Difference between String literal and New String object in Java

String is a special class in Java API and has so many special behaviours which is not obvious to many programmers. In order to master Java, first step is to master String class, and one way to explore is checking what kind of String related questions are asked on Java interviews. Apart from usual questions like why String is final, or equals vs == operator, one of the most frequently asked question is *what is difference between String literal and String object in Java*. For example, what is the difference between String object created in following two expression :

```
String strObject = new String("Java");
```

and

```
String strLiteral = "Java";
```

Both expression gives you String object, but there is subtle difference between them. When you create String object using `new()` operator, it always create a new object in heap memory. On the other hand, if you create object using String literal syntax e.g. `"Java"`, it may return an existing object from String pool (a cache of String object in Perm gen space, which is now moved to heap space in recent Java release), if it's already exists. Otherwise it will create a new string object and put in string pool for future re-use. In rest of this article, why it is one of the most important thing you should remember about String in Java.

OOP Tutorials

- oop - polymorphism
- oop - object
- oop - inheritance
- oop - free courses
- oop - class vs interface

- oop - inheritance questions
- oop- interview questions
- oop - class vs object
- oop - overriding
- oop - abstraction
- oop - encpasulation
- oop - abstract class vs interface
- oop - method
- oop - constructor chaining
- oop - static method
- oop - java
- oop - abstract class
- oop - overloading vs overriding
- oop - overloading
- oop - interface
- oop - constructor

String Tutorials

- string - 101 guide
- string - contains
- string - check unique
- string - rotation
- string - count words
- string - join
- string - substring
- string - split
- string - palindrome
- string - reverse words
- string - byte array

What is String literal and String Pool

Since String is one of the most used type in any application, Java designer took a step further to optimize uses of this class. They know that Strings will not going to be cheap, and that's why they come up with an idea to cache all String instances created inside double quotes e.g. `"Java"`. These double quoted literal is known as *String literal* and the cache which stored these String instances are known as as *String pool*. In earlier version of Java, I think up-to Java 1.6 String pool is located in permgen area of heap, but in Java 1.7 updates its moved to main heap area. Earlier since it was in PermGen space, it was always a risk to create too many String object, because its a very limited space, default size 64 MB and used to store class metadata e.g. .class files. Creating too many String literals can cause java.lang.OutOfMemory: permgen space. Now because String pool is moved to a much larger memory space, it's much more safe. By the way, don't misuse memory here, always try to minimize temporary String object e.g. `"a"`, `"b"` and then `"ab"`. Always use StringBuilder to deal with temporary String object.

Difference between String literal and String object

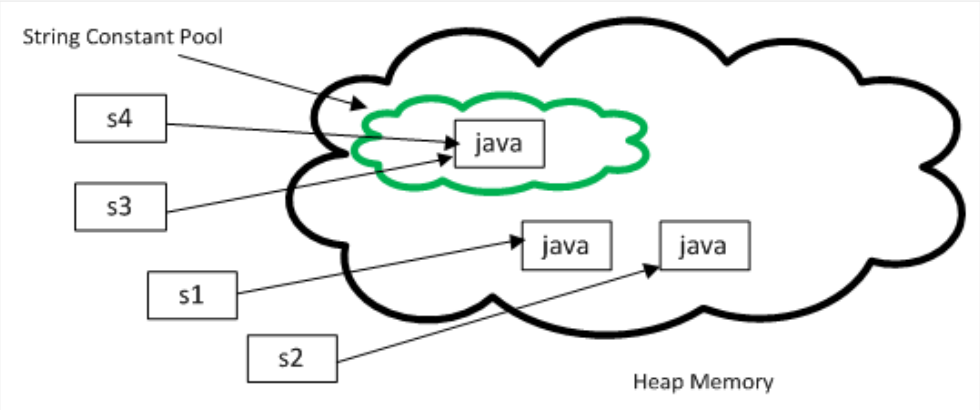
Interview Questions

- core java interview questions
- SQL interview questions
- data structure interview question
- coding interview questions
- java collection interview questions
- java design pattern interview questions
- thread interview questions
- hibernate interview questions
- j2ee interview questions
- Spring Interview Questions
- object oriented programming questions

- string - to enum
- string - compare
- string - empty
- string - stringbuffer
- string - duplicate
- string - immutable
- string - split regex
- string - remove whitespace
- string - toLowerCase
- string - reverse

