●常见对象(String 类的概述)(掌握)

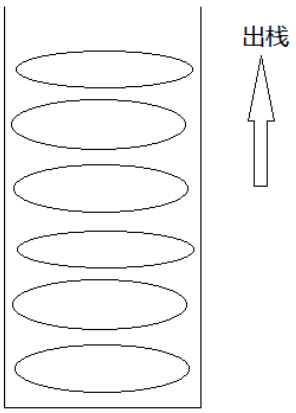
- A:String 类的概述
  -
  - 可以看到这样的两句话。
    - a:字符串字面值"changsha"也可以看成是一个字符串对象。
    - b:字符串是常量，一旦被赋值，就不能被改变。

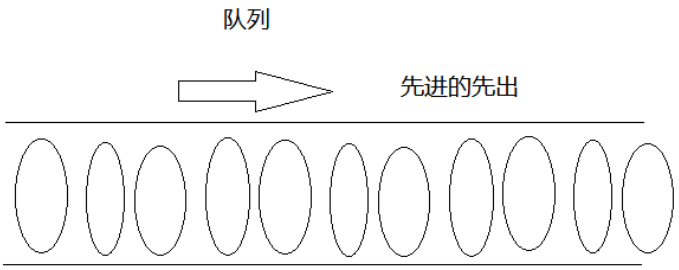
```
package com.sxt.stringdemo;

public class Demo {
    public static void main(String[] args) {
        String s1 = "changsha";// 等同于String s1 = new String("changsha")
        System.out.println(s1.hashCode());
        s1 = "广州";  //等同于String s1 = new String("广州")
        System.out.println(s1.hashCode());
        /*
        上面输出的哈希码不一样,说明以上两个s1 已经不是同一个对象了
         */
    }
}
```

方法区:
其实也在堆内存中,用来存储字节码,静态数据,静态区,常量池,jvm 在运行中即时编译产生的数据,是为了与堆内存区别开,它也叫 no_heap
栈内存:
是方法运行时存储的空间,这是一个动态空间

栈内存-有点像枪的弹夹,先进的后出





堆内存:heap
用来储存 new 出来的对象

●12.04_常见对象(String 类的构造方法)(掌握)

- A:常见构造方法

- public String():空构造
- public String(byte[] bytes):把字节数组转成字符串
- public String(byte[] bytes,int index,int length):把字节数组的一部分转成字符串
- public String(char[] value):把字符数组转成字符串
- public String(char[] value,int index,int count):把字符数组的一部分转成字符串
- public String(String original):把字符串常量值转成字符串

- B:案例演示
  - 演示 String 类的常见构造方法

```
package com.sxt.stringdemo;

public class Demo {
    public static void main(String[] args) {
       /*
        * public String():空构造
        * public String(byte[] bytes):把字节数组转成字符串
        * public String(byte[] bytes,int index,int length):把字节数组的一部分转成字符串
        * public String(char[] value):把字符数组转成字符串
        * public String(char[] value,int index,int count):把字符数组的一部分转成字符串
        * public String(String original):把字符串常量值转成字符串
        */
       String s1 = new String();//等同于:String s1 = "";
        System.out.println(s1.toString());//没有打印出这个地址***@234ba343,说明String 类重写了toString

        byte[] b = {97,98,99,100};
        String s2 = new String(b);
        System.out.println(s2);
        String s3 = new String(b,1,3);//从下标索引1开始转化3个元素
      System.out.println("s3="+s3);
        char[] c = {'d','e','m','o'};
        String s4 = new String(c);
        System.out.println("s4="+s4);
        String s5 = new String(c,2,2);//从下标索引2开始转化2个元素
      System.out.println(s5);
        String s6 = new String("湖南省"); //开发中推荐直接String s6 = "湖南省";
        System.out.println(s6);
    }
}
```

```
String s7 = "湖南省";
System.out.println(s6==s7);
boolean bo = s6.equals(s7);//比较两个字符串中,所有的字符序列
System.out.println(bo);
```

●12.05_常见对象(String类的常见面试题)(掌握)

- 1.判断定义为 String 类型的 s1 和 s2 是否相等
  - String s1 = "abc";

