

**OOP mit Java**

**Gültigkeitsbereiche**

**von**

**Attributen, Variablen, Parametern**

# Gültigkeitsbereich: Attribut

```
class Konto  
{  
    double stand;  
  
    . . .  
  
}
```

# Gültigkeitsbereich: Attribut

```
class Konto
```

```
{
```

```
double stand;
```

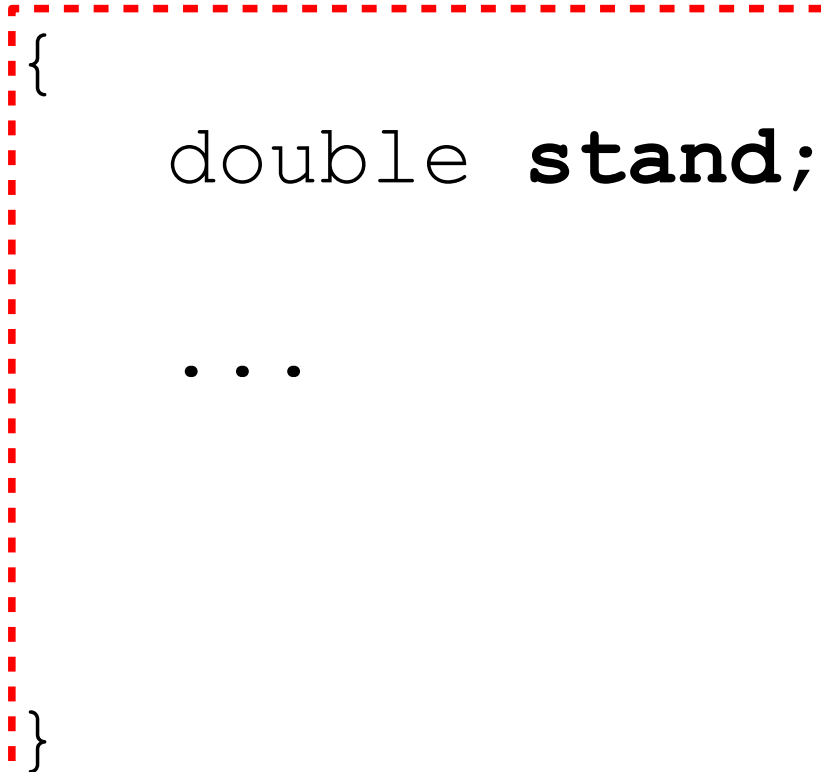
```
...
```

Deklaration

```
}
```

# Gültigkeitsbereich: Attribut

```
class Konto
{
    double stand;
    ...
}
```



Gültigkeitsbereich:  
die ganze Klasse, von { bis } → „**global**“



# Gültigkeitsbereich: Attribut

```
class Konto
{
    double stand;

    void zinsen()
    {
        stand = stand * 1.05;
    }
}
```



Attribut auch „gültig“ in Methoden,  
da sie innerhalb der Klasse stehen

# Gültigkeitsbereich: Parameter

```
class Konto
{
    ...
    void einzahlen(double pb)
    {
        ...
    }
    ...
}
```

# Gültigkeitsbereich: Parameter

```
class Konto  
{
```

```
...
```

```
void einzahlen(double pb)
```

```
{
```

```
...
```

```
}
```

