Академија струковних
студија Шумадија
одсек у Крагујевцу

Увод у програмирање

Презентација 13

Академија струковних студија Шумадија

Одсек у Крагујевцу

Студијски програм Информатика

Крагујевац, 2021. година

Објектно оријентисано програмирање

Класа String

- Класа **String** је предефинисана класа у Јава библиотеци која омогућава формирање текста који чини скуп знаковних типова, тј. **char**.
- Низ или скуп знаковних карактера можемо представити и овако, међутим то није практично јер се губи велики број манипулација:

```
char[] stringIme = new char[10];
```

- Класа **String** је предефинисана у Јавином пакету **java.lang**.

```
String ime = "Aleksandar";
```

- Класа **String** не дозвољава директнзу манипулацију карактерима стринга.
- Класа **String** пружа мноштво метода не само за основне операције над стринговима, већ и напредније процесирање текстуалних података.

Објектно оријентисано програмирање

- Класа **String** је непроменљива (енгл. *immutable*) што значи, када се декларишу, њихове вредности се не могу мењати, тј. немогуће је променити садржај постојећег стринга у меморији.
- Ако је резултат неке операције неки други стринг, та инстанца класе **String** биће изнова направљена.
- Класа **String** је коначна, `public final class String`, што значи да се не може наследити.
- Објекте типа **String** могуће је креирати без коришћења оператора `new`:

```
String ime = "Aleksandar";
```

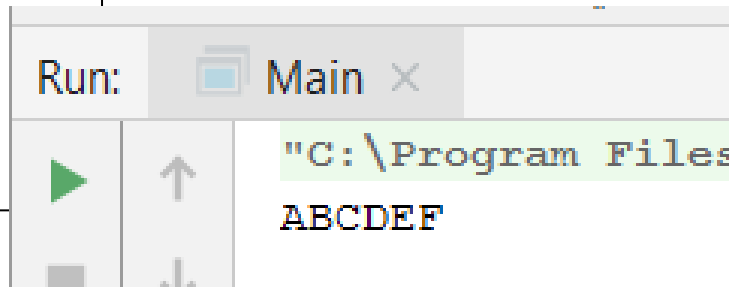
- Стрингови се могу спајати коришћењем оператора + без позива специјалног метода:

```
String ime = "Aleksandar" + "Mišković";
```

Објектно оријентисано програмирање

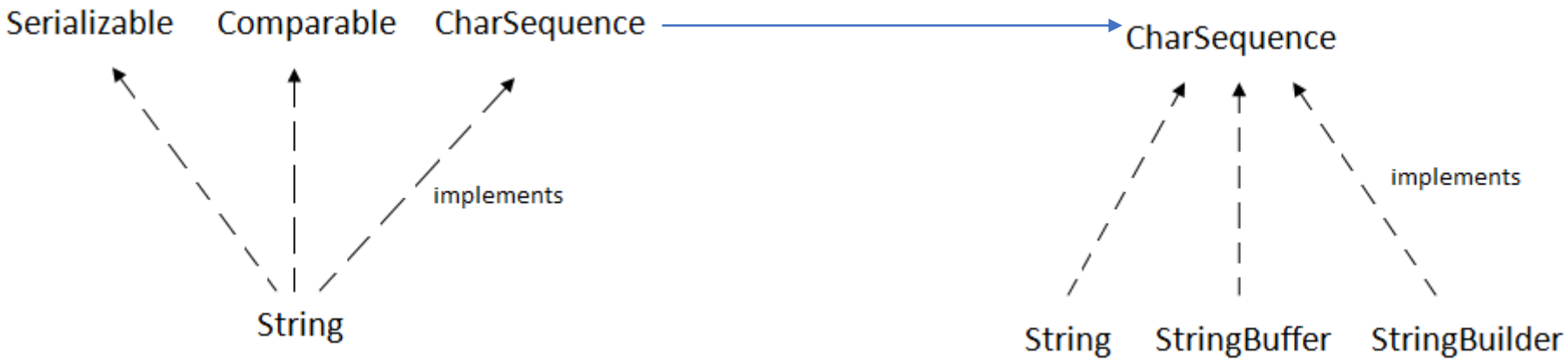
- Уколико нам затреба стринг који може да се мења, онда користимо класе **StringBuffer** и **StringBuilder**.
- Иако се објекат класе **String** може направити без позивања оператора **new**, то не значи да класа **String** нема конструктор(е). Напротив!
- Према *Oracle* документацији **String** класа има преко 10 конструктора.
- Нпр.: `String ime = new String("Aleksandar");`
- А може и овако:

```
public static void main(String[] args) {
    byte ascii[] = {65, 66, 67, 68, 69, 70};
    String ime = new String(ascii);
    System.out.println(ime);
}
```



