Как я заново открыл для себя инкапсуляцию в java.

* [Java](https://habr.com/hub/java/)

Я всегда считал, что Java — лаконичный и красивый (в плане концепции) язык с четкой структурой, позволяющей расширять эту структуру и на всевозможные фреймворки, тем самым помогающая привнести порядок и в код конечного программиста. И, прежде всего, я считал, что java — это 100% ОО язык! Но недавно мне попался код, после которого я вечер ходил возмущался. Код совершенно несложный для понимания даже людей несведующий в java.  
Допустим у нас есть класс А в некотором пакете:

package test;

public class A {

private int x = 33;

public void test(){

System.out.println(x);

}

}

Как видно поле x является защищенным и, согласно [доке от sun](http://java.sun.com/docs/books/tutorial/java/javaOO/accesscontrol.html), это поле не видимо ни в подклассах А, ни в пакете test, ни тем более в других пакетах. Поэтому у нас есть все основания считатать, что метод test() возвращает всегда число 33. Более того, мы бы могли использовать это предположение в своем коде, завязав на этом какую-либо логику.   
Далее, имеем класс в любом другом пакете (но в пределах этой же JVM):

package access;

public class X {

public void getAccess() throws Exception{

A a = new A();

Field x = a.getClass().getDeclaredField("x");

x.setAccessible(true);

x.setInt(a, x.getInt(a)+1);

System.out.print("protected field: ");

a.test();

}

}

Вот и все. Все наши предположения о инкапсуляции и неизменности поля А.х развеяны. Метод А.test() теперь возвращает число 34 (и, если мы завязали на этом какую-нибудь логику, то наш код вряд ли теперь будет работать корректно). И даже может показаться, что нарушена одна из трех заповедей ООП — инкапсуляция. Во всем виновен ~~программист~~ пакет [java.lang.reflect](http://java.sun.com/docs/books/tutorial/reflect/index.html). Этот пакет предоставляет возможности для работы с объектами на этапе выполнения программы. Т.к. практически все классы этого объекта работают с динамическими структурами, то он довольно медленно работает и должен использоваться лишь в исключительных случаях. Однако возможность его применения остается и нарушение инкапсуляции налицо.   
  
Впоследствии, покапавшись с этим пакетом, я нашел, что за соблюдение безопасности и доступа отвечает специальный класс SecurityManager и, если его запустить, то все становится на свои места. Если нашему классу А добавить статический инициализатор:

public class A {

static{

if(System.getSecurityManager()==null)

System.setSecurityManager(new SecurityManager());

}

То финт ушам, какой получился в предыдущем примере, уже не пройдет. Т.ч. все в порядке, Java по-прежнему ОО язык :) Единственное, для чего я написал этот пост — ***предупредить*** (это не сигнал к действию, просто нужно это тоже иметь в виду) всех явистов:  
  
*РАЗРАБОТЧИК, БДИ!  
При отсутствии SecurityManager на этапе выполнения в твой класс может залезть кто-угодно и изменить что-угодно!*

Комментарии 68

* [**Digit**](https://habr.com/users/Digit/)3 ноября 2008 в 15:51

**+14**

Достаточно известная фишка java reflection, странно что вы о ней не знали.  
Статья хороша как напоминание, но её можно сделать лучше, если добавить описание класса SecurityManager и рассказать почему подобное поведение не является нарушением java security model.  
Хотя, конечно, кто захочет — найдёт всё в гугле.

