# 案例一、

[兼容性] JDK1.5及以上

## 工程1主程序

**package** com.hugh.demo;[[1]](#footnote-1)

**public** **class** MyProgram {

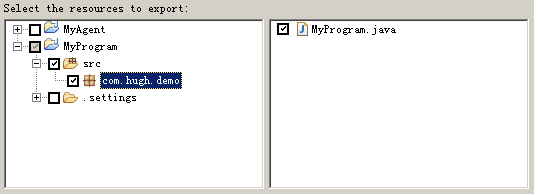
**public** **static** **void** main(String[] args) {

System.***out***.println("Calling main(String[]) ...");

}

}

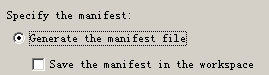
## 工程1导出为myapp.jar，步骤如下：



P1. 选择导出的源文件，只导出class



P2. 导出jar包到当前主程序工程根目录



P3. 自动生成manifest文件，无需保存



P4. 选择程序执行入口

## 工程2代理程序

**package** com.hugh.demo.agent;

**import** java.lang.instrument.Instrumentation;

**public** **class** MyAgent {

**public** **static** **void** premain(String agentOps, Instrumentation inst) {

System.***out***.println("Calling public static void premain(String, Instrumentation) ...");

System.***out***.println(agentOps);

}

**public** **static** **void** premain(String agentOps) {

System.***out***.println("Calling public static void premain(String) ...");

System.***out***.println(agentOps);

}

}

## 工程2的manifest配置文件

Manifest-Version: 1.0

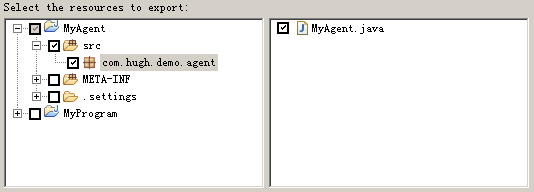
Premain-Class: com.hugh.demo.agent.MyAgent

Can-Redefine-Classes: true

## 工程2的工程结构



## 工程2导出为myagent.jar，步骤如下：



P5. 仅导出class



P6. 导出到主程序根目录，和myapp.jar同一个目录下



P7. 使用自定义的manifest配置文件

## 命令行

"d:\Program Files\Java\jdk1.5.0\_22\bin\java" -javaagent:myagent.jar=Hello1 -javaagent:myagent.jar=Hello2 -jar myapp.jar

*Calling public static void premain(String, Instrumentation) ...*

*Hello1*

*Calling public static void premain(String, Instrumentation) ...*

*Hello2*

*Calling main(String[]) ...*

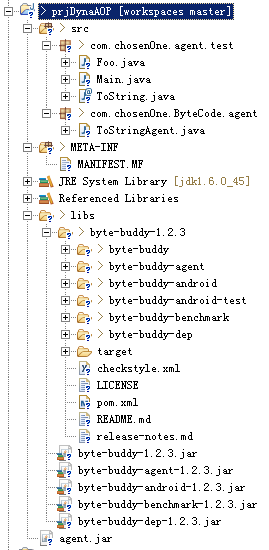
## 工程源代码



# 案例二、

[兼容性] JDK1.6及以上

[依赖] byte-buddy组件，版本1.2.3，<https://github.com/raphw/byte-buddy/archive/byte-buddy-1.2.3.tar.gz>

