1. What is String in Java?

A String is a sequence of characters. In the Java programming language, strings are objects. A String is immutable so once a String object is created, its content can't be changed.

From Java API, the String is a Java Class defined in **java.lang package**. It's not a **primitive data type** like `int` and `long` .

Below is the String Class hierarchy diagram for quick reference -

2. What are different ways to create String Object?

There are two ways to create a String object:

1. By string literal
2. By new keyword

Using String Literal

Java String literal is created by using double quotes.

For Example:

```
String s="javaguides";
```

Using new Keyword

Let's create a simple example to demonstrate by creating String objects using the new keyword.

```
    String str = new String("Java Guides");
```

3. What do you mean by mutable and immutable objects?

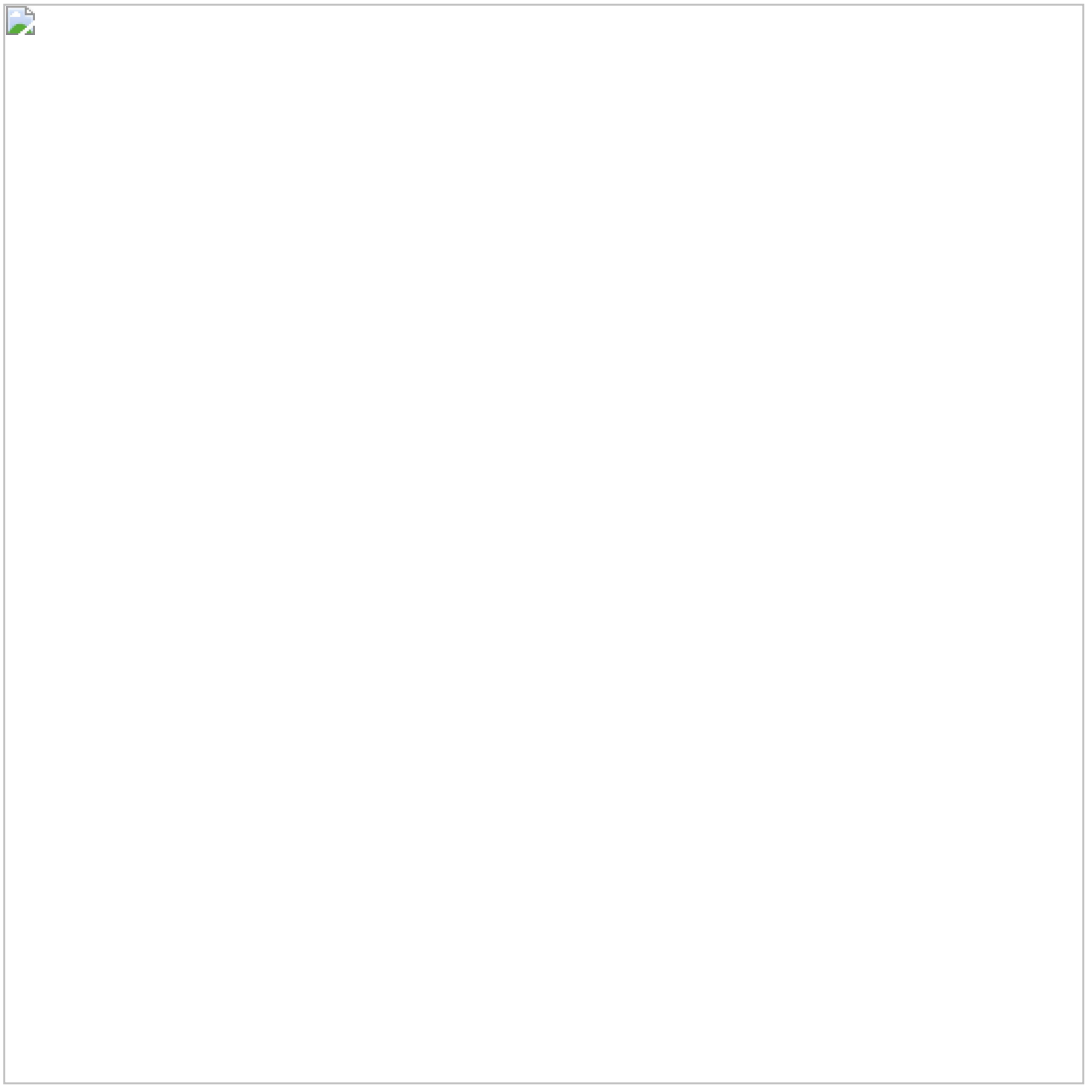
Immutable objects are like constants. You can't modify them once they are created. They are final in nature. Whereas mutable objects are concerned, you can perform modifications to them.

For example, String is immutable in Java, once you created a String object then you can't modify it. Whereas StringBuilder or StringBuffer is a mutable in Java so once you create an object of StringBuilder or StringBuffer class then you can modify it.

4. What is String Constant Pool?

String Constant Pool is the memory space in heap memory specially allocated to store the string objects created using string literals. In String Constant Pool, there will be no two string objects having the same content.

String objects are stored in a special memory area known as `String Constant Pool` .



- In the above example, only one object will be created. Firstly JVM will not find any string object with the value `"Helloworld"` in the string constant pool so it will create a new object. After that it will find the string with the value `"Helloworld"` in the pool, it will not create the new object but will return the reference to the same instance.

```
String s1 = "Helloworld";
String s2 = "Helloworld";
String s5 = "Helloworld";
```

- The "Greeting" object is created by using the new keyword, this "Greeting" object is stored in the heap memory.

```
String s4 = new String("Greeting");
```

Read more at **http://www.javaguides.net/2018/07/guide-to-java-string-constant-pool.html**

5. Does String is thread-safe in Java?

Yes, Strings are immutable in Java so once we create a String object then we can't change its content. Hence it's thread-safe and can be safely used in a multi-threaded environment.

