

```
public class Beitrag
{
    private String username;
    private long timestamp;
    private int likes;
    private ArrayList<String> comments;

    /**
     * Initialisiere die Felder von Beitrag
     */
    public Beitrag(String author)
    {
        username = author;
        timestamp = System.currentTimeMillis();
        likes = 0;
        comments = new ArrayList<>();
    }

    // weitere Methoden
}
```

Der Konstruktor der Superklasse initialisiert die in der Superklasse festgelegten Attribute – logo!

```

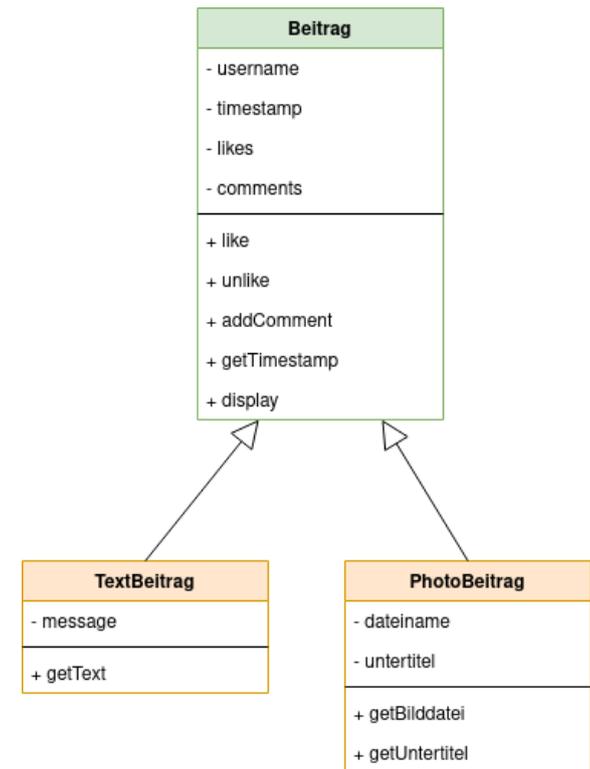
public class TextBeitrag extends Beitrag
{
    private String message;

    /**
     * Konstruktor fuer TextBeitrag-Objekte
     */
    public TextBeitrag(String author, String text)
    {
        // username muss irgendwie auf author
        // gesetzt werden?!

        // das ist klar.
        message = text;
    }

    // methods omitted
}

```



Bei der Instanziierung eines Objekts einer abgeleiteten Klasse wird zunächst **immer der Konstruktor der abgeleiteten Klasse** aufgerufen

**Problem:** Auch der `TextBeitrag` hat einen Autor – bei der Instanziierung eines `TextBeitrags` muss also das von der Superklasse geerbte Attribut `username` mit dem Parameter `author` initialisiert werden?! → Wie?!

```

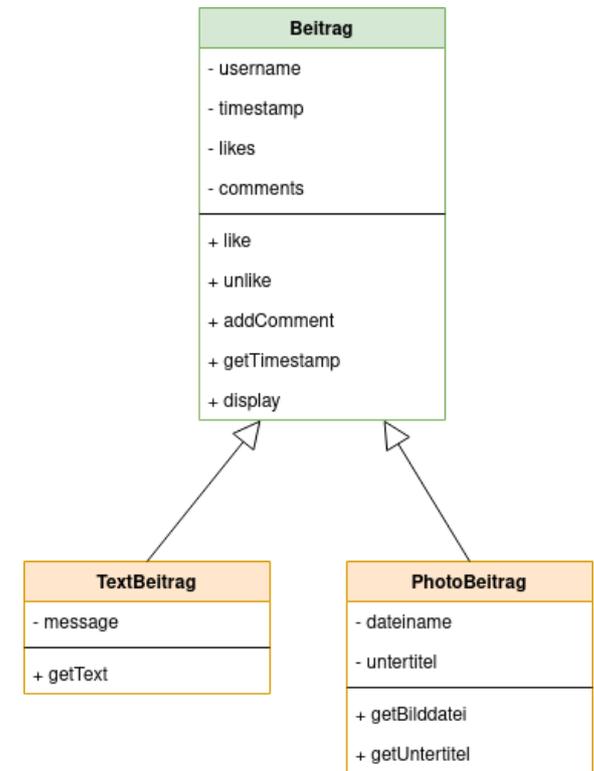
public class TextBeitrag extends Beitrag
{
    private String message;

    /**
     * Konstruktor fuer TextBeitrag-Objekte
     */
    public TextBeitrag(String author, String text)
    {
        // Konstruktor der Superklasse aufrufen:
        super(author);

        // das ist klar.
        message = text;
    }

    // methods omitted
}

```



**Lösung:** der Konstruktor der abgeleiteten Klasse ruft **immer** den Konstruktor der Superklasse auf:

```
super (...parameter...);
```

So werden alle geerbten Attribute initialisiert, wenn ein Objekt der abgeleiteten Klasse instanziiert wird.

**Achtung:** Wenn der `super`-Aufruf nicht explizit angegeben wird, wird `super` implizit ohne Argumente aufgerufen!

- Die Konstruktoren abgeleiteter Klassen müssen **immer** einen Aufruf des Konstruktors der Superklasse (**super**) beinhalten.
- Wenn der Programmierer keinen `super`-Aufruf in seinen Code einfügt, macht das der Compiler. Dann wird `super` ohne Parameter aufgerufen – **das geht schief, wenn der Konstruktor der Superklasse Parameter benötigt.**
- Der `super`-Aufruf muss das **erste Statement** im Konstruktor der abgeleiteten Klasse sein.