Objekte

Ziele der Verwendung von Objekten

für große, komplexe Programme nötig:

- viele Algorithmen zu Einheit integrieren
- inkrementelle Softwareentwicklung, einfache Wartbarkeit
- viele unterschiedliche Strukturierungsmöglichkeiten

Objekte helfen dabei durch:

- menschliche Fähigkeiten im Umgang mit Objekten
- Trennung zwischen Innen- und Außenansicht
- Verständnis als abstrakte Maschinen, verteilte Kontrolle

Faktorisierung

- Faktorisierung = Art der Zerlegung des Ganzen in Teile
- gute Fatorisierung $\Rightarrow$ einfach änderbar
- Faktorisierungsmöglichkeiten:
  - seiteneffektfreie Funktionen (Ausdrücke, ~~Zuweisungen~~)
  - Prozeduren mit Seiteneffekten (wie auf Arrays)
  - Prozeduren und Variablen zu Objekten zusammenfügen

Paradigmen und Programmzustände

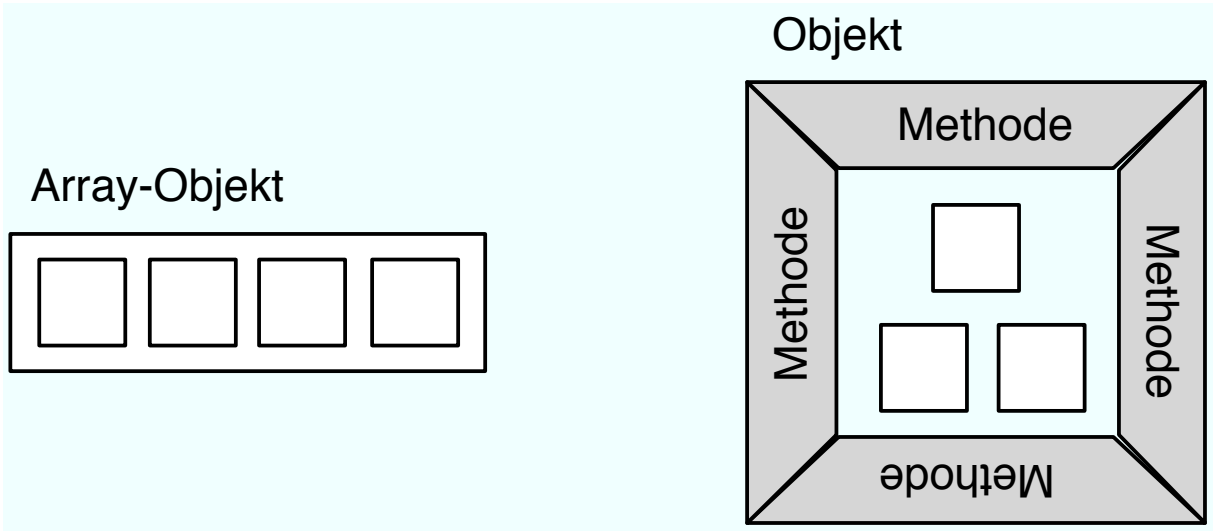
funktional: kein Programmzustand sichtbar

