Grundlagen der Programmierung

Dr. Christian Herzog
Technische Universität München

Wintersemester 2007/2008

Kapitel 11: Zugriffskontrolle

Inhalt dieses Kapitels

- Die genaue Kontrolle des Zugriffs auf
  - Klassen,
  - Schnittstellen und
  - Merkmale

  ist wichtige Voraussetzung für sichere Programmierung.
- Die in Java vorhandenen Möglichkeiten zur Zugriffskontrolle werden in diesem Kapitel kurz vorgestellt:
  - Pakete
  - die Modifikatoren **`private`**, **`protected`** und **`public`** und der Standard-Modifikator

Pakete

- In der Regel umfasst ein Java-Programm viele Klassen und Schnittstellen.
- Um Programme strukturierter zu gestalten, können Klassen und Schnittstellen in sog. Paketen (engl.: *packages*) zusammengefasst werden.
- Die Pakete in Java entsprechen damit den Bibliotheken anderer Programmiersprachen.
  - Auch die Java-Standardklassen werden in Paketen untergebracht.
- Ein Paket kann Klassen, Schnittstellen und Unterpakete enthalten. Letztere sind selbst Pakete.
  - Pakete sind also hierarchisch angeordnet.
- Bei Betriebssystemen wie Windows oder UNIX entspricht ein Paket einem Dateiverzeichnis mit allen in diesem Verzeichnis befindlichen Klassen- und Schnittstellendateien
  - Den Unterpaketen entsprechen dann Unterverzeichnisse.

Vollständig qualifizierte Bezeichner

- Ähnlich wie über absolute Pfadangaben in UNIX- oder Windows-Dateisystemen können Klassen und Schnittstellen über ihren **vollständig qualifizierten Bezeichner** direkt angesprochen werden.
- Ein vollständig qualifizierter Bezeichner besteht dabei aus dem Namen des Pakets bzw. der Unterpakete und dem Bezeichner der Klasse bzw. Schnittstelle selbst.
- Die einzelnen Bezeichner werden dabei jeweils durch einen Punkt getrennt.
- **Beispiel**:
  - Der vollständig qualifizierte Bezeichner der Klasse **`FileWriter`** aus dem Paket **`java.io`** lautet **`java.io.FileWriter`**
  - **`io`** ist ein Unterpaket von **`java`**
  - Das Paket **`java`** und seine Unterpakete enthalten die vordefinierten Klassen/Schnittstellen des Java-Systems (Standard-Klassen)

Das namenlose Paket

- Generell müssen alle Klassen und Schnittstellen in Java einem Paket zugeordnet sein.
- Der Zugriff auf sie ist dann nur noch über ihren vollständig qualifizierten Bezeichner möglich.
- Wird die Paketzugehörigkeit nicht explizit vereinbart (wie in allen bisherigen Beispielen im Rahmen der Vorlesung), so werden die Klassen und Schnittstellen einem namenlosen Paket zugeordnet, d.h. der einfache Bezeichner einer Komponente ist auch ihr vollständig qualifizierter Bezeichner.
- In der Regel ist das namenlose Paket auf das Verzeichnis begrenzt, in dem die jeweilige Klassen- bzw. Schnittstellendatei sich befindet.
  - Die Bezeichner von evtl. vorhandenen Unterpaketen entsprechen dann den jeweiligen Unterverzeichnisnamen.

Erstellung von Paketen

- Um eigene Pakete zu erstellen, wird zu Beginn des Programmtextes (vor der Deklaration von Klassen und Schnittstellen) der Paketname explizit angegeben:
  `package` *<Paketname>* **`;`**
- Damit wird festgelegt, dass alle Klassen und Schnittstellen innerhalb der Programm-Datei zu dem Paket *<Paketname>* gehören.
- *<Paketname>* muss ein vollständig qualifizierter Bezeichner sein;
  - bei der Deklaration eines Unterpakets müssen auch die Namen aller Oberpakete (jeweils durch Punkte getrennt) angegeben werden.
- Die Klassen-Dateien (mit Endung .class) müssen sich dann in entsprechend bezeichneten Verzeichnissen und Unterverzeichnissen befinden.