6. Difference between == and equals() method in Java

In general, both `equals()` and "==" operator in Java are used to compare objects to check equality but here are some of the differences between the two:

1. The main difference between the `equals()` method and `==` operator is that one is a method and the other is an operator.
2. We can use `==` operators for reference comparison (address comparison) and `equals()` method for content comparison. In simple words, == checks if both objects point to the same memory location whereas `equals()` evaluates to the comparison of values in the objects.

```
public class Test {
    public static void main(String[] args)
    {
        String s1 = new String("HELLO");
        String s2 = new String("HELLO");
        System.out.println(s1 == s2);
        System.out.println(s1.equals(s2));
    }
}
```

Output:

```
false
true
```

7. Difference between String and StringBuilder?

1. A String is immutable in Java, while a StringBuilder is mutable in Java
2. A String is thread-safe, whereas StringBuilder is not a thread-safe
3. The performance of the String is slower than StringBuilder in the case of multiple concatenation operations. This is because the string is immutable in Java, and concatenation of two string objects involves creating a new object
4. String class overrides the equals() method of Object class. So you can compare the contents of two strings by the equals() method. StringBuilder class doesn't override the equals() method of Object class.

8. Difference between String and StringBuffer

1. String class is immutable and whereas StringBuffer class is mutable.
2. String is slow and consumes more memory when you perform too many concatenation String operations because every time it creates new instance. But StringBuffer is fast and consumes less memory when you cancat strings.

3. String class overrides the equals() method of Object class. So you can compare the contents of two strings by the equals() method. StringBuffer class doesn't override the equals() method of Object class.
4. String class uses String constant pool to store the objects whereas StringBuffer class uses heap memory to store its objects.

9. Difference between StringBuilder and StringBuffer?

1. A StringBuilder is not a thread-safe, whereas StringBuffer is a thread-safe
2. A StringBuffer is slower than StringBuilder. Because StringBuffer is a thread-safe implementation and therefore slower than the StringBuilder.

10. Which one will you prefer among "==" operator or equals() method to compare two string objects?

Say, equals() method because it compares two string objects based on their content. That provides more logical comparison of two string objects. If you use "==" operator, it checks only references of two objects are equal or not. It may not be suitable in all situations. So, rather stick to equals() method to compare two string objects.

11. What does the String intern() method do?

The method `intern()` creates an exact copy of a String object in the heap memory and stores it in the String constant pool.

Note that, if another String with the same contents exists in the String constant pool, then a new object won't be created and the new reference will point to the other String.

We can call the intern() method to tell the JVM to add it to the string pool if it doesn't already exist there, and return a reference of that interned string:

```
String s1 = "abc";
String s2 = new String("abc");
String s3 = new String("foo");
String s4 = s1.intern();
String s5 = s2.intern();

System.out.println(s3 == s4);
System.out.println(s1 == s5);
```

Output:

```
false
```

```
true
```

12. Why String is a popular HashMap key in Java?

Since String is immutable, its hashcode is cached at the time of creation and it doesn't need to be calculated again. This makes it a great candidate for key in a Map and it's processing is fast than other HashMap key objects. This is why String is mostly used Object as HashMap keys.

13. How many objects will be created in the following code and where they will be stored?

```
String s1 = new String("javaguides");

String s2 = "javaguides";
```

Answer:

Here, two string objects will be created. An object created using a new operator(s1) will be stored in the heap memory. The object created using a string literal(s2) is stored in the string constant pool.

14. How many objects will be created in the following code and where they will be stored?

```
String s1 = new String("javaguides");

String s2 = new String("javaguides");
```

Answer

As we have used a `new` keyword to create String objects so there are two objects will be created and they will be stored in the heap memory.

15. Write a Java program to Count the Number of Duplicate Words in String

Solution: **Write a Java program to Count the Number of Duplicate Words in String**

16. Write a Java Program to Count Number of Words in Given String

Solution: **Write a Java Program to Count Number of Words in Given String**

17. Write a Java Program to Count the Number of Occurrences of Substring in a String

Solution: **Write a Java Program to Count the Number of Occurrences of Substring in a String**

18. Write a Java Program to Count the Occurrences of Each Character in String

Solution: **Write a Java Program to Count the Occurrences of Each Character in String**

19. Write a Java Program to Check if Input String is Palindrome

Solution: **Write a Java Program to Check if Input String is Palindrome**

20. Write a Java Program to Find all Permutations of String

Solution: **Write a Java Program to Find all Permutations of String**

21. Write a Java Program to Merge two String Arrays

Solution: **Write a Java Program to Merge two String Arrays**

22. Write a Java Program to Remove Duplicate Words from String

Solution: **Write a Java Program to Remove Duplicate Words from String**

23. Write a Java Program to Reverse a String(5 ways)

Solution: **Write a Java Program to Reverse a String(5 ways)**

24. Write a Java Program to Reverse Each Word of a String

Solution: **Write a Java Program to Reverse Each Word of a String**

25. Write a Java Program to Swap Two Strings

Solution: **Write a Java Program to Swap Two Strings**

26. How to Check if the String Contains only Digits?

Solution: **How to Check if the String Contains only Digits?**

27. How to Check if the String Contains only Letters?

Solution: **How to Check if the String Contains only Letters?**

28. How to Check If the String Contains Only Letters or Digits?

Solution: **How to Check If the String Contains Only Letters or Digits?**