```
class Animal {
}.....
```

Anmerkung: Klasse wird nur an einer einzigen Stelle benutzt

```
class Monster extends Animal {

    public boolean likes(String food) { return food.equals("blood"); }
    public void speak() { getAngry(); }
    public void getAngry() { out.println("grrrhhh"); }

}
```

Besser:

abstract

```
Animal monster = new Animal() {

    public boolean likes(String food) { return food.equals("blood"); }
    public void speak() { getAngry(); }
    public void getAngry() { out.println("grrrhhh"); }

};
```

anonyme Klasse

- wird direkt bei der new-Operation angegeben
- definiert eine namenlose Unterklasse von Animal
- erlaubt es, in dieser Unterklasse Methoden zu überschreiben
- Anonyme Klassen haben keinen Konstruktor (man kann aber Parameter für den Oberklassenkonstruktor angeben)
- erspart die Deklaration einer Klasse Monster
- für einmalige Verwendung an einer einzigen Stelle
- geht auch zur Implementierung von Interfaces (siehe später)