* [[https://habrastorage.org/getpro/habr/avatars/721/f3c/0cf/721f3c0cff1484385bc33067963065a6_small.jpg](https://habr.com/users/YasonBy/)**YasonBy**](https://habr.com/users/YasonBy/)3 ноября 2008 в 16:13

**+6**

Java позволит вам выстрелить себе в ногу, если вы произнесёте заклинание «import java.lang.reflect» :)

* + [[https://habrastorage.org/getpro/habr/avatars/ddd/ed3/93b/ddded393b6e7de1880279795856fa521_small.jpg](https://habr.com/users/iZENfire/)**iZENfire**](https://habr.com/users/iZENfire/)3 ноября 2008 в 16:34

**+1**

Не всегда. Нужна ещё «правильная» политика. ;)

* [[https://habrastorage.org/getpro/habr/avatars/82b/849/b7a/82b849b7a3d5191977650f1a1ac18dc0_small.jpg](https://habr.com/users/Zordhauer/)**Zordhauer**](https://habr.com/users/Zordhauer/)3 ноября 2008 в 16:21

**+1**

Обойти защиту вроде можно было всегда и везде) нельзя все предусмотреть, а иногда абсолютной защите мешает наличие противоречий (т.е. при организации такой защиты нельзя не пожертвовать функциональностью)

* [[https://habrastorage.org/getpro/habr/avatars/ddd/ed3/93b/ddded393b6e7de1880279795856fa521_small.jpg](https://habr.com/users/iZENfire/)**iZENfire**](https://habr.com/users/iZENfire/)3 ноября 2008 в 16:29

**+13**

*/path/to/demo.policy*:

grant {

permission java.lang.reflect.ReflectPermission "suppressAccessChecks";

};

*test.A.java*:

package test;

public class A {

static {

if (System.getSecurityManager() == null)

System.setSecurityManager(new SecurityManager());

}

private int x = 33;

public void test() {

System.out.println(x);

}

}

*access.X.java*:

package access;

import java.lang.reflect.Field;

import test.A;

public class X {

public void getAccess() throws Exception {

A a = new A();

Field x = a.getClass().getDeclaredField("x");

x.setAccessible(true);

x.setInt(a, x.getInt(a) + 1);

System.out.print("private field: ");

a.test();

}

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

new X().getAccess();

}

}

> java access.X  
Exception in thread «main» java.security.AccessControlException: access denied (java.lang.reflect.ReflectPermission suppressAccessChecks)  
at java.security.AccessControlContext.checkPermission(AccessControlContext.java:323)  
at java.security.AccessController.checkPermission(AccessController.java:546)  
at java.lang.SecurityManager.checkPermission(SecurityManager.java:532)  
at java.lang.reflect.AccessibleObject.setAccessible(AccessibleObject.java:107)  
at access.X.getAccess(X.java:10)  
at access.X.main(X.java:17)  
  
> java -Djava.security.policy=/path/to/demo.policy access.X  
private field: 34

* [[https://habrastorage.org/getpro/habr/avatars/83d/31d/ec4/83d31dec409f7cbb7d7a80be8db2a7bd_small.png](https://habr.com/users/Q2W/)**Q2W**](https://habr.com/users/Q2W/)3 ноября 2008 в 16:34

**–4**

Ппц, сначала какие-то костыли, чтоб разрушить стандарт, потом костыли на костыли, чтоб он всё-таки остался стандартом. Мдя уж.

* + [**nikolaikopernik**](https://habr.com/users/nikolaikopernik/)3 ноября 2008 в 16:44

**+1**

смотри пред пост iZENfire — это уже костыли на костыли на костыли :)

* + [**Calvrack**](https://habr.com/users/Calvrack/)3 ноября 2008 в 20:23

**+2**

Посмотрите на это под другим углом, рефлекшон в этом смысле можно воспринимать как работу с кодом — нечто эквивалентное динмаически разобрать classfile, подправить и собрать, и залоадить. Тоесть эта «фича» будет в любом языке с динамической загрузкой.

* [**corp**](https://habr.com/users/corp/)3 ноября 2008 в 16:39

**+8**

эмм, что то мне подсказывает, что любой java/.net-разработчик знает(должен знать) что такое рефлекшн и как его (не)использовать. а за вас конечно рад — никогда не поздно узнавать что-то новое :)  
  
По этому примеру — можно запускать jvm с SecurityManager, и без всякого доп. кода она будет стучать по рукам при попытке сделать setAccessible(true), или прописать нормальные security policy для jvm(в большинстве случаев так и будет).