[工程结构] 见左图，下载byte-buddy源代码到libs目录下并编译byte-buddy的jar

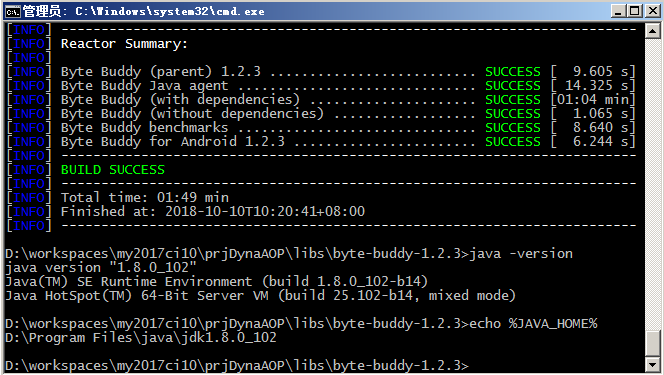
[依赖] apache maven 3.5.4，下载并解压到D:\programs\apache-maven-3.5.4，将D:\programs\apache-maven-3.5.4\bin添加到环境变量Path，maven启用代理，修改conf\settings.xml的<proxies/>标签，注释掉<username/>、<password/>和<nonProxyHosts/>标签，修改其他配置项，运行：

**mvn.cmd clean**

[注意] JDK版本可以无需考虑，本案例使用JDK 1.8.102

[编译] 进入代码目录：libs\byte-buddy-1.2.3，并运行

**mvn.cmd package**



[注意] 将打包好的byte-buddy\*.jar拷贝到libs/下，选中后右键Build Path🡪Add to build path…

## 1、测试程序

**package** com.chosen0ne.agent.test;

**import** java.lang.annotation.Retention;

**import** java.lang.annotation.RetentionPolicy;

@Retention(RetentionPolicy.***RUNTIME***)[[2]](#footnote-2)

**public** **@interface** ToString {

}

**package** com.chosen0ne.agent.test;

@ToString

**public** **class** Foo {

}

**package** com.chosen0ne.agent.test;

**public** **class** Main {

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

Foo foo = **new** Foo();

System.***out***.println(foo.toString());

}

}

## 2、代理程序

**package** com.chosen0ne.ByteCode.agent;

**import** java.lang.instrument.Instrumentation;

**import** com.chosen0ne.agent.test.ToString;

**import** net.bytebuddy.agent.builder.AgentBuilder;

**import** net.bytebuddy.description.type.TypeDescription;

**import** net.bytebuddy.dynamic.DynamicType.Builder;

**import** net.bytebuddy.implementation.FixedValue;

**import** net.bytebuddy.matcher.ElementMatchers;

**public** **class** ToStringAgent {

**public** **static** **void** premain(String args, Instrumentation instrumentation) {

System.***out***.println("print pre main");

**new** AgentBuilder.Default().type(ElementMatchers.*isAnnotatedWith*(ToString.**class**))

.transform(**new** AgentBuilder.Transformer() {

@Override

**public** Builder<?> transform(Builder<?> builder, TypeDescription typeDescription,

ClassLoader classLoader) {

**return** builder.method(ElementMatchers.*named*("toString")).intercept(FixedValue.*value*("test"));

}

}).installOn(instrumentation);

}

}

## 3、打包agent

仅打包agent程序即可，MANIFEST.MF定义如下：

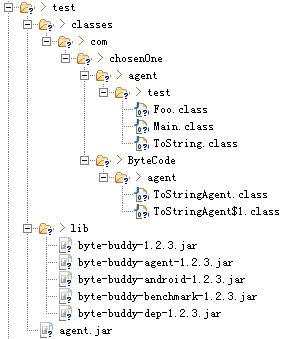
*Manifest-Version: 1.0*

*Premain-Class: com.chosen0ne.ByteCode.agent.ToStringAgent*

*Can-Redefine-Classes: true*

## 4、测试目录结构

需要将测试程序放在test/classes目录下，byte-buddy\*.jar放在test/lib目录下，agent.jar放在test/目录下



运行：

"d:\Program Files\java\jdk1.6.0\_45\bin\java.exe" -cp classes;lib/byte-buddy-dep-1.2.3.jar -javaagent:agent.jar com.chosen0ne.agent.test.Main

## 5、工程代码



# 案例三、动态修改字节码，添加方法耗时监控

## 1、目标程序

**package** com.jdktest.SayHello;

**public** **class** Target {

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

**new** Target().sayHello();

}

**public** **void** sayHello() {

System.***out***.println("Hello, guys!");

