今日学习目标

* 能够明确API的使用步骤
* 能够使用Scanner类获取键盘输入的字符串 next() nextLine();
* 能够使用String类的构造方法创建字符串对象
* 能够明确String类的构造方法创建对象,和直接赋值创建字符串对象的区别
* 能够使用文档查询String类的常用方法 equals, split
* 能够使用类名作为形式参数和返回值
* 能够使用StringBuilder类的添加功能(append)和反转功能(reverse)

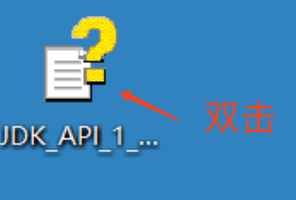
# API文档

## 概述

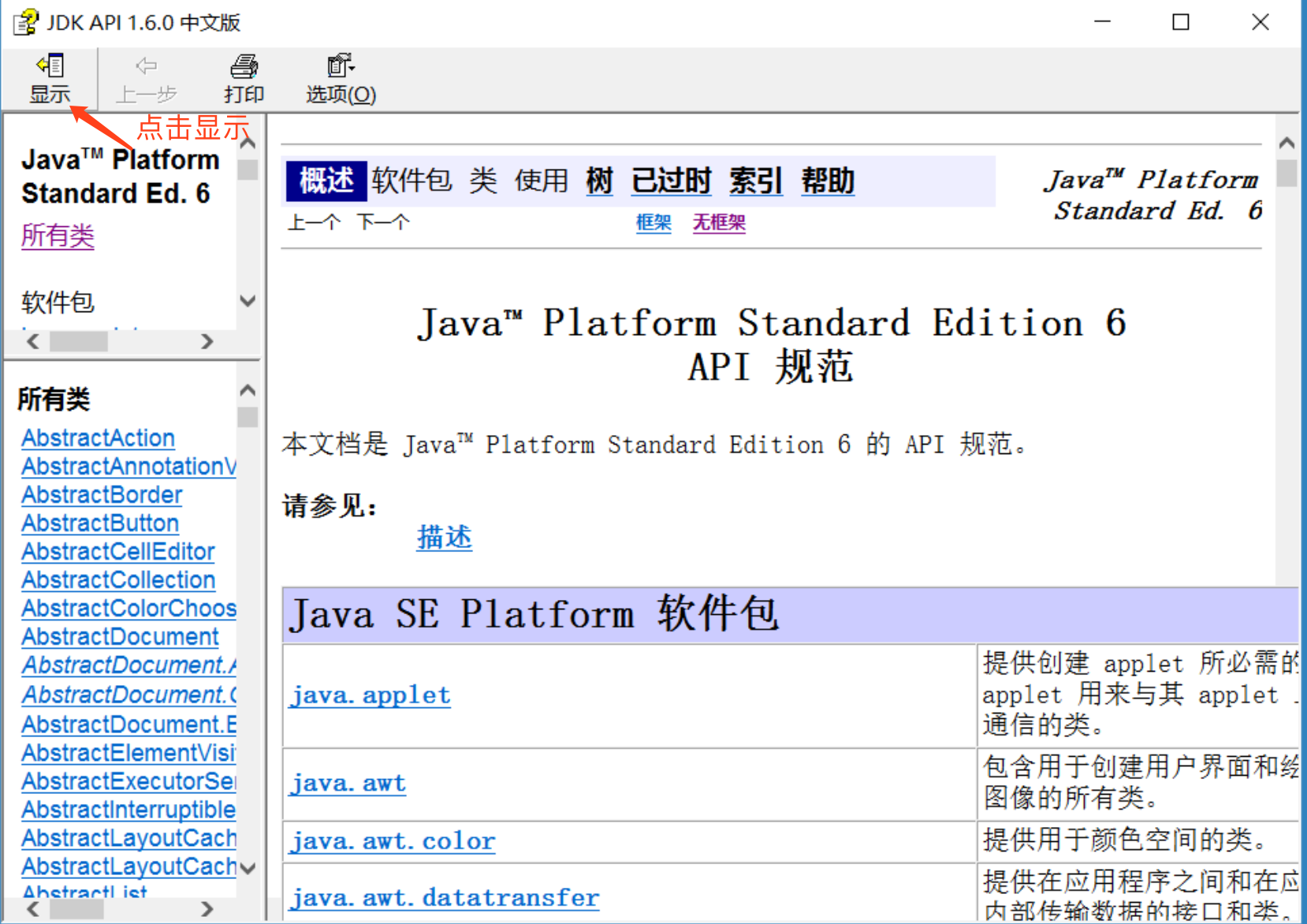
* API(Application Programming Interface)
  + 应用程序编程接口
  + 编写一个机器人程序去控制机器人踢足球，程序就需要向机器人发出向前跑、向后跑、射门、抢球等各种命令，没有编过程序的人很难想象这样的程序如何编写。但是对于有经验的开发人员来说，知道机器人厂商一定会提供一些用于控制机器人的Java类，这些类中定义好了操作机器人各种动作的方法。其实，这些Java类就是机器人厂商提供给应用程序编程的接口，大家把这些类称为API。
  + Java API就是JDK中提供给我们使用的类，这些类将底层的代码实现封装了起来，我们不需要关心这些类是如何实现的，只需要学习这些类如何使用即可。
  + 在JDK安装目录下有个src.zip文件，这个文件解压缩后里面的内容是所有Java类的源文件。可以在其中查看相对应的类的源码。
  + 我们在每次查看类中的方法时，都打开源代码进行查看，这种方式过于麻烦。其实，我们可以通过查帮助文档的方式，来了解Java提供的API如何使用。

