Vorteile von Klassen

- **Kompaktheit**
- **Zusammenfassung komplexer Datenstrukturen**
- **Information Hiding**

Definition von Klassen

z.B.:

```
class datum
{
        byte tag;
        byte monat;
        int jahr;
...
public void naechstertag();
...
public string monatsname();
...
}
```

Klasse Auto1

```
class Auto1{
      public
       String eigentuemer;
       String farbe;
       String marke;
       double geschwindigkeit;
       double richtung;

       Auto1() {}  //Konstruktor ohne void
                   //bei neueren Versionen

}
```

Programmcode Autofahren1

```
public class Autofahren1 {
        public static void main(String args[]) {
                Auto1  meinAuto, deinAuto;

                meinAuto = new Auto1();
                deinAuto = new Auto1();

                meinAuto.eigentuemer="JG";
                meinAuto.farbe="schwarz";
                meinAuto.marke="Fiat";

                deinAuto.marke=“Ferrari“;

                deinAuto.geschwindigkeit=350;

        }
}
```

Klasse Auto2

```
class Auto2{
      public
      String eigentuemer;
      String farbe;
      double geschwindigkeit;
      double richtung;

      void lenken(double a) {richtung=a;};

      void beschleunigen(double a, double b)
            {richtung=a; geschwindigkeit=b;};

      Auto2() {};
}
```

Programmcode Autofahren2

```
public class Autofahren2 {
        public static void main(String args[]) {
            Auto2  meinAuto, deinAuto;
            deinAuto = new Auto2();
            meinAuto =  new Auto2();
            ...
            deinAuto.lenken(15);
            ...
            meinAuto.beschleunigen(4.3, 8.0);
            ...
        }
}
```

Der Verweis this

Innerhalb einer Klasse kann ohne die Angabe eines Instanzennamens direkt auf die Variablen der individuellen Instanz zugegriffen werden.

z. B.

```
public void setPoint(int x_uebergabe, int y)
      this.x=x_uebergabe;
      this.y=y;
      }
```

Aufgabe

Klassen

Lösung (1)

```
public class MainPunktStrecke {

	public static void main(String[] args)
{
		Punkt p1=new Punkt();
		Punkt p2=new Punkt(3,4);

		Strecke s1;
		s1=new Strecke();

		p1.x=1;
		p1.y=2;

		s1.abstand(p1,p2);
		System.out.print(s1.s);
	}
}
```

Lösung (2)

```
public class Punkt {
      int x,y;

Punkt(){}

Punkt(int x, int y){
this.x=x;
this.y=y;
}
}
```

Lösung (3)

```
import java.lang.Math;

public class Strecke {
double s;

void abstand(Punkt p1, Punkt p2){
     s=Math.sqrt((p1.x-p2.x)*(p1.x-
p2.x)+(p1.y-p2.y)*(p1.y-p2.y));
}
}
```

Kopieren von Instanzen (1) (Klasse)

```
public class Werte {
    int x;
    int y;

public Werte() {
    }

}
```

Kopieren von Instanzen (2)

```
public static void main(String[] args) {
        Werte w1,w2;
        w1= new Werte();
        w2= new Werte();
        w1.x=1;  w1.y=2;
        w2=w1;
        System.out.println(w1.x);
        System.out.println(w1.y);
        w2.x=3;
        w2.y=4;
        System.out.println(w1.x);
        System.out.println(w1.y);
}
```

Kopieren von Instanzen (3)

```
public static void main(String[] args) {
        Werte w1,w2;
        w1= new Werte();
        w2= new Werte();
        w1.x=1;      w1.y=2;
        w2.x=w1.x;   w2.y=w1.y;
        System.out.println(w1.x);
        System.out.println(w1.y);
        w2.x=3;
        w2.y=4;
        System.out.println(w1.x);
        System.out.println(w1.y);
}
```

Das Konzept der Vererbung ist zentrale Eigenschaft von Klassen

- Klassen können Eigenschaften anderer Klassen *erben*.
- Klassen können durch Vererbung *weiterverwendet* und
- zusätzlich *verfeinert* werden.

Klasse und Object

Jede Klasse ist eine Unterklasse der Klasse Object.

In der Klasse Object sind bereits verschiedene Methoden definiert. Diese können mit neuem Verhalten überschrieben werden. Hierzu gehören

- **toString – Umwandlung in einen String**

```
//Standardimplementierung
public String toString() {
      return getClass().getName() + "@" +
      Integer.toHexString(hashCode()); }
```

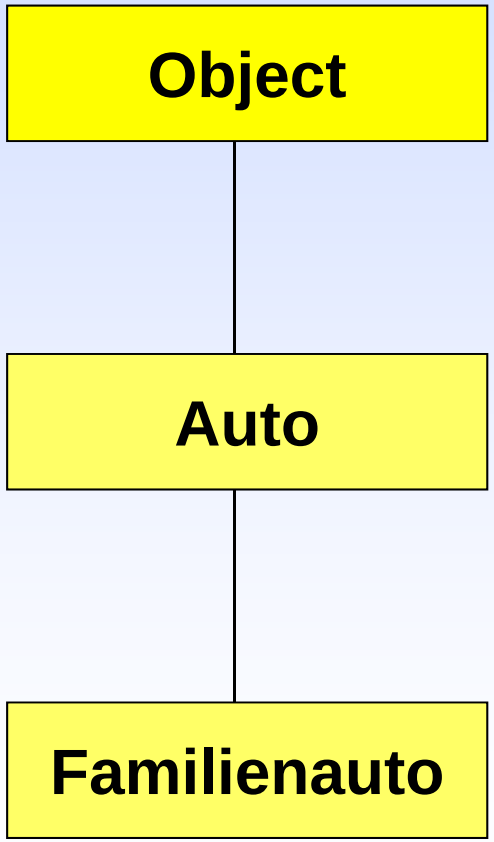
- **clone - zum Kopieren**

Muss individuell überschrieben werden

Basisklasse und Subklasse

Basisklasse,
Superklasse

Subklasse,
erweiterte Klasse,
abgeleitete Klasse



Programmcode Familienauto

```
class Familienauto extends Auto1 {

      public boolean kindersitz;

}
```

Aufgabe

Vererbung: Erweiterung der Autoklasse