23/3/2022 version 1.3  
Appunti iniziali

the interface of a subroutine (o metodo) consists of the name of the subroutine, its return type, and the number and types of its parameters (signature, firma di un metodo).

This is the information you need to know if you want to call il metodo -the subroutine.

public Contatto cerca(String cognome)  oppure public Contact search(String cognome)

A subroutine/metodo  also has an implementation: the block of code which defines it and which is executed when the subroutine is called.

public Contatto cerca(String cognome){

// codice che serve a fare questa ricerca

}

In Java, **interface** is a reserved word with an additional, technical meaning.

An "interface" in this sense consists of a set of instance method interfaces, without any associated implementations.

A class can implement an interface by providing an implementation for each of the methods specified by the interface.

Here is an example of a very simple Java interface AstrazioneDiRubrica:

public interface AstrazioneDiRubrica {

public Contatto cerca(String cognome);

}

This looks much like a class definition, except that the implementation of the

Contatto cerca(String cognome) method is omitted.

A class that implements the interface AstrazioneDiRubrica must provide an implementation for Contatto cerca(String cognome).

Of course, the class can also include other methods and variables.

Note that to implement an interface, a class must do more than simply provide an implementation for each method in the interface;  it must also state(dichiarare) that it implements the interface, using the reserved word **implements** as in this example:

.

Any concrete class that implements the AstrazioneDiRubrica interface must define a Contatto cerca(String cognome) instance method.

import java.util.ArrayList;

public class Rubrica implements AstrazioneDiRubrica{

private ArrayList<Contatto> contatti;

//fare costruttore, toString, get

public Contatto cerca(String cognome){

Contatto c;

c = null;

// da completare

return c;

}

}

public interface AstrazioneDiRubrica {

public Contatto cerca(String cognome);

}

public class Contatto {

private String nome;

private String congome;

private String telefono;

private String email;

public Contatto(String nome, String congome, String telefono, String email) throws Exception{

valida(nome, congome, telefono, email);

this.nome = nome;

this.congome = congome;

this.telefono = telefono;

this.email = email;

}

private void valida(String nome, String congome, String telefono, String email) throws Exception{

if (nome == null){

throw new NullPointerException("nome null");

}

if (congome == null){

throw new NullPointerException("congome null");

}

if (telefono == null){

throw new NullPointerException("telefono null");

}

if (email == null){

throw new NullPointerException("email null");

}

//manca altri controlli. (che ci sia solo una @ nella mail) (numero di telefono sia 11)

}

//fare get(), equals()

//non fare costruttore di copia

//fare toString()

}