## -使用步骤

* 打开帮助文档



* 点击显示



* 找到索引，看到输入框



* 你要找谁 ?
  + 在输入框里输入，然后回车,再回车.



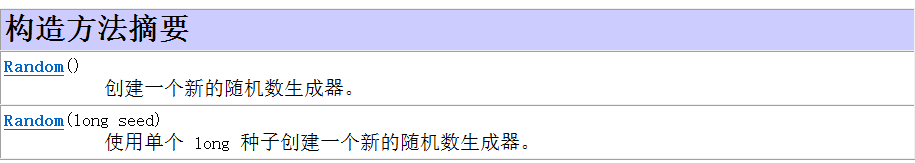
* 看包
  + java.lang下的类不需要导包，其他需要



* 看类的解释和说明



* 学习构造方法
  + 使用构造方法我们可以创建该类的对象



* 使用成员方法
  + 看方法名,不要写错了.
  + 看参数列表,参数是什么数据类型,你就给什么类型的数据,有几个就你就传入几个.
  + 看返回值类型,返回类类型是什么就定义什么类型的变量接收, 返回值类型void就单独调用.



调用上面的框起来的方法: int num = r.nextInt(10); 说明: r是Random的对象名

## API的使用（Scanner类）

* 获取键盘录入的字符串数据

**public class**  DemoScanner {

**public static void** main(String[] args) {

//创建键盘录入对象

Scanner sc = **new** Scanner(System.***in***);

//接收数据

System.***out***.println("请输入一个字符串数据：");

String s = sc.nextLine();

//输出结果

System.***out***.println("s:"+s);

}

}

# String类

## String概述

* 字符串是由多个字符组成的一串数据
* 字符串可以看成是字符数组

## 构造方法

* 构造方法
  + public String(String original)把字符串数据封装成字符串对象
  + public String(char[] value)把字符数组的数据封装成字符串对象
  + public String(char[] value,int offset,int count)把字符数组中的一部分数据封装成字符串对象
  + 直接赋值也可以是一个对象(定义一个字符串变量)

注意：字符串是一种比较特殊的引用数据类型，直接输出字符串对象输出的是该对象中的数据。

**public class**  Demo01String {

**public static void** main(String[] args) {

//方式1

//String(String original):把字符串数据封装成字符串对象

Strings1 = **new** String("hello");

System.***out***.println("s1:"+s1);

System.***out***.println("---------");

//方式2

//String(char[] value):把字符数组的数据封装成字符串对象

**char**[] chs = {'h','e','l','l','o'};

String s2 = **new** String(chs);

System.***out***.println("s2:"+s2);

System.***out***.println("---------");

//方式3

//String(char[] value, int index, int count):把字符数组中的一部分数据封装成字符串对象

//String s3 = new String(chs,0,chs.length);

String s3 = **new** String(chs,1,3);

System.***out***.println("s3:"+s3);

System.***out***.println("---------");

//方式4

String s4 = "hello";

System.***out***.println("s4:"+s4);

}

}

* String的特点（区别）
  + 通过构造方法创建字符串对象

String s = new String(“hello”);

* + 直接赋值创建字符串对象

String s = “hello”;

区别是什么?

