Java中的System类

1. 对于一个类如果没有在API文档中提供**构造方法**，那么有两种可能：
2. **此类不能产生对象，即不能实例化**；这种类中**提供的字段和方法**都是**静态的**；如Math、System等。
3. **对象单例化：**此类**至少**存在一个静态方法且此方法就返回此类的一个对象，这称为**对象单例化，此类不允许通过new创建对象，而是通过调用此类的静态方法产生对象**。这种类中**存在非静态的字段和方法**。如**Runtime类**。

注意： 不能生成对象的类有两种，一是抽象类，抽象类中可能有构造方法，供**子类实例化**使用，可以不存在**抽象方法，**也可以不存在**非抽象方法，存在的非抽象方法供子类继承使用。二是没有构造方法也不是对象单例化模式的类，这种类就像一个工具类一样，只是提供某一个特定功能，没有必要生成对象。**

1. **System类**是直接继承于Object类的，**没有构造方法**，不可以被实例化即不可以生成对象，用**final**修饰的，**没有子类**。其提供的一些重要的字段（Field）和方法，这些字段和方法都是静态的，可以直接利用**类名System**进形**调用**。
2. Java语言的一个**好处**在于：**其具有很好的跨平台性**。
3. 例如： 换行符 \r\n 具有换行功能，但是在不同的操作系统下，作用不同，故不可使用这样的**转义字符**直接进行编程。 Java 语言的处理方法是先获取系统信息，根据系统信息从而调用**相应的转义字符代码**。既然这些比较常用，可以使用**固定字符**代表此功能， 如下面获取换行：

**获取相应系统下的换行字符**：private static final String LINE\_SEPARATOR =System.getProperty("line.separator");//常量是利用static和final修饰

1. System的Fields字段：三个。
2. **out ：**



调用System的out属性可以获得一个**标准打印输出流**：PrintStream流；然后可以直接调用PrintStream流提供的各种**方法**。PrintStream流为打印输出流，提供最重要方法就是打印方法print方法及println方法，可以**直接打印任何数据**。如平常使用最多的就是 **System.out.println();**

至于**PrintStream流**可见笔记IO流技术介绍。

1. **in ：**



调用System的in属性可以获得一个**标准输入流：**InputStream流。

1. **err：**



调用System的err属性可以获得一个**标准打印输出流：**PrintStream流，只不过是用于打印输出**错误**而已了。

**注意： 标准输入流和标准输出流都只能有一个对象，不像其他IO流一样可以通过new创建多个流对象。标准输入流和标准输出流一旦关闭，就不能再次使用了，一般不需要关闭。**

1. **Properties集合**属于一种**Map集合**，里面存的都是**键值对**，且键和值都是String类型，如使用Properties时不需要指定**泛型**。
2. **通过System类的getProperties可以获得的属性有：**

**对这些键值对的遍历代码：**

Properties p = System.getProperties();

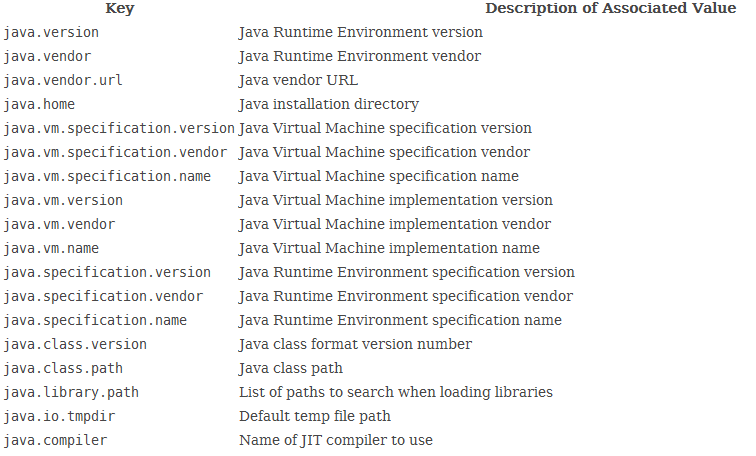
Set<String> propertiesNames = p.stringPropertyNames();

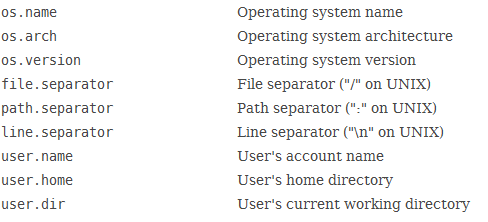
for(String name : propertiesNames) {

String value = p.getProperty(name);

System.out.println(name+" :: "+value);

}



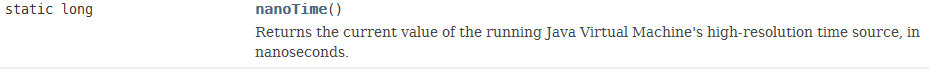


1. System类的主要方法：
2. **获取当前系统的时间：利用毫秒值表示，有了毫秒值就可以创建Date对象，从而进一步获取年月日时分秒。（Calendar是抽象类）**



**如：**System.out.println(System.currentTimeMillis());//1451569077294

**方法：nanoTime（）** 获取的是运行的JVM时间，**纳秒值**。



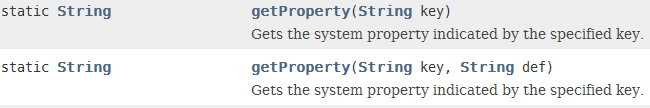
System.out.println(System.nanoTime());//295919839159762

1. **获取系统属性信息方法：**

获取所有的属性信息，存放到Properties集合中：



获取指定的**某个属性信息**，返回String类型的value值：



示例：

String value1 = System.getProperty("user.language");

String value2 = System.getProperty("java.home");

String value3 = System.getProperty("file.encoding");

System.out.println(value1);

System.out.println(value2);

System.out.println(value3);

1. **添加或修改信息属性信息方法：**

给系统设置一些属性信息，这些信息时全局的，其他的应用也可以使用。

**setProperty（String key，String value） 方法，返回一个String值，如果这个键以前存在，就返回以前的值，如果这个键以前不存在，就返回null。**

**如：**String previousValue =System.setProperty("user.language","zh");

System.out.println(previousValue);



