Kapselung

openHPI-Java-Team

Hasso-Plattner-Institut

Kapselung

HPI Hasso Plattner Institut

```java
public class Parrot {
    String name;
    int age;
    boolean alive;
}
```

Kapselung

- Verhindert, dass direkter Zugriff von außen erfolgt
- Schützt Attribute vor unkontrollierter Veränderung

Kapselung

```
1 class Detective {
2     private String pinCode;
3 }
4
```

Kapselung

- Verhindert, dass ungewollter Zugriff auf Werte erfolgt

Getter und Setter

```
1  public class Parrot {
2     private String name;
3
4     public String getName() {
5         return this.name;
6     }
7     public void setName(String value) {
8         this.name = value;
9     }
10 }
```

Private

- Konvention: Alle Attribute `private`
- Zugriff optional über `public` Getter/Setter möglich
  - Ermöglichen es Entwicklern anderen Objekten Lese- bzw. Schreibrechte auf Attribute einzuräumen
  - Vollständige Kontrolle des Zugriffs

Getter und Setter

Hasso Plattner Institut

```java
public class Parrot {
    private String name;

    public String getName() {
        return this.name;
    }
    public void setName(String value) {
        this.name = value;
    }
}
```

Getter und Setter

- Das Attribut kann von außen gelesen und geschrieben werden

Getter und Setter

```
1 public class Parrot {
2    private String name;
3
4    public String getName() {
5        return this.name;
6    }
7 }
```

Getter

- Das Attribut kann von außen gelesen werden

Getter und Setter

Hasso Plattner Institut

```java
public class Parrot {
    private String name;

    public void setName(String value) {
        this.name = value;
    }
}
```

Setter

- Das Attribut kann von außen geschrieben werden

Validierung von Werten

```
1  public class Parrot {
2     protected int age;
3
4     public void setAge(int newAge) {
5         if (newAge > 0) {
6                 this.age = newAge;
7         } else {
8                 this.age = 1;
9         }
10    }
11 }
```

Kapselung

- Ermöglicht die Validierung von Attribut-Werten

Kapselung

```
1  public class Robot {
2          private boolean stealthMode = false;
3
4          public void enableStealthMode() {
5          stealthMode = true;
6                  this.log("Tarnmodus aktiviert");
7          }
8
9          private void log(String logMessage) {
10                 sendDataToDuke(logMessage);
11         }
12 }
```

- Die Implementierung kann geändert werden ohne die Zusammenarbeit mit anderen Klassen zu beeinträchtigen
- Kapselung sorgt für Modularisierung

Kapselung

openHPI-Java-Team

Hasso-Plattner-Institut