Categories

- core java (451)
- core java interview question answer (124)
- programming (123)
- Java collection tutorial (111)
- Coding Problems (74)
- interview questions (65)
- data structure and algorithm (53)
- Java 8 (51)
- String (51)
- error and exception (46)
- object oriented programming (41)
- books (39)
- ArrayList (36)
- array (33)
- coding (32)
- Java Multithreading Tutorial (31)
- spring framework (30)
- date and time (23)
- sql (22)
- java io tutorial (21)
- SQL interview Question (20)



At high level both are String object, but main difference comes from the point that new() operator always creates a new String object. Also when you create String using literal they are interned. This will be much more clear when you compare two String objects created using String literal and new operator, as shown in below example :

```
String a = "Java";
String b = "Java";
System.out.println(a == b);  // True
```

Here two different objects are created and they have different references:

```
String c = new String("Java");
String d = new String("Java");
System.out.println(c == d);  // False
```

Similarly when you compare a String literal with an String object created using new() operator using == operator, it will return false, as shown below :

```
String e = "JDK";
String f =  new String("JDK");
System.out.println(e == f);  // False
```

In general you should use the string literal notation when possible. It is easier to read and it gives the

Recomm

- 10 M... All L...
- 10 P... Sho...
- 10 B... Rea...
- 10 Open Source Lit... Framework for Java Developers
- 10 Programming language to Learn in 2018
- The Best Book to Learn Java in 30 days
- 10 Java Web Service Interview Questions
- Top 10 Android Interview Questions for Java Programmers
- 10 Books Every Programmer Should Read
- 5 Great Books to Learn Java 8

Books and Resources

- Best Book to Learn Java in 2017
- 5 Books to Learn Spring MVC and Core in 2017
- 2 books to learn Hibernate in 2017
- 12 Advanced Java Programming Books for Experienced Programmers
- 5 Free JavaScript Books to download
- 5 Books to Improve Your Coding Skill
- Books Every Programmer Should Read
- Top 10 Computer Algorithm Books

- J2EE (18)
- JDBC (18)
- Linux (18)
- java (18)
- JavaScript (16)
- database (16)
- Eclipse (15)
- Java programming Tutorial (15)
- JSP (14)
- Servlet (14)
- thread interview questions (13)
- OCAJP (11)
- binary tree (10)
- Unix (9)
- design pattern (8)
- Java Interview Question (7)
- OCPJP (7)
- java concurrency tutorial (7)
- JSON (6)
- homework (6)
- jQuery (6)
- Hibernate (4)
- Web Service (4)
- Hibernate interview Question (3)
- Java 5 tutorial (3)
- Struts (3)
- debugging (3)
- thread (3)
- Java Enum (2)
- garbage collection (2)
- xml (2)

compiler a chance to optimize your code. By the way any answer to this question is incomplete until you explain what is *String interning*, so let's see that in next section.

String interning using inter() method

Java by default doesn't put all String object into String pool, instead they gives you flexibility to explicitly store any arbitrary object in String pool. You can put any object to String pool by calling `intern()` method of `java.lang.String` class. Though, when you create using *String literal notation* of Java, it automatically call `intern()` method to put that object into String pool, provided it was not present in the pool already. This is another *difference between string literal and new string*, because in case of new, interning doesn't happen automatically, until you call `intern()` method on that object. Also don't forget to use StringBuffer and StringBuilder for string concatenation, they will reduce number

That's all about this question, **what is difference between String literal and String object in Java**. Always remember that literal Strings are returned from string pool and Java put them in pool if not stored already. This difference is most obvious, when you compare two String objects using equality operator (==). That's why it's suggested as always compare two String object using `equals()` method and never compare them using == operator, because you never know which one is coming from pool and which one is created using `new()` operator. If you know the *difference between string object and string literal*, you can also solve questions from Java written test, which also test this concept. It's something, every Java programmer should know. of temporary String object in heap space.

