Packages

Packages

Mehrere Klassen können in Java zu einem Paket (engl. **package**) zusammengefasst werden.

- Jede Datei des Pakets muss mit `package <paketname>;` beginnen, wobei `<paketname>` der **Paketname** ist.
- Die Klassen des Pakets müssen alle in einem Unterverzeichnis liegen, dessen Name dem Paketnamen entspricht.
- Der Paketname sollte nur aus Kleinbuchstaben bestehen.
- Da Paketnamen einzigartig sein sollen, wird empfohlen, die umgekehrte Domain des Herstellers voranzustellen, z.B. `de.lmu.tcs.GraphicsWindow`
- Jeder Punkt im Paktenamen entspricht einem Verzeichnis.

Kompilieren von Packages

Der Java Kompiler erwartet Einhaltung der Verzeichnisstrukturen: Wenn z.B. die Klasse `Main` die Deklaration `package de.lmu.tcs;` enthält, dann muss `Main.java` im Unterverzeichnis `de/lmu/tcs/` gespeichert sein. Entsprechend im Verzeichnis (Ordner, Directory) darüber kompilieren:

```
./> javac ./de/lmu/tcs/Main.java
./> java de.lmu.tcs.Main
```

- `.` bezeichnet aktuelles Verzeichnis
- *Windows:* `/` durch `\` ersetzen

Wie wir hier sehen, muss man sich auch zum Ausführen im Verzeichnis darüber befinden, und die Klasse mit der `main`-Methode mit Paketnamen angeben.

Meistens genügt Kompilieren der Datei mit der `main`-Methode, da benutzte Dateien automatisch mit kompiliert werden.

Import

- Auf die Klassen des Pakets greift man durch Vorschalten von `<paketname>.` zu: Klasse `java.util.Arrays` ist völlig unabhängig von Klasse `my.package.Arrays`
  $\Rightarrow$ Pakete teilen den Namensraum auf!
- Wenn man anfangs `import <paketname>.*;` schreibt, kann man alle *eindeutigen* Klassennamen auch direkt ohne vorangestellten Paketnamen verwenden.
- Verzeichnisse implizieren dabei *keine* Hierarchie: `import javafx.*;` importiert nicht automatisch auch noch `import javafx.scene.paint.*;`
- Klassen ohne explizite `package`-Deklaration befinden sich in der **default package**. Seit Java 1.4. können solche nicht innerhalb einer echten package verwendet werden.

Export Man kann Pakete in `.jar`-Dateien zusammenpacken, um diese auszutauschen. Direkte Ausführung ebenfalls möglich.

Sichtbarkeit und Pakete

Pakete unterstützen ebenfalls die Aufteilung eines komplexen Systems durch Gruppierung der Einheiten.

- Wird eine gewisse Funktionalität in verschiedenen Softwareprojekten benötigt, so sollte man versuchen, gemeinsame Teile in unabhängige Pakte aufzuteilen.
- Die Klassen eines Paketes sollten daher möglichst untereinander abgeschlossen sein, so dass man jedes Paket einzeln betrachten und verwenden kann.
  $\Rightarrow$ Pakete sollten sich *nicht gleichzeitig gegenseitig* voraussetzen!

Dies wird durch die Qualifikatoren unterstützt:
Ist eine Klasse, Methode oder Instanzvariable weder `public` noch `private`, so ist sie nur innerhalb ihres Pakets sichtbar.

Zusammenfassung Packages

- Namen des Pakets muss mit Verzeichnisstruktur übereinstimmen
- Pakete dienen zur Aufteilung des Namensraums
- Pakete reduzieren die Schnittstelle durch Einschränkung der Sichtbarkeit:

private	Nur innerhalb der Klasse sichtbar
kein Qualifikator	Nur innerhalb des Pakets sichtbar auch als „package-private“ bekannt
public	Überall sichtbar

IDEs bieten inzwischen eine gute Unterstützung an:
Dateien werden automatisch in die richtigen Verzeichnisse abgelegt; Package Deklaration werden automatisch bereinigt.