**public class** Demo02String {

**public static void** main(String[] args) {

String s1 = **new** String("hello");

String s2 = "hello";

System.***out***.println("s1:"+s1);

System.***out***.println("s2:"+s2);

System.***out***.println("s1==s2:"+(s1==s2));

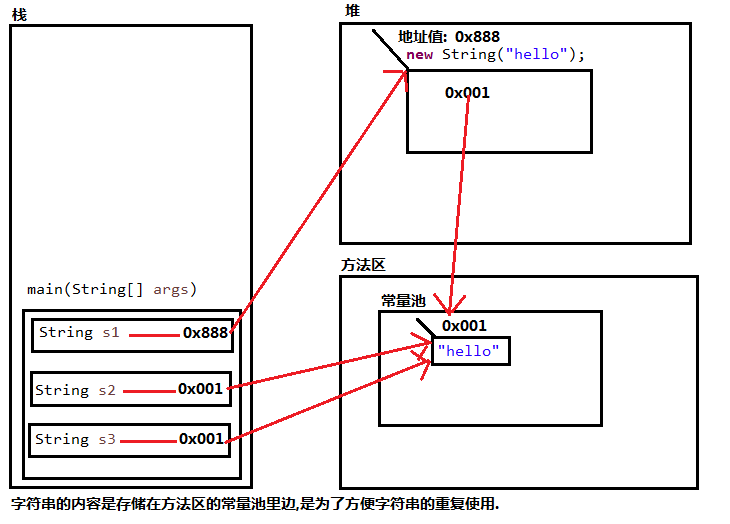
String s3 = "hello";

System.***out***.println("s1==s3:"+(s1==s3));

System.***out***.println("s2==s3:"+(s2==s3));

}

}



## 常用方法

* 判断功能方法
  + boolean equals(Object obj):比较字符串的内容是否相同
  + boolean equalsIgnoreCase(String str):比较字符串的内容是否相同,忽略大小写。
  + boolean startsWith(String str):判断字符串对象是否以指定的str开头。
  + boolean endsWith(String str):判断字符串对象是否以指定的str结尾。
  + boolean contains(String str)：判断字符串对象是否包含指定的str。

**public class** Demo01String {

**public static void** main(String[] args) {

//创建字符串对象

String s1 = "hello";

String s2 = "hello";

String s3 = "Hello";

//boolean equals(Object obj):比较字符串的内容是否相同

System.***out***.println(s1.equals(s2));

System.***out***.println(s1.equals(s3));

System.***out***.println("-----------");

//boolean equalsIgnoreCase(String str):比较字符串的内容是否相同,忽略大小写

System.***out***.println(s1.equalsIgnoreCase(s2));

System.***out***.println(s1.equalsIgnoreCase(s3));

System.***out***.println("-----------");

//boolean startsWith(String str):判断字符串对象是否以指定的str开头

System.***out***.println(s1.startsWith("he"));

System.***out***.println(s1.startsWith("ll"));

}

}

* 练习：模拟用户登录
* 获取功能方法
  + int length():获取字符串的长度，其实也就是字符个数
  + char charAt(int index):获取指定索引处的字符
  + int indexOf(String str):获取str在字符串对象中第一次出现的索引,没有返回-1
  + String substring(int start):从start开始截取字符串到字符串结尾
  + String substring(int start,int end):从start开始，到end结束截取字符串;包括start，不包括end

**public class** Demo01String {

**public static void** main(String[] args) {

//创建字符串对象

String s = "helloworld";

//int length():获取字符串的长度，其实也就是字符个数

System.***out***.println(s.length());

System.***out***.println("--------");

//char charAt(int index):获取指定索引处的字符

System.***out***.println(s.charAt(0));

System.***out***.println(s.charAt(1));

System.***out***.println("--------");

//int indexOf(String str):获取str在字符串对象中第一次出现的索引,没有返回-1

System.***out***.println(s.indexOf("l"));

System.***out***.println(s.indexOf("owo"));

System.***out***.println(s.indexOf("ak"));

System.***out***.println("--------");

//String substring(int start):从start开始截取字符串到字符串结尾

System.***out***.println(s.substring(0));

System.***out***.println(s.substring(5));

System.***out***.println("--------");

//String substring(int start,int end):从start开始，到end结束截取字符串。包括start，不包括end

System.***out***.println(s.substring(0, s.length()));

System.***out***.println(s.substring(3,8));

}

}

* 练习：字符串的遍历

**public class** Demo02StringTest {

**public static void** main(String[] args) {

//创建一个字符串对象

String s = "abcde";

//原始做法

System.***out***.println(s.charAt(0));

