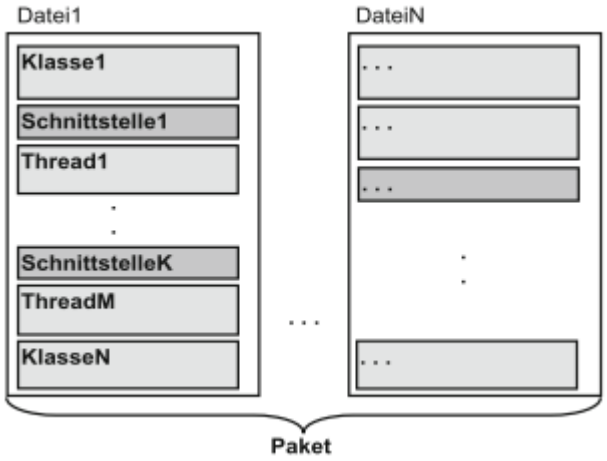
Pakete

SS 2019

Gruppierung

- Ein Java-Programm kann grob strukturiert werden mit Hilfe der folgenden Sprachmittel von Java :
  - Klasse bzw. Thread (spezielle Klasse)
  - Schnittstelle (Interface)
  - Paket
- Paket
  - ist ein Mittel zur Gruppierung von Quellcode
  - trägt einen Namen
  - kann Klassen/Threads, Schnittstellen und Unterpakete als Komponenten enthalten
  - kann aus einer oder aus mehreren Quellcode-Dateien bestehen
  - in jeder dieser Dateien ist deklariert, dass sie zum Paket gehört



Strukturierung der Sichtbarkeit

- Komponenten, die im **selben** Paket liegen
  - haben mehr Zugriffsrechte aufeinander (wenn keine Angabe von Zugriffsmodifikatoren) als ein externer Nutzer, der diese Komponenten des Pakets benutzen will.
- Ein **externer** Nutzer eines Pakets
  - kann die Teile eines Pakets nutzen, die explizit zur externen Benutzung frei gegeben wurden (Schlüsselwort `public`)
- Pakete bilden eigene Bereiche für den Zugriffsschutz
  - Information Hiding
  - mit Paketen kann man kapseln

Namensraum

- Jedes Paket bildet einen eigenen Namensraum
  - Namenskonflikte werden vermieden
  - identische Namen für Klassen bzw. Schnittstellen in verschiedenen Paketen können vergeben werden
  - zwei Klassen bzw. zwei Schnittstellen mit identischem Namen können nicht in einem gemeinsamen Paket liegen

Pakete erstellen

- Deklaration des Paketnamens mit Schlüsselwort `package`:
- Bsp:Klasse `Artikel` gehört zum Paket `lagerverwaltung`:

```
// Datei: Artikel.java

package lagerverwaltung;  // Deklaration des Paketnamens


public class Artikel      // Definition der Komponente Artikel des
{                         // Pakets lagerverwaltung
   private String name;
   private float preis;
   public Artikel (String name, float preis)
   {
      this.name = name;
      this.preis = preis;
   }
   // …Klassen-Methoden
}
```

Erstellung von Paketen - Paketdeklaration

- Paketnamen werden klein geschrieben
- Jede Quellcode-Datei, die zu einem Paket gehört, muss mit derselben Paketdeklaration beginnen
- Alle Programmeinheiten einer Quellcode-Datei gehören auf jeden Fall zum gleichen Paket

  =>für eine Datei darf es nur eine einzige Paketdeklaration geben

- Pakethierarchie:
  - Ein Paket selbst kann wiederum Unterpakete enthalten
  - Deklaration des Unterpaketnamens mit Punktoperator

