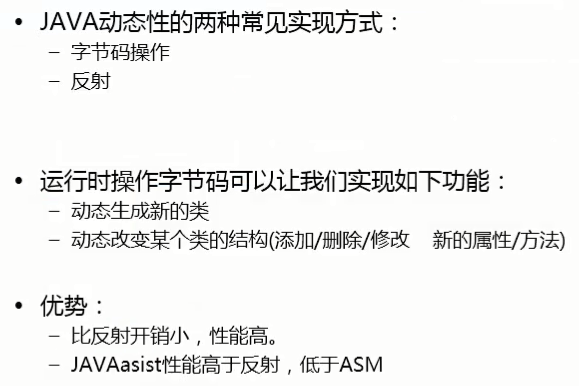
**字节码操作\_javassist库\_动态创建新类\_属性\_方法\_构造器\_API详解JAVA216-217**

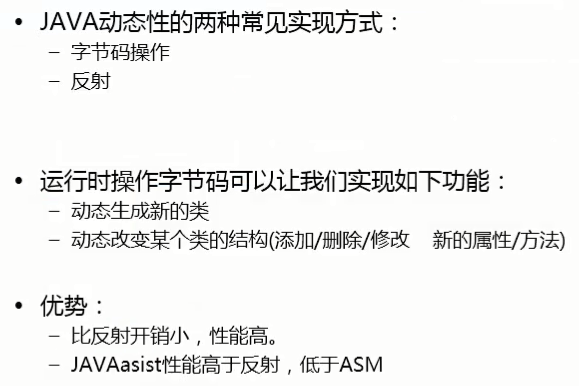


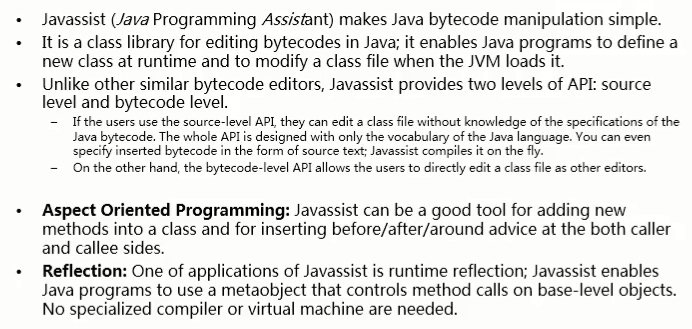
[WT-TW](https://me.csdn.net/u013292493) 2016-04-02 23:57:14  3807  收藏

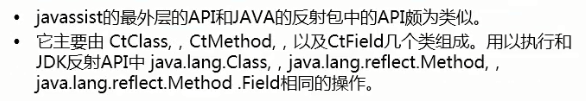
展开

来源：<http://www.bjsxt.com/>  
**一、S02E216\_01字节码操作\_javassist库、介绍、动态创建新类、属性、方法、构造器**

**字节码操作**  


**常见的字节码操作类库**  


**JAVAssist库**  




package com.test.javassist;

import javassist.CannotCompileException;

import javassist.ClassPool;

import javassist.CtClass;

import javassist.CtConstructor;

import javassist.CtField;

import javassist.CtMethod;

/\*\*

\* 测试使用javassist生成一个新的类

\*/

public class Demo01 {

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

ClassPool pool = ClassPool.getDefault();

CtClass cc = pool.makeClass("com.test.bean.Emp");

*//创建属性*

CtField f1 = CtField.make("private int empno;",cc);

CtField f2 = CtField.make("private String ename;",cc);

cc.addField(f1);

cc.addField(f2);

*//创建方法*

CtMethod m1 = CtMethod.make("public int getEmpno(){return empno;}", cc);

CtMethod m2 = CtMethod.make("public void setEmpno(){this.empno = empno;}", cc);

cc.addMethod(m1);

cc.addMethod(m2);

*//添加构造器*

CtConstructor constructor = new CtConstructor(new CtClass[]{CtClass.intType, pool.get("java.lang.String")},cc);

constructor.setBody("{this.empno = empno; this.ename = ename;}");

cc.addConstructor(constructor);

cc.writeFile("g:/java/test");*//将上面构造好的类写入到指定的工作空间中*

System.out.println("生成类，成功！");