System.***out***.println(s.charAt(1));

System.***out***.println(s.charAt(2));

System.***out***.println(s.charAt(3));

System.***out***.println(s.charAt(4));

System.***out***.println("---------");

//用for循环改进

**for**(**int**x=0; x<5; x++) {

System.***out***.println(s.charAt(x));

}

System.***out***.println("---------");

//用length()方法获取字符串的长度

**for**(**int**x=0; x<s.length(); x++) {

System.***out***.println(s.charAt(x));

}

}

}

* 练习：统计字符串中大写、小写及数字字符个数

**public class** Demo03StringTest {

**public static void** main(String[] args) {

//键盘录入一个字符串数据

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("请输入一个字符串数据：");

String s = sc.nextLine();

//定义三个统计变量，初始化值都是0

**int** bigCount = 0;

**int** smallCount = 0;

**int** numberCount = 0;

//遍历字符串，得到每一个字符

**for**(**int**x=0; x<s.length(); x++) {

**char** ch = s.charAt(x);

//拿字符进行判断

**if**(ch>='A'&&ch<='Z') {

bigCount++;

}**else if**(ch>='a'&&ch<='z') {

smallCount++;

}**else if**(ch>='0'&&ch<='9') {

numberCount++;

}**else** {

System.***out***.println("该字符"+ch+"非法");

}

}

//输出结果

System.***out***.println("大写字符："+bigCount+"个");

System.***out***.println("小写字符："+smallCount+"个");

System.***out***.println("数字字符："+numberCount+"个");

}

}

* 转换功能方法
  + char[] toCharArray():把字符串转换为字符数组
  + String toLowerCase():把字符串转换为小写字符串
  + String toUpperCase():把字符串转换为大写字符串

**public class** Demo01String {

**public static void** main(String[] args) {

//创建字符串对象

String s = "abcde";

//char[] toCharArray():把字符串转换为字符数组

**char**[] chs = s.toCharArray();

**for**(**int**x=0; x<chs.length; x++) {

System.***out***.println(chs[x]);

}

System.***out***.println("-----------");

//String toLowerCase():把字符串转换为小写字符串

System.***out***.println("HelloWorld".toLowerCase());

//String toUpperCase():把字符串转换为大写字符串

System.***out***.println("HelloWorld".toUpperCase());

}

}

* 练习：键盘录入一个字符串，把该字符串的首字母转成大写，其余为小写。

**public class** Demo02StringTest {

**public static void** main(String[] args) {

//键盘录入一个字符串

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("请输入一个字符串：");

String s = sc.nextLine();

//截取首字母

String s1 = s.substring(0, 1);

//截取除了首字母以外的字符串

String s2 = s.substring(1);

//2转大写+3转小写

String s3 = s1.toUpperCase()+s2.toLowerCase();

//输出即可

System.***out***.println("s3:"+s3);

}

}

* 去除空格和分割功能方法
  + String trim()去除字符串两端空格
  + String[] split(String str)按照指定符号分割字符串
  + String[] repalce(String oldStr,String newStr) 使用newStr替换所有的oldStr

**public class** Demo01String {

**public static void** main(String[] args) {

//创建字符串对象

String s1 = "helloworld";

String s2 = " helloworld ";

String s3 = " hello world ";

System.***out***.println("---"+s1+"---");

System.***out***.println("---"+s1.trim()+"---");

System.***out***.println("---"+s2+"---");

System.***out***.println("---"+s2.trim()+"---");

System.***out***.println("---"+s3+"---");

System.***out***.println("---"+s3.trim()+"---");

System.***out***.println("-------------------");

//String[] split(String str)

//创建字符串对象

String s4 = "aa,bb,cc";

String[] strArray = s4.split(",");

**for**(**int** x=0; x<strArray.length; x++) {

System.***out***.println(strArray[x]);

}

}

}

* 练习：把数组中的数据按照指定格式拼接成一个字符串

举例：int[] arr = {1,2,3}; 输出结果：[1, 2, 3]

**public class** Demo01StringTest {

**public static void** main(String[] args) {

//定义一个int类型的数组

**int**[] arr = {1,2,3};

//调用方法

String s = *arrayToString*(arr);

//输出结果

System.***out***.println("s:"+s);

}

/\*

\* 写方法实现把数组中的元素按照指定的格式拼接成一个字符串

\* 两个明确：

\* 返回值类型：String

\* 参数列表：int[] arr

\*/

**public static** String arrayToString(**int**[] arr) {

String s = "";

s += "[";