Објектно оријентисано програмирање

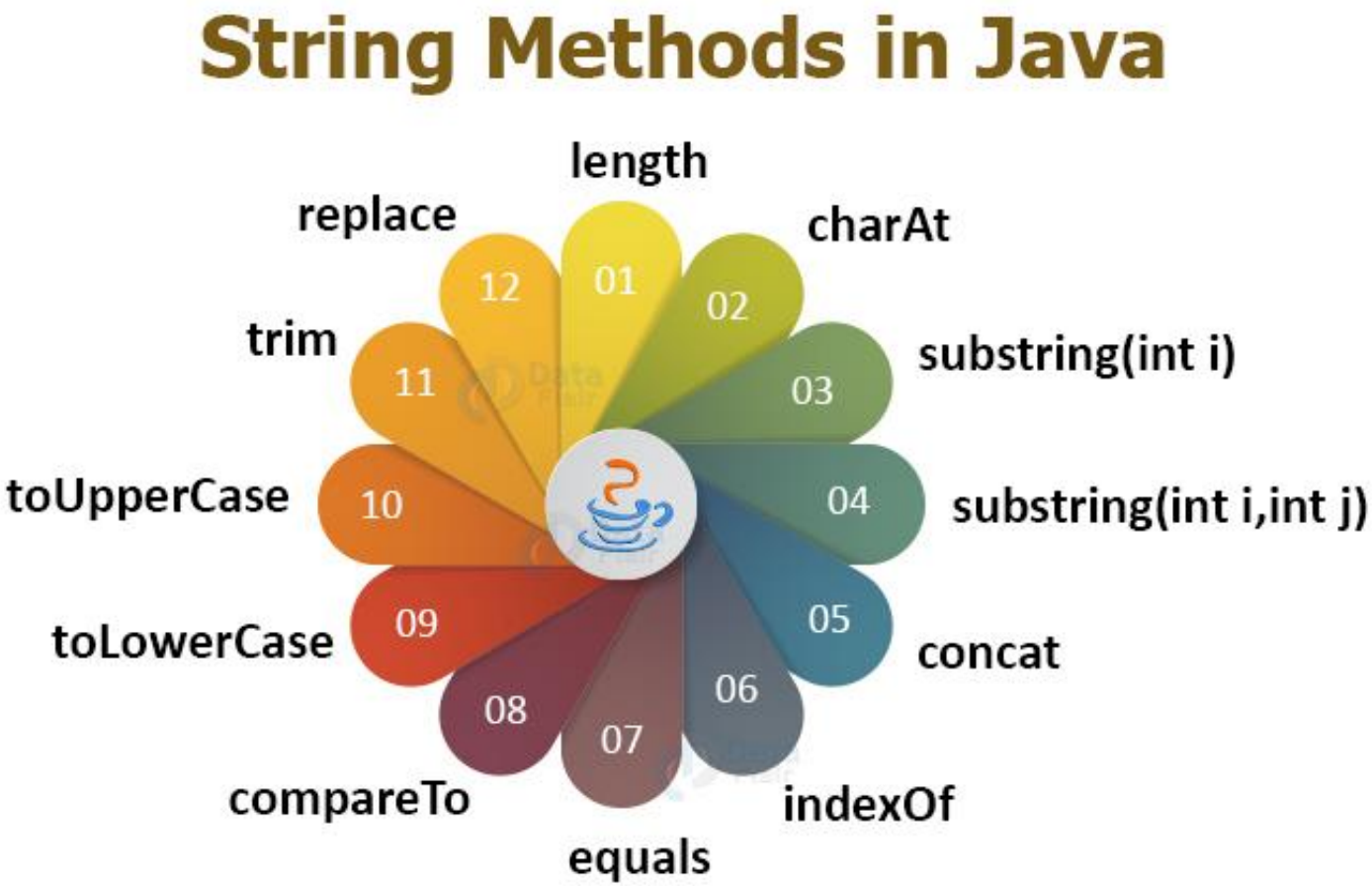
- Класа **String** имплементира три интерфејса: **Serializable**, **Comparable** и **CharSequence**.



- Интерфејс **CharSequence** користи се за представљање низа знакова.
- Класе **String**, **StringBuffer** и **StringBuilder** то примењују.

Објектно оријентисано програмирање

- Класа **String** уз друге помоћне класе, пружа **много метода** које омогућавају лако и ефикасно манипулисање стринговима.



Објектно оријентисано програмирање

Пример 1: Написати програм који омогућава унос стринга, а потом исписује његову дужину.

```
1  package zadatak_1;
2
3  import java.util.Scanner;
4
5  public class DužinaStringa {
6
7      private static Scanner ulaz;
8
9      public static void main(String[] args) {
10         ulaz = new Scanner(System.in);
11         System.out.println("Unesite tekst :");
12         String tekst = ulaz.nextLine();
13         System.out.println(tekst);
14         System.out.println("Broj karaktera :" + tekst.length());
15     }
16 }
```

Објектно оријентисано програмирање

Пример 2: Написати програм који омогућава унос стринга, а потом га исписује, слово по слово. Свако слово треба да буде одвојено зарезом.

```java
package zadatak_2;

import java.util.Scanner;

public class SlovoPoSlovo {

    private static Scanner ulaz;

    public static void main(String[] args) {
        ulaz = new Scanner(System.in);

        System.out.println("Unesite tekst :");
        String tekst = ulaz.nextLine();
        System.out.println(tekst);

        //koristimo foreach petlju:
        for(char slovo: tekst.toCharArray()){
            System.out.print(slovo + " , ");
        }

        System.out.print("\n");

        //isto to, samo što koristimo for petlju:
        for (int i = 0; i < tekst.length(); i++) {
            System.out.print(tekst.charAt(i) + " , ");
        }
    }
}
```