}

}

## 2、代理程序

**package** com.jdktest.instrument;

**import** java.lang.instrument.ClassFileTransformer;

**import** java.lang.instrument.Instrumentation;

**public** **class** AopAgentTest {

**static** **private** Instrumentation *\_inst* = **null**;

**public** **static** **void** premain(String agentArgs, Instrumentation inst) {

System.***out***.println("AopAgentTest.premain() was called.");

*\_inst* = inst;

ClassFileTransformer trans = **new** AopAgentTransformer();

System.***out***.println("Adding a AopAgentTest instance to the JVM.");

*\_inst*.addTransformer(trans);

}

}

**package** com.jdktest.instrument;

**import** java.lang.instrument.ClassFileTransformer;

**import** java.lang.instrument.IllegalClassFormatException;

**import** java.security.ProtectionDomain;

**import** javassist.CannotCompileException;

**import** javassist.ClassPool;

**import** javassist.CodeConverter;

**import** javassist.CtClass;

**import** javassist.CtMethod;

**import** javassist.NotFoundException;

**import** javassist.expr.ExprEditor;

**import** javassist.expr.MethodCall;

**public** **class** AopAgentTransformer **implements** ClassFileTransformer {

@SuppressWarnings("unused")

**public** **byte**[] transform(ClassLoader loader, String className, Class<?> classBeingRedefined,

ProtectionDomain protectionDomain, **byte**[] classfileBuffer) **throws** IllegalClassFormatException {

**byte**[] transformed = **null**;

System.***out***.println("Transforming " + className);

ClassPool pool = **null**;

CtClass cl = **null**;

**try** {

pool = ClassPool.*getDefault*();

cl = pool.makeClass(**new** java.io.ByteArrayInputStream(classfileBuffer));

CodeConverter convert = **new** CodeConverter();

**if** (cl.isInterface() == **false**) {

CtMethod[] methods = cl.getDeclaredMethods();

**for** (**int** i = 0; i < methods.length; i++) {

**if** (methods[i].isEmpty() == **false**) {

AOPInsertMethod(methods[i]);

}

}

transformed = cl.toBytecode();

}

} **catch** (Exception e) {

System.***err***.println("Could not instrument " + className + ", exception : " + e.getMessage());

} **finally** {

**if** (cl != **null**) {

cl.detach();

}

}

**return** transformed;

}

**private** **void** AOPInsertMethod(CtMethod method) **throws** NotFoundException, CannotCompileException {

method.instrument(**new** ExprEditor() {

**public** **void** edit(MethodCall m) **throws** CannotCompileException {

m.replace("{ long stime = System.currentTimeMillis(); $\_ = $proceed($$);System.out.println(\""

+ m.getClassName() + "." + m.getMethodName()

+ " cost:\" + (System.currentTimeMillis() - stime) + \" ms\");}");

}

});

}

}

## 3、导出jar

"d:\Program Files\java\jdk1.6.0\_45\bin\java.exe" -javaagent:agent.jar -cp target.jar;D:\workspaces\javaassist\javassist-3

.15.0-GA.jar com.jdktest.SayHello.Target

*AopAgentTest.premain() was called.*

*Adding a AopAgentTest instance to the JVM.*

*Transforming com/jdktest/SayHello/Target*

*Hello, guys!*

*java.io.PrintStream.println cost:1 ms*

*com.jdktest.SayHello.Target.sayHello cost:1 ms*

*Transforming java/lang/Shutdown*

*Transforming java/lang/Shutdown$Lock*

*java.lang.Shutdown.runHooks cost:0 ms*

*java.lang.Shutdown.sequence cost:1 ms*

## 4、附件



# 案例四、Instrument外部class文件

## 1、目标程序

**package** com.hugh.example4.target;

**public** **class** TransClass {

**public** **int** getNumber() {

**return** 1;

}

}

**package** com.hugh.example4.target;

**public** **class** TestMainInJar {

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

System.***out***.println(**new** TransClass().getNumber());

