Ereditarietà e Documentazione

Alessandro Midolo, Ph.D. Student
alessandro.midolo@phd.unict.it

Tutorato Ingegneria del Software

A.A. 2021/2022

Ereditarietà nella OOP

L’ereditarietà è una proprietà della OOP che permette di strutturare le classi secondo una gerarchia

Definire una nuova classe definendo gli attributi e i metodi aggiuntivi rispetto alla classe madre oppure modificando i metodi già esistenti

Una classe può quindi:

- ereditare da un’altra classe (può essere anche una classe astratta)
- implementare un interfaccia

Sottoclassi

- Eredita tutti gli attributi e i metodi della classe madre; questo le permette di poterli usare senza doverli reimplementare
- Può implementare nuovi metodi
- Può modificare i metodi della classe madre ridefinendoli
- Non può eliminare metodi o attributi della classe madre

Visibilità

- **Private**: visibile solo alla classe (no sottoclassi)
- **Protected**: visibile alla classe e alle sue sottoclassi
- **Public**: visibile a tutte le classi

Interfaccia e Classe Astratta

Un’**interfaccia** definisce un tipo e dà la struttura delle classi che la implementeranno

- Non implementa i metodi, elenca solo le signature di questi con la rispettiva visibilità
- No costruttori
- No attributi
- Non può essere istanziata

Una **classe astratta** è una classe non del tutto implementata, alcuni metodi **abstract**

- forza le sottoclassi ad implementare i metodi abstract
- una classe astratta non può essere istanziata, come le interfacce

Documentazione

In java è possibile generare automaticamente la documentazione del proprio codice

E’ sufficiente inserire all’interno del proprio codice dei commenti con delle specifiche annotazioni per istruire ***javadoc*** nel generare la documentazione

Il comando per generare la documentazione:

javadoc src\main\java\org\javaClasses\negozio* -d docs\negozio

1. il primo parametro sono i file per generare la documentazione
2. il secondo parametro indica la cartella dove caricare la documentazione

E’ possibile eseguire il comando dal terminale di VisualStudioCode

Diagrammi UML delle classi

Un diagramma UML permette di rappresentare le classi con le loro caratteristiche e le loro relazioni con una notazione standardizzata

E’ possibile definire gli attributi e i metodi di ogni classe

- attributi: visibilità nome: tipo
- metodi: visibilità nome(par: tipo): tipo di ritorno

Le interfacce si identificano con lo stereotipo **<<interface>>**

Le classi astratte si identificano con lo stereotipo **<<abstract>>**

PlantUML è uno strumento per la generazione dichiarativa di diagrammi UML. Permette la definizione di diagrammi tramite un file di testo invece che mediante un editor grafico