* [[https://habrastorage.org/getpro/habr/avatars/895/06f/a8f/89506fa8f9b17d23882295328339b8a0_small.png](https://habr.com/users/lam0x86/)**lam0x86**](https://habr.com/users/lam0x86/)3 ноября 2008 в 18:09

**0**

В Java Persistance API доступ к скрытым полям entity-классов активно используется. Это полезно с точки зрения производительности. Чем дёргать каждый раз методы-аксессоры, проще работать с полями напрямую. На сколько я знаю, .NET таких вещей не позволяет.

* + [**corp**](https://habr.com/users/corp/)3 ноября 2008 в 18:52

**+2**

хм,   
FieldInfo fieldInfo = typeof(Test).GetField(«xProperty», BindingFlags.Instance | BindingFlags.NonPublic);  
fieldInfo.SetValue(test, 48);  
вполне работает(xProperty — private), да и вообще System.Reflection сильно похож на java.lang.reflect   
(поправьте если ошибаюсь, с .net по работе не сталкиваюсь)

* + - [[https://habrastorage.org/getpro/habr/avatars/895/06f/a8f/89506fa8f9b17d23882295328339b8a0_small.png](https://habr.com/users/lam0x86/)**lam0x86**](https://habr.com/users/lam0x86/)3 ноября 2008 в 18:59

**+1**

Действительно работает. Извиняюсь =) Тоже не дотнетчик…

* + НЛО прилетело и опубликовало эту надпись здесь
* [[https://habrastorage.org/getpro/habr/avatars/8f2/379/bcd/8f2379bcd48dd074310362d89b52a1da_small.jpg](https://habr.com/users/ultimate_darkness/)**ultimate\_darkness**](https://habr.com/users/ultimate_darkness/)3 ноября 2008 в 19:49

**0**

Есть еще один замечательный способ отстрелить себе ногу — класс Unsafe, который позволяет напрямую работать со структурами в памяти.

* [[https://habrastorage.org/getpro/habr/avatars/39b/709/429/39b7094298337f54c9f0f09002427bc4_small.jpg](https://habr.com/users/kolupaeva/)**kolupaeva**](https://habr.com/users/kolupaeva/)3 ноября 2008 в 19:58

**+2**

>РАЗРАБОТЧИК, БДИ!  
>При отсутствии SecurityManager на этапе выполнения в твой класс может залезть кто-угодно и изменить что-угодно!  
Я надеюсь это не руководство к действию?  
  
У страха глаза велики. Не нужно *так* защищать свой код без веской на то причины.  
  
\* Не всегда наследование хорошо спроектировано. Представьте, что вы переопределяете в своем классе метод родителя, использующий несколько приватных методов, без которых вам при переопределении не обойтись. Варианта два: или дублировать код (если есть исходники, а есть они не всегда), или использовать рефлекшн. Допустим, исходников нет, а предусмотрительный автор кода включил СекьюритиМенеджер. Ваши пожелания автору? :)  
  
\* Волшебные фреймворк Seam активно использует рефлекшн для реализации своей dependency bijection (если у поля есть аксессоры, то используются они, иначе — рефлекшн). Что будет с вашим Seam-приложением в после включения СекьюритиМенеджера?  
  
Мораль: подумайте дважды, прежде чем бдеть

* + [[https://habrastorage.org/getpro/habr/avatars/00e/d8b/37e/00ed8b37ebaffa09bae347ed0304496f_small.jpg](https://habr.com/users/Murkt/)**Murkt**](https://habr.com/users/Murkt/)3 ноября 2008 в 21:02

**–1**

Включение SecurityManager'а напоминает всякие методы мега-защиты дельфовских программ, которые применяют 14-летние школьники — «А вдруг кто-то сможет влезть в мой код?!!».

