零、前言

在Java中可用于执行系统命令的方式有三种，分别是java.lang.Runtime，java.lang.ProcessBuilder以及java.lang.UNIXProcess/ProcessImpl。

这三种方式都是JDK原生提供的，并不需要再额外引入。

下面我们通过代码示例来进一步理解这几种方式是如何执行系统命令的。

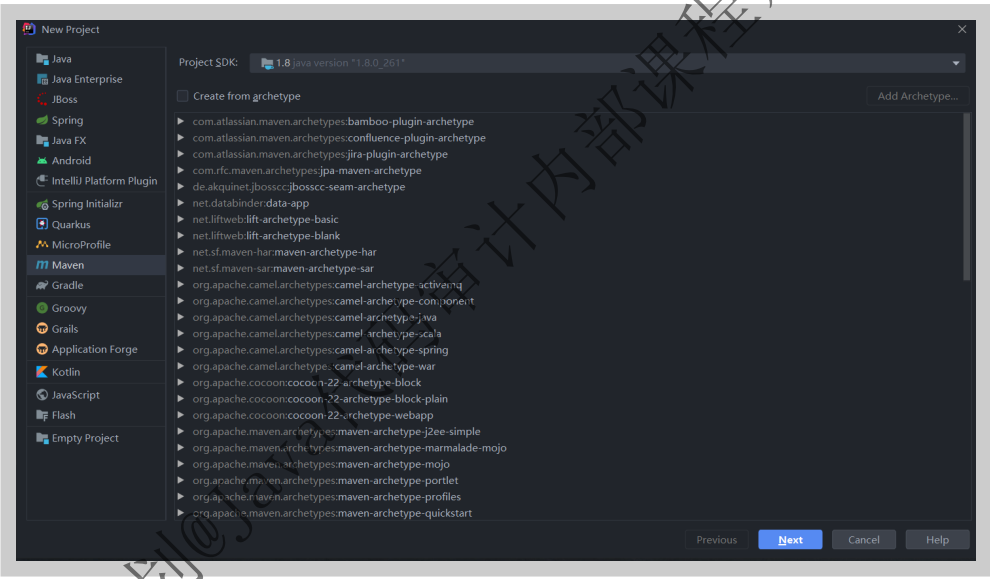
在我们针对Java系统进行代码审计以及渗透测试时，目标系统如果存在执行命令的功能，大概率是这三种方法之一。