}

}

## 2、代理程序

**package** com.hugh.example4.agent;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.IOException;

**import** java.io.InputStream;

**import** java.lang.instrument.ClassFileTransformer;

**import** java.lang.instrument.IllegalClassFormatException;

**import** java.security.ProtectionDomain;

**class** Transformer **implements** ClassFileTransformer {

**public** **static** **final** String ***classNumberReturns2*** = "TransClass2.class[[3]](#footnote-3)";

@SuppressWarnings("resource")

**public** **static** **byte**[] getBytesFromFile(String fileName) {

**try** {

File file = **new** File(fileName);

InputStream is = **new** FileInputStream(file);

**long** length = file.length();

**byte**[] bytes = **new** **byte**[(**int**) length];

**int** offset = 0;

**int** numRead = 0;

**while** (offset < bytes.length && (numRead = is.read(bytes, offset, bytes.length - offset)) >= 0) {

offset += numRead;

}

**if** (offset < bytes.length) {

**throw** **new** IOException("Could not completely read file " + file.getName());

}

is.close();

**return** bytes;

} **catch** (Exception e) {

System.***out***.println("error occurs in \_ClassTransformer!" + e.getClass().getName());

**return** **null**;

}

}

**public** **byte**[] transform(ClassLoader l, String className, Class<?> c, ProtectionDomain pd, **byte**[] b)

**throws** IllegalClassFormatException {

**if** (!className.equals("com/hugh/example4/target/TransClass")) {

**return** **null**;

}

**return** *getBytesFromFile*(***classNumberReturns2***);

}

}

**package** com.hugh.example4.agent;

**import** java.lang.instrument.Instrumentation;

**import** java.lang.instrument.UnmodifiableClassException;

**public** **class** Premain {

**public** **static** **void** premain(String agentArgs, Instrumentation inst)

**throws** ClassNotFoundException, UnmodifiableClassException {

inst.addTransformer(**new** Transformer());

}

}

## 3、命令行

"d:\Program Files\java\jdk1.6.0\_45\bin\java.exe" -javaagent:agent.jar -cp target.jar com.hugh.example4.target.TestMainInJar

2

## 4、附件



# 案例五、动态Instrument

代理程序采用agentmain做为入口，对类的重定义参照案例4。

**package** com.hugh.example5.agent;

**import** java.lang.instrument.Instrumentation;

**import** java.lang.instrument.UnmodifiableClassException;

**import** com.hugh.example5.target.TransClass;

**public** **class** AgentMain {

**public** **static** **void** agentmain(String agentArgs, Instrumentation inst)

**throws** ClassNotFoundException, UnmodifiableClassException, InterruptedException {

inst.addTransformer(**new** Transformer(), **true**);

inst.retransformClasses(TransClass.**class**);

System.***out***.println("Agent Main Done");

}

}

MANIFEST.MF定义如下：

Manifest-Version: 1.0

Agent-Class: com.hugh.example5.agent.AgentMain

Can-Redefine-Classes: true

动态依附并加载代理，重定义类：

**public** **static** **void** main(String[] args) **throws** InterruptedException, AttachNotSupportedException, IOException,

AgentLoadException, AgentInitializationException {

// new AttachThread("agent.jar", VirtualMachine.list()).start();

**for** (VirtualMachineDescriptor vmd : VirtualMachine.*list*()) {

**if** (vmd.displayName().equals("com.hugh.example5.target.TestMainInJar")) {

VirtualMachine vm = VirtualMachine.*attach*(vmd);

vm.loadAgent("agent.jar");

vm.detach();

}

}

}



1. 此处代码从MyEclipse中粘贴过来，带格式粘贴要求所有代码必须展开。 [↑](#footnote-ref-1)
2. 此注解表明ToString会由虚拟机保留，以便它可在运行时通过反射读取 [↑](#footnote-ref-2)
3. 用以替换TransClass的类文件需要在原有代码TransClass上修改后，拷贝出，类名和原有名称TransClass必须保持一致 [↑](#footnote-ref-3)