**for**(**int**x=0; x<arr.length; x++) {

**if**(x==arr.length-1) {

s += arr[x];

s += "]";

}**else** {

s += arr[x];

s += ", ";

}

}

**return** s;

}

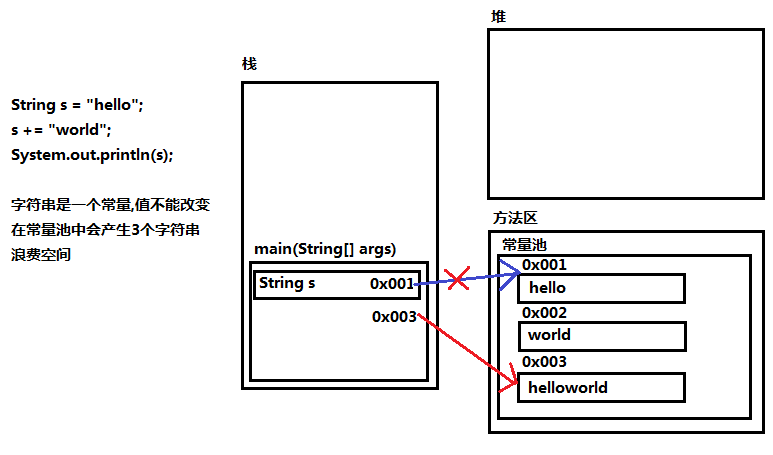
}

* 练习：字符串反转（两种方式）

# StringBuilder类

## StringBuilder概述

我们如果对字符串进行拼接操作，每次拼接，都会构建一个新的String对象，既耗时，又浪费空间。而StringBuilder就可以解决这个问题。



## -StringBuilder和String的区别?

* String的内容是固定的。
* StringBuilder的内容是可变的。

## 方法

* 构造方法与相关的方法
  + public StringBuilder():默认无参构造方法
  + public int capacity():获取底层数组的总长度（理论长度）
  + public int length():获取底层数组的已使用长度（实际长度）
* 常用方法
  + public StringBuilder append(任意类型):添加数据，并返回自身对象
  + public StringBuilder reverse():反转所有字符内容

**public class**  Demo02StringBuilder {

**public static void** main(String[] args) {

//创建对象

StringBuilder builder = **new** StringBuilder();

//public StringBuilder append(任意类型)

//StringBuilder builder2 = builder.append("hello");

/\*

System.out.println("builder:"+builder);

System.out.println("builder2:"+builder2);

System.out.println(builder == builder2); //true

\*/

/\*

builder.append("hello");

builder.append("world");

builder.append(true);

builder.append(100);

\*/

//链式编程

builder.append("hello").append("world").append(**true**).append(100);

System.***out***.println("builder:"+builder);

//public StringBuilder reverse()

builder.reverse();

System.***out***.println("builder:"+builder);

}

}

* 与String的相互转换

## 练习

* 数组元素格式拼接

举例：int[] arr = {1,2,3};结果：[1, 2, 3]

**public class**  Demo04StringBuilderTest {

**public static void** main(String[] args) {

//定义一个数组

**int**[] arr = {1,2,3};

//调用方法

String s = *arrayToString*(arr);

//输出结果

System.***out***.println("s:"+s);

}

/\*

\* 写方法实现拼接

\* 两个明确：

\* 返回值类型：String

\* 参数列表：int[] arr

\*/

**public static** String arrayToString(**int**[] arr) {

StringBuilder builder = **new** StringBuilder();

builder.append("[");

**for**(**int**x=0; x<arr.length; x++) {

**if**(x==arr.length-1) {

builder.append(arr[x]).append("]");;

}**else** {

builder.append(arr[x]).append(", ");

}

}

String result = builder.toString();

**return**result;

}

}

* 字符串反转

**public class** Demo05StringBuilderTest {

**public static void** main(String[] args) {

//1:键盘录入一个字符串

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("请输入一个字符串：");

String s = sc.nextLine();

//3:调用方法

String result = *myReverse*(s);

//4:输出结果

System.***out***.println("result:"+result);

}

/\*

\* 2:写方法实现反转

\* 两个明确：

\* 返回值类型：String

\* 参数列表：String

\*/

**public static** String myReverse(String s) {

//String -- StringBuilder -- reverse() -- String

StringBuilder builder= **new** StringBuilder(s);

builder.reverse();

String result = builder.toString();

**return** result;

}

}

* 判断对称字符串（使用StringBuilder的简单方式）