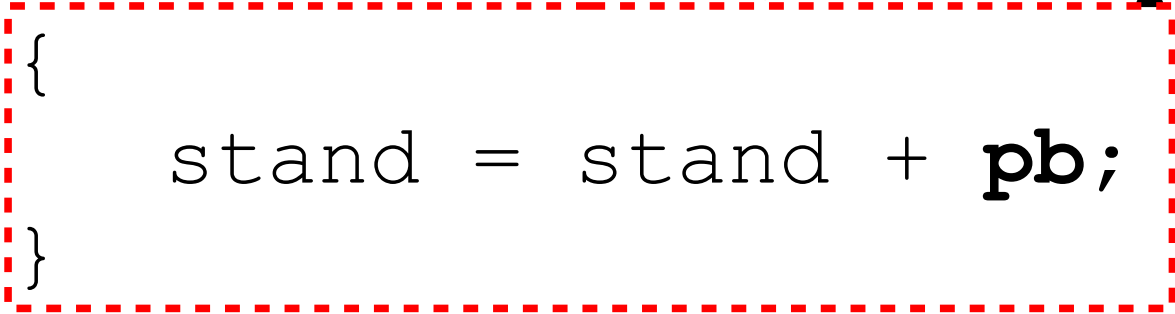
```
...
```



Deklaration

# Gültigkeitsbereich: Parameter

```
class Konto
{
    ...
    void einzahlen(double pb)
    {
        stand = stand + pb;
    }
    ...
}
```



Gültigkeitsbereich:  
der Methodenkörper,  
von { bis }



# Gültigkeitsbereich: Parameter

```
class Konto  
{
```

```
    ...
```

```
    void einzahlen(double pb)
```

```
    {
```

```
        stand = stand + pb;
```

```
    }
```

```
    void bonus()
```

```
    {
```

```
        stand = stand + pb;
```

```
    }
```

hier nicht gültig,  
Compilerfehler



# Gültigkeitsbereich: Parameter

```
class Konto  
{
```

```
...
```

```
void einzahlen(double pb)
```

```
{
```

```
    stand = stand + pb;
```

```
}
```

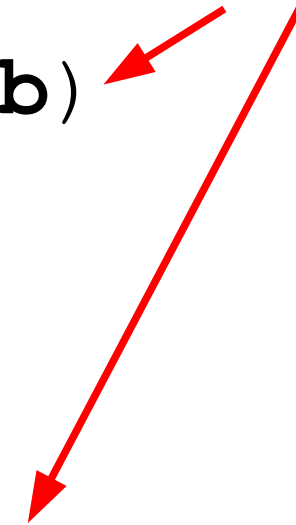
```
void auszahlen(double pb)
```

```
{
```

```
    stand = stand - pb;
```

```
}
```

zwei verschiedene  
Parameter mit dem  
gleichen Namen



# Gültigkeitsbereich: lokale Variable

```
class Game
{
    ...
    void aufgabe1()
    {
        int zähler;
        zähler = 0;
        ...
    }
    ...
}
```

# Gültigkeitsbereich: lokale Variable

```
class Game
{
    ...
    void aufgabe1()
    {
        int zähler;
        zähler = 0;
        ...
    }
    ...
}
```

← Deklaration

# Gültigkeitsbereich: lokale Variable

```
class Game
{
    ...
    void aufgabe1()
    {
        int zähler;
        zähler = 0;
        ...
    }
    ...
}
```

← Gültigkeitsbereich:  
Methodenkörper,  
von { bis }

→ „**lokal**“  
d.h. nicht global  
in der Klasse

# Gültigkeitsbereich: lokale Variable

```
...  
void aufgabe1()  
{  
    int zähler;  
    zähler = 0;  
    ...  
}
```

```
void aufgabe2()  
{  
    zähler = 20;  
    ...  
}
```

**hier nicht gültig,  
Compilerfehler**



# Global (Attribut) vs. Lokal (Variable)

**Global** (als Attribut) sollten nur Daten deklariert werden, deren Werte ein Objekt **dauerhaft** behalten soll.

Beispiel:

Kontostand

Wird durch Methoden mal erhöht, mal vermindert, darf aber zwischendurch nicht „vergessen“ werden.

# Global (Attribut) vs. Lokal (Variable)

Daten, die nur **kurzfristig** benötigt werden, sollten als **lokale** Variablen deklariert werden.

Beispiele:

Zähler für Schleifen

kurzfristig benötigte Farben

Die Werte dieser Variablen werden nur für eine Methode gebraucht und können ansonsten „vergessen“ werden.



# Autor / Quellen

Autor:

- Christian Pothmann (cpothmann.de)  
Freigegeben unter CC BY-NC-SA 4.0, März 2021