一、创建Demo工程

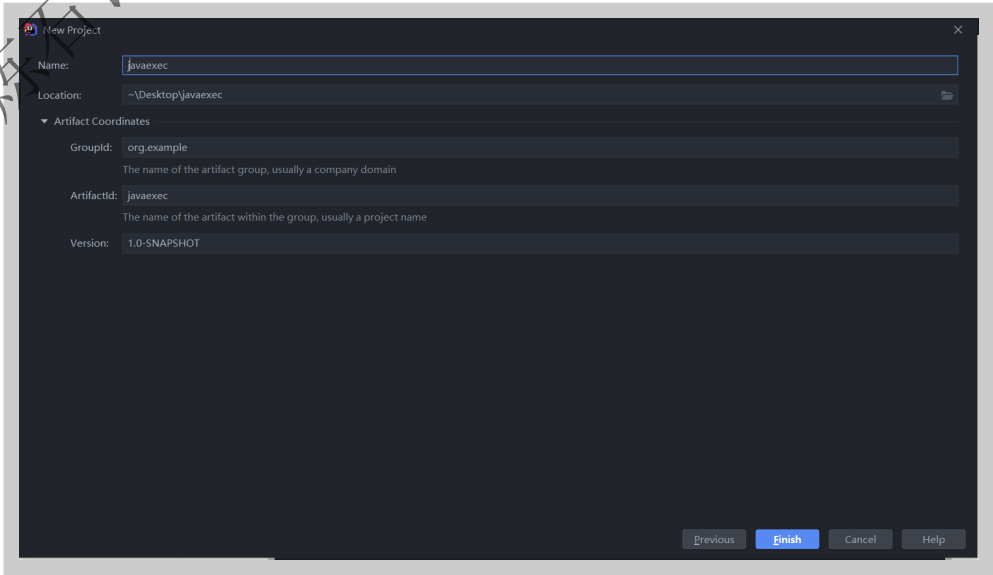
使用IDEA创建一个基础项目工程，用于下面几种命令执行方法的调试。

①、双击IDEA启动，点击 `Create New Project` 。

②、左侧选择Maven，模板就不用选择了，点击Next。如下图所示：



③、随便起个名字，最后点击Finish即可。如下图所示：



到此项目创建完成，下面根据每一个实现方式编写对应的示例代码。

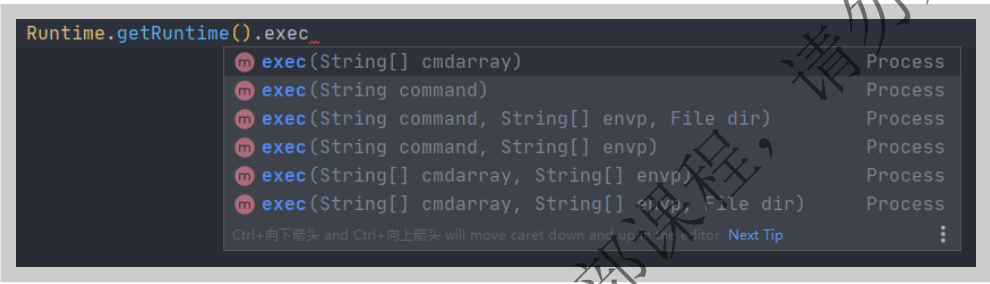
二、java.lang.Runtime执行命令

1、使用说明

java.lang提供了一系列对Java编程至关重要的类。具体提供的类可看：https://docs.oracle.com/javase/7/docs/api/java/lang/package-summary.html

其中Runtime是java.lang中的一个类，主要是与操作系统交互执行操作命令。

而在 `java.lang.Runtime` 中我们主要关注 `exec()` 方法，使用该方法执行具体的命令，而执行exec()方法有以下六种形式，如下图所示：



方法	英文释义	中文释义（非标准）
exec(String[] cmdarray)	Executes the specified command and arguments in a separate process.	在单独的进程中执行指定的命令和参数。
exec(String command)	Executes the specified string command in a separate process.	在单独的进程中执行指定的字符串命令。
exec(String command, String[] envp, File dir)	Executes the specified string command in a separate process with the specified environment and working directory.	在具有指定环境和工作目录的单独进程中执行指定的字符串命令。
exec(String command, String[] envp)	Executes the specified string command in a separate process with the specified environment.	在具有指定环境的单独进程中执行指定的字符串命令。
exec(String[] cmdarray, String[] envp)	Executes the specified command and arguments in a separate process with the specified environment.	在具有指定环境的单独进程中执行指定的命令和参数。
exec(String[] cmdarray, String[] envp, File dir)	Executes the specified command and arguments in a separate process with the specified environment and working directory.	在具有指定环境和工作目录的单独进程中执行指定的命令和参数。

java.lang.Runtime的API文档 https://docs.oracle.com/javase/7/docs/api/java/lang/Runtime.html

目前我们只先关注 `exec(String command)` 和 `exec(String[] cmdarray)` 这两种执行方式。

下面通过示例代码来理解。

1.1、exec(String command)

在单独的进程中执行指定的字符串命令。

简单来说就是直接执行字符串命令。

```
//简易示例代码
String command = "whoami";
Runtime.getRuntime().exec(command)
```

1.2、exec(String[] cmdarray)

在单独的进程中执行指定的命令和参数。

简单来说就是以数组的形式接收多个字符串然后执行。

```
//简易示例代码
String[] command = { "cmd", "/c", "whoami" };
Runtime.getRuntime().exec(command)
```