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38

使用XJad反编译后生成的.java文件

*// Decompiled by Jad v1.5.8e2. Copyright 2001 Pavel Kouznetsov.*

*// Jad home page: http://kpdus.tripod.com/jad.html*

*// Decompiler options: packimports(3) fieldsfirst ansi space*

*// Source File Name: Emp.java*

package com.test.bean;

public class Emp

{

private int empno;

private String ename;

public int getEmpno()

{

return empno;

}

public void setEmpno()

{

empno = empno;

}

public Emp(int i, String s)

{

empno = empno;

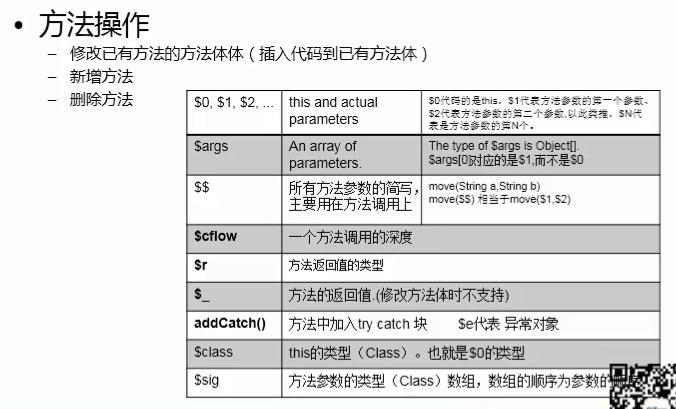
ename = ename;

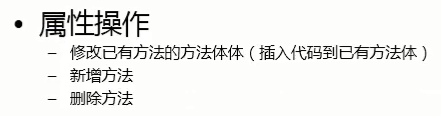
}

}

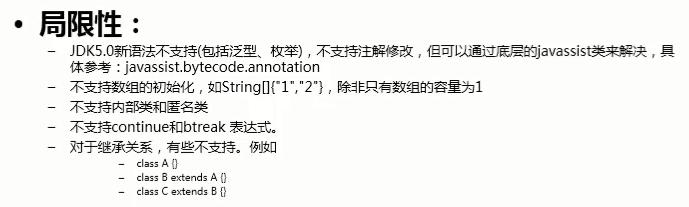
* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30

**一、S02E217\_01字节码操作*javaassist库*介绍\_API详解**









package com.test.javassist;

import java.lang.reflect.Method;

import java.util.Arrays;

import javassist.ClassPool;

import javassist.CtClass;

import javassist.CtConstructor;

import javassist.CtField;

import javassist.CtMethod;

import javassist.CtNewMethod;

import javassist.Modifier;

/\*\*

\* 测试javassist的API

\*/

public class Demo2 {

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

test06();

}

/\*\*

\* 处理类的基本用法

\* @throws Exception

\*/

public static void test01() throws Exception{

ClassPool pool = ClassPool.getDefault();

CtClass cc = pool.get("com.test.javassist.Emp");*//获取已有的类*

byte[] bytes = cc.toBytecode();

System.out.println(Arrays.toString(bytes));

System.out.println(cc.getName());*//获取类名*

System.out.println(cc.getSimpleName());*//获取简要类名*

System.out.println(cc.getSuperclass());*//获取父类*

System.out.println(cc.getInterfaces());*//获取接口*

}

/\*\*

\* 测试产生新的方法

\*/

public static void test02() throws Exception{

ClassPool pool = ClassPool.getDefault();

CtClass cc = pool.get("com.test.javassist.Emp");*//获取已有的类*

*//CtMethod m = CtNewMethod.make("public int add(int a,int b){return a+b;}", cc);*

*//另一种方式，参数：返回类型，方法名，可变参数类型，CtClass对象*

CtMethod m = new CtMethod(CtClass.intType, "add",

new CtClass[]{CtClass.intType,CtClass.intType}, cc);

m.setModifiers(javassist.Modifier.PUBLIC);

m.setBody("{System.out.println(\"返回方法体中的打印信息\");return $1+$2;}");

cc.addMethod(m);

