先从一个面试题谈起：

```
public class StringInternTest {

    public static void main(String[] args) {
        String s1 = new String("ab");
        s1.intern();
        String s2 = "ab";
        System.out.println(s1 == s2);
        //jdk1.6:false jdk1.7/1.8:false

        String s3 = new String("1") + new String("1");
        s3.intern();
        String s4 = "11";
        System.out.println(s3 == s4);
        //jdk1.6:false jdk1.7/1.8:true
    }
}
```

1、new String(“ab”)创建了几个对象？

两个：一个对象是 new 关键字在堆中创建的，一个对象是字符串常量池中的对象（不存在则创建，存在则复用）

2、new String(“a”) + new String(“b) 创建了几个对象？

对象一：new StringBuilder()
对象二：new String(“a”)

对象三：字符串常量池中的”a”
对象四：new String(“b);
对象五：字符串常量池中的”b”

继续深入：StringBuilder.toString()
对象六：new String(“ab”);

补充：toString()的调用，没有在字符串常量池中生成”ab”

3、String#intern()

Jdk1.6 中，将这个字符串对象尝试放入字符串常量池
1、如果字符串常量池中有，则并不会放入，返回已有的字符串常量池中的对象地址
2、如果没有，则会把此对象复制一份（相当于创建了一个新对象），放入字符串常量池，并返回串池中的对象地址

Jdk1.7/jdk1.8 中，将这个字符串对象尝试放入字符串常量池
1、如果字符串常量池中有，则并不会放入，返回已有的字符串常量池中的对象地址
2、如果没有，则会把对象的引用地址复制一份，放入字符串常量池，并返回串池中的引用地址

4、回到面试题

```
public class StringInternTest {

    public static void main(String[] args) {
        String s1 = new String("ab");//创建了两个对象，一个在堆中，一个在字符串常量池中，s1 指向堆中的”ab”
        s1.intern();//调用 intern 方法之前，字符串常量池中已存在”ab”
        String s2 = "ab";//s2 指向 new String(“ab”)操作放入常量池中的”ab”
        System.out.println(s1 == s2);
        //jdk1.6:false jdk1.7/1.8:false


        String s3 = new String("1") + new String("1");
        //执行完上一行代码后，字符串常量池中不存在”11”，s3 指向堆中创建的”ab”
        s3.intern();//在字符串常量池中生成”11”
//jdk1.6 中，字符串常量池存在永久代中，在串池中创建一个新的对象”11”
//jdk1.7/1.8 中，字符串常量池存在堆中，此时串池中并没有创建一个新的对象”11”，而是创建一个指向堆中 new String(“11”)的地址
        String s4 = "11";
//s4 记录的地址，使用的是上一行代码执行后，在常量池中生成的”11”的地址，即 s3 指向的堆中床架的”ab”
        System.out.println(s3 == s4);
        //jdk1.6:false jdk1.7/1.8:true
    }
}
```