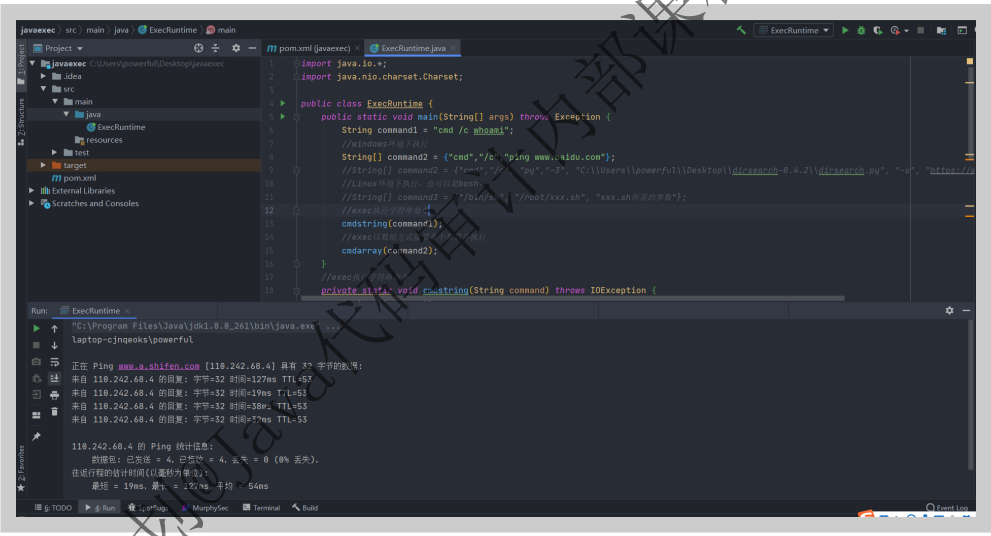
2、代码示例

①、在 `src/main/java` 目录下新建一个Java Class，名叫 `ExecRuntime` ，并键入以下代码，最终如下图所示：

```
import java.io.*;
import java.nio.charset.Charset;

public class ExecRuntime {
    public static void main(String[] args) throws Exception {
        String command1 = "cmd /c whoami";
        //windows环境下执行
        String[] command2 = {"cmd","/c","ping www.baidu.com"};
        //String[] command2 = {"cmd","/c", "py","-3",
"C:\\Users\\powerful\\Desktop\\dirsearch-0.4.2\\dirsearch.py", "-u",
"https://www.baidu.com", "-e *"};
        //Linux环境下执行，也可以是bash，
        //String[] command3 = {"/bin/sh", "/root/xxx.sh", "xxx.sh所需的参
数"};
        //exec执行字符串命令
        cmdstring(command1);
        //exec以数组方式接受多个参数并执行
        cmdarray(command2);
    }
    //exec执行字符串命令
    private static void cmdstring(String command) throws IOException {
```

```
        String line = null;
        Process process = Runtime.getRuntime().exec(command);
        //使用BufferedReader设置编码解决执行命令响应中文乱码问题
        BufferedReader bufferedReader =
                    new BufferedReader(new
InputStreamReader(process.getInputStream(), Charset.forName("GBK")));
        while ((line = bufferedReader.readLine()) != null) {
            System.out.println(line);
        }
    }
    //exec以数组方式接受多个参数并执行
    private static void cmdarray(String[] command) throws IOException {
        String line = null;
        Process process = Runtime.getRuntime().exec(command);
        //使用BufferedReader设置编码解决执行命令响应中文乱码问题
        BufferedReader bufferedReader =
                new BufferedReader(new
InputStreamReader(process.getInputStream(), Charset.forName("GBK")));
        while ((line = bufferedReader.readLine()) != null) {
            System.out.println(line);
        }
    }
}
```



②、在代码注释中直接以字符串形式执行命令和以字符串数组形式执行命令给出了几个样例，大家可以关闭注释调试下。

③、为了便于理解字符串数组形式执行命令，我举了个执行python脚本的例子，命令如下：`String[] command2 = {"cmd","/c", "py","-3", "C:\\Users\\powerful\\Desktop\\dirsearch-0.4.2\\dirsearch.py", "-u", "https://www.baidu.com", "-e *"};`，大家可以关闭该注释可以启动调试一下，观察下结果。

④、我们的执行命令样例都是在Windows环境下执行的，等到练习命令执行注入漏洞时，会进一步讲解在Linux系统下遇见的各种问题。现在先熟悉下这些函数方法吧。

三、java.lang.ProcessBuilder执行命令

1、使用说明

在上面介绍了java.lang。java.lang.Runtime是其中的一个API。而 `java.lang.ProcessBuilder` 也是java.lang中的一个API。该类主要用于创建操作系统进程。具体介绍可查看： https://docs.oracle.com/javase/7/docs/api/java/lang/ProcessBuilder.html 。

