# Java 9 改进的进程 API

在 Java 9 之前，Process API 仍然缺乏对使用本地进程的基本支持，例如获取进程的 PID 和所有者，进程的开始时间，进程使用了多少 CPU 时间，多少本地进程正在运行等。

Java 9 向 Process API 添加了一个名为 ProcessHandle 的接口来增强 java.lang.Process 类。

ProcessHandle 接口的实例标识一个本地进程，它允许查询进程状态并管理进程。

ProcessHandle 嵌套接口 Info 来让开发者逃离时常因为要获取一个本地进程的 PID 而不得不使用本地代码的窘境。

我们不能在接口中提供方法实现。如果我们要提供抽象方法和非抽象方法（方法与实现）的组合，那么我们就得使用抽象类。

ProcessHandle 接口中声明的 onExit() 方法可用于在某个进程终止时触发某些操作。

## 实例

import java.time.ZoneId; import java.util.stream.Stream; import java.util.stream.Collectors; import java.io.IOException; public class Tester { public static void main(String[] args) throws IOException { ProcessBuilder pb = new ProcessBuilder("notepad.exe"); String np = "Not Present"; Process p = pb.start(); ProcessHandle.Info info = p.info(); System.out.printf("Process ID : %s%n", p.pid()); System.out.printf("Command name : %s%n", info.command().orElse(np)); System.out.printf("Command line : %s%n", info.commandLine().orElse(np)); System.out.printf("Start time: %s%n", info.startInstant().map(i -> i.atZone(ZoneId.systemDefault()) .toLocalDateTime().toString()).orElse(np)); System.out.printf("Arguments : %s%n", info.arguments().map(a -> Stream.of(a).collect( Collectors.joining(" "))).orElse(np)); System.out.printf("User : %s%n", info.user().orElse(np)); } }

以上实例执行输出结果为：

Process ID : 5800Command name : C:\Windows\System32\notepad.exeCommand line : Not PresentStart time: 2017-11-04T21:35:03.626Arguments : Not PresentUser: administrator