Sichtbarkeit - Regeln

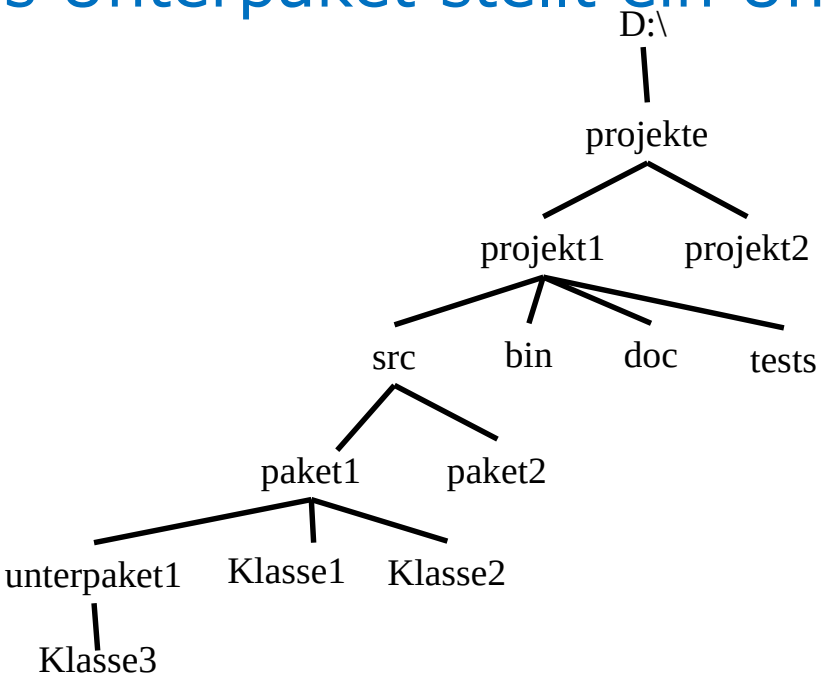
- Enthält eine Datei eine Klasse mit der Sichtbarkeit `public`, so muss der Dateiname gleich sein wie der Name der `public` Klasse
- Maximal eine Klasse einer Datei kann `public` sein
- Enthält eine Datei keine Klasse mit der Sichtbarkeit `public`, so kann der Dateiname beliebig sein
- Soll eine Klasse von einem anderen Paket aus nutzbar sein, so muss sie `public` sein, sonst ist sie nur innerhalb ihres eigenen Pakets als Hilfsklasse verwendbar

Pakete benutzen

- Paketkomponenten (Klassen, Schnittstellen oder Unterpakete) können mit Hilfe des Punktoperators angesprochen werden
  - Analogie zu Attribute und Methoden in Klassen
- Mit public deklarierte Klassen können mittels der import-Vereinbarung in anderen Paketen sichtbar gemacht werden
- Die import-Vereinbarung(en) stehen
  - hinter der package-Deklaration, aber
  - vor dem Rest des Programms
- Import zweier gleichnamiger Klassen od. Schnittstellen
  - voll qualifizierter Name wg. Eindeutigkeit (Paket.Klassenname)

Paketnamen und Verzeichnisstruktur

- alle Dateien, die zu einem Paket gehören
  - liegen in einem Verzeichnis
  - Verzeichnisname identisch zum Paketnamen
- jedes Unterpaket stellt ein Unterverzeichnis dar



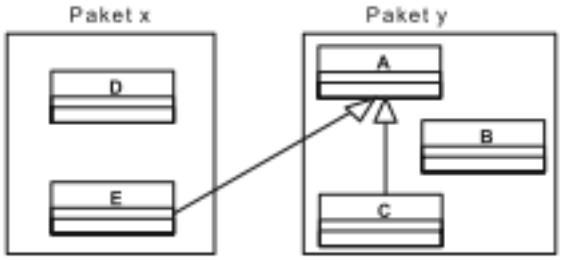
Eindeutige Paketnamen

- Konvention:
  - Bei großen Projekten
    - Der Internet-Domain-Name ist in umgekehrter Reihenfolge vor den Rest des Namens zu stellen.
    - aus Domain-Namen oracle.com wird der Paketname com.oracle
  - Bei kleineren Projekten nicht notwendig
- Anonyme Pakete
  - Wird in einer einer Quellcode-Datei kein Paketname deklariert
    - gehört sie zu einem anonymen Paket (Default-Paket)
    - Alle Klassen dieser Datei gehören also zu einem anonymen Paket.
  - Dateien ohne Paketzuordnung innerhalb desselben Verzeichnisses gehören dann automatisch zum gleichen anonymen Paket
  - Z.B. bei kleinen Programmen
  - Bei größeren Projekten -> Aufteilung der Anwendung in Pakete