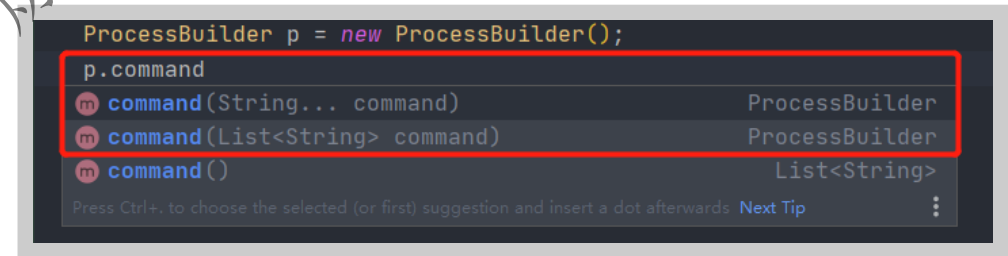
在 `java.lang.ProcessBuilder` 中我们要关注 `command()` 方法，可通过该方法设置要执行的命令参数。以及 `start()` 方法，简单来说使用该方法可以执行命令。以及

1.1、command()方法

官方原文介绍： https://docs.oracle.com/javase/7/docs/api/java/lang/ProcessBuilder.html#command()

`command()` 方法主要用于设置要执行的命令。

`command()` 方法传参有两种方式，一种是普通的字符串，另一种是字符串列表，如下图所示：




```
//简易示例代码
ProcessBuilder p = new ProcessBuilder();
p.command("calc");
```

1.2、start()方法

官方原文介绍：https://docs.oracle.com/javase/7/docs/api/java/lang/ProcessBuilder.html#start()

使用 `start()` 方法可以创建一个新的具有命令，或环境，或工作目录，或输入来源，或标准输出和标准错误输出的目标，或redirectErrorStream属性的进程。
新进程中调用的命令和参数有 `command()` 方法设置，工作目录将由 `directory()` 方法设置，进程环境将由 `environment()` 设置。

在使用 `command()` 方法设置执行命令参数后，然后由 `start()` 方法创建一个新的进程进而在系统中执行了我们设置的命令。

这么看来 `java.lang.ProcessBuilder#start()` 和 `Runtime.exec(String[] cmdarray, String[] envp, File dir)` 有些相似。

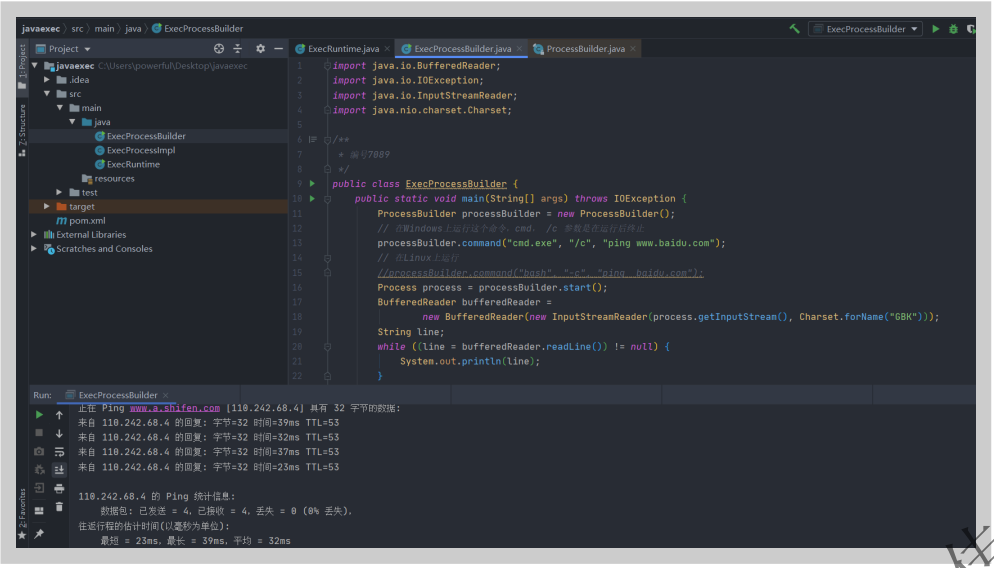
```
//简易示例代码
Process cmd = new ProcessBuilder(command).start();
```