Further Learning

Data Structures and Algorithms: Deep Dive Using Java
Java Fundamentals: The Java Language
Complete Java Masterclass

If you like this article and eager to learn more about String type in Java, check out these amazing articles from Java67 blog :

- How to create array from ArrayList of String in Java
- How to count number of Vowels and Consonants in Java String
- How to replace characters on String in Java
- How to use substring method in Java
- How to use Regular Expression to Search in String
- Difference between StringBuilder and StringBuffer in Java
- How to find if String is Empty in Java
- Difference between String and StringBuffer in Java
- How to convert array to String in Java

- 10 Free Java Programming Books
- 5 B
- Boo Rea
- Top

- How to convert Enumeration type to String in Java
- Best way to compare two Strings in Java

Posted by javin paul

Labels: core java, core java interview question answer, Java Interview Question, programming, String

44 comments:

Unknown August 5, 2014 at 3:29 PM

Thank you for this question I was very confuse about this string literal and string object but your tutorial give me clear idea about that thank you once again.....

Reply

Anonymous August 12, 2014 at 8:16 AM

Thanks for your article. Could you please tell me. How many objects are created in memory and at which place? when we create objects using new keyword?

Reply

Replies

Blog Archive

- ► 2020 (208)
- ► 2019 (148)
- ► 2018 (32)
- ► 2017 (57)
- ► 2016 (109)
- ► 2015 (87)
- ▼ 2014 (46)
  - ► December (6)
  - ► November (4)
  - ► October (2)
  - ► September (4)
  - ▼ August (3)

HelloWorld - My First jQuery Program

What is the difference between a Class and an Obje...

Difference between String literal and New String o...

► July (1)
► June (3)
► May (3)
► April (4)
► March (4)
► February (5)
► January (7)

► 2013 (41)
► 2012 (59)

Anonymous October 2, 2014 at 3:02 AM

Good question. For example if you execute following two lines :

String language = "Java";
String anotherLanguage = new String("Java")

how many objects are created in where i.e. heap, stack and pool?

In first statement one String is created using literal syntax so that will go to the String pool, which is also part of heap from Java 1.7 but was part of permgen area in Java 1.6. In second statement, "Java" will not create another object instead it will reuse the object created in previous step, had it been any other word e.g. "java" it would have created another object. By the way new String() will create another object in Java heap space. No object in stack but reference variables language and anotherLanugage will be created and stored in stack.

Anonymous January 11, 2015 at 10:24 AM

I'm a little bit confused about the second statement in the expressions above executed:

String language = "Java";
String anotherLanguage = new String("Java");

After the execution of first statement, the String Literal "Java" is created in String Pool.But then in second statement, I think that the "Java" will be created in Heap area, although I may be wrong in this but to the point I know I think that's the way it is. Although I'm not confirm about where the "Java"in the second statement parenthesis will be created/or not.

Can anyone please clarify/

SaiVemuri February 19, 2015 at 6:51 PM

This is how 2nd statement works
1) Scan SCP . if it has already same string then return old one, else create new object in SCP.
2) Create one String object in Heap(due to new)

Totally 2 objects in general.

Tanveer September 26, 2015 at 3:59 PM

Don't think so, Only one object will be created .If we call intern then the object should be created in String pool.

Anonymous April 9, 2019 at 5:03 AM

new String("..") always creates an object in heap we all know
but the question is ,is it also check in String pool.???

Reply

Anonymous August 18, 2014 at 9:52 PM

I think main difference between String object and String literal still remains the fact that former is automatically added into String pool while later is added using intern() method.

Reply

Anonymous January 11, 2015 at 12:36 PM

I have a question, can a String Constant Pool be Garbage Collected. If so when & how?

Reply

Replies

Anonymous July 24, 2015 at 2:03 AM

Yes, both interned String and literals are subject to garbage collection. when a String literl doesn't have any active reference it becomes eligible for garbage collection and this is true for all Java versions. In Java 7, String pool is moved to heap space, but even when it was in PermGen space until Java 6, String constants were garbage collected.

Reply

Unknown May 7, 2015 at 12:39 PM

good clarification

Reply

Anonymous July 19, 2015 at 6:17 PM

very good explanation of the String objects & their differences. Thanks a lot.

Reply

Anonymous July 30, 2015 at 2:03 AM

I am still confused about how many objects created when u create string using new operator.
as per me if I write String s = new String("ABC") then two objects are created
one in heap and one in String pool.
Please provide your feedback.

Reply

Replies

Anonymous November 1, 2015 at 9:59 PM

@Anonymous, that is correct, two objects, "ABC" on String pool and new String() on heap.

Anonymous July 3, 2017 at 3:15 AM

Nope, until you explicitly call intern() method, it will create one object in Heap space only.

