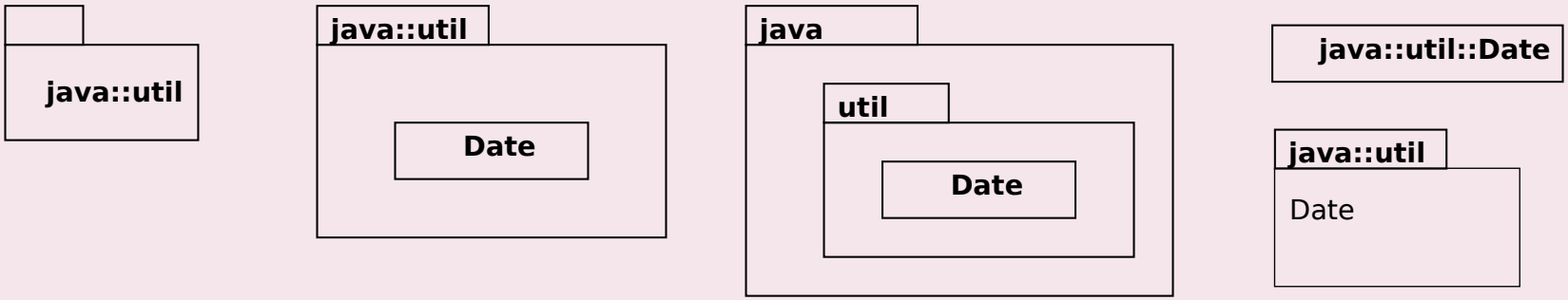
Diagrammi di Package
Package Diagram

a cura di **Angelo Furfaro**
da “UML Distilled”
Martin Fowler

Dipartimento di Ingegneria Informatica Elettronica Modellistica e Sistemistica
Università della Calabria, 87036 Rende(CS) - Italy
Email: a.furfaro@dimes.unical.it
Web: http://angelo.furfaro.dimes.unical.it

Diagrammi di Package

- Un **package** è un costrutto che permette di raggruppare un insieme di elementi UML in unità di livello più alto.
- In genere si utilizzano per raggruppare classi.
- Ogni classe fa parte di un solo package.
- Un package può essere membro di un altro package.
- Al livello della programmazione i package UML corrispondono a costrutti di raggruppamento quali gli omonimi package Java o i namespace in C++.
- Ogni package introduce uno *spazio di nomi* (**namespace**): classi differenti membre dello stesso package devono avere nomi distinti.
- I package sono rappresentati come dei box dotati di linguetta (*tab*) in alto a sinistra. Il nome è riportato all'interno del box o sul tab se si illustrano i contenuti interni.



Diagrammi di Package

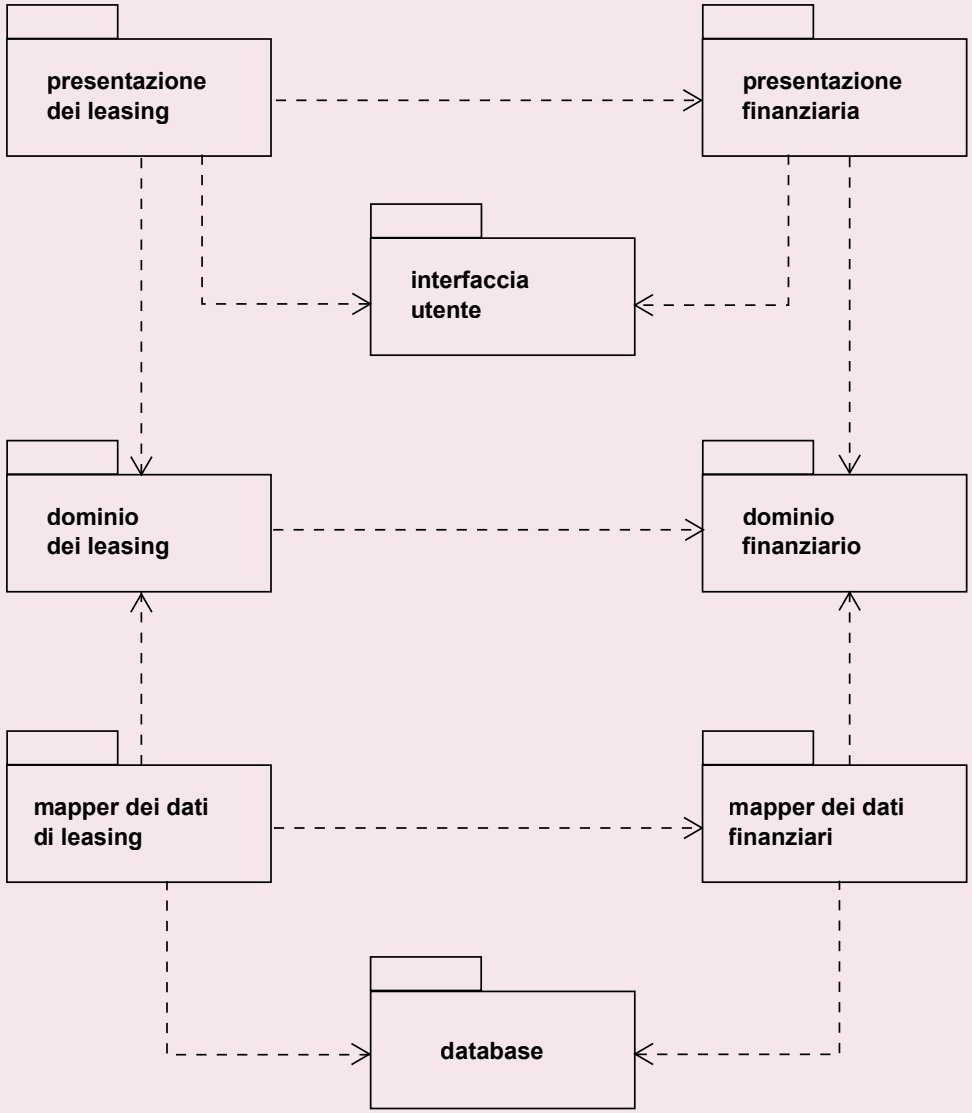
- Nei casi in cui esistono classi con lo stesso nome in package differenti per risolvere l'ambiguità si utilizzano i nomi completamente qualificati che includono i nomi dei package separati da una coppia di due punti (**::**) (Ad esempio `MioPackage::Date` e `java::util::Date`).
- Le classi contenute in un package UML possono essere pubbliche o private.
- Le classi pubbliche costituiscono l'interfaccia del package in quanto possono essere utilizzate all'esterno di esso.
- Per ridurre l'interfaccia di un package si può esportare solo un piccolo sottoinsieme delle operazioni delle classi (pattern *Façade*):
  - si dichiarano private tutte le classi;
  - si introducono solo poche classi pubbliche che rendono visibile l'interfaccia desiderata.

