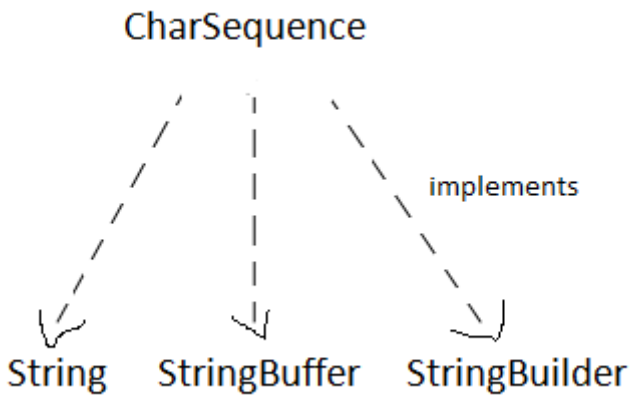
Chapter 4 – Unit 2
Strings

- **CharSequenceInterface : The CharSequence interface is used to represent the sequence of characters.**
- **It implements String, StringBuffer and StringBuilder classes.**
- It means, we can create strings in Java by using these three classes.



- **Strings, which are widely used in Java programming, are a sequence of characters. In Java programming language, strings are treated as objects.**

String class

- **String is a sequence of characters. In java, objects of String are immutable which means a constant and cannot be changed once created.**

String class methods

- The java.lang.String class provides many useful methods to perform operations on sequence of char values.

No.	Method	Description
1	char charAt(int index)	returns char value for the particular index
2	int length()	returns string length
3	static String format(String format, Object... args)	returns a formatted string.
4	static String format(Locale l, String format, Object... args)	returns formatted string with given locale.
5	String substring(int beginIndex)	returns substring for given begin index.
6	String substring(int beginIndex, int endIndex)	returns substring for given begin index and end index.
7	boolean contains(CharSequence s)	returns true or false after matching the sequence of char value.
8	static String join(CharSequence delimiter, CharSequence... elements)	returns a joined string.
9	static String join(CharSequence delimiter, Iterable<? extends CharSequence> elements)	returns a joined string.
10	boolean equals(Object another)	checks the equality of string with the given object.
11	booleanisEmpty()	checks if string is empty.
12	String concat(String str)	concatenates the specified string.
13	String replace(char old, char new)	replaces all occurrences of the specified char value.
14	String replace(CharSequence old, CharSequence new)	replaces all occurrences of the specified CharSequence.
15	static String equalsIgnoreCase(String another)	compares another string. It doesn't check case.

16	String[] split(String regex)	returns a split string matching regex.
17	String[] split(String regex, int limit)	returns a split string matching regex and limit.
18	String intern()	returns an interned string.
19	int indexOf(int ch)	returns the specified char value index.
20	int indexOf(int ch, int fromIndex)	returns the specified char value index starting with given index.
21	int indexOf(String substring)	returns the specified substring index.
22	int indexOf(String substring, int fromIndex)	returns the specified substring index starting with given index.
23	String toLowerCase()	returns a string in lowercase.
24	String toLowerCase(Locale l)	returns a string in lowercase using specified locale.
25	String toUpperCase()	returns a string in uppercase.
26	String toUpperCase(Locale l)	returns a string in uppercase using specified locale.
27	String trim()	removes beginning and ending spaces of this string.
28	static String valueOf(int value)	converts given type into string. It is an overloaded method.

Stringbuffer class

- **StringBuffer class is used to create mutable (modifiable) String objects. The StringBuffer class in Java is the same as String class except it is mutable i.e. it can be changed.**

Important Constructors of StringBuffer Class

Constructor	Description
StringBuffer()	It creates an empty String buffer with the initial capacity of 16.
StringBuffer(String str)	It creates a String buffer with the specified string..
StringBuffer(int capacity)	It creates an empty String buffer with the specified capacity as length.

Important methods of StringBuffer class

Sr.No.	Method & Description
1	StringBuffer append(boolean b) This method appends the string representation of the boolean argument to the sequence.
2	StringBuffer append(char c) This method appends the string representation of the char argument to this sequence.
3	StringBuffer append(char[] str) This method appends the string representation of the char array argument to this sequence.
4	StringBuffer append(char[] str, int offset, int len) This method appends the string representation of a subarray of the char array argument to this sequence.
5	StringBuffer append(CharSequence s) This method appends the specified CharSequence to this sequence.
6	StringBuffer append(CharSequence s, int start, int end) This method appends a subsequence of the specified CharSequence to this sequence.
7	StringBuffer append(double d) This method appends the string representation of the double argument to this sequence.

