@Retention(RetentionPolicy.RUNTIME)

@Target({ElementType.TYPE})

public @interface Skip {}

@Skip

public class RWCounter {

private HashMap<String, int[]> \_\_rwCounters = .. .;

public void incRead(String key) { .. .. .. }

public void incWrite(String key) { .. .. .. }

public void putIfAbsent(String key) { .. .. .. }

public void printProfiles() { .. .. .. }

}

public class Main {

public static void main(String[] args) {

.. .. ..

}

}

class Main {

public static RWCounter \_\_rwCounters = new RWCounter();

public static void main(String[] args) {

.. .. ..

{ \_\_rwCounters.printProfiles(); }

}

}

public class ProfilerTranslator implements Translator {

private String mainClassName;

public ProfilerTranslator(String mainClassName) {

this.mainClassName = mainClassName;

}

public void start(ClassPool pool) throws .. {

CtClass mainClass = pool.get(mainClassName);

mainClass.addField(

CtField.make(

"public static RWCounter \_\_rwCounters = RWCounter();"

, mainClass));

mainClass

.getDeclaredMethod("main")

.insertAfter(" { \_\_rwCounters.printProfiles(); } ");

}

.. .. .. ..

}

public void onLoad(ClassPool pool, String className) throws .. .. {

CtClass ctClass = pool.get(className);

try {

if(ctClass.isInterface()) return;

if(ctClass.hasAnnotation(Skip.class)) return;

profile(pool, ctClass, ctClass.getDeclaredConstructors());

profile(pool, ctClass, ctClass.getDeclaredMethods());

} catch (ClassNotFoundException e) {

throw new RuntimeException(e);

}

}

String className = ctClass.getName();

addIfNotPresent(ctClass, "\_\_rwCounter");

for(CtBehavior ctBehavior : ctBehaviors) {

ctBehavior.instrument(new ExprEditor() {

public void edit(FieldAccess fa) .. .. {

.. .. ..

}

});

if(ctBehavior instanceof CtConstructor)

ctBehavior.insertAfter(String.format(

" { %s.\_\_rwCounters.putIfAbsent(\"%s\", \_\_rwCounter); } "

, mainClassName, className));

}

public void edit(FieldAccess fa) throws CannotCompileException {

if(fa.isStatic()) return;

String fieldClassName = fa.getClassName();

CtClass fieldCtClass = get(pool, fieldClassName);

addIfNotPresent(fieldCtClass, "\_\_rwCounter");

if(fa.isReader())

fa.replace(String.format(

" { $\_ = $proceed($$); %s.\_\_rwCounter[0] += 1; } ", fieldClassName));

if(!(ctBehavior instanceof CtConstructor

&& fieldClassName.equals(className))

&& fa.isWriter())

fa.replace(String.format(

" { $\_ = $proceed($$); %s.\_\_rwCounter[1] += 1; } ", fieldClassName));

}