Java中Runtime类和Process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. **Runtime类 （运行时类）直接继承于java.lang.Object 类，存在于java.lang包中。Runtime类没有提供构造方法，采用单例设计模式，依靠它的静态方法getRuntime（），返回一个Runtime对象。单例设计模式的作用就是保证对象的唯一性。**



1. 每一个java应用程序都有一个**Runtime对象**，使应用程序能够与其运行的环境相连接。通过Runtime类自己的静态方法创建一个Runtime类对象，而应用程序是不能创建自己的**Runtime对象**的。
2. Runtime类的主要方法：
3. **exec方法**：返回的都是**Process（进程）对象**。

**execute**  英 ['eksɪkjuːt] vt. **执行，运行**



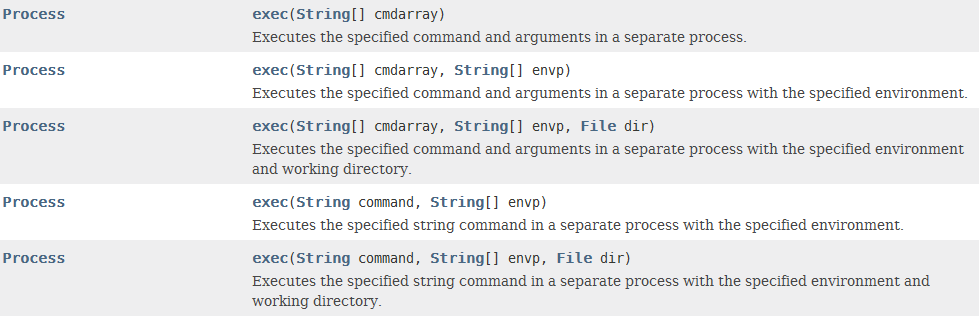
如：

Runtime r = Runtime.getRuntime();

r.exec("notepad");//打开记事本

r.exec("notepad e:\\111.txt");//利用记事本打开111.txt文件

还有一些其他的参数：



1. **Process类**: 直接继承于java.lang.Object类的，存在于java.lang包中。
2. **构造方法：**



1. **方法**：
   1. **destroy方法**：关闭进程。

