Atitit 动态代理 Cglib javassist对实现比较

Cglib对api更加ok一些。。。

package util;

import java.lang.reflect.Method;

import java.util.Arrays;

import javassist.util.proxy.MethodHandler;

import javassist.util.proxy.Proxy;

import javassist.util.proxy.ProxyFactory;

import net.sf.cglib.proxy.Enhancer;

import net.sf.cglib.proxy.MethodInterceptor;

import net.sf.cglib.proxy.MethodProxy;

/\*\*

\* 使用 Cglib 代理 Created by YongXin Xue on 2020/05/05 15:03 only for dyn metho

\* d,catn use in static method

\*/

public class CglibProxyUtil {

// 测试

public static void main(String[] args) throws Exception {

Object target=new ShellUtil();

ShellUtil cglibProxy = (ShellUtil) getProxyObj2(target);

cglibProxy.main(null);

cglibProxy.dynMeth("p111");

}

//javassist

private static Object getProxyObj2(Object target) throws Exception {

ProxyFactory f = new ProxyFactory();

f.setSuperclass(target.getClass());

MethodHandler mi = new MethodHandler() {

public Object invoke(

Object self, Method m, Method proceed, Object[] args)

throws Throwable {

System.out.printf("Method %s called with %s%n",

m.getName(), Arrays.toString(args));

// call the original method

return proceed.invoke(self, args);

}

};

f .setHandler(mi);

// ((Proxy) foo).setHandler(mi);

return f.createClass().newInstance();

}

private static Object getProxyObj(Object target) {

// 是 Cglib 用于创建代理对象的增强工具类

Enhancer enhancer = new Enhancer();

// Cglib需要对目标对象的Class字节码进行修改.

// Cglib产生的代理对象实例.是目标对象的子类

enhancer.setSuperclass(target.getClass());

// setCallback() 设置用于增强 操作的实现类( MethodInterceptor对代理方法进行拦截 )

enhancer.setCallback(new MethodInterceptor() {

// \* \* @param proxy Cglib代理对象实例

// \* @param methodProxy 代理方法的method代理对象

public Object intercept(Object proxy, Method method, Object[] args, MethodProxy methodProxy)

throws Throwable {

Object result = null;

try {

System.out.println(

" LogUtils\_logBefore methd:" + method.getName() + " args:" + Arrays.toString(args));

// 调用目标方法 [加 / 减 / 乘 / 除 / 或具体方法]

result = method.invoke(target, args);

// 执行增强代码

System.out.println(

" LogUtils\_logAfterReturning methd:" + method.getName() + " args:" + Arrays.toString(args));

} catch (Exception e) {

e.printStackTrace();

System.out.println(" LogUtils\_logAfterReturning methd:" + method.getName() + " args:" + e);

}

return result;

}

});

// 创建 Cglib 代理对象实例

// ShellUtil cglibProxy = (ShellUtil) ;

return enhancer.create();

}

}