*//通过反射调用新生成的方法*

Class clazz = cc.toClass();

Object obj = clazz.newInstance();*//调用Emp无参构造器，创建新的Emp对象*

Method method = clazz.getDeclaredMethod("add", int.class,int.class);

Object result = method.invoke(obj, 200,300);

System.out.println(result);

}

/\*\*

\* 修改已有方法的方法体内容

\* @throws Exception

\*/

public static void test03() throws Exception{

ClassPool pool = ClassPool.getDefault();

CtClass cc = pool.get("com.test.javassist.Emp");*//获取已有的类*

CtMethod cm = cc.getDeclaredMethod("sayHello", new CtClass[]{CtClass.intType});

cm.insertBefore("System.out.println($1);System.out.println(\"start!!!\");");

cm.insertAt(9, "int b=3;System.out.println(\"b=\"+b);");

cm.insertAfter("System.out.println(\"end!!!\");");

*//通过反射调用新生成的方法*

Class clazz = cc.toClass();

Object obj = clazz.newInstance();*//调用Emp无参构造器，创建新的Emp对象*

Method method = clazz.getDeclaredMethod("sayHello", int.class);

method.invoke(obj, 300);

}

/\*\*

\* 测试产生新的属性

\*/

public static void test04() throws Exception{

ClassPool pool = ClassPool.getDefault();

CtClass cc = pool.get("com.test.javassist.Emp");*//获取已有的类*

*//CtField f1 = CtField.make("private int salary;", cc);*

*//另一种方式*

CtField f1 = new CtField(CtClass.intType, "salary",cc);

f1.setModifiers(Modifier.PRIVATE);

cc.addField(f1);

*//cc.getDeclaredField("salary");//获取指定的属性*

*//另一种方式*

cc.addMethod(CtNewMethod.getter("getSalary", f1));

cc.addMethod(CtNewMethod.getter("setSalary", f1));

*//通过反射调用，省略。。。*

}

/\*\*

\* 测试构造器

\*/

public static void test05() throws Exception{

ClassPool pool = ClassPool.getDefault();

CtClass cc = pool.get("com.test.javassist.Emp");*//获取已有的类*

CtConstructor[] cs = cc.getConstructors();

for (CtConstructor c : cs) {

System.out.println(c.getLongName());

}

}

/\*\*

\* 测试注解

\*/

public static void test06() throws Exception{

ClassPool pool = ClassPool.getDefault();

CtClass cc = pool.get("com.test.javassist.Emp");*//获取已有的类*

Object[] all = cc.getAnnotations();

Author a = (Author) all[0];

String name = a.name();

int year = a.year();

System.out.println("name:" + name + ",year:" + year);

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 59
* 60
* 61
* 62
* 63
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73
* 74
* 75
* 76
* 77
* 78
* 79
* 80
* 81
* 82
* 83
* 84
* 85
* 86
* 87
* 88
* 89
* 90
* 91
* 92
* 93
* 94
* 95
* 96
* 97
* 98
* 99
* 100
* 101
* 102
* 103
* 104
* 105
* 106
* 107
* 108
* 109
* 110
* 111
* 112
* 113
* 114
* 115
* 116
* 117
* 118
* 119
* 120
* 121
* 122
* 123

javabean

package com.test.javassist;

@Author(name = "test",year= 2016)

public class Emp {

private int empNo;

private String empName;

public Emp() {

}

public Emp(int empNo, String empName) {

super();

this.empNo = empNo;

this.empName = empName;

}

public int getEmpNo() {

return empNo;

}

public void setEmpNo(int empNo) {

this.empNo = empNo;

}

public String getEmpName() {

return empName;

}

public void setEmpName(String empName) {

this.empName = empName;

}

public void sayHello(int a){

System.out.println("sayHello," + a);

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32

注解

package com.test.javassist;

public @interface Author {

String name();

int year();

}

* 1
* 2
* 3
* 4
* 5
* 6