

# Programmieren I

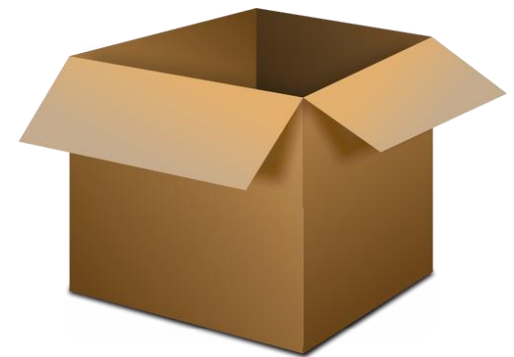
## Java Packages

Institut für Automation und angewandte Informatik

```
final List<String> allResults = new ArrayList<String>();  
final Map<String, Integer> typeWordResultCount = new HashMap<String, Integer>();  
final Map<String, Integer> typePoints = new HashMap<String, Integer>();  
evaluation.put(type, typePoints);  
  
for (final Sheet sheet : this.sheets) {  
    final String sheetResult = sheet.getPlayerInput(type);  
    if (sheetResult.startsWith(start) && this.isValidWord(sheetResult, type)) {  
        validWordCountForType++;  
        allResults.add(sheetResult);  
    }  
}
```

# Java Packages – Programme strukturieren

- Strukturierung der unterschiedlichen Aufgaben mittels Paketen (engl: `package`)
- Ein Paket ist eine Gruppe von thematisch zusammengehörigen Klassen.  
Diese befinden sich normalerweise in *einem* Verzeichnis.  
Der *Name des Verzeichnisses* ist gleich dem *Paketnamen*.

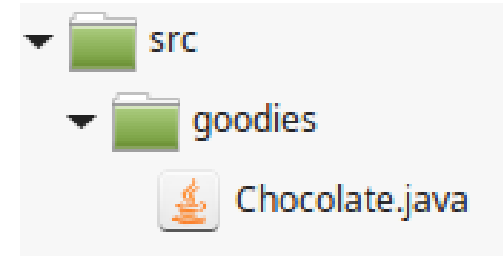


Bildquelle: <http://publicdomainvectors.org>

# Pakete deklarieren

## ■ Beispiel:

```
package goodies;  
  
public class Chocolate {  
  
}
```



Die Klasse, die in dieser Datei implementiert wird, gehört zum Paket `goodies`. Diese Zugehörigkeit wird durch das Schlüsselwort `package` ausgedrückt.

# Pakete importieren

- Um in einem *anderen* Paket die Klassen zu nutzen, wird der Compiler mit `import` auf die Klassen im Paket hingewiesen.

Die Klasse kann dann einfach mit dem Klassennamen (d.h. ohne Angabe des Paketnamens) in der Datei referenziert werden:

```
package treat;

import goodies.Chocolate;

class ChocolateBunny {
    Chocolate choc;    // sonst: goodies.Chocolate
}
```

```
package goodies;

public class Chocolate {
}
```

- Damit nicht alle genutzten Klassen eines Pakets einzeln aufgeführt werden müssen, lässt sich mit dem *Sternchen* (“\*”) als einer Art Wildcard auf *alle* `public`-Klassen des Pakets hinweisen.
- Häufig Gebrauch machen davon Programme mit grafischer Benutzeroberfläche. In den ersten Zeilen steht dann z.B.:

```
import javax.swing.*;
```

# Paket-Hierarchien

- Pakete sind oft in *Hierarchien* geordnet (*hauptpaket.unterpaket* etc.).  
Die Pakethierarchie wird dann auch auf die Verzeichnisstruktur des Dateisystems abgebildet.
- Zu einem Paket gehören oft *mehrere* Unterpakete.
- Mit der Anweisung

```
import hauptpaketname.*;
```

werden *nicht* automatisch auch die Klassen der *Unterpakete* mit eingebunden.  
Die `import`-Anweisung bezieht sich nur auf *ein* (Haupt- bzw. Unter-)Paket und schließt dessen Unterpakete nicht mit ein.



Bildquelle: <http://publicdomainvectors.org>

# Namenskonvention für Pakete

- Um eine bessere Unterscheidung zu Klassennamen zu gewährleisten, werden Paketnamen per Konvention komplett *klein* geschrieben.
  
- Z.B.
  - `java.io`
  - `java.lang`
  - `java.lang.annotation`
  - ...
  - `java.util`
  - `java.util.concurrent`
  - `de.dhbw.karlsruhe.programmieren`

# Vorschlag: Struktur für Übungsaufgaben

- Ein Projekt in der IDE für sämtliche Übungsaufgaben
- Pro Aufgabe ein Basis-Package
- Vorschlag: `de.dhbwka.java.exercise.aufgabe`  
→ steht auch auf den Übungsblättern

**DHBW Karlsruhe, Angewandte Informatik**  
Programmieren in JAVA – <https://www.iai.kit.edu/~javavorlesung>  
T. Schlachter, C. Schmitt, J. Sidler, W. Süß

  
DHBW  
Duale Hochschule  
Baden-Württemberg

Bereich: IDE einrichten, erstes Java-Programm	
Hallo Welt	
Package: <code>de.dhbwka.java.exercise.helloworld</code>	Klasse: HelloWorld
Aufgabenstellung:  Installieren Sie (falls nicht vorhanden) eine Entwicklungsumgebung Ihrer Wahl (Java Standard-Edition SE ist ausreichend), z.B. <ul style="list-style-type: none"><li>• Eclipse (<a href="http://www.eclipse.org">www.eclipse.org</a>)</li></ul>	