

Dist 2

Normal

- #fleeting

- Skener (učitavanje input-a sa standardnog ulaza)

- java

```
Scanner scanner = new Scanner(System.in);  
scanner.nextLine();
```

- `volatile` promenljive su promenljive čije vrednosti se ne keširaju, već se učitavaju iz glavne memorije

- java

```
static volatile int count = 0;
```

- Jednostavan / primitivan mehanizam za sinhronizaciju

- Koristan kada samo jedna nit piše u takvu promenljivu, dok ostale čitaju iz nje, i kada više niti upisuje u zajedničku promenljivu tako da upisana vrednost ne zavisi od prethodnog upisa

- Više informacija: <https://www.baeldung.com/java-volatile-variables-thread-safety>

• Niti h1

- Izvedena klasa klase `Thread`

- Kreiranje niti

- java

```
public class Nit extends Thread {  
    public void run() {  
        // radi nesto  
    }  
}
```

- Pokretanje niti

- java

```
Nit n = new Nit();  
n.start(); // !!!
```

-



Obratiti pažnju! Ne koristi se metod `run` , već metod `start` jer je reč o klasi koja nasleđuje `Thread`

- Kreiranje worker-a koji instanciranja klasu `Thread`

- Kreiranje

-

java

```
public class Worker { Worker  
    public void run() {  
        Thread t1 = new Thread(new Runnable() {  
            public void run() {  
                // radi nesto  
            }  
        });  
  
        t1.start();  
  
        try {  
            t1.join();  
        } catch (InterruptedException e) {  
            e.printStackTrace();  
        }  
    }  
}
```

- Pokretanje

-

java


```
Worker w = new Worker();  
w.run();
```

-



Pošto ne nasleđuje `Thread` , klasa `Nit` nema metod `start` već koristi metod `run` u kome kreira

- `Thread` -u se proseleđuje `Runnable` koji sadrži metod `run` . Pomoću `Thread` - ovog metoda `start` pozivamo metod `run`

- `synchronized` na metodu znači da ni jedan drugi thread za dati objekat ne može da pristupi drugim `synchronized` metodima
 - Takodje, za dati objekat samo jedan thread može da pristupi datom metodu i datom trenutku
- `synchronized` blok može da bude izvršavan od strane samo jednog threada u datom trenutku
- `synchronized` metodi se sinhronizuju na osnovu monitora
- statički metodi se sinhronizuju po monitoru za klasu, a nestatički sinhronizovani metodi na osnovu monitora za datu instancu
- stringovi ne bi trebali da se koriste kao monitor objekti (zbog optimizacija)
 - najbolje je da se za to koriste objekti ili instance user-defined klase
 - izbegavati primitivne tipove
- Java VM ostaje da radi sve dok je neka nit aktivna, bila ona glavna niti ili ne
- -  | `wait()` oslobađa lock koji je `synchronized` zauzeo
- `nit.join()` čeka da se nit završi i preuzima izlazni kod