Gültigkeitsbereich und Sichtbarkeit von Klassen

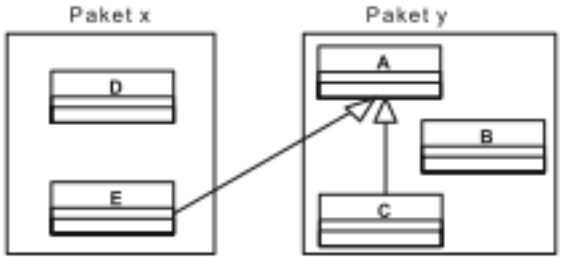
- Klassennamen
  - Gültig in einem Paket
  - übergreifend über verschiedene Dateien im Paket
- Compiler geht in Java mehrfach über den Quellcode
  - bis er alle Klassendeklarationen gefunden hat
- Regelung des Zugriffsschutzes
  - Zugriffsmodifikatoren public, protected und private.
- Eine Klasse in einem Paket
  - ist für Klassen aus anderen Paketen nur sichtbar und importierbar, wenn sie `public` ist
  - ist nur für Klassen desselben Paketes sichtbar, wenn der Zugriffsschutz einer Klasse default ist (ohne Zugriffsmodifikator)

Zugriffsschutz für Methoden und Attribute



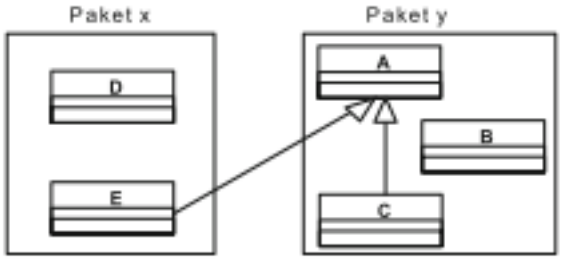
- Klasse A ist public
- Zugriff auf `private`-Attribut/Methode von A
  - B - Nein
  - C - Nein
  - D - Nein
  - E - Nein

Zugriffsschutz für Methoden und Attribute



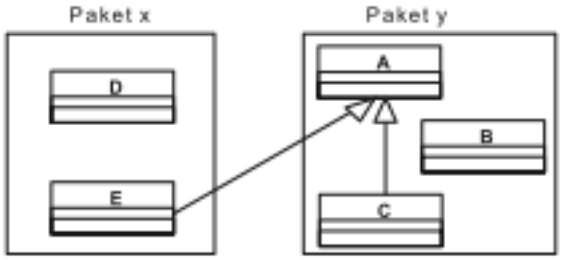
- Klasse A ist public
- Zugriff auf default-Attribut/Methode von A
  - B - Ja
  - C - Ja
  - D - Nein
  - E - Nein

Zugriffsschutz für Methoden und Attribute



- Klasse A ist public
- Zugriff auf `protected`-Attribut/Methode von A
  - B - Ja
  - C - Ja
  - D - Nein
  - E - Ja

Zugriffsschutz für Methoden und Attribute



- Klasse A ist public
- Zugriff auf `public`-Attribut/Methode von A
  - B - Ja
  - C - Ja
  - D - Ja
  - E - Ja

Zugriffsschutz für Konstruktoren

- Klasse stellt zur Verfügung
  - nur Konstruktor ohne Zugriffsmodifikator
    - Konstruktor nur von Klassen innerhalb des eigenen Pakets aufrufbar
  - Konstruktoren mit dem Zugriffsmodifikator protected
    - so können von allen Klassen aus, die im selben Paket liegen, Objekte erzeugt werden
    - Abgeleitete Klassen in anderen Paketen
      - können keine Objekte erzeugen,
      - können aber Konstruktor der Vaterklasse mit super() aufrufen
  - alle Konstruktoren als private
    - Keine andere Klasse aus kann ein Objekt dieser Klasse erzeugen
  - überhaupt kein Konstruktor
    - so existiert der vom Compiler zur Verfügung gestellte Default-Konstruktor
    - Default-Konstruktor hat den Zugriffsschutz der Klasse.