2.4.3. The String Constant Pool

In the previous post, we have learned about string immutability. Another interesting thing about strings in java is **The String Constant Pool.** So what is it exactly? It is a special place where the collection of references to string objects are placed. What makes this so special? we will try to understand it now.

In our tutorial about String immutability, we have learned that string literals cannot be modified and multiple reference variable can refer to the same string literal. Let us first write a program to understand object comparison and references

```
public class StringConstantPool {
public static void main(String[] args) {
String s = "prasad";
String s2 = "prasad";

System.out.println(s.equals(s2));
System.out.println(s == s2);
}
}
```

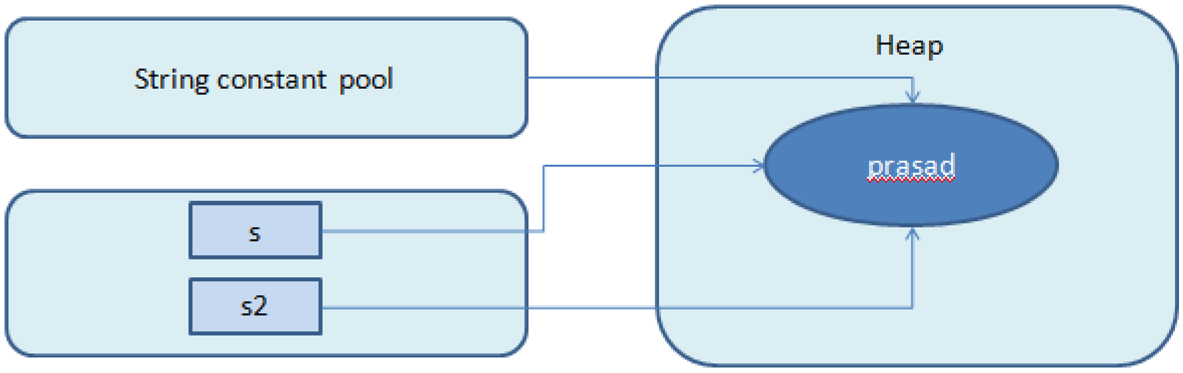
The output of the above code is

```
true
true
```

Now lets know what happens here step by step

- The class is loaded when JVM is invoked.
- JVM will look for all the string literals in the program
- First, it finds the variable `s` which refers to the literal "`prasad`" and it will be created in the memory
- A reference for the literal "`prasad`" will be placed in the string constant pool memory.
- Then it finds another variable `s2` which is referring to the same string literal "`prasad`".
- Now that JVM has already found a string literal "`prasad`", both the variables `s` and `s2` wil refer to the same object i.e. "`prasad`".

The diagram below demonstrates this



String literals referred

Now we have looked into the case when string literals are created without using the `new` operator. What happens if `String s2 = new ("prasad");`

As we are invoking the new keyword, **The object "`prasad`" will be created when the new String("prasad") is invoked. This is unlike the string literal "`prasad`" which is created when class is loaded.**

Now the values of objects referenced by variable `s` and variable `s2` are the same i.e. "prasad" but those are not the same objects. They refer to different objects. We will verify this with a program

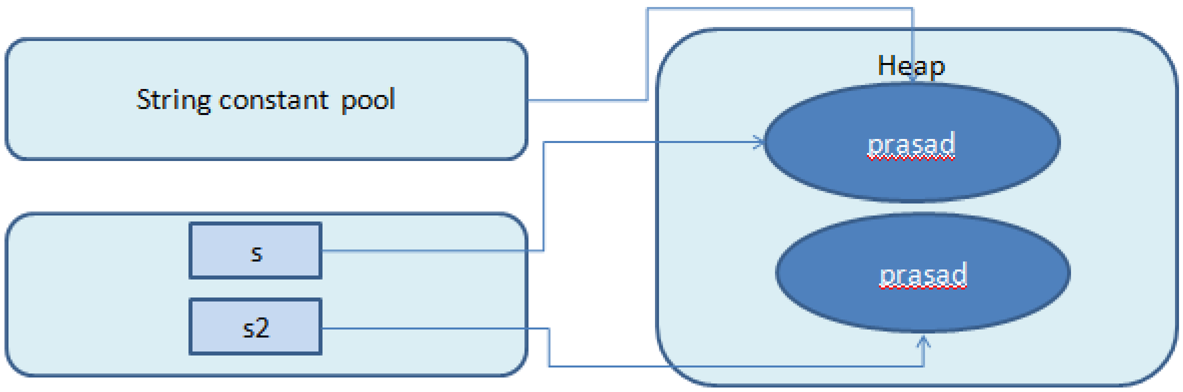
```java
public class StringConstantPool {
public static void main(String[] args) {
String s = "prasad";
String s2 = new String("prasad");
System.out.println(s.equals(s2));
System.out.println(s == s2);

}
}
```

This outputs,

```
true
false
```

- The contents of both objects are the same so `equals` method returns `true`
- The objects referred by both variables are different so == operator returns `false`

This is elaborated in this diagram



String with new keyword

Points to be remembered

- String literals with same values will always refer to the same String object
- String objects created using new operator will be different from literals