String

Java™

Java.lang.String

- Strings are objects that contains a char array
- They are immutable and its state can never be changed. It produces new String Objects when you ’update’ a String.

Here your String is stored char by char in a char Array

```
/** The value is used for character storage. */
private final char value[];
```

charAt() and length() methods

```
String name = "Erik Svensson";

for(int i=0; i<name.length(); i++) {

    System.out.println(name.charAt(i));

}
```

length() method returns an int representing number of chars in the String

charAt() method returns the char at given index

indexOf() method

indexOf() looks through the char values and return the index of the first match.
Can also seach from a start index if specified. The search term can be both char and String

```
String string = "Hello Strings";

int index1 = string.indexOf("llo");      // index1 == 2
int index2 = string.indexOf('H');        // index2 == 0
int index3 = string.indexOf('l', 3);     // index3 == 3
int index4 = string.indexOf("Hell", 1); // index4 == -1
```

Will return -1 when match is not found

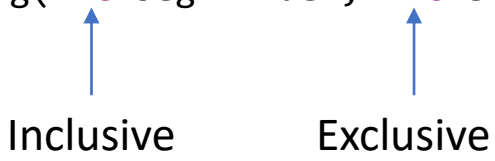
substring() method

This method return parts of a String based on passed in index value(s).

Method signatures defined in the String class:

```
public String substring(int beginIndex)

public String substring(int beginIndex, int endIndex)
```



Index	0	1	2	3	4	5	6
char	w	e	l	c	o	m	e

```
String greeting = "Welcome";
System.out.println(greeting.substring(3));     //come
System.out.println(greeting.substring(1, 5));  //elco
System.out.println(greeting.substring(4, 10)); //StringIndexOutOfBoundsException
```

toLowerCase() and toUpperCase()

These methods to exactly like they say.

Method signatures defined in the String class:

```
public String toLowerCase()
public String toUpperCase()
```

```
String str1 = "A1,b2,c3,D4";
str1 = str1.toLowerCase(); //a1,b2,c3,d4
String str2 = "e5,F6,G7,h8";
str2 = str2.toUpperCase(); //E5,F6,G7,H8
```

equals() and equalsIgnoreCase()

These two methods checks if two String objects have exactly the same characters in the same order. equalsIgnoreCase() will do case convertion if needed.

Method signatures defined in the String class:

```
public boolean equals(Object obj)
public boolean equalsIgnoreCase(String str)
```

```
String str1 = "nisse";
String str2 = "Nisse";

System.out.println(str1.equals("nisse"));              //true
System.out.println(str1.equals(str2));                 //false
System.out.println(str1.equalsIgnoreCase(str2));       //true
```

startsWith() and endsWith()

These methods checks if the value passed in matches the start and end of String object in a case sensitive manner.

Method signatures defined in the String class:

```
public boolean startsWith(String prefix)
public boolean endsWith(String suffix)
```

```
String name = "Erik Svensson";

System.out.println(name.startsWith("Erik"));          //true
System.out.println(name.endsWith("sson"));            //true
System.out.println(name.startsWith("e"));             //false
System.out.println(name.endsWith("svensson"));        //false
```

contains()

Looks for matches in the String in a case sensitive manner

Method signature defined in the String class:

```
public boolean contains(String str)
```

```
String name = "Ulf Bengtsson";

System.out.println(name.contains("ulf"));      //false
System.out.println(name.contains(" "));        //true
System.out.println(name.contains("engts"));    //true
```

replace()

Does simple search and replace on the String object. There are two versions of this method. One takes chars and the other two CharSequence objects.

Method signatures defined in the String class:

```
public String replace(char oldChar, char newChar)
public String replace(CharSequence oldChar, CharSequence newChar)
```

```
String text = "abracadabra";

System.out.println(text.replace('a', 'A'));              //AbrAcAdAbrA
System.out.println(text.replace("abra", "dabra"));       //dabracaddabra
System.out.println(text.replace("cad"," "));             //abra abra
```

trim()

The trim method removes whitespaces along with special characters like \t (tab) and \n (newline) from the beginning and the end of the String.

Method signature defined in the String class:

```
public String trim()
```

```
String simple = "  java ";
String str = "\ta\tb\tc\t  ";

System.out.println(simple.trim());    //”java”
System.out.println(str);              //”    a    b    c       "
System.out.println(str.trim());       //"a    b    c"
```