Datenkapselung: vollständiger Überblick

AvHG, Inf, My

Datenkapselung

Bisher kennen wir für die Datenkapselung nur die Schlüsselworte `public` und `private`. Mit `private` gekennzeichnete Methoden können jedoch in einer abgeleiteten Klasse nicht benutzt werden. Wie kann man dafür sorgen, dass eine Methode zwar von abgeleiteten Klassen benutzt werden kann, aber für andere Klassen (zumindest aus anderen Packages) versteckt ist? Antwort: man benutzt das Schlüsselwort **`protected`**.

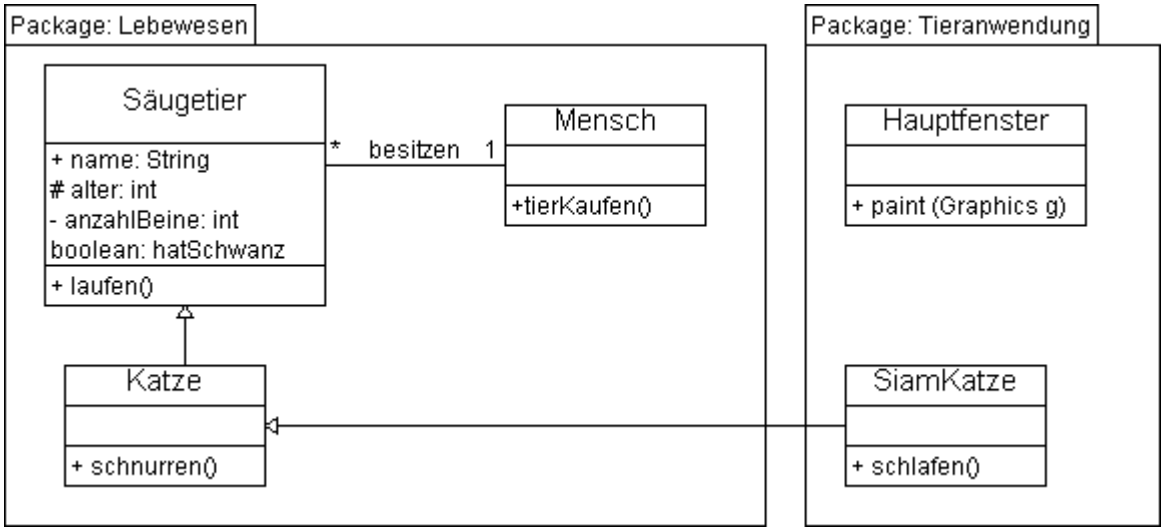
Die folgende Tabelle gibt einen vollständigen Überblick über die Möglichkeiten der Datenkapselung in Java:

Attribut (UML-Zeichen)	**Klasse**	**abgeleitete Klasse**	**Package**	**andere Packages**
private (–)	+	–	–	–
protected (#)	+	+ (*)	+	–
public (+)	+	+	+	+
<kein Angabe>	+	–	+	–

(*) Ableitung in fremdem Paket: Zugriff nur bei Objekten vom Typ der abgeleiteten Klasse nicht in Objekten vom Typ der Basisklasse.

Übung zur Datenkapselung

Das UML-Klassendiagramm zeigt mehrere Klassen, die sich in zwei verschiedenen Packages befinden. Vereinfacht kann man sagen, dass ein Package aus allen Dateien in einem Verzeichnis besteht. Wenn man auf Klassen aus einem fremden Package zugreifen möchte, muss man die Klassen mit Hilfe einer `import`-Anweisung in die Datei einbinden.



Auf welche der Variablen aus der Klasse Säugetier kann man von den anderen Klassen aus zugreifen?

Variable	Methode **schnurren()** für ein Katzen-Objekt	Methode **tierKaufen()** für ein Mensch-Objekt	Methode **paint(g)** für ein Hauptfenster-Objekt	Methode **schlafen()** für ein SiamKatze-Objekt
name	ja	ja	ja	ja
alter	ja	ja	nein	ja
anzahlBeine	nein	nein	nein	nein
hatSchwanz	ja	ja	nein	nein