Reply

Amila Nandana August 2, 2015 at 11:40 PM

When executing String s = new String("ABC") , only one object is created in the heap. This doesn't not deal with String pool.

Reply

Replies

Anonymous November 1, 2015 at 9:58 PM

Actually two objects are created, first the "ABC" object which is created in String pool and second the new String() object which is created in heap. String pool is also now part of Java heap from JDK 7 release update 40 I guess.

Sridhar Kandula April 11, 2016 at 8:23 PM

SO in this case how intern() method is used? and what is the advantage of having intern() method if string is already in pool even using with new operator

Unknown June 5, 2018 at 5:45 AM

yes Sridhar , me too having same point can anyone please clear me.

Reply

Unknown December 11, 2015 at 10:33 PM

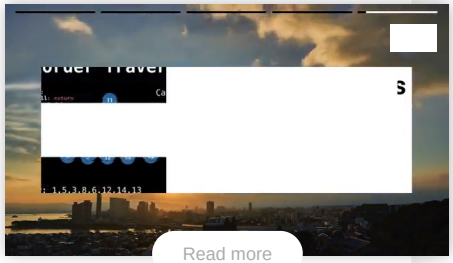
String str = "ABC" + "DEF"; How many object will be created in this.

Reply

Replies

javin paul December 13, 2015 at 4:05 AM

Hi @Davinder, four objects will be created



Read more

1) "ABC"
2) "DEF"
3) The StringBuilder object which will used for concatenation
4) The String object which will be created after concatenation e.g. "ABCDEF"

Java internally uses StringBuilder or StringBuffer when you use + operator to join string together.

Reply

Unknown January 5, 2016 at 8:43 AM

Hi.
What is the use of creating two object i.e using new keyword .
For Ex. String S=new String("abc");
In this example there will be 2 objects. 1 in string Pool and 1 in NonPool Memory.
But what is the relevance of String pool object when using new Keyword. Is it available for next time when we create another object using new keyword with same value and if not then why it is there?

Reply

Replies

javin paul July 26, 2016 at 5:41 PM

hello @Himanshu, it will be returned when you create another object using same character but by using String literal e.g. "abc" will return the same object.

Reply

Unknown January 16, 2016 at 12:12 PM

The same question...
Are 2 objects created ?

String s=new String("abc");

Or only 1 object in non-pool heap memory is created?

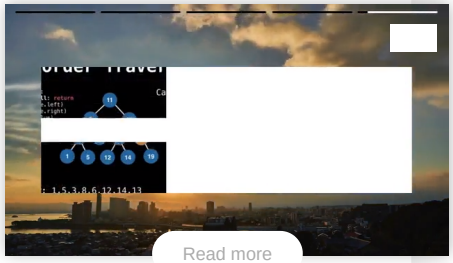
Reply

Replies

javin paul January 16, 2016 at 10:16 PM

Hello Aditya, Yes, two objects are created if "abc" is not in the String pool already, if it is then just one. First object is "abc" and second is the new String().

Yes, only "abc" will be created in pool, s will not go to pool until you call s.intern()



Read more

true taller February 3, 2016 at 9:34 AM

there will be two objects created s will be in heap and by "abc" literal object will be go in the string constant pool.
if there is already a "abc" present than it will not create new and give the reference of previous "abc".
Take A example:

```
class intern
{
public static void main(String... s)
{
String s6 = "arpan";
String s5 = "arpan";
if(s5==s6)
System.out.println("true");
}}
```

Reply

true taller February 3, 2016 at 9:24 AM

is it possible to put a string only in heap memory not in string constant pool.

Reply

Replies

javin paul July 26, 2016 at 5:40 PM

@true taller yes, if you create an object using new String() without using String literal e.g. via byte array then that object only remain in heap space, not created in String pool.

Reply

Anonymous May 30, 2016 at 4:45 AM

Can you tell me how many objects created here

```
String ab="ab"+"ba";
String ba="ba"+"ab";
```

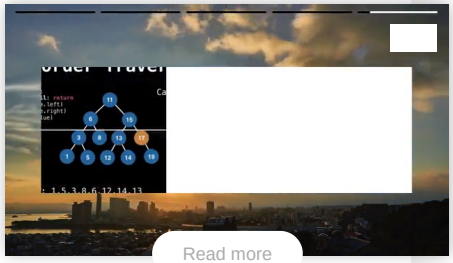
Reply

Replies

javin paul July 26, 2016 at 5:39 PM

@Anonymous, 4 objects, two String literal and two created by String concatenation.

