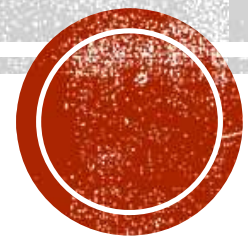
STRING IN JAVA



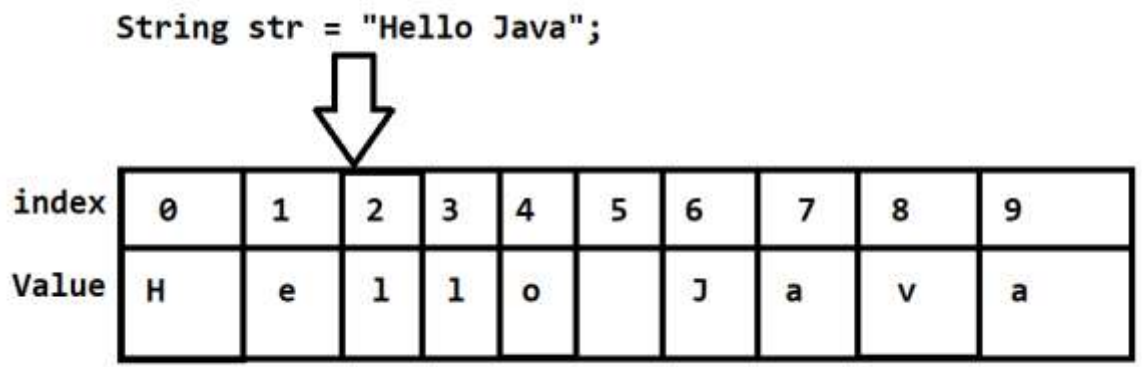
STRING IN JAVA

1. String is a non-primitive Data type.
2. String is an array of character. (combination of multiple characters)
3. In Java String values are always in the double quotes(“Value”)
4. To work with string java has provided multiple classes
    a. String class
    b. StringBuffer class
    c. StringBuilder class
5. These classes are the build-in classes. And there is multiple build-in function provided in this class using which you can perform operations on existing string values.

JavaInBeats
https://javainbeats.com/

STRING CLASS

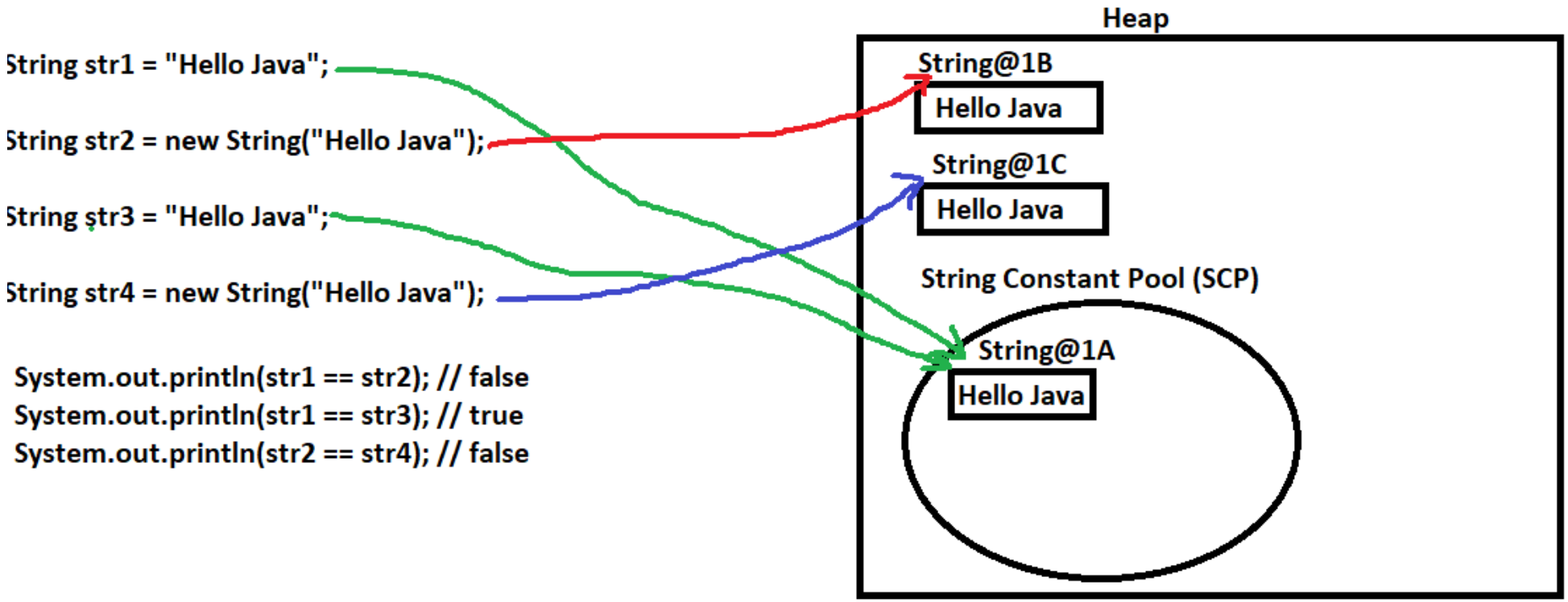
1. String is a **build-in class**.
2. String class present inside **java.lang package**.
3. String class is a **final class**.
4. String objects are **immutable**. That is once you assign a value to a string it never changes by applying any of the string function.
5. In String the value internally store in the form for index and values format (array format).
6. To use a String you have to create object of String class. Object creation can be achieve by 2 ways.
   a. With new operator
      i. The object will be store inside Heap directly.
      ii. Syntax:
         - **String str = new String("Value");**
   a. Without new Operator
      i. The object will be store in the SCP which is part of Heap.
      ii. Example:
         - **String str = "value";**



