Chapter 6

String Methods

String in Java is an object that represents a sequence of characters. It is a great way to store information. Because character strings are so vital to programming, Java dedicated an entire class to them. String class provides a lot of useful methods to help us perform operations on strings and data manipulation.

String methods that you should know:

- length()
- concat()
- equals()
- indexOf()
- charAt()
- substring()
- toUpperCase() / toLowerCase()

Try the following code and guess what each method does.

```
public class HelloString {
   public static void main(String[] args) {
    String str = "President University!";
    System.out.println(str.length());
    System.out.println(str.substring(10));
    System.out.println(str.toUpperCase());

  }
}
```

1. length()

In Java, the length() string method returns the length — total number of characters — of a String.

```
public class CharCounter {
   public static void main(String[] args) {
     String slogan = "Where tomorrow\'s leaders come together.";
     System.out.println(slogan.length());
```

```
  }
}
```

2. concat()

The concat() method joins two string together.

Suppose we have a String called str1 and another String called str2, using str1.concat(str2) would return str1 with str2 appended to the end of it.

```
public class CombineNames {
   public static void main(String[] args) {
     String firstName = "Ziggy";
     String lastName = "Stardust";
     System.out.println(firstName.concat(" " + lastName));
  }
}
```

3. equals()

With objects, such as Strings, we can't use the primitive equality operator == to check for equality between two strings. To test equality with strings, we use a built-in method called equals().

```
public class Password {
  public static void main(String[] args) {
    String password = "correcthorsebatterystaple";
 System.out.println(password.equals("Tr0ub4dor&3"));
  }
}
```

4. indexOf()

If we want to know the index of the first occurence of a character in a string, we can use the indexOf() method on a string.

Remember that the indices in Java start with 0.

```
public class President {
  public static void main(String[] args) {
    String code = "President University";
```

```
    System.out.println(code.indexOf("side"));
  }
}
```

5. charAt()

The charAt() method returns the character located at a String's specified index.

Example:

```
public class Initials {
   public static void main(String[] args) {

    // Add a first name and a last name:
    String firstName = "Sandy ";
    String lastName = "Darmowinoto";

    // What are the initials?
    System.out.println(firstName.charAt(0));
    System.out.println(lastName.charAt(0));

  }

}
```

6. substring()

There may be times when we only want a part of a string. In such cases, we may want to extract a substring from a string.

The substring() method does exactly that.

Example:

```
public class Poetry {

   public static void main(String[] args) {

    String line = "The Heav'ns and all the Constellations rung";
    // Change the arguments:
    System.out.println(line.substring(4, 11));
```

```
  }
}
```

7. toUpperCase() / toLowerCase()

There will be times when we have a word in a case other than what we need it in.

toUpperCase(): returns the string value converted to uppercase

toLowerCase(): returns the string value converted to lowercase

Example:

```
public class Hashtag {
  public static void main(String[] args) {

    String hashtag = "#100DaysOfCode";

    // Make the hashtag lowercase:
    System.out.println(hashtag.toLowerCase());
  }
}
```

8. toCharArray()

What does this method do?

```
public class StringtoChar{

   public static void main(String[] args) {
         String someText = "Welcome to PresUniv.";
         char[] myArray = someText.toCharArray();

         for (char output : myArray) {
               System.out.println(output);
         }
   }
}
```