Atitit 方法运行器methodRunnerV3 方法虚拟机 vm 新特性

\* v4 url cls loader v3 lib jar 2 -cp v2 ast

java -cp C:\0wkspc\methodRunner/bin;C:\0wkspc\methodRunner/WebContent/WEB-INF/lib/\* methodRunnerV4urlClsLdr -cpx "C:\0wkspc\hislog\WebContent\WEB-INF\classes;C:\0wkspc\hislog\WebContent\WEB-INF\lib\\*" -new -class com.attilax.autoup.autoupdate -method exe -paramtypes str,str,str -arguments c:\home\cnhis\cnhisShell\1.0.102.20180613.zip,utf8,c:\home\cnhis\cnhisShell

java -cp /elk/methodRunner/bin:/elk/methodRunner/WebContent/WEB-INF/lib/\* methodRunnerV4urlClsLdr -cpx "/home/cnhis/managerweb/webapps/webcon/WEB-INF/classes;/home/cnhis/managerweb/webapps/webcon/WEB-INF/lib/\*" -new -class com.attilax.autoup.autoupdate -method exe -paramtypes str,str,str -arguments \home\cnhis\cnhisShell\1.0.102.20180613.zip,utf8,/home/cnhis/cnhisShell

import java.io.File;

import java.lang.reflect.InvocationTargetException;

import java.util.List;

import org.apache.commons.cli.CommandLine;

import org.apache.commons.cli.CommandLineParser;

import org.apache.commons.cli.DefaultParser;

import org.apache.commons.cli.Options;

import org.apache.commons.cli.ParseException;

import org.apache.commons.lang3.reflect.ConstructorUtils;

import org.apache.commons.lang3.reflect.MethodUtils;

import com.alibaba.fastjson.JSON;

import com.attilax.ast.ClassInstanceCreation;

import com.attilax.ast.Expression;

import com.attilax.ast.MethodInvocation;

import com.attilax.ast.SimpleName;

import com.attilax.core.ClassLoader\_StandardExecutor;

import com.attilax.core.astExpressInterpreter;

import com.attilax.core.classLoaderUtil;

import com.attilax.parser.Token;

import com.attilax.ref.refServiceV2s530;

import com.google.common.collect.Lists;

//com.attilax.ref.methodRunnerByclassfile

/\*\*

\* v4 url cls loader v3 lib jar 2 -cp v2 ast

\*

\* @author attilax

\*

\*/

@SuppressWarnings("all")

public class methodRunnerV4urlClsLdr {

// 、、C:\0wkspc\webdavroot\bin\EmbeddedTomcat.class

// -cpx "C:\\0wkspc\\webdavroot\\bin" jardir

// "C:\\0wkspc\\webdavroot\\libtomcart" -class EmbeddedTomcat -method

// openWebdav -paramtypes 'str,int' -arguments "C:\\0wkspc" 1315

// -cpx "C:\\0wkspc\\webdavroot\\bin" -jardir

// "C:\\0wkspc\\webdavroot\\libtomcart" -class EmbeddedTomcat -method

// openWebdav -paramtypes 'str,int' -arguments "C:\\0wkspc" 1315

public static void main(String[] args) throws NoSuchMethodException, IllegalAccessException,

InvocationTargetException, ParseException, InstantiationException {

Options options = new Options();

options.addOption("cpx", true, "class folder path.");

options.addOption("jardirx", true, "jardir.");

options.addOption("class", true, "class need.");

options.addOption("method", true, "method need.");

options.addOption("paramtypes", true, "paramtypes need.");

options.addOption("arguments", true, "arguments need.");

options.addOption("new", false, "class need.");

options.addOption("conparamtypes", true, "class need.");

options.addOption("conargs", true, "class need.");

CommandLineParser parser = new DefaultParser(); // new GnuParser();

CommandLine CommandLine1 = parser.parse(options, args);

String cpx = CommandLine1.getOptionValue("cpx");

// if (cmd.hasOption("dir"))

// String

// classfile="C:\\0wkspc\\ftpserverati\\WebContent\\WEB-INF\\classes\\com\\attilax\\util\\clsNodep.class";

String clsname = CommandLine1.getOptionValue("class");

String jardir = CommandLine1.getOptionValue("jardirx");

String[] cp\_a = cpx.split(";");

ClassLoader\_StandardExecutor clsldr1 = new ClassLoader\_StandardExecutor();

for (String cpItem : cp\_a) {

String cpitem\_trim = cpItem.trim();

if (cpitem\_trim.endsWith("\*")) {

String jardir\_s = cpitem\_trim.substring(0, cpitem\_trim.length() - 2);

System.out.println(" load jar dir:" + jardir\_s);

// refServiceV2s530.loadJarFromDirtrave\_urlclassloader(jardir\_s);

clsldr1.addurl\_byJarDir(jardir\_s );

} else

clsldr1.addURL(new File(cpitem\_trim.trim()));

// refServiceV2s530.loadclass\_urlclassloader( cpitem\_trim.trim());

}

MethodInvocation mi = new MethodInvocation();

Expression miexp;

if (CommandLine1.hasOption("new")) {

ClassInstanceCreation cic = new ClassInstanceCreation();

cic.jsonname = "ClassInstanceCreation";

cic.name = clsname;

cic.arguments = get\_con\_paramsList(CommandLine1);

miexp = cic;

} else {

SimpleName sn = new SimpleName(clsname);

miexp = sn;

}

mi.Exp = miexp;

mi.Name = CommandLine1.getOptionValue("method");

mi.arguments = getParams(CommandLine1);

mi.jsonname = "MethodInvocation";

System.out.println(JSON.toJSONString(mi, true));

Object interpret\_rzt = new astExpressInterpreter().interpret(mi, clsldr1);

System.out.println("rzt:" + interpret\_rzt);

// mi.

System.out.println("--f");

}

private static List getParams(CommandLine cmd) {

List<Object> paramsList = Lists.newArrayList();

if (cmd.hasOption("arguments")) {

String arguments = cmd.getOptionValue("arguments");

String paramtypes = cmd.getOptionValue("paramtypes");

String[] a = arguments.split(",");

int n = 0;

for (String string : a) {

Object p = refServiceV2s530.getArg(string, n, paramtypes.split(","));

paramsList.add(p);

n++;

}

}

return paramsList;

}

private static List get\_con\_paramsList(CommandLine cmd) {

if (cmd.hasOption("conargs")) {

List con\_paramsList;// = get\_con\_paramsList();

return Lists.newArrayList();

} else

return Lists.newArrayList();

}

}