Објектно оријентисано програмирање

Пример 3: Написати програм који омогућава унос стринга, а потом га исписује уназад.

```
1  package zadatak_3;
2
3  import java.util.Scanner;
4
5  public class IspisiUnazad {
6
7      private static Scanner ulaz;
8
9      public static void main(String[] args) {
10         ulaz = new Scanner(System.in);
11
12         System.out.println("Unesite tekst :");
13         String tekst = ulaz.nextLine();
14         System.out.println(tekst);
15
16         for(int i = tekst.length() - 1; i >= 0; i-- ){
17             System.out.print(tekst.charAt(i));
18         }
19     }
20 }
```

Објектно оријентисано програмирање

Пример 4: Неке од метода класе **String**

```
1  package zadatak_4;
2
3  /* Nakon svakog bloka naredbi možete da pokrenete program
4   * i da vidite rezultate. Skrolujte u konzoli na početak,
5   * pa pogledajte date primere.
6   */
7
8  public class MetodeString {
9
10     public static void main(String[] args) {
11         // Spajanje stringova:
12         String S1 = "Visoka";
13         String S2 = "tehnička škola";
14         System.out.println(S1 + " " + S2);
15         System.out.println("\n");
```

Објектно оријентисано програмирање

```
17          // Poređenje stringova:
18          String prviString = "Kragujevac";
19          String drugiString = "KRAGUJEVAC";
20
21          if(prviString.equals(drugiString)){
22              System.out.println("Stringovi su jednaki!");
23          }
24          else{
25              System.out.println("Stringovi nisu jednaki!");
26          }
27          System.out.println("\n");
28
29          System.out.println("Ako se ignorišu mala i velika slova, dobija se: ");
30
31          if(prviString.equalsIgnoreCase(drugiString)){
32              System.out.println("Stringovi su jednaki!");
33          }
34          else{
35              System.out.println("Stringovi nisu jednaki!");
36          }
37          System.out.println("\n");
```

Објектно оријентисано програмирање

```
39          // Mala i velika slova
40          String S3 = S1 + " " + S2 + " " + prviString;
41          System.out.println("String S3 izgleda ovako: " + S3);
42          System.out.println("String S3 sada izgleda ovako: " + S3.toUpperCase());
43          System.out.println("A može da izgleda i ovako: " + S3.toLowerCase());
44          System.out.println("\n");
45
46          // Dužina stringa i karakter na određenom mestu
47          System.out.println("Naš string: " + S3);
48          System.out.println("je dužine: " + S3.length() + " karaktera.");
49          System.out.println("Karakter na mestu 3 je: " + S3.charAt(2));
50          System.out.println("Prvo pojavljivanje slova a je na mestu: "
51                             + S3.indexOf("a"));
52          System.out.println("\n");
53
54          // Podniz stringova
55          System.out.println("Naš substring od pozicije 7: " + S3.substring(7));
56          System.out.println("Naš substring između pozicije 7 i 15: "
57                             + S3.substring(7, 15));
58          System.out.println("\n");
59      }
60  }
```

Објектно оријентисано програмирање

- Класа **StringBuffer** представља измењиви знаковни низ.
- За разлику од објеката класе **String**, објекти класе **StringBuffer** су секвенце знакова промењљиве дужине који се могу мењати.
- Објекат класе **StringBuffer** аутоматски се проширује да би прихватио додатне податке.
- Конструктори класе **StringBuffer**:
  - **`StringBuffer()`** - Резервише простор за 16 карактера.
  - **`StringBuffer(int capacity)`** - Резервише простор за прослеђени број карактера.
  - **`StringBuffer(String string)`** - Резервише простор за дужину прослеђеног стринга и смешта стринг у тај простор.
  - **`StringBuffer(CharSequence seq)`** – Резервише простор за исте карактере као наведени интерфејс **CharSequence.**
- Методе класе **StringBuffer**:
  https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/lang/StringBuffer.html

Објектно оријентисано програмирање

- Класа **StringBuilder** је слична класи **StringBuffer** уз једну веома битну разлику: она није синхронизована, тј. није безбедна за вишенитни рад.
- Вишенитно програмирање у Јави учићемо на предмету ООП.
- Класа **StringBuilder** је има исте методе као и класа **StringBuffer**.
- Класа **StringBuilder** има боље перформансе у односу на класу **StringBuffer**.
- Међутим, уколико је потребно користити вишенитно програмирање треба имати на уму да класа **StringBuilder** није безбедна за рад у таквом окружењу.

Литература

- Одабрана поглавља из књиге: Јава JDK9: Комплетан приручник
  - Аутор: *Herbert Schildt (може и стариje издање JDK7).*
- https://data-flair.training/blogs/java-string-methods-and-constructor/
- https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/lang/String.html
- https://www.javatpoint.com/java-string