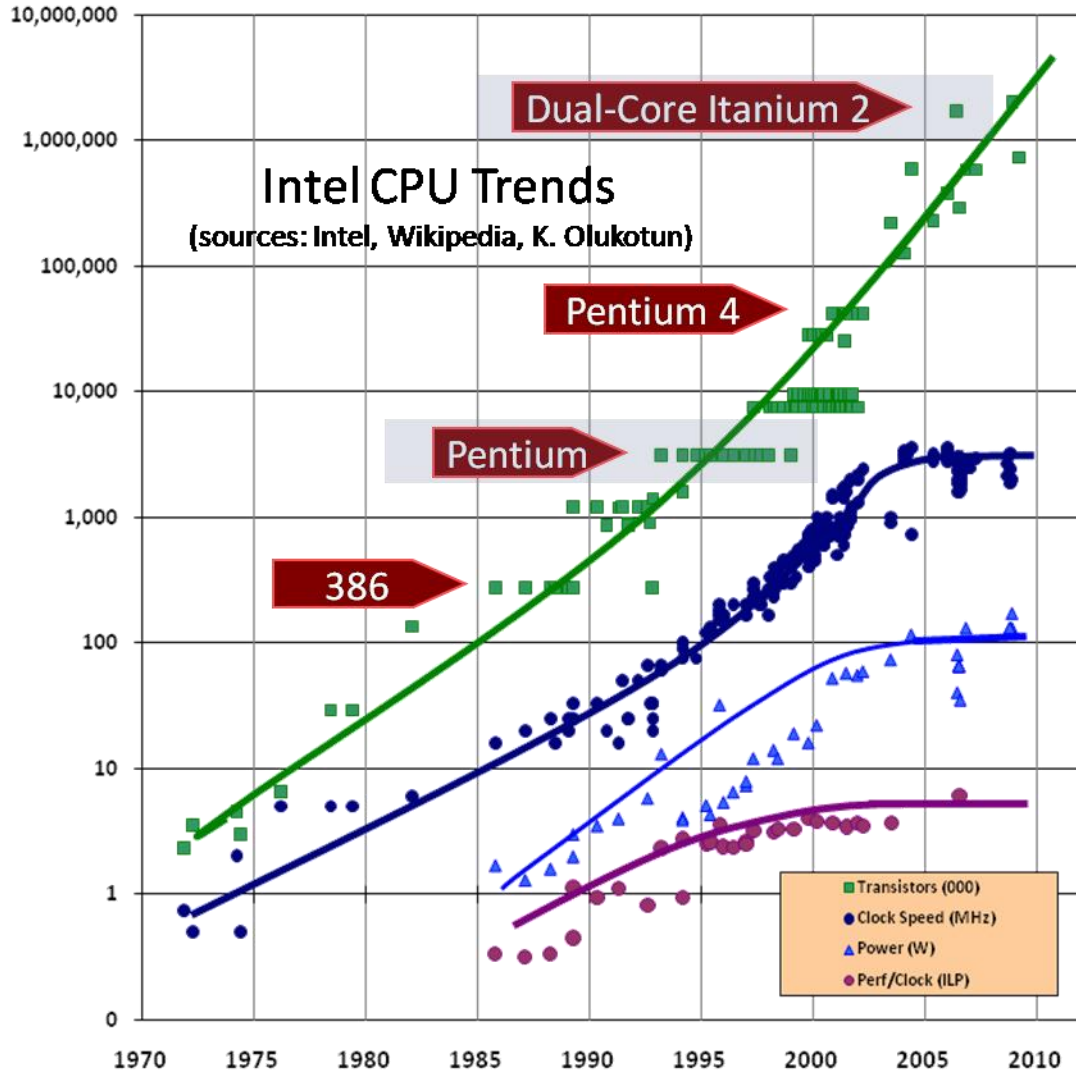
# Anwendung

- Desktopanwendungen
  - Event-Dispatching
- Webanwendungen
  - Langlaufende Benutzeraktionen
- Batchanwendungen
  - Performance



# Anwendung

- Performance
  - Can't squeeze out more of 1 CPU





# Anwendung

- Performance
  - The Free Lunch Is Over
    - Steigende Belastung wurde durch schnellere Prozessoren ausgeglichen
  - Nebenläufigkeit in Anwendungen gewinnt an Bedeutung

# Umsetzung in Java

- Green Threads
  - Keine Unterstützung für native Threads vom BS
  - Simuliertes Threading von JVM
  - Keine parallele Ausführung

# Umsetzung in Java

- Native Threads
  - Verwaltet vom Betriebssystem
  - Mögliche Ausführung auf verschiedenen Prozessoren
  - Windows ab JDK 1.1
  - Solaris ab JDK 1.2
  - Linux ab JDK 1.3
    - Native POSIX Thread Library

# Umsetzung in Java

- Thread
  - OOP-Darstellung eines Threads der JVM
- Runnable
  - Trennt Anwendungslogik von dem Thread selbst
- Synchronized Keyword
  - atomare Ausführung eines Programmabschnitts
- wait() & notify()
  - Kommunikation zwischen Threads

# Umsetzung in Java

- Volatile Keyword
  - Atomarer Zugriff auf eine Variable oder Referenz
- ThreadLocal
  - Eigene Version einer Variablen pro Thread

# Umsetzung in Java

- Java Memory Model
  - Seit Java 1.5 (JSR 133)
  - Definition und Modernisierung der Unterstützung für Nebenläufigkeit

# Umsetzung in Java

- Lock
  - Ersatz für synchronized
  - Bietet erweiterte Möglichkeiten zum Synchronisieren von Code
- Condition
  - Ersatz für wait() / notify()
  - Erweiterung der Thread-Koordination

# Umsetzung in Java

- Threadsafe Collections
- Executor Interface
  - Allgemeine Schnittstelle für das Ausführen von Tasks
- Executors Klasse
  - Factory für vorgefertigte Threadpools und Scheduler

# Umsetzung in Java

- Futures
  - Rückgabewert eines Tasks
- Callable
  - Runnable mit Rückgabewert



# Umsetzung in Java

- Beispiel – Kontoüberweisung

# Praxis

- Deadlocks
  - Speisende Philosophen
  - Java Monitoring & Management Console
  - Erfahrungen



# Praxis

- Scheduling
  - Garten Beispiel
  - Scheduling mit dem Spring Framework

# Praxis

- Quartz – Job Scheduling Framework
  - Entwicklung von Terracotta
  - Open Source Framework
  - Aktuelle Version: 1.8.0
  - Job Persistence
  - Unterstützung für JTA Transactions
  - Clustering



# Praxis

- Threadpooling
  - Autofabrik Beispiel

# Quellen

- <http://www.gotw.ca/publications/concurrency-ddj.htm>
- Wikipedia