Princìpi di suddivisione di classi in package

- *Common Closure*: classi dello stesso package dovrebbero condividere le cause di un eventuale cambiamento.
- *Common Reuse*: classi dello stesso package dovrebbero essere riusate insieme.

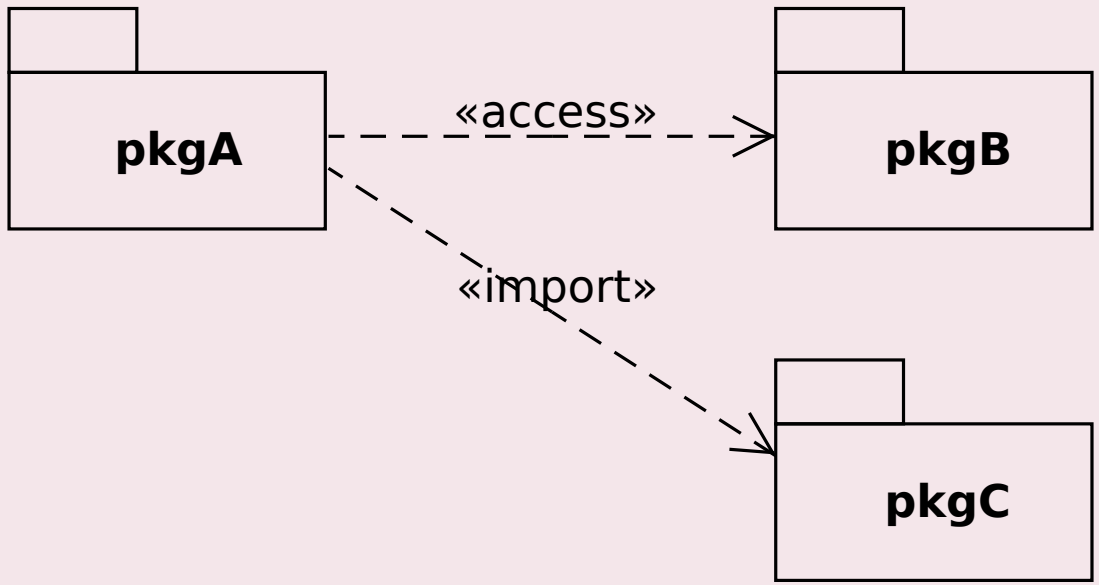
Package e dipendenze

- Un diagramma dei package documenta i package e le dipendenze tra di essi.
- Nell'esempio le classi di presentazione dipendono da quelle del dominio.
- Le dipendenze tra package riassumono quelle tra gli elementi contenuti.
- Un diagramma generale dei package è utile a tenere sotto controllo la complessità strutturale del codice.
- Man mano che le dipendenze entranti in un package aumentano la sua interfaccia deve essere sempre più stabile (*Dependencies principle*).
- I package più stabili tendono a contenere una percentuale maggiore di classi astratte e interfacce (*Stable abstraction principle*).



Dipendenze ≪access≫ e ≪import≫

- Entrambe le parole chiavi sono utilizzate per indicare che un package include nel proprio namespace i nomi definiti nell'altro package.
- La parola chiave ≪import≫ indica un'importazione di tipo *public*: gli elementi importati sono aggiunti al namespace e resi visibili anche all'esterno di esso.
- La parola chiave ≪access≫ indica un'importazione di tipo *private*: gli elementi importati sono aggiunti al namespace ma non sono visibili dall'esterno.



Gerarchia di package

Visualizzazione tramite struttura ad albero