2、代码示例

①、在 `src/main/java` 目录下新建一个Java Class，名叫 `ExecProcessBuilder` ，并键入以下代码，最终如下图所示：

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.nio.charset.Charset;


public class ExecProcessBuilder {
    public static void main(String[] args) throws IOException {
        ProcessBuilder processBuilder = new ProcessBuilder();
        // 在Windows上运行这个命令，cmd， /c 参数是在运行后终止
        processBuilder.command("cmd.exe", "/c", "ping www.baidu.com");
        // 在Linux上运行
        //processBuilder.command("bash", "-c", "ping  baidu.com");
        Process process = processBuilder.start();
        BufferedReader bufferedReader =
                new BufferedReader(new
InputStreamReader(process.getInputStream(), Charset.forName("GBK")));
        String line;
        while ((line = bufferedReader.readLine()) != null) {
            System.out.println(line);
        }
        }
    }
```



②、建议大家私下多试一些命令进行调试，天马行空的想法都可以尝试一下，另外在代码中还给出了linux系统下运行的命令方法。

四、java.lang.UNIXProcess/ProcessImpl 执行命令

1、使用说明

首先对于 `UNIXProcess` 和 `ProcessImpl` 可以理解本就是一个东西，因为在JDK9的时候把 `UNIXProcess` 合并到了 `ProcessImpl` 当中了。具体可查看：`https://hg.openjdk.java.net/jdk-updates/jdk9u/jdk/rev/98eb910c9a97` 。

`UNIXProcess` 和 `ProcessImpl` 最终都是调用 `native` 执行系统命令的类，这个类提供了一个叫 `forkAndExec` 的native方法，如方法名所述主要是通过 `fork&exec` 来执行本地系统命令。

UNIXProcess类是*nix系统在java程序中的体现，可以使用该类创建新进程，实现与”fork”类似的功能（对于Windows系统，使用的是java.lang.ProcessImpl类）

ProcessImpl是更为底层的实现，Runtime和ProcessBuilder执行命令实际上也是调用了ProcessImpl这个类。

对于ProcessImpl类，我们不能直接调用需要配合使用反射。

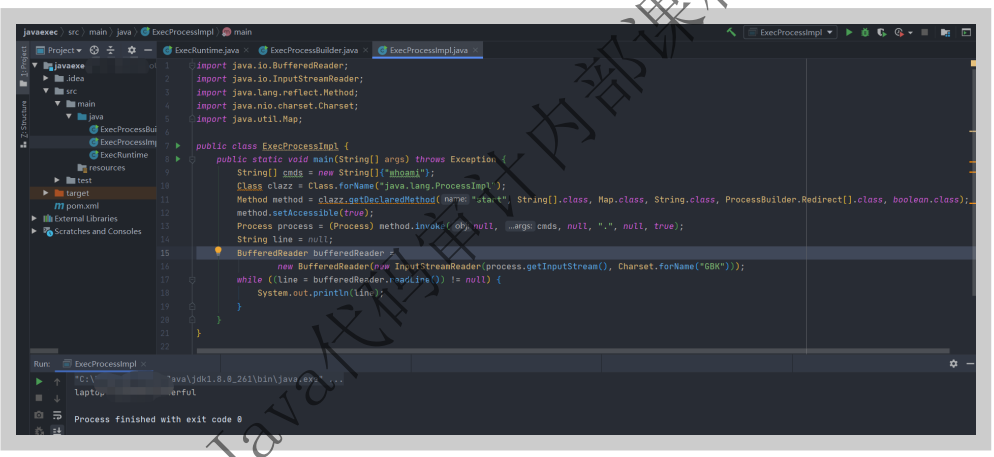
2、代码示例

①、在 `src/main/java` 目录下新建一个Java Class，名叫 `ExecProcessImpl` ，并键入以下代码，最终如下图所示：

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
```

```
import java.lang.reflect.Method;
import java.nio.charset.Charset;
import java.util.Map;

public class ExecProcessImpl {
    public static void main(String[] args) throws Exception {
        String[] cmds = new String[]{"whoami"};
        Class clazz = Class.forName("java.lang.ProcessImpl");
        Method method = clazz.getDeclaredMethod("start",
String[].class, Map.class, String.class,
ProcessBuilder.Redirect[].class, boolean.class);
        method.setAccessible(true);
        Process process = (Process) method.invoke(null, cmds, null,
".", null, true);
        String line = null;
        BufferedReader bufferedReader =
                new BufferedReader(new
InputStreamReader(process.getInputStream(), Charset.forName("GBK")));
        while ((line = bufferedReader.readLine()) != null) {
            System.out.println(line);
        }
    }
}
```