prozedural: nur globaler Programmzustand verständlich

objektorientiert: jedes einzelne Objekt verständlich

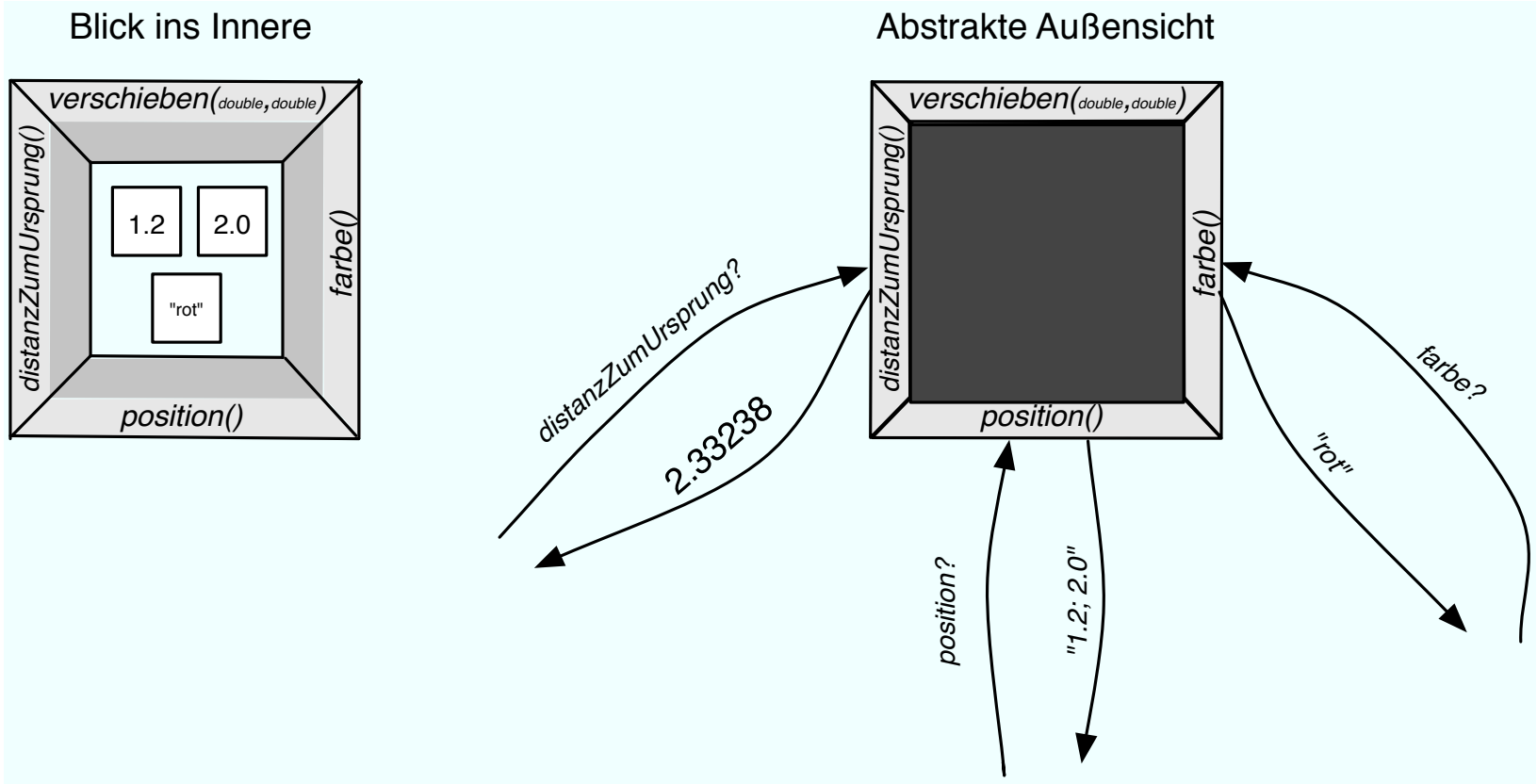
- objektorientiert ideal für große, komplexe Programme
- alles andere besser für kleine, einfache Programme

Kapselung



Kapselung = Objektvariablen und Methoden als Einheit

Datenabstraktion



Datenabstraktion = Kapselung + Data Hiding

Objekteigenschaften

Gleichheit: gleicher Zustand, kann sich jederzeit ändern

Vergleich: `s.equals(t)` (nur für Referenztypen)

Identität: unveränderlich, entsteht bei Objekterzeugung

Vergleich: `s == t` (nicht für String-Vergleich)

bei elementaren Typen entspricht Gleichheit der Identität

Verhalten: unveränderlich, nur von Klasse abhängig

abstrakt, kein dynamischer Vergleich möglich