JavaInBeats
https://javainbeats.com/

STRING CONSTANT POOL (SCP)

1. All the string values store in side SCP which is created without new operator.
2. While storing any values inside SCP it will first check of the same value is present or not. If same value is present then no new object will be created of it instead it will returns a same object.



JavaInBeats
https://javainbeats.com/

STRING METHODS

Method	Description
public int length();	Return the total number of character in the string
public boolean isEmpty();	Return true if string is empty else returns false
public char charAt(int);	Returns the character at specific index
public boolean equals(java.lang.Object);	Returns true if both the string has same values else returns false
public boolean equalsIgnoreCase(java.lang.String);	Returns true if both the string has same values by ignoring the spaces else returns false
public boolean startsWith(java.lang.String);	Returns true if the string starts with the given value else returns false.
public boolean endsWith(java.lang.String);	Returns true if the string ends with the given value else returns false.
public java.lang.String toLowerCase();	To convert string into small/lower case
public java.lang.String toUpperCase();	To convert string into capital case.
public java.lang.String trim();	Remove the spaces from the start and end of string.
public char[] toCharArray();	Return the array of character from the string. Every character of the string is in the separate index.
public java.lang.String[] split(java.lang.String);	Split the string into multiple words.
public boolean contains(java.lang.CharSequence);	Return true if the string word is present inside the targeted string.
public java.lang.String concat(java.lang.String);	Append the string (word) value at the end of existing string.

JavaInBeats
https://javainbeats.com/

STRINGBUILDER CLASS

1. Can store string value using this class.
2. StringBuilder is a build-in class
3. StringBuilder is a final class which is present inside java.lang package.
4. StringBuilder object is mutable. That is if you made any changes inside the string value from the StringBuilder then it will update the original value.
5. In StringBuilder no SCP concept applicable.
6. You can use StringBuilder and its functionality by creating an Object of class.
7. Syntax:
   - **StringBuilder object = new StringBuilder(“Value”);**

JavaInBeats
https://javainbeats.com/

STRINGBUFFER CLASS

1. Can store string value using this class.
2. StringBuffer is a build-in class
3. StringBuffer is a final class which is present inside java.lang package.
4. StringBuffer object is mutable. That is if you made any changes inside the string value from the StringBuffer then it will update the original value.
5. In StringBuffer no SCP concept applicable.
6. You can use StringBuffer and its functionality by creating an Object of class.
7. All the methods of the StringBuffer are synchronized.
8. The Object of StringBuffer is thread safe.
9. It is slower in the performance than StringBuilder.
10. Syntax:
    - **StringBuffer object = new StringBuffer(“Value”);**

JavaInBeats
https://javainbeats.com/