Inhaltsverzeichnis

1. Access Modifier (Sichtbarkeit)

[Inhalt | Demo | Übungen]

Die Standard **Access Modifier** - mittels Schlüsselwort - sind die folgenden:

- `public`,
- `protected` und
- `private`.

Fehlt eine solche Angabe, wirkt ein Standard, der als *default* bezeichnet wird.

Die "Access Modifier" bedeuten:

1.1. *default* (Fehlen eines Sichtbarkeits-Schlüsselwortes)

Wenn kein Schlüsselwort explizit verwendet wird, legt Java einen Standardzugriff auf eine bestimmte Klasse, Methode oder Eigenschaft fest. Der **Standardzugriffs**-Modifikator wird auch `package-private` genannt. Dies bedeutet, dass alle Mitglieder (= Konstruktoren, Methoden & Felder) innerhalb desselben **Pakets** sichtbar sind, aber aus Klassen in anderen Paketen heraus nicht darauf zugegriffen werden kann.

1.2. `public`

Wenn einer Klasse, Methode oder Eigenschaft das Schlüsselwort `public` hinzufügt wird, wird es auf der **"ganzen Welt"** verfügbar/sichtbar, d.h. alle anderen Klassen in allen Paketen können es verwenden. Dies ist der am wenigsten restriktive Zugriffsmodifikator.

1.3. `protected`

Wenn eine Methode, Eigenschaft oder ein Konstruktor mit dem Schlüsselwort `protected` deklariert wird, kann auf das Mitglied aus demselben **Paket** zugegriffen werden (wie bei der Zugriffsebene *package-private*) und zusätzlich aus allen **Unterklassen** dieser Klasse, auch wenn diese in anderen Paketen liegen.

1.4. `private`

Auf jede Methode, Eigenschaft oder jeden Konstruktor mit dem Schlüsselwort `private` kann nur von derselben **Klasse**, also von *"innen"*, zugegriffen werden. Dies ist der restriktivste Zugriffsmodifikator und bildet den Kern des Konzepts der Kapselung. Alle Daten werden vor der Außenwelt verborgen.

*Das "Verstecken" von Methoden oder Feldern durch Nutzung vom `private` Modifizierer ist häufig gemeint, wenn vom allgemeinen Prinzip **Information Hiding** die Rede ist.*

Demo:

```
→ src/test/java/de/dhbw/demo/VisibilityDemoTest.java
```

```
    @Test
    @DisplayName("Demo 3: Sichtbarkeiten")
    public void canCheckVisibility() {
        // given
        VisibilityExampleClass someClass = new VisibilityExampleClass();

        // What is the reason for fields A, C and D being not accessible?

        //someClass.fieldA
        someClass.fieldB = "some value for field B";
        //someClass.fieldC
        //someClass.fieldD

        // when
        String fieldBValue = someClass.fieldB;

        // then
        assertNotNull(fieldBValue);
    }
```

Übungen:

keine