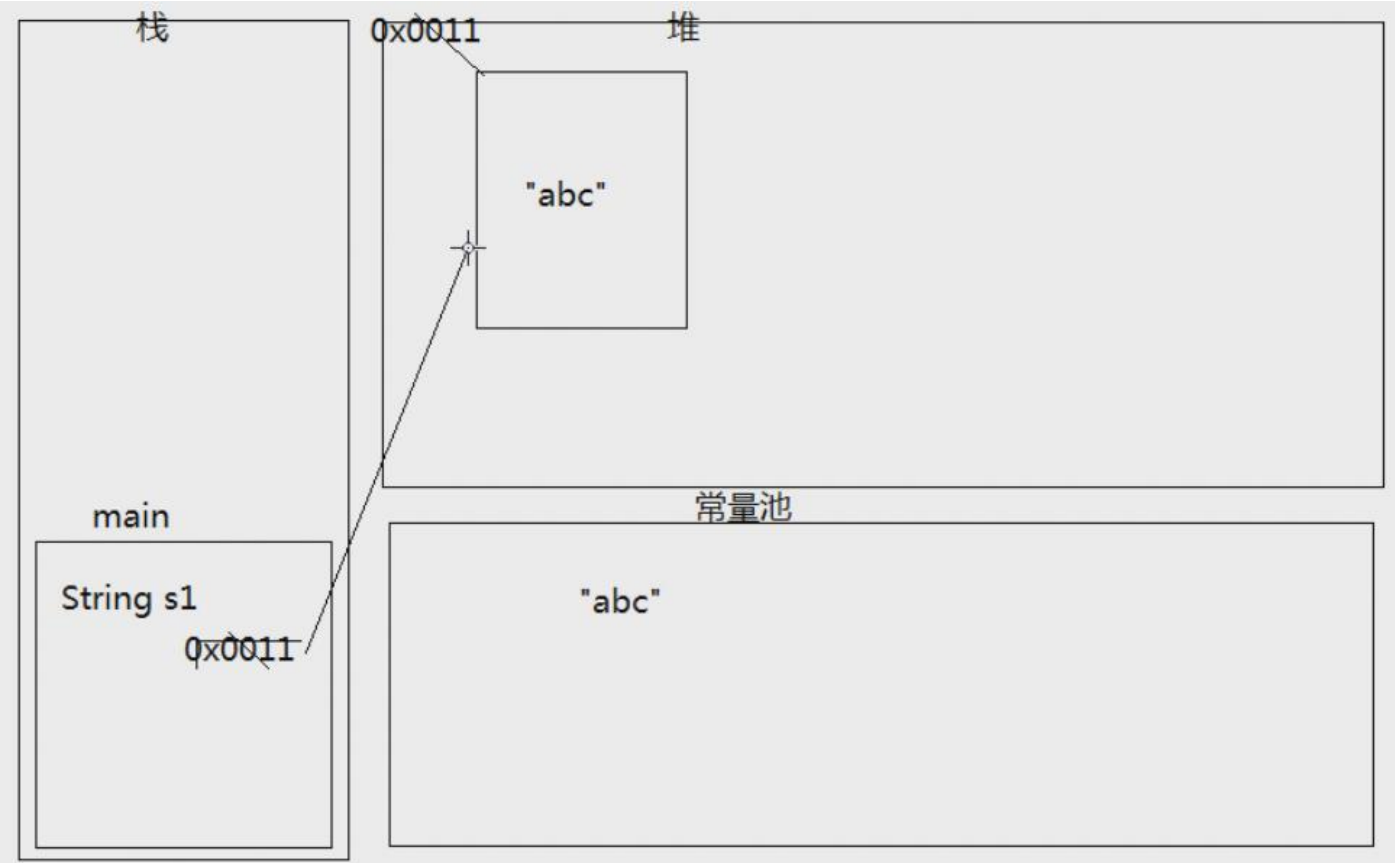
- String s2 = "abc";
- System.out.println(s1 == s2);
- System.out.println(s1.equals(s2));

```
package com.sxt.stringdemo;

public class Demo {
    public static void main(String[] args) {
        String s1 = "abc";
        String s2 = "abc"; //"abc 都存在常量池中,jvm 底层为了节约内存,如果常量池中已经存在同一个常量,就直接拿过来用"
        System.out.println(s1 == s2); //true
       System.out.println(s1.equals(s2));     //true

    }
}
```

- 2.下面这句话在内存中创建了几个对象?(面试题)
  - String s1 = new String("abc"); //2 个



- 3.判断定义为 String 类型的 s1 和 s2 是否相等

```
String s1 = "岳麓区";  //在方法区
String s2 = new String("岳麓区"); //在堆内存
System.out.println(s1 == s2);  //false
System.out.println(s1.equals(s2));// true
```