* [**egorinsk**](https://habr.com/users/egorinsk/)3 ноября 2008 в 20:09

**–10**

Мда, слава богу, я не программирую на Яве) Всегда не любил этот язык, и как пользователь, и как программист, выясняется — не зря. Нафик в якобы правильном ООП языке такие извраты — не понять.

* + [**Scala**](https://habr.com/users/Scala/)3 ноября 2008 в 21:05

**+1**

Вам — «не понять», а другим оно нужно. В PHP который у вас написан в профиле это тоже есть — [ru2.php.net/oop5.reflection](http://ru2.php.net/oop5.reflection)

* + - [**egorinsk**](https://habr.com/users/egorinsk/)3 ноября 2008 в 21:17

**+3**

Блин, мне стыдно (( Я догадываюсь, чот он полезн для отладки? Полез в Гугл искать «зачем нужен рефлекшн»)

* + - * [[https://habrastorage.org/getpro/habr/avatars/66d/b1d/00a/66db1d00a985fbf66c10a09b55be55f8_small.jpg](https://habr.com/users/oleganza/)**oleganza**](https://habr.com/users/oleganza/)3 ноября 2008 в 22:06

**0**

Не только для отладки. А вообще для метапрограммирования, для генерации кода. Посмотрите на API ActiveRecord, DataMapper — отличные примеры грамотной (в меру :-) генерации кода.

* [**henu3detb**](https://habr.com/users/henu3detb/)3 ноября 2008 в 20:33

**+9**

Мдааа, выдержит ли у автора сердце если он откроет для себя манипулирование байт-кдом в рантайме?

* + [**nikolaikopernik**](https://habr.com/users/nikolaikopernik/)3 ноября 2008 в 22:05

**0**

я его время от времени тренирую на подобных вот примерах :)

* + [[https://habrastorage.org/getpro/habr/avatars/04f/385/143/04f385143eacf606d27b850d88ea5461_small.jpg](https://habr.com/users/Goodkat/)**Goodkat**](https://habr.com/users/Goodkat/)4 ноября 2008 в 01:50

**0**

не, ну это можно было ещё на Ассемблере делать

* [**DbLogs**](https://habr.com/users/DbLogs/)3 ноября 2008 в 21:01

**+1**

Известная фича. Очень часто используется при написании тесткейсов. Проблем, если честно, никаких не вижу, так как если в рантайме только твой код, то ничего страшного, а если подключается чужой, то самоубийц не включающих SecurityManager я не видел:)

* [**blezalex**](https://habr.com/users/blezalex/)3 ноября 2008 в 21:28

**+6**

Инкапсуляция это прежде всего **сокрытие** а не **защита**.  
  
Инкапсуляция позволяет правильно работать с объектом, если же хочется надругаться над классом — reflaction в помощь, но при этом никто не гарантирует что вы получите правильный результат, кроме того, класс вовсе не обязан обрабатывать такие ситуации

* + [[https://habrastorage.org/getpro/habr/avatars/06c/dd9/500/06cdd950054947954c29f7eb8e3ce969_small.png](https://habr.com/users/mace/)**mace**](https://habr.com/users/mace/)3 ноября 2008 в 23:25

**0**

Именно так. Программисту дают возможность сделать это, а воспользуется ли он ей — это его дело. Понятно, что перед использованием рефлекшена для такой цели нужно несколько раз подумать и быть готовым к возможным багам. «Правильный» язык должен дать возможность сделать все, что угодно, но при этом при написании кода нужно понимать, что делаешь. К сожалению, пока что никто не придумал язык, на котором мог бы успешно писать совершенно безмозглый человек. А если новичек решил все же залезть рефлекшеном в приватные поля, то нужно ему обьяснить, почему так делать плохо и разобратся, а вдруг использование этого инструмента в даном случае действительно оправдано?

* + [**shuron**](https://habr.com/users/shuron/)20 марта 2009 в 20:23

**0**

Наконец-то кто-то обратил на это внимание!  
Абсолютно верно!