8	StringBuffer append(float f) This method appends the string representation of the float argument to this sequence.
9	StringBuffer append(int i) This method appends the string representation of the int argument to this sequence.
10	StringBuffer append(long lng) This method appends the string representation of the long argument to this sequence.
11	StringBuffer append(Object obj) This method appends the string representation of the Object argument.
12	StringBuffer append(String str) This method appends the specified string to this character sequence.
13	StringBuffer append(StringBuffersb) This method appends the specified StringBuffer to this sequence.
14	StringBufferappendCodePoint(int codePoint) This method appends the string representation of the codePoint argument to this sequence.
15	int capacity() This method returns the current capacity.
16	char charAt(int index) This method returns the char value in this sequence at the specified index.
17	int codePointAt(int index) This method returns the character (Unicode code point) at the specified index
18	int codePointBefore(int index) This method returns the character (Unicode code point) before the specified index.
19	int codePointCount(int beginIndex, int endIndex) This method returns the number of Unicode code points in the specified text range of this sequence.
20	StringBuffer delete(int start, int end) This method removes the characters in a substring of this sequence.
21	StringBufferdeleteCharAt(int index) This method removes the char at the specified position in this sequence.

22	void ensureCapacity(int minimumCapacity) This method ensures that the capacity is at least equal to the specified minimum.
23	void getChars(int srcBegin, int srcEnd, char[] dst, int dstBegin) This method characters are copied from this sequence into the destination character array dst.
24	int indexOf(String str) This method returns the index within this string of the first occurrence of the specified substring.
25	int indexOf(String str, int fromIndex) This method returns the index within this string of the first occurrence of the specified substring, starting at the specified index.
26	StringBuffer insert(int offset, boolean b) This method inserts the string representation of the boolean argument into this sequence.
27	StringBuffer insert(int offset, char c) This method inserts the string representation of the char argument into this sequence.
28	StringBuffer insert(int offset, char[] str) This method inserts the string representation of the char array argument into this sequence.
29	StringBuffer insert(int index, char[] str, int offset, int len) This method inserts the string representation of a subarray of the str array argument into this sequence.
30	StringBuffer insert(int dstOffset, CharSequence s) This method inserts the specified CharSequence into this sequence.
31	StringBuffer insert(int dstOffset, CharSequence s, int start, int end) This method inserts a subsequence of the specified CharSequence into this sequence.
32	StringBuffer insert(int offset, double d) This method inserts the string representation of the double argument into this sequence.
33	StringBuffer insert(int offset, float f) This method inserts the string representation of the float argument into this sequence.
34	StringBuffer insert(int offset, int i

	This method inserts the string representation of the second int argument into this sequence.
35	StringBuffer insert(int offset, long l) This method inserts the string representation of the long argument into this sequence.
36	StringBuffer insert(int offset, Object obj) This method inserts the string representation of the Object argument into this character sequence.
37	StringBuffer insert(int offset, String str) This method inserts the string into this character sequence.
38	int lastIndexOf(String str) This method returns the index within this string of the rightmost occurrence of the specified substring.
39	int lastIndexOf(String str, int fromIndex) This method returns the index within this string of the last occurrence of the specified substring.
40	int length() This method returns the length (character count).
41	int offsetByCodePoints(int index, int codePointOffset) This method returns the index within this sequence that is offset from the given index by codePointOffset code points.
42	StringBuffer replace(int start, int end, String str) This method replaces the characters in a substring of this sequence with characters in the specified String.
43	StringBuffer reverse() This method causes this character sequence to be replaced by the reverse of the sequence.
44	void setCharAt(int index, char ch) The character at the specified index is set to ch.
45	void setLength(int newLength) This method sets the length of the character sequence.
46	CharSequencesubSequence(int start, int end) This method returns a new character sequence that is a subsequence of this sequence.
47	String substring(int start)

	This method returns a new String that contains a subsequence of characters currently contained in this character sequence.
48	String substring(int start, int end) This method returns a new String that contains a subsequence of characters currently contained in this sequence.
49	String toString() This method returns a string representing the data in this sequence.
50	void trimToSize() This method attempts to reduce storage used for the character sequence.

Difference between String and StringBuffer

- There are many differences between String and StringBuffer. A list of differences between String and StringBuffer are given below:

No.	String	StringBuffer
1)	The String class is immutable.	The StringBuffer class is mutable.
2)	String is slow and consumes more memory when we concatenate too many strings because every time it creates new instance.	StringBuffer is fast and consumes less memory when we concatenate t strings.
3)	String class overrides the equals() method of Object class. So you can compare the contents of two strings by equals() method.	StringBuffer class doesn't override the equals() method of Object class.

4)	String class is slower while performing concatenation operation.	StringBuffer class is faster while performing concatenation operation.
5)	String class uses String constant pool.	StringBuffer uses Heap memory