Abstrakte Methoden

- Abstrakte Methoden legen lediglich die Signatur der Methode fest, ohne sie zu implementieren.

- Sie sind durch das Schlüsselwort `abstract` gekennzeichnet und haben keinen Rumpf (Anweisungsblock).

```
public abstract void aendernKlang (String klang);
```

- Abstrakte Methoden müssen in Unterklassen überschrieben werden.

Abstrakte Klassen

- Abstrakte Klassen bestimmen lediglich die Struktur einer Klasse. Es können keine Exemplare von abstrakten Klassen erzeugt werden, nur von deren (nicht-abstrakten) Unterklassen, die alle abstrakten Methoden implementieren.

- Eine abstrakte Klasse darf abstrakte Methoden besitzen. Daneben darf es auch implementierte Methoden und Datenelemente geben.

- Kommt in einer Klasse eine abstrakte Methode vor, so muss die Klasse selbst ebenfalls abstrakt vereinbart sein.

Abstrakte Klassen (2)

- Abstrakte Klassen werden mit dem Schlüsselwort `abstract` vereinbart.

```
public abstract class Bewegliches {

protected String klang;

protected boolean lebend;

public abstract void aendernKlang(String klang);
public abstract void rede();

}
```

Casting von Verweisdatentypen

1. **Indikation:**

   - Variable `var` vom Typ Verweis auf Instanzen der Klasse `A`
   - Zuweisung eines Exemplars `exemplarB` einer Unterklasse `B` von `A`
   - Aufruf einer Methode `methode()`, die in `B`, nicht aber in `A` existiert

   Syntax:
   ```
   A var = exemplarB;
   ((B)var).methode();
   ```

2. **Indikation:** Zuweisung eines Verweises auf eine Instanz der Klasse `A` an eine Variable vom Typ `B`, wobei `B` Unterklasse von `A` ist

   Syntax:
   ```
   A varA = exemplarA;
   B varB = (B)varA;
   ```

Interfaces

- Interfaces sind reine Schnittstellen, die keinerlei Implementierung enthalten.
- Alle Methoden sind implizit `abstract`, alle Datenelemente sind implizit final static (ohne Angabe dieser Schlüsselwörter!).

```
public interface Bewegliches {

public void aendernKlang(String klang);
public void rede();

}
```

Interfaces (2)

- Eine Klasse kann ein oder mehrere Interfaces implementieren.

  ```
  public class Tier implements Bewgliches, Lebend { ... }
  ```

  Dann müssen alle Methoden dieser Interfaces überschrieben werden.

- Die Eigenschaft einer Klasse, ein Interface zu implementieren, wird an ihre Unterklassen vererbt.

- Ob ein Objekt `objekt` ein Exemplar einer Klasse `klasse` ist, die ein Interface `interf` implementiert, kann zur Laufzeit mit dem `instanceof`-Operator geprüft werden:

  ```
  object instanceof interf   // true falls klasse interf implementiert
  ```