🈸 String a = "123"; String b = "123"; a==b 吗？为什么？？

学过编程语言的应该没人不会做吧，答案是 true 。

很多人都觉得这个问题是不是太简单了，其实不然。这里面包含的内存，String存储方式等知识点。

现在我以我的理解来分析、解释一下这个问题的底层原因。如有不足，还请大家留言指出。

String a = "123";
String b = "123";
System.out.println(a == b);

答案：true
学习Java的人都知道，JVM的重要性。在JVM中有一块区域叫做常量池，具体虚拟机的结构可以看我的另外一篇博客（https://blog.csdn.net/qq_40722827/article/details/103236038）。常量池中的数据是那些在编译期间被确定的，并被保存在已编译的.class文件中的一些数据。

我们定义的String a = "123"; String b = "123"; 这些语句，我们拆分开来看：

编译器：指类中成员变量

1. 123，等号右边的指的是编译期间可以被确定的内容，都维护在常量池中。
2. str ，等号左边的指的是一个引用，引用的内容是等号右边数据在常量池中的地址
3. String 这是引用类型

栈有一个特点，就是数据共享。回到最初的问题，String a = "123"，编译的时候，在常量池中创建了一个常量"123"，然后String b= "123"，先去常量池中找有没有这个"123"，发现常量池中有这个“123”，然后b也指向常量池中的"123"，所以a==b返回的是true，因为a和b指向的都是常量池中的"123"这个字符串的地址。其实其他基本数据类型也都是一样的：先看常量池中有没有要创建的数据，有就返回数据的地址，没有就创建一个。

看完上面这个，我们再来看一下下面的这个：

```
String a = new String("234");
String b = new String("234");
System.out.println(a == b);
```

答案：false
那么，这儿的答案为什么是 false 呢？？？

原因：Java虚拟机的解释器每遇到一个new关键字，都会在堆内存中开辟一块内存来存放一个String对象，所以a,b指向的堆内存虽然存储的都是“234”，但是由于两块不同的堆内存，因此 a==b 返回的仍然是false。
————————————————