②、由于ProcessImpl类不能直接调用，需要配合反射来进行命令执行，因此此部分中的反射知识点可先自行学习。到后面Java反射章节会在进一步讲解。

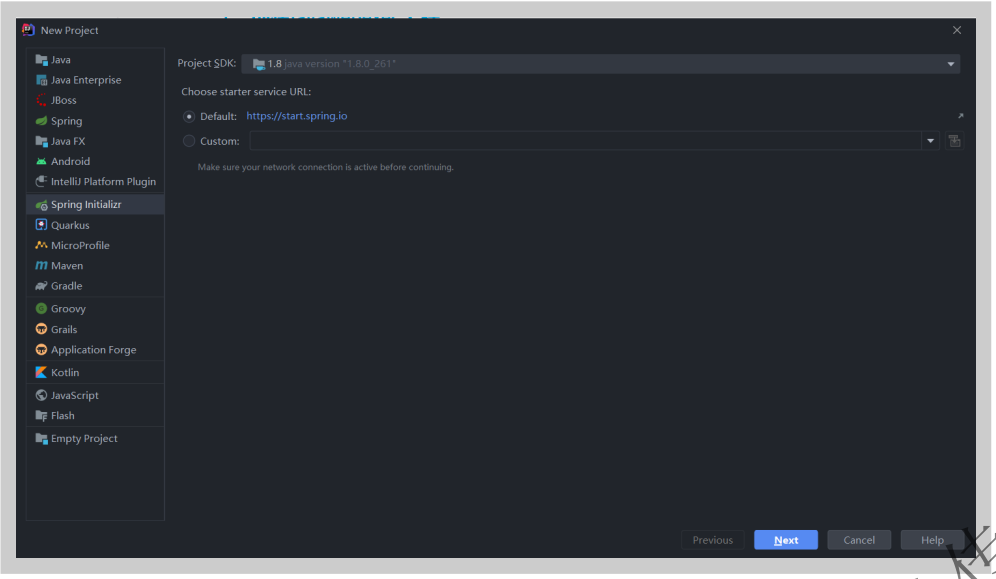
五、执行命令Java Web项目

1、创建javawebexec工程

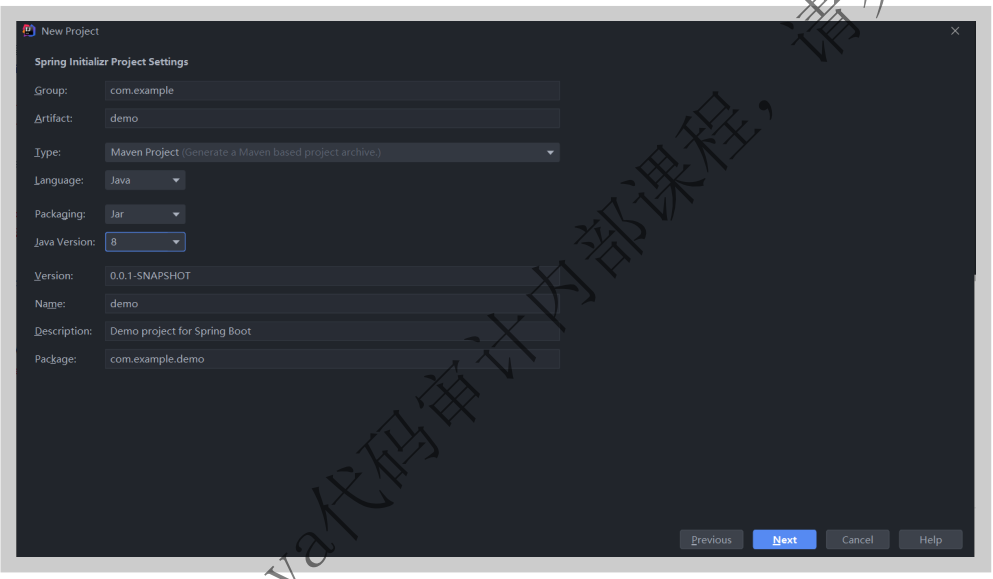
我们创建一个基于SpringBoot的执行系统命令的JavaWeb工程，这回我们从WEB视角调试上面几种方法。