Reply



Read more

Anonymous July 6, 2016 at 2:19 AM

String s="abc";
if(s=="abc")
{
System.out.print("true");
}

Does the result will be true in this casse?

Reply

Replies

javin paul July 26, 2016 at 5:38 PM

@Anonymous, yes result will be true because both are pointing to same String literal.

Reply

Anonymous July 26, 2016 at 6:21 AM

when you say if we call itern() on string object it moves to string constant pool.

System.out.println("Testing intern");
String test = "intern";
String test1 = new String("intern");
System.out.println(test == test1);// it will return false no doubt about it
test1.intern();// moved string object into pool having same literal
System.out.println(test == test1); // it should have written true but its not

Reply

Replies

javin paul July 26, 2016 at 5:37 PM

Hello @Anonymous, the intern() method has no effect on existing object. Now, if you create another object with literal e.g. "intern" then same object is returned instead of creating a new one. See my post when to use intern() method of String in Java to learn more about intern

Reply

Anonymous November 16, 2016 at 3:47 AM

String a = "one";
String b = "two";
String c = a+b;

```
String d = "one"+"two";
String e = "onetwo";
if(c==d){
System.out.println(true);//1st
}else if(c==e){
System.out.println(true);//2nd
}else if(d==e){
System.out.println(true);//3rd
}else{
System.out.println(true);//4th
}
```

Explain Mr. Javin Paul,
1.Why 3rd will be true and why not 1/2?
2. Afetr concatenation will new object will be created in heap/pool?

Reply

Anonymous August 31, 2017 at 11:08 PM

Very good explanation, thank you!

But this is an exception, a trick and any exception makes our life harder and it pales the beauty of Java.
Why Java designers did not automatically decide if a string in PermGen to use it no matter it is literal or new object?
So if after a
String s1="Java";
//coming with
String s2 = new String("Java");
to return also s1 as it is already there?

This in idea that Java is a tool in our hands not vice-verse.
And we prefer simple but effective tools.

Reply

Replies

javin paul September 8, 2017 at 11:58 PM

agreed.

Reply

niraj September 14, 2017 at 12:04 AM

why they need it
why they create two different type String object..

in case of String str=new String("Hello");
two String object is created...why it need...

Reply

Unknown August 31, 2019 at 8:48 AM

String a="a";
String b="b";
String c=a+b+"c";
String d=new String ();

How many string objects are created in this code?

Reply

Unknown December 11, 2019 at 2:58 AM

String j = "Java";
String f = new String("Java");

System.out.println(j.equals(f));
System.out.println(j==f);

Reply

Unknown December 11, 2019 at 3:19 AM

String j = "Java";
String f = new String("Java");

System.out.println(j.equals(f)); // true
System.out.println(j==f); // false Why? If f doesn't create a new object and uses reference of j. Then why false?

Reply

Replies

javin paul December 17, 2019 at 5:07 AM

Please read the article, they are different object and == only return true for same object, not same content of different object, which equals does.

Reply

OMER May 1, 2020 at 3:42 PM

I think there are false sentences in the article. Or I may have misunderstood me. When I search on the internet, when I say new String ("abc"), 2 values are created in memory, including a heap and a String pool. I

agree that there is no value in the pool.
Then the following sentences in the article attract my attention. And I can not understand. It says that it will not be put in the pool without calling the intern () method. Do I get it wrong?

"Java by default doesn't put all String object into String pool, instead they gives you flexibility to explicitly store any arbitrary object in String pool. You can put any object to String pool by calling intern() method of java.lang.String class."

According to these sentences, when I say new String ("abc"), if I don't call the intern () method, it will not be put in the pool. However, it does not write like this on the Internet.Can you explain it to me?

Reply

javin paul May 8, 2020 at 6:12 AM

Hello OMER,
When you write String str = new String ("abc") there are two objects created here, string literal "abc" which is created in String pool and an object refereed by str, both contains "abc" but one is in heap and other is in string pool. if you do str.intern() then it will refer to same object as "abc". You can try it yourself by comparing using == operator to find out more.

Reply

Enter your comment...

Comment as: shreyanshsinha ▾ Sign out

Publish Preview Notify me

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)