

Poređenje stringova

Java dozvoljava poređenje stringova **po vrijednosti** i **po referenci**!

Poređenje stringova je potrebno pri:

- autentikaciji (metoda `equals()`),
- sortiranju (metoda `compareTo()`) i
- podudaranju referenci (operator `==`).

Poređenje stringova u javi je moguće izvesti na 3 načina:

- metoda `equals()` poredi vrijednosti dva stringa (ako se žele ignorisati mala i velika slova, koristi se metoda `equalsIgnoreCase()`)
- operator `==` poredi reference stringova (a ne njihove vrijednosti)
- metoda `compareTo()` vrši leksikografsko poređenje i vraća vrijednost da li je prvi string manji, jednak ili veći od drugog stringa

Metoda equals ()

equals() metoda poredi logičku jednakost vrijednosti dva stringa (vrijednosti stringova su jednake/nisu jednake). Metoda za zadani string i drugi string (naveden kao parametar) vraća vrijednost True ako i samo ako drugi string nije null string i ako drugi string sadrži iste znakove (karaktere) kao i zadani string.

```
/**
 * A Java program to compare two strings using equals()
 * and equalsIgnoreCase() method of the String.
 *
 */

public class CompareUsingEquals {
    public static void main(String[] args) {
        String prviString = "javatekst";
        String drugiString = "javakod";
        String treciString = "javatekst";
        String cetvrtiString = "javatekst";

        System.out.println("Poredjenje stringova upotrebom metoda equals()
i equalsIgnoreCase() \n");

        // Using equals() method
        System.out.print("prviString.equals(drugiString) : ");
        System.out.println(prviString.equals(drugiString));
        System.out.print("prviString.equals(treciString) : ");
        System.out.println(prviString.equals(treciString));

        /*
         * Using equalsIgnoreCase() method to ignore
         * case consideration (i.e. Capital or small) of both the strings.
         */
        System.out.print("prviString.equalsIgnoreCase(cetvrtiString) : ");
        System.out.println(prviString.equalsIgnoreCase(cetvrtiString));
    }
}
```

OUTPUT :

```
Poredjenje stringova upotrebom metoda equals() i
equalsIgnoreCase()

prviString.equals(drugiString) : false
prviString.equals(treciString) : true
prviString.equalsIgnoreCase(cetvrtiString) : true
```

Operator ==

U klasi String operator == se koristi za poređenje referenci navedenih stringova, da li se odnose na isti objekat. Kada se dva stringa porede operatorom == rezultat će biti True ako varijable pokazuju prema istom Java objektu, inače vraća vrijednost False.

```
/**
 * A Java program to compare two strings using equals()
 * and equalsIgnoreCase() method of the String.
 */
public class CompareUsingEqualsToOperator {
    public static void main(String[] args) {
        String prviString = "javatekst";
        String drugiString = "javakod";
        String treciString = "javatekst";
        String cetvrtiString = new String ("javatekst");

        System.out.println("Poredjenje stringova upotrebom operatora ==");

        // Using operator ==
        System.out.print("prviString == drugiString : ");
        System.out.println(prviString == drugiString);
        System.out.print("prviString == treciString : ");
        System.out.println(prviString == treciString);

        System.out.print("prviString == cetvrtiString : ");
        System.out.println(prviString == cetvrtiString );
    }
}
```

OUTPUT:

```
Poredjenje stringova upotrebom operatora ==

prviString == drugiString : false
prviString == treciString : true
prviString == cetvrtiString : false
```

Kada varijabli stringa dodijelimo vrijednost niza, JVM će provjeriti da li niz s jednakom vrijednošću već postoji u registru nizova ili ne. Ako nije prisutan u skupu nizova, dodaće ga u registar konstanti i vraća se referenca na taj objekt niza. Ako je prisutan u registru nizova, vraća se referenca na memorijsku adresu postojećeg string objekta.

Kada kreiramo string pomoću operatora new, stvara se novi objekat stringa i pohranjuje u Java heap prostor.

Metoda compareTo ()

compareTo () metoda vrši leksikografsko poređenje i vraća vrijednost da li je prvi string manji, jednak ili veći od drugog stringa.

Vraća negativni cijeli broj ako je string naveden kao parametar leksikografski veći od zadanog stringa, tj ako string naveden kao parametar slijedi zadani niz. (parametarski string > zadani string)

Vraća pozitivan cijeli broj ako je string naveden kao parametar leksikografski manji od zadanog stringa, tj. ako string naveden kao parametar prethodi zadanom nizu. (parametarski string < zadani string)

Metoda vraća nulu ako su oba stringa leksikografski jednaka. (parametarski string = zadani string).

Ako se želi zanemariti velika/mala slova u oba stringa, koristi se metoda compareToIgnoreCase().

```
public class CompareUsingCompareTo {
    public static void main(String[] args) {
        String prviString = "javarar";
        String drugiString = "javakod";
        String treciString = "javarolls";
        String cetvrtiString = "javaRolls";
        String petiString = "javarar";
        System.out.println("Poredjenje stringova upotrebom metoda
compareTo() and compareToIgnoreCase() \n");

        // Using compareTo() method
        System.out.print("prviString.compareTo(drugiString) : ");
        System.out.println(prviString.compareTo(drugiString));
        System.out.print("prviString.compareTo(treciString) : ");
        System.out.println(prviString.compareTo(treciString));
        System.out.print("prviString.compareTo(cetvrtiString) : ");
        System.out.println(prviString.compareTo(cetvrtiString));
        System.out.print("prviString.compareTo(petiString) : ");
        System.out.println(prviString.compareTo(petiString));

        /*
        * metoda compareToIgnoreCase() se koristi kada se želi ignorisati
        * postojanje malih ili velikih slova
        */
        System.out.print("treciString.compareTo(cetvrtiString) : ");
        System.out.println(treciString.compareTo(cetvrtiString));

        System.out.print("treciString.compareToIgnoreCase(cetvrtiString):");
        System.out.println(treciString.compareToIgnoreCase(cetvrtiString));
    }
}
```

OUTPUT:

```
prviString.compareTo(drugiString) : 7
prviString.compareTo(treciString) : -14
prviString.compareTo(cetvrtiString) : 32
prviString.compareTo(petiString) : 0
treciString.compareTo(cetvrtiString) : 32
treciString.compareToIgnoreCase(cetvrtiString) : 0
```