Interfeisai (interface)

Interfeisas - tai sąrašas metodų, kurie bus realizuoti klasėje (klasėse). Jokių metodų realizacijų (kūnų) interfeise nėra. Interfeisai turi savo hierarchiją, kuri nesikerta su klasių paveldėjimo hierarchija. Tai leidžia realizuoti tą patį interfeisą klasėse, nesusijusiose hierarchiškai pagal klasių paveldėjimo liniją.

Interfeisas yra panašus į abstrakčią klasę, bet turi keletą esminių skirtumų :

1. Klasė gali realizuoti kelis interfeisus, kai tuo tarpu paveldėti gali tik vieną.
2. Interfeiso metodai pasirenkami dinamiškai, t.y., programos vykdymo metu (rekia nepamiršti, jog tai pailgina vykdymo laiką).
3. Interfeisas gali būti realizuotas kitoje programavimo kalboje.
4. Interfeisą galima sėkmingai panaudoti tų pačių konstančių importavimui į atskiras klases (panašiai kaip *#define* h tipo failuose C++ kalboje).
5. Interfeisas negali turėti ir realizuotų metodų, o abstract klasė gali.

Interfeiso sintaksė :

[public] **interface** *InterfeisoVardas* [extends Interfeisas1 [, Interfeisas2]…] {
    rezultatoTipas *metodoVardas1* ([*parametrų sąrašas*]);
    rezultatoTipas *metodoVardasN* ([*parametrų sąrašas*]);
    tipas *finalTipoKintamojoVardas* = *reikšmė*;
    tipas *finalTipoKintamojoVardasN* = *reikšmė*;
}

Interfeisas automatiškai tampa **public** , metodai - **public**, o interfeiso kintamieji – **public static final.**

Interfeisą realizuojancios klasės sintaksė :

[modifikatorius] class *KlasėsVardas*
                implements *InterfeisoVardas* [, *InterfeisoVardas2*…] {
    // klasės kūnas, realizuojantis metodus
    // ir naudojantis jo konstantes (final tipo kintamuosius)
}

Keletas pastabų :

1. Interfeisas neturi konstruktoriaus.
2. Interfeiso metodai neturi kūnų.
3. Klasė, realizuojanti interfeisą, privalo realizuoti visus jo metodus ir visus super-interfeisų metodus (interfeise ir klasėje metodų antraštės turi idealiai sutapti). Kitaip ji turės būti paskelbta abstract.
4. Interfeiso metodai turi būti atviri. Jiems negalimi modifikatoriai : static, final, private, protected, native, synchronized.
5. Interfeiso kintamieji savaime įgauna final tipą (realizuojanti klasė negali jų pakeisti). Kintamieji privalo būti inicializuoti konstantomis.
6. Jei klasė realizuoja kelis interfeisus, jų vardai atskiriami kableliais (žodis implements nedubliuojamas).
7. Tas pats metodas gali pasikartoti keliuose intefeisuose.
8. Interfeisas gali paveldėti kitą interfeisą (bet klasės paveldėti negali). Šiu atveju jį realizuojanti klasė privalo perdengti abiejų interfeisų metodus.

Kintamasis gali būti interfeiso tipo. Tada jam galima priskirti bet kurios jį realizuojančios klasės objektą. Bus vykdomos tos klasės interfeiso metodų realizacijos, kurios vardą šiuo momentu saugo interfeiso tipo kintamasis.

PVZ1 :

```
interface Inter {
   public void geras();
   void blogas();
   public static final int JIS = 1;
   int JI = 2;
}

class A implements Inter {
   private int tipas = JIS;
   public void geras() {
      if (tipas == JIS) {
         System.out.println("Esu geras");
      } else if (tipas == JI) {
         System.out.println("Esu gera");
      } else {
         System.out.println("Tipas itartinas");
      }
   }
   public void blogas() {
      if (tipas == JIS) {
         System.out.println("Esu blogas");
      } else if (tipas == JI) {
         System.out.println("Esu bloga");
      } else {
         System.out.println("Tipas itartinas");
      }
   }
   public void keistiTipa(int tipas) {
      this.tipas = tipas;
   }
}

class B implements Inter { // kita interfeiso metodu realizacija
   private int tipas = JI;
   public void geras() {
      if (tipas == JIS) {
         System.out.println("Esu labai geras");
      } else if (tipas == JI) {
         System.out.println("Esu labai gera");
      } else {
         System.out.println("Tipas itartinas");
      }
   }
   public void blogas() {
      if (tipas == JIS) {
         System.out.println("Esu labai blogas");
      } else if (tipas == JI) {
         System.out.println("Esu labai bloga");
```

```
        } else {
            System.out.println("Tipelis itartinas");
        }
    }
    public void keistiTipa(int tipas) {
        this.tipas = tipas;
    }
}

class Testas {
    public static void main(String argumentai[]) {
        A a = new A();
        B b = new B();
        Inter i;
        System.out.println("     Tiesiogiai");
        i = a;
        i.geras();
        i.blogas();
        i = b;
        i.geras();
        i.blogas();
        System.out.println("     Call-Back");
        kviecia(a);
        b.keistiTipa(b.JIS);
        //i.keistiTipa(b.JIS);  // klaida, nes keistiTipa() nera interfeiso metodas
        //((B)i).keistiTipa(b.JIS);  // sitaip gerai
        kviecia(b);
        System.out.println("     blablabla");
        b.keistiTipa(5);
        b.blogas();
    }
    public static void kviecia(Inter i2) {
      /////   CallBack
        i2.geras();
        i2.blogas();
    }
}

/* Atsakymai
     Tiesiogiai
Esu geras
Esu blogas
Esu labai gera
Esu labai bloga
     Call-Back
Esu geras
Esu blogas
Esu labai geras
Esu labai blogas
    blablabla
Tipelis itartinas
*/
```