**OOP in** 

**Interfaces**

**+ Vergleich Abstrakte Klassen**

Über das kürzlich kennengelernte Vererbungskonzept (in Java) hinaus gibt es noch eine weitere Möglichkeit, Vererbungsstrukturen zu implementieren, und zwar über ein so genanntes „Interface“, was so viel wie „Schnittstelle“ bedeutet und unter anderem das Problem der sonst nicht möglichen Mehrfachvererbung beseitigt. Ohne auf Details einzugehen kann man sich ein Interface als eine Abstrakte Klasse vorstellen, die keine üblichen Klassenattribute enthält (nur Konstanten sind erlaubt) und bei der alle Methoden abstrakt sind. Im Folgenden aber Näheres dazu:

### Deklaration eines Interface

- Interfaces definieren das Verhalten von Objekten.
- Interfaces enthalten:
  - Instanzmethoden
  - Konstanten
- Interfaces haben keine Konstruktoren.
- Interfaces enthalten keine Implementierung.

```
interface ICustomer {
    public String getName();
    public void setName(String name);
}
```

### Verwendung eines Interface

- Man sagt: *Eine Klasse implementiert ein Interface*, wenn sie jede einzelne Methode des Interfaces bereitstellt.
- Eine Klasse kann beliebig viele Interfaces implementieren.
- Ein Interface kann von beliebig vielen Klassen implementiert werden.
- Eine Referenz auf ein Interface kann überall dort im Code eingesetzt werden, wo auch eine Referenz auf eine Klasse erlaubt ist.

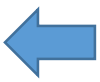
```
class MyCustomerImpl implements ICustomer {
    public MyCustomerImpl(String name) {...};
    public String getName() {...};
    public void setName(String name) {...};
}
```

**Interface vs. Abstrakte Klasse**

- Abstrakte Klasse:
  - Methoden können vollständig (und für alle Unterklassen damit gleichermaßen gültig) „ausprogrammiert“ oder (nur) als „abstract“ deklariert werden (, wobei die konkrete Implementierung dann ausbleiben kann oder in den Unterklassen erfolgt)
  - für das Erstellen von Instanzen sind erbende Subklassen von Nöten
- Interface:
  - nichts kann „ausprogrammiert“ werden
  - für das Erstellen von Instanzen sind Klassen notwendig, die das Interface mithilfe des Schlüsselwortes „**implements**“ einbinden – in diesen Klassen **müssen** alle im Interface definierten Methoden implementiert werden (dies kann bei mehrfachem Implementieren eines Interfaces in ähnlichen Klassen zu Redundanzen führen)

**Vererbung von Klassen bzw. Interfaces**

- eine Klasse kann nur von einer Klasse erben, aber
- **eine Klasse kann mehrere Interfaces implementieren**
- **ein Interface kann von mehreren Interfaces erben**



beides ist auch problemlos gleichzeitig möglich!