



Principet e Avancuara të Gjuheve të Programimit

Lamir Shkurti, Can. PhD.
lamir.shkurti@gmail.com

VARIABLAT STATIKE



- Variablat e klasës të njohur gjithashtu si variabla statike deklarohen me fjalën kyçe **static** në një klasë, por jashtë një metode, konstruktori ose një blloku.
- Variablat statike krijohen kur fillon programi dhe shkatërrohen kur programi ndalet.
- Dukshmëria është e ngjashme me variablat e instancës. Sidoqoftë, shumica e variablave statike deklarohen publike pasi ato duhet të jenë të disponueshme për përdoruesit e klasës

VARIABLAT STATIKE



- Variabla statike mund të përdoret për t'iu referuar pronës së përbashkët të të gjitha objekteve (e cila nuk është unike për secilin objekt), për shembull, emri i ndërmarrjes së të punësuarve, emri i kolegjit të studentëve, etj.
- Variabla statike merr memorie vetëm një herë në zonën e klasës në kohën e ngarkimit të klasës.

PËRPARËSITË E VARIABLAVE STATIKE



- Kjo e bënë memorien e programit më efikas (d.m.th kursen memorien)

```
class Student{  
    int ID;  
    String name;  
    String college="UPZ";  
}
```

Supozojmë se ka 500 studentë në një kolegji. Të gjitha variablat do të marrin memorie çdo here që krijohet një objekt. Të gjithë studentët kanë emrin dhe mbiemrin e tyre unik.

Këtu, "kolegji" i referohet pronës së përbashkët të të gjitha objekteve. Nëse e përdorim si statike, kjo fushë do të marrë memorien vetëm një herë.

SHEMBULLI 1

```
public class Student{
    int ID;
    String name;
    static String college="UPZ"; //static variable

    public Student (int i, String n) {
        ID = i;
        name = n;
    }
    public void display () {
        System.out.println(ID+" "+name+" "+college);
    }
}

public class TestStaticVariable{
    public static void main(String args[]) {
        Student s1 = new Student (192048888,"Filan Fisteku");
        Student s2 = new Student (192048889, "Emri Mbiemri");
        //ne mund ta ndryshojmë kolegjin e të gjitha objekteve me anë të këtij rreshti
        //Student.college="Upz - Prizren";
        s1.display();
        s2.display();
    }
}
```

SHEMBULLI 2



```
public class Counter{
    int count=0;
    public Counter () {
        count++;
        System.out.println(count);
    }
    public static void main(String args[])
        //Creating objects
        Counter c1=new Counter();
        Counter c2=new Counter();
        Counter c3=new Counter();
    }
}
```

Rezultati = ?

```
public class CounterStatic{
    static int count=0;
    public CounterStatic() {
        count++;
        System.out.println(count);
    }
    public static void main(String args[]) {
        //Creating objects
        CounterStatic c1=new CounterStatic();
        CounterStatic c2=new CounterStatic();
        CounterStatic c3=new CounterStatic();
    }
}
```

Rezultati = ?

BLLOKU STATIK



- Përdoret për inicimin e anëtarit të të dhënave statike.
- Ekzekutohet para metodës main në kohën e ngarkimit të klasës
- Kodi brenda bllokut statik ekzekutohet vetëm një herë: herën e parë që klasa ngarkohet në memorie.
- Gjithashtu, blloqet statike ekzekutohen para konstruktorëve.

BLLOKU STATIK – SHEMBULLI 1

```
public class Test {
    static int i;
    int j;
    // blloku statik
    static {
        i = 10;
        System.out.println("static block called ");
    }
}

public class MainClass {
    public static void main(String args[]) {
        //Edhe pse nuk kemi krijuar nje objekt te Test, blloku statik thirret
        //dhe mund te shtypet vlera e i
        System.out.println(Test.i);
    }
}
```


BLLOKU STATIK – SHEMBULLI 2

```
public class Test {
    static int i;
    int j;
    public Test () {
        System.out.println("Constructor called");
    }
    // blloku statik
    static {
        i = 10;
        System.out.println("static block called ");
    }
}

public class MainClass {
    public static void main(String args[]) {
        //Edhe pse kemi dy objekte, blloku statik ekzekutohet vetem njehere.
        Test t1 = new Test ();
        Test t2 = new Test();
    }
}
```

BLLOKU INICIALIZUES



- Blloku inicializues përdoret për të inicializuar atributet e klasës. Ekzekutohet çdo herë kur krijohet një objekti klasës.
- Inicializimi i variablave të instancës mund të bëhet direkt, por mund të kryhen edhe veprime shtesë në rast që përdorim bllokun inicializues
- Ka qasje në variablat dhe metodat e instancës dhe thirret në fillim të konstruktorit (gjatë krijimit të instancës), pasi të jetë thirrur super konstruktori.