Die `import`-Deklaration

- Im allgemeinen muss eine Klasse oder Schnittstelle mit ihrem vollständig qualifizierten Namen angegeben werden.
- Da solche Bezeichner oft recht lang werden können, kann man **`import`**-Deklarationen einsetzen.
- Dadurch können die Bezeichner auch ohne Angabe ihres Paketes verwendet werden.
  - **`import`** *<Paketname>* **`.`** *<Klassenname>* **`;`**
  - **`import`** *<Paketname>* **`.*;`**

Alle Klassen des angegebenen Pakets können mit ihrer Kurzbezeichnung verwendet werden.

Der angegebene Klassenname kann in seiner Kurzform verwendet werden, solange kein Namenskonflikt auftritt.

- Die Klassen und Schnittstellen des Pakets **`java.lang`** sind immer implizit importiert! Eine **`import`**-Deklaration kann unterbleiben.

Zugriffsbeschränkungen durch Modifikatoren

- Java bietet zur Zugriffsbeschränkung auf Klassen und Merkmale verschieden Modifikatoren an.
- Bisher haben wir nur Modifikatoren für Merkmale verwendet und zwar **`public`**, **`private`** und **`protected`**.
- Es gibt zusätzlich noch den Standard-Modifikator, der immer dann als gesetzt gilt, wenn kein anderer Modifikator angegeben ist.
- Zur exakten Definition der Modifikatoren wird der Begriff des Pakets benötigt.
  - Deshalb werden wir sie erst jetzt nachliefern.

Zugriffskontrolle für Klassen

- Für Klassen und Schnittstellen kann nur **`public`** oder der Standard-Modifikator (d.h. kein Modifikator) angegeben werden.
- Falls eine Klasse oder Schnittstelle mit dem Modifikator **`public`** (vor dem Schlüsselwort **`class`** bzw. **`interface`**) versehen ist, dann kann auf sie von jeder Stelle des Programms zugegriffen werden.
- Beim Standard-Modifikator können Zugriffe nur **innerhalb des Paketes** erfolgen, zu dem die Klasse bzw. Schnittstelle gehört.

Zugriffskontrolle für Merkmale (Attribute, Konstruktoren, Operationen)

- Der Modifikator **`public`** wird für uneingeschränkte Zugriffsrechte verwendet.
  - Auch Zugriffe aus anderen Paketen heraus sind ohne Einschränkung möglich.
- Beim Modifikator **`protected`** sind Zugriffe auf die Merkmale nur noch von **innerhalb desselben Pakets** möglich.
  - Die einzige Ausnahme sind Unterklassen der Klasse, zu der das Merkmal gehört.
  - Diese können auf das Merkmal zugreifen, auch wenn sie selbst einem anderen Paket angehören.

Zugriffskontrolle für Merkmale (cont'd)

- Wird einem Merkmal keiner der Modifikatoren **`public`**, **`protected`** oder **`private`** vorangestellt, so spricht man von Standard-Zugriff oder auch der „**Paket-Sichtbarkeit**".
  - In diesem Fall ist der Zugriff auf das Merkmal erlaubt, sofern er innerhalb desselben Pakets stattfindet.
- Durch **`private`** wird die schärfste Variante der Zugriffskontrolle gekennzeichnet.
  - Merkmale mit diesem Modifikator sind nur innerhalb der Klasse verwendbar, in der sie deklariert werden.
  - Diese Merkmale bleiben auch für Unterklassen unsichtbar.
    - Deshalb haben wir in Kapitel 8 das Merkmal **`protected`** eingeführt, damit aber nicht nur den Zugriff von allen Unterklassen sondern von allen Klassen desselben Pakets erlaubt!