```java
@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() { .. .. .. }
}
```

```java
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(); } ");
    }


    .. .. .. ..
}
```

```java
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);
        }
    }
```

```java
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));
}
```