Designer

ISTITUTO TECNICO TECNOLOGICO STATALE
"S. Fedi - E. Fermi"

Guglielmo Bartelloni, Francesco Bellezza

24 Dicembre, 2017
10 febbraio 2018

4IB
Luigi Vestri e Davide Caramelli

Laboratorio di Informatica

Indice

1 Scopo dell'esercitazione

Lo scopo dell'esercitazione e quella di realizzare un programma che consenta di disegnare figure geometriche e grafici di funzioni permettendo di salvare le figure sul un file di testo.

2 Cenni Storici

2.1 Ereditarieta'

L'ereditarietà è una relazione di tipo is-A, dove una superclasse mette a disposizione di una sottoclasse i suoi metodi e attributi non privati (a eccezione del costruttore).

In Java, per estendere una superclasse e per creare quindi la sottoclasse, si utilizza la parole chiave extends accanto al nome della sottoclasse e accanto alla parola extends si inserisce il nome della superclasse.

Nel caso delle interfacce, ovvero particolari classi che al loro interno contengono solamente metodi astratti, si utilizza la parola chiave implements. Fiorenzo, Giorgio e Ivan (*Corso di Informatica*)

2.2 Polimorfismo

Il polimorfismo è una tecnica che consente di utilizzare metodi polimorfici, ovvero metodi che hanno lo stesso nome, ma implementazioni diverse. Esistono due tipi di polimorfismi principali: per overloading e per overriding.

Nel polimorfismo per overloading si opera a un livello locale, ovvero all'interno di una classe.

Il metodo polimorfico in questo caso deve:

- Avere lo stesso nome degli altri metodi polimorfici.
- Avere numero di parametri diversi o avere tipi di parametri diversi o avere ordine di parametri diversi rispetto agli altri metodi polimorfici.
- Può avere un tipo di ritorno diverso se i due punti qua sopra sono rispettati.

Nel polimorfismo per overriding si opera a un livello di superclassi e sottoclassi.

Nelle sottoclassi viene ridefinito il metodo presente nelle superclassi.

Il metodo polimorfico in questo caso deve:

- Avere la stessa signature del metodo della superclasse.
- Avere lo stesso tipo di ritorno della superclasse o un sottotipo del tipo di ritorno della superclasse (stesso discorso vale per i parametri).
- I metodi private non vengono ereditati alla classe figlia, quindi non si può effettuare l'overriding del metodo.
- Le clausole come native, strictfp possono essere incluse nel metodo della classe figlia.
- Un metodo statico può essere solo adombrato e non sovrascritto.

Wikipedia (*Polimorfismo*)

3 Analisi Funzionale

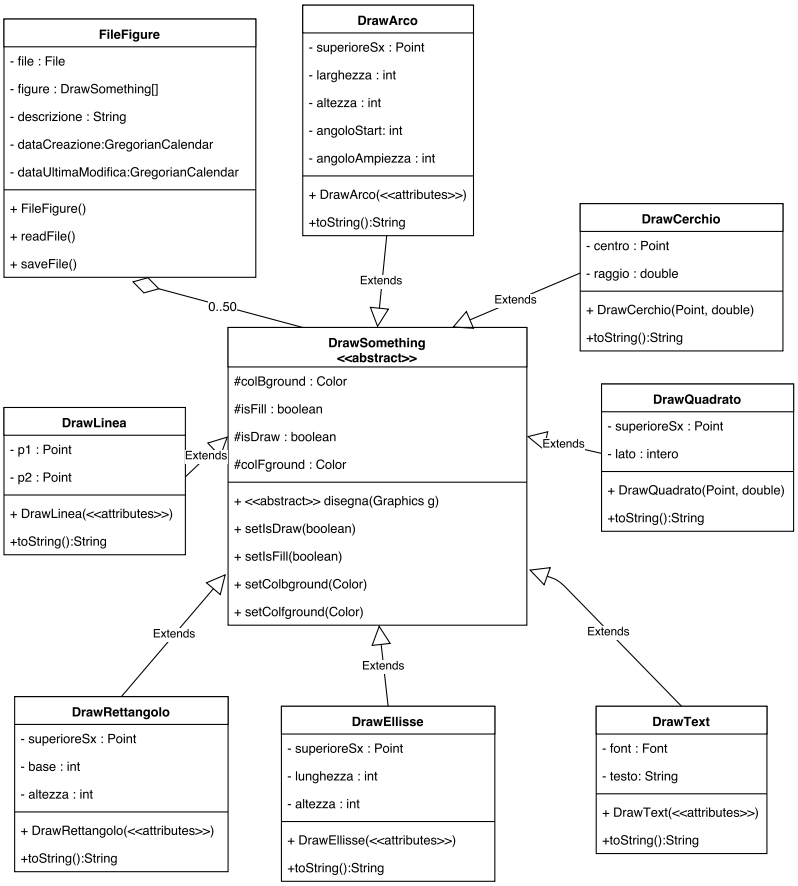
3.1 Ipotesi Risolutiva

3.2 Funzionalitá del programma

4 Analisi Tecnica

4.1 Scomposizione Top-Down

4.2 UML



4.3 Descrizione Classi

4.3.1 MainFrame

4.3.2 DrawSomething

La classe *DrawSomething* é una classe astratta che dichiara alcuni metodi per impostare i colori e un metodo astratto per essere riscritto nelle classi figlie. É pensata all'unico scopo di essere estesa dalle altre classi *draw* che implementeranno il metodo *toString()* e il metodo *draw(Graphics g)* come mostrato in figura 4.2. Gli attributi di questa classe sono i seguenti:

```
protected Color colBground;
protected Color colFground;
```

```
protected boolean isFill;
protected boolean isDraw;
```

4.3.3 FileFigure

La classe *FileFigure* é la classe che si occupa di gestire di gestire il ripristino e il salvataggio delle figure nel file di testo.

Il metodo *readFile()* si occupa di leggere il file di testo e salvare le figure nell'array.

Il metodo *saveFile()* si occupa di salvare l'array di figure all'interno del file di testo. Per il salvataggio delle figure viene utilizzato il metodo *toString()* di ogni figura.

5 Test Data Set/Debug

Riferimenti bibliografici

Fiorenzo, Formichi, Meini Giorgio e Venuti Ivan. *Corso di Informatica*. Zanichelli, 2017.

Wikipedia. *Polimorfismo*. https://it.wikipedia.org/wiki/Polimorfismo_(informatica).