①、双击IDEA启动，点击 `Create New Project` 。

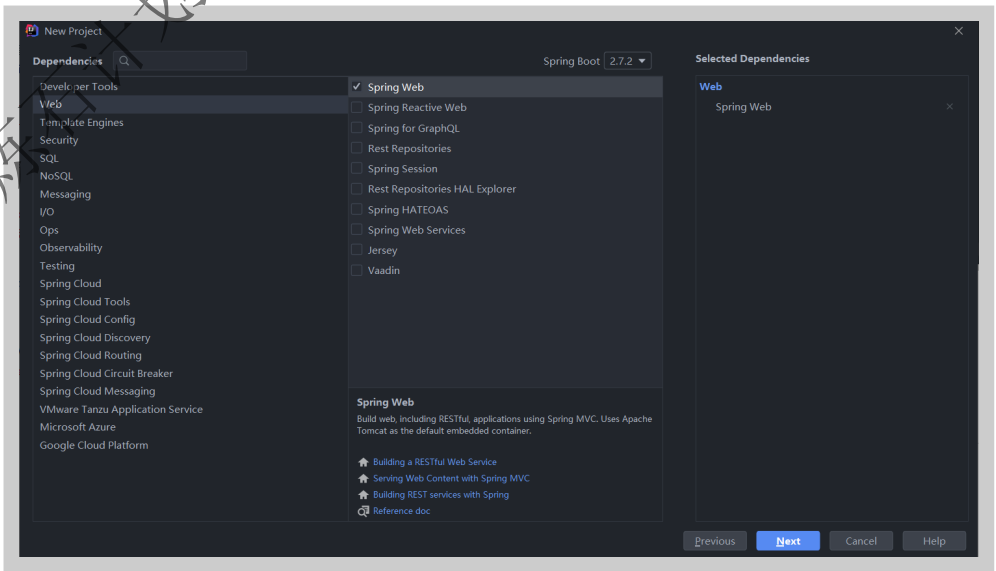
②、左侧选择 `Spring Initializr` ，内容默认即可，点击Next，如下图所示：



