University of Basel

Software Engineering

Marcel Lüthi, Universität Basel

Modularität

Was ist ein Modul

Ein Modul ist ein wohldefinierter Teil einer Software

Beispiele

- Eine Funktion/Methode
- Eine Sammlung von Funktionen/Methoden
- Eine Sammlung von Daten
- Eine Klasse / Ein Objekt
- Ein Package

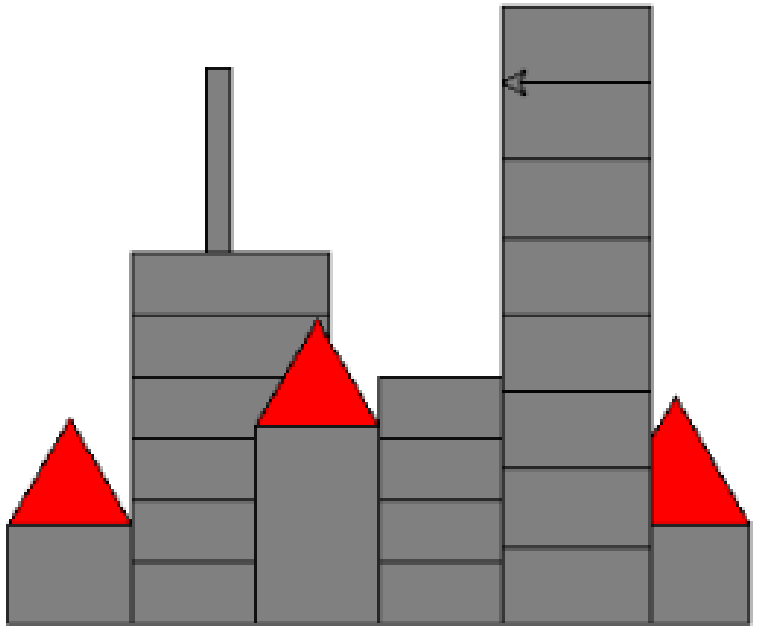
```
public final class Math {
    public static final double E = 2.718281828459045235 4;
    public static double sin(double a) { /* …*/ }
    public static double tan(double a) { /* …*/ }
    …
}
```

Dekomposition und Komposition

Modularität erlaubt, ein System

1. in einfachere Teile zu zerlegen
2. aus einfacheren Teilen zusammenzubauen (Lego)
3. in kleinen Blöcken verstehen zu können.
4. zu verändern, indem nur ein kleiner Teil (ein Modul) verändert wird.

Modularisierung ist im Zentrum der Software-Entwicklung



Modularität: Top-down vs bottom up

Top-down design

Modularität erlaubt “Teilen der Verantwortlichkeiten” in zwei Phasen:

1. Ausarbeiten der Details (eines Moduls)
   – Andere Module werden ignoriert

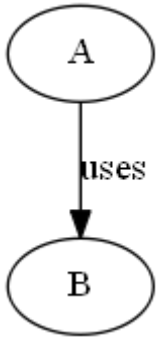
2. Integration in Gesamtes unter Berücksichtigung der Beziehungen zwischen den Modulen.
   – Vernachlässigung der Details der Module.

Bottom-up design

Modulbeziehungen: “Uses” Beziehung

A "uses" B

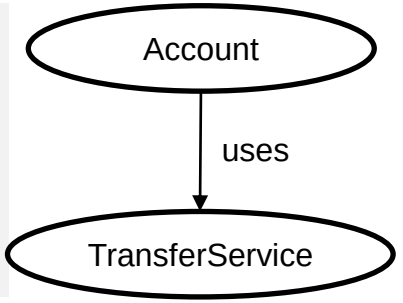
- Module A nutzt Funktionalität von B
- B stellt Funktionalität zur Verfügung



auch

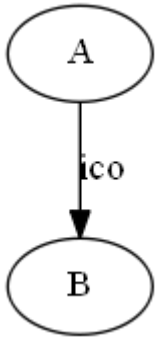
- A ist der "client", B der "server"

```
class Account {
       public static void transferMoney(int amount, Account to) {
           TransferService.transfer(this, to, amount);
       }
}
```



Modulbeziehungen 2, Is component of (Komposition)

- Modul ist aus einfacheren Modulen zusammengesetzt
- Auch Aggregation oder Komposition genannt.
- Beschreibung eines Moduls auf höherer Abstraktionsebene.




```
class Browser {
	JavaScriptEngine jsEngine;
	RenderingEngine renderingEngine;

	void renderHTML(HTML doc) {
		jsEngine.executeJS(doc);
		renderingEngine.render(doc);
	}
}
```