BLLOKU INICIALIZUES – SHEMBULLI 1



```
public class A{
    public A() {
        System.out.println("parent class constructor invoked");
    }
}
public class B2 extends A{
    public B2 () {
        super();
        System.out.println("child class constructor invoked");
    }
    {
        System.out.println("instance initializer block is invoked");
    }
    public static void main(String args[]) {
        B2 b = new B2 ();
    }
}
```

KLASAT E NDËRTHURURA

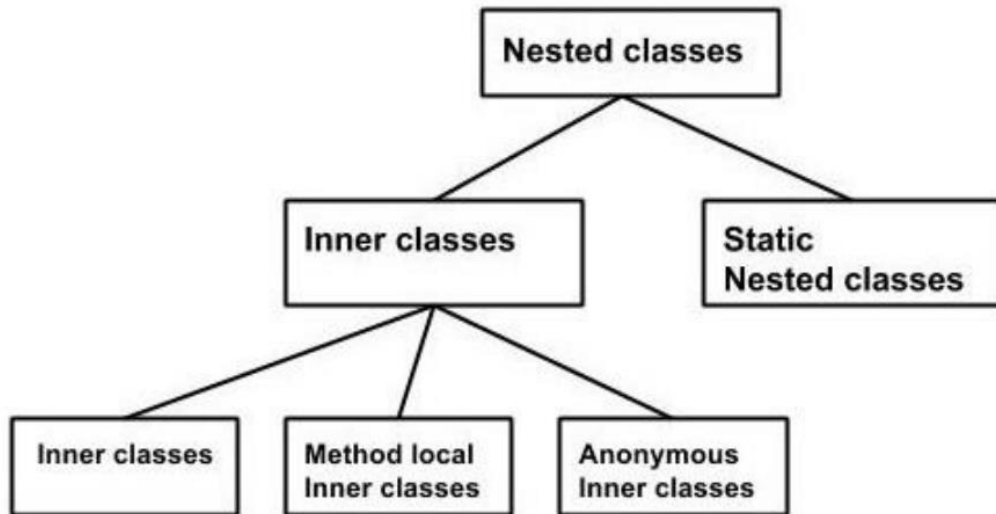


- Në Java, ashtu si metodat dhe variablat, një klasë mund të ketë edhe një klasë tjetër si onëtare të saj.
- Krijimi i një klase brenda një klase tjetër lejohet në Java.
- Klasa e krijuar brenda një klase quhet klasë e ndërthurur (nested Class) dhe klasa që mban klasën e brendshme quhet klasë e jashtme.
- Shembull

```
class Outer_Demo{  
    class Inner_Demo{  
    }  
}
```

KLASAT E NDËRTHURURA

- Klasat e ndërthurura i ndajmë në dy lloje :
- Non-static nested classes - Klasat e brendshme jo statike
- Static nested classes - Klasat e brendshme statike



KLASA E NDËRTHURUR



- Klasat e brendshme janë një mekanizëm sigurie në Java. Ne e dimë që një klasë nuk mund të ketë modifikuesin private, por nëse e kemi klasën si anëtare të klasës tjetër, atëherë klasa e brendshme mund të bëhet private. Kjo përdoret gjithashtu për të ju çasur anëtarëve private të klasës së jashtme
- Klasat e brendshme ndahen në tri lloje:
 - Klasa e brendshme
 - Klasa e Brendshme lokale
 - Klasa e brendshme anonime

KLASA E BRENDSHME

- Krijimi i një klase të brendshme është mjaft e thjeshtë. Ne vetëm duhet të shkruajmë një klasë brenda një klase tjetër. Për dallim nga një klasë e jashtme, një klasë e brendshme mund të deklarohet private dhe kjo klasë nuk mund të arrihet nga një objekt jashtë klasës.

```
public class Outer_Demo {  
    int num;  
    // inner class  
    private class Inner_Demo {  
        public void print() {  
            System.out.println("This is an inner class");  
        }  
    }  
  
    public static void main(String args[]) {  
        // Instantiating the outer class  
        Outer_Demo outer = new Outer_Demo ();  
        Outer_Demo.Inner_Demo inner = outer.new Inner_Demo();  
        inner.print();  
    }  
}
```

KLASAT E BRENDSHME STATIKE



- Klasa e brendshme statike është një klasë e ndërthurur e klasës së jashtme.
- Klasa e brendshme është anëtar static i klasës së jashtme.
- Mund të kemi çasje pa krijuar instancën e klasës së jashtme.
- Një klasë statike e ndërthurur ka qasje në variablat dhe metodat statike të klasës së jashtme.
- Sintaksa e klasës së brendshme statike është:

```
class MyOuter {  
    static class Nested_Demo {  
    }  
}
```


KLASAT E BRENDSHME STATIKE

- Inicializimi i një klase të brendshme statike është pak më ndryshe nga inicializimi i një klase të brendshme.
- Kodi i mëposhtëm tregon se si të përdoret një klasë e brendshme statike.

```
public class Outer {  
    static class Nested_Demo {  
        public void my_method() {  
            System.out.println("This is my nested class");  
        }  
    }  
    public static void main(String args[]) {  
        Outer.Nested_Demo nested = new Outer.Nested_Demo();  
        nested.my_method();  
    }  
}
```

KLASAT E BRENDSHME LOKALE



- Një klasë e brendshme lokale përcaktohet brenda një metode të klasës.
- Në Java, ne mund të shkruajmë një klasë brenda një metode. Ashtu si variablat, fushëveprimi i klasës së brendshme është i kufizuar brenda metodës.
- Një klasë e brendshme lokale-metodë mund të inicializohet vetëm brenda metodës ku përcaktohet klasa e brendshme. Shembulli i mëposhtëm tregon mënyrën e përdorimit të një klase të brendshme lokale.

KLASAT E BRENDSHME LOKALE - SHEMBULL

```
// Klasa e jashtme
public class MyOuterClass {
    private int x= 1;
    void myMethod () {
        int num =23;
        // Klasa e brendshme lokale
        class MethodInnerDemo {
            public void print(){
                System.out.println("This is method inner class "+num);
            }
        } // Klasa e brendshme perfundon ketu
        // Inicializimi i klases se brendshme lokale
        MethodInner_Demo inner = new MethodInner_Demo();
        inner.print();
    }

    public static void main(String args[]) {
        MyOuterClass outer = new MyOuterClass();
        outer.my_Method();
    }
}
```

Principet e Avancuara të Gjuhëve të Programimit