③、稍等片刻， Srping Initializr Project Settings页面配置内容将Java Version选择为8，其他默认即可。如下图所示：

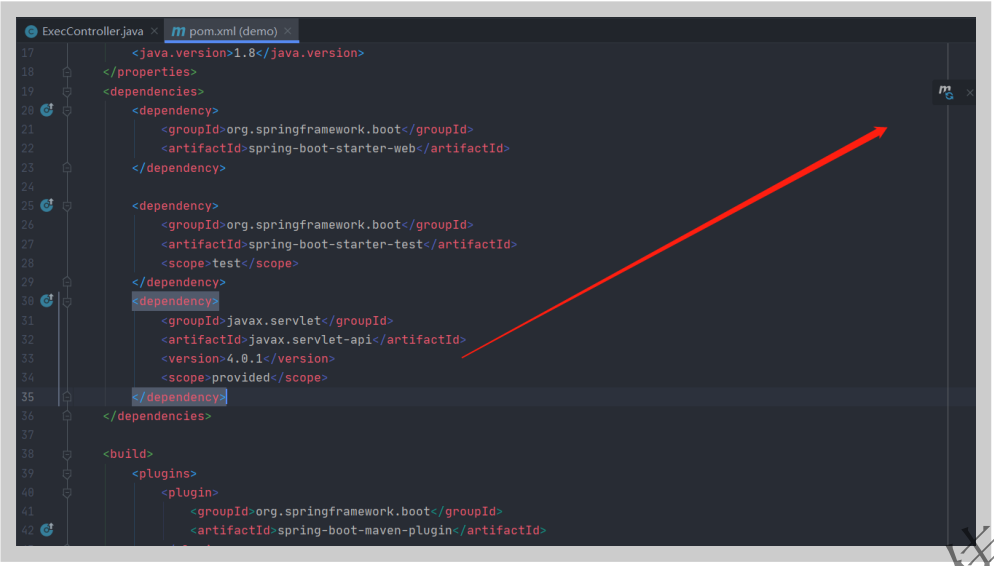


④、点击Next，进入依赖项选择页面，我们选择 `Web -> Spring Web` 这一个即可，如下图所示：



⑤、起个项目名称为 `javawebexec` ，最后点击Finish，稍等片刻进入项目工程。

⑥、在 `pom.xml` 中引入servlet依赖，并重载maven，如下图所示：



2、示例代码

①、在 `src/main/java/com/example/demo` 目录下创建一个名为 `ExecController` 的Java Class，并键入以下代码。

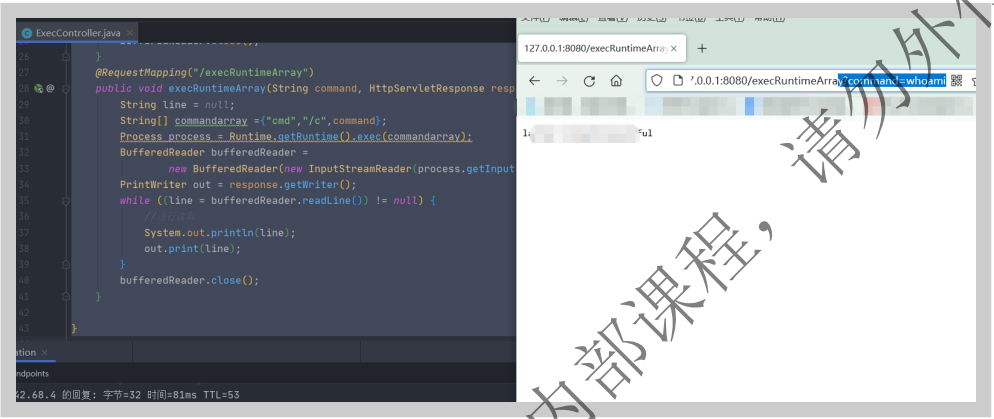
```
package com.example.demo;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import javax.servlet.http.HttpServletResponse;
import java.io.*;
import java.nio.charset.Charset;

@Controller
@ResponseBody
public class ExecController {

    @RequestMapping("/execRuntimeString")
    public void execRuntimeString(String command, HttpServletResponse
response) throws IOException {
        String line = null;
        Process process = Runtime.getRuntime().exec(command);
        BufferedReader bufferedReader =
                new BufferedReader(new
InputStreamReader(process.getInputStream(), Charset.forName("GBK")));
        PrintWriter out = response.getWriter();
        while ((line = bufferedReader.readLine()) != null) {
            //逐行读取
            System.out.println(line);
            out.print(line);
        }
        bufferedReader.close();
    }
    @RequestMapping("/execRuntimeArray")
    public void execRuntimeArray(String command, HttpServletResponse
response) throws IOException {
        String line = null;
```

```
        String[] commandarray ={"cmd","/c",command};
        Process process = Runtime.getRuntime().exec(commandarray);
        BufferedReader bufferedReader =
                new BufferedReader(new
InputStreamReader(process.getInputStream(), Charset.forName("GBK")));
        PrintWriter out = response.getWriter();
        while ((line = bufferedReader.readLine()) != null) {
            //逐行读取
            System.out.println(line);
            out.print(line);
        }
        bufferedReader.close();
    }

}
```



②、在这个Controller里面我只写了java.lang.Runtime下执行命令的两种方式。在此留作业，请各位将 `java.lang.ProcessBuilder` 和 `java.lang.ProcessImpl` 根据上面形式写出对应的Controller，并成功运行。