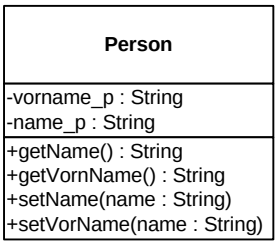
Interface

Eine Klasse enthält Methoden und Datenfelder. Methoden bestehen aus Methodenköpfen und Methodenrümpfen. Methodenköpfe stellen die **Schnittstellen** eines Objektes zu seiner Aussenwelt dar



Klassenname

Datenfelder

Methodenköpfe

```
public class Person {

  public Person() {
  }

  private String name_p;
  private String vorname_p;

  public String getName() {
   return name_p;
  }
  public String getVorName() {
   return vorname_p;
  }
  public void setName(String name) {
   this.name_p = name;
  }

  public void setVorName(String name) {
   this.vorname_p = name;
  }
}
```

Aus der Theorie über die Datenkapselung oder auch Information Hiding wissen wir:

> In einer guten Implementation sind die Daten im Innern des Objektes verborgen und nach aussen sind nur die öffentlichen Methoden sichtbar.

Trennung von Spezifikation und Implementation

Beim Entwurf, auch Design genannt, interessiert man sich zunächst mehr für die Schnittstellen einer Klasse und erst später kümmert man sich wie die einzelnen Methoden implementiert werden. Dieses Konzept wird durch die speziell in Java vorhandenen Sprachmittel **Interface** und **implements** unterstützt und gefördert.

Interface (deklariert die Schnittstelle)

In Java werden Interfaces sehr häufig angewendet, auch in der Java-Klassenbibliotheken wird reger Gebrauch von diesen Sprachelementen gemacht. (z.B. die *Interfaces java.lang.Runnable*, *java.util.Enumeration*)

Beispiel:

```
// Datei: PunktSchnittstelle.java

interface PunktSchnittstelle {

  public int getX(); // öffentliche Schnittstelle für Zugriff auf X (lesen)

  public void setX ( int i ) ;   // öffentliche Schnittstelle für Zugriff auf X (schreiben)

}
```

implements (Methoden werden ausprogrammiert)

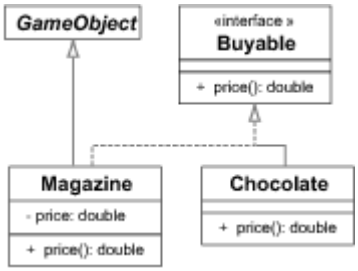
```
// Datei: Punkt.java
public class Punkt implements PunktSchnittstellen {

  int x; // x-Koordinate

  public int getX (  ) {   // Methode um x-Werte zu bekommen
     return x;
  }
  public void setX ( int i ) {   // Methode um x-Werte zu setzen
     x = i;
  }
  public static void main ( String[] args )  {
     Punkt p =  new Punkt( );
    p.setX(3);

   System.out.println ( "Die Koordinate des Punktes p ist:" );
   System.out.println ( p.getX());
  }
```

}

Ausgabe:

Die Koordinate des Punktes p ist: 3

Darstellung in UML (Unified Modelling Language)




```
public class Magazine extends GameObject implements Buyable
{
  double price;

  @Override public double price()
  {
    return price;
  }
}
```