- 4.判断定义为 String 类型的 s1 和 s2 是否相等

```
String s1 = "a" + "b" + "c";
String s2 = "abc";
System.out.println(s1 == s2);//true
System.out.println(s1.equals(s2));//true
```

- 5.判断定义为 String 类型的 s1 和 s2 是否相等

```
String s1 = "ab";
String s2 = "abc";
String s3 = s1 + "c";
System.out.println(s3 == s2);//false
System.out.println(s3.equals(s2));//true
```

●12.06_常见对象(String 类的判断功能)(掌握)

- A:String 类的判断功能
  - boolean equals(Object obj):比较字符串的内容是否相同,区分大小写
  - boolean equalsIgnoreCase(String str):比较字符串的内容是否相同,忽略大小写
  - boolean contains(String str):判断大字符串中是否包含小字符串
  - boolean startsWith(String str):判断字符串是否以某个指定的字符串开头
  - boolean endsWith(String str):判断字符串是否以某个指定的字符串结尾
  - boolean isEmpty():判断字符串是否为空。

```
package com.sxt.stringdemo;

public class Demo {
    public static void main(String[] args) {

        String s1 = "abdfD";
        String s2 = "abdfd";
        System.out.println(s1.equals(s2));
        System.out.println(s1.equalsIgnoreCase(s2));
        System.out.println(s1.contains("a"));//判断 s1 中是否包含了 a
        System.out.println(s1.startsWith("b"));//判断是否以 b 开头
      System.out.println(s1.endsWith("D"));//判断是否以 D 结尾
      System.out.println(s1.isEmpty());
        System.out.println("".isEmpty());//字符串常量可以直接当做对象使用,所以可以直接调用 String 类中的方法
      System.out.println("张三".startsWith("张"));
    }
}
```

●12.07_常见对象(模拟用户登录)(掌握)

- A:案例演示

- 需求：模拟登录,给三次机会,并提示还有几次。
- 用户名和密码都是 admin

```
package com.sxt.stringdemo;
import java.util.Scanner;

public class Demo {
    public static void main(String[] args) {
        check();

    }
    public static void check(){
        Scanner sc = new Scanner(System.in);

        for (int i = 3; i >=1 ; i--) {
            System.out.println("请输入名字:");
            String name = sc.next();
            System.out.println("请输入密码:");
            String password = sc.next();
            if("admin".equals(name) && "admin".equals(password)){
                System.out.println("欢迎您,"+name+"登录成功!");
            }else{
                System.out.println("你还有"+ (i-1) +"次机会");
                System.out.println("用户名或密码有误,请重试");
                if(i ==0){
                    System.exit(0);
                }
            }
        }
    }
}
```

●12.08_常见对象(String 类的获取功能)(掌握)

- A:String 类的获取功能
  - int length():获取字符串的长度。
  - char charAt(int index):获取指定索引位置的字符
  - int indexOf(int ch):返回指定字符在此字符串中第一次出现处的索引。
  - int indexOf(String str):返回指定字符串在此字符串中第一次出现处的索引。
  - int indexOf(int ch,int fromIndex):返回指定字符在此字符串中从指定位置后第一次出现处的索引。
  - int indexOf(String str,int fromIndex):返回指定字符串在此字符串中从指定位置后第一次出现处的索引。
  - lastIndexOf
  - String substring(int start):从指定位置开始截取字符串,默认到末尾。
  - String substring(int start,int end):从指定位置开始到指定位置结束截取字符串。

```
package com.sxt.stringdemo;

import java.util.Scanner;

public class Demo {
    public static void main(String[] args) {
```

```
        String s1 = "长沙人民欢迎你民欢迎你民";
        int len = s1.length();
        System.out.println(len);
        char c = s1.charAt(1);
         System.out.println(c);
         int index = s1.indexOf("民");
         System.out.println(index);
         System.out.println("abca".indexOf(98));
         System.out.println(s1.indexOf("民",4));//这个可以处理查找同一个字符第二次出现的索引
       System.out.println(s1.lastIndexOf("民"));

         String s2 = s1.substring(3);//从指定位置开始截取字符串,默认到末尾
       System.out.println(s2);
         String s3 = s1.substring(2,5);//截取字符串从索引包含 2 开始到 4 结束,不包含 5
         System.out.println(s3);

    }
}
```