* [[https://habrastorage.org/getpro/habr/avatars/66d/b1d/00a/66db1d00a985fbf66c10a09b55be55f8_small.jpg](https://habr.com/users/oleganza/)**oleganza**](https://habr.com/users/oleganza/)3 ноября 2008 в 21:29

**+2**

Подскажите, а так скрывать поля нужно по соображениям заповедей или практической необходимости? Хакиры меняют вам значения констант?

* + [**egorinsk**](https://habr.com/users/egorinsk/)3 ноября 2008 в 21:56

**0**

Скрывать надо чтоб у недопрограмеров не было соблазна обращаться к полям объектов напрямую, проще скрыть чем объяснять каждому все паттерны программирования.

* + - [[https://habrastorage.org/getpro/habr/avatars/66d/b1d/00a/66db1d00a985fbf66c10a09b55be55f8_small.jpg](https://habr.com/users/oleganza/)**oleganza**](https://habr.com/users/oleganza/)3 ноября 2008 в 22:04

**+2**

т.е. недопрограммеры так и норовят загрузить util.reflect и залезть в private-поля?

* + - * [[https://habrastorage.org/getpro/habr/avatars/ddd/ed3/93b/ddded393b6e7de1880279795856fa521_small.jpg](https://habr.com/users/iZENfire/)**iZENfire**](https://habr.com/users/iZENfire/)4 ноября 2008 в 14:34

**0**

Скорее да, чем если бы они хорошенько изучили предоставленный разработчиком «закрытого» класса интерфейс и воспользовались его функциональностью правильно.

* + - * + [[https://habrastorage.org/getpro/habr/avatars/66d/b1d/00a/66db1d00a985fbf66c10a09b55be55f8_small.jpg](https://habr.com/users/oleganza/)**oleganza**](https://habr.com/users/oleganza/)4 ноября 2008 в 15:00

**0**

Иными словами, вы относитесь к своим программерам почти как к внешним хакерам, а не к товарищам по команде? Жестко у вас там.

[**nikolaikopernik**](https://habr.com/users/nikolaikopernik/)4 ноября 2008 в 15:11

**+1**

Подождите, вы ведь сейчас говорите об инкапсуляции? При чем тут товарищи или хакеры? Инкапсуляция — это один из базисов ООП. Это подход общий для всех. Его сила — в том, чтобы ВСЕ без исключения его соблюдали (и дело совершенно не в товарищеских настроениях или личных мотивах). Представте, что будет, если отметить инкапсуляцию «Вась, а сделай лично для меня вот в этом классе такую-то переменную — мне так легче, а выводи все свои результаты вот в таком виде — так легче будет Костяну. А для Ирочки заведи еще переменную i\_love\_you ..» Детский сад не устраивайте. Разработка — это разработка, и к личным предпочтениям дела не имеет.

[[https://habrastorage.org/getpro/habr/avatars/66d/b1d/00a/66db1d00a985fbf66c10a09b55be55f8_small.jpg](https://habr.com/users/oleganza/)**oleganza**](https://habr.com/users/oleganza/)4 ноября 2008 в 15:25

**0**

Я имею в виду, что предлагается писать System.setSecurityManager(new SecurityManager()) чтобы не дать Васе завести свою переменную через java.util.reflect. Возникает вопрос: почему бы Васю не уволить, или научить, что делать a.getClass().getDeclaredField(«x») не следует. Или, что лучше, поинтересоваться, по какой такой причине ему нужно залезть в класс именно так. Мож, он баг какой так хитро патчит.  
К инкапсуляции и ооп мое любопытство не имеет прямого отношения.

[**nikolaikopernik**](https://habr.com/users/nikolaikopernik/)4 ноября 2008 в 15:32

**0**

>>Подскажите, а так скрывать поля нужно по соображениям заповедей или практической необходимости?<<  
— из соображений инкапсуляции.  
  
>>Я имею в виду, что предлагается писать System.setSecurityManager(new SecurityManager()) <<  
— не предлагается, а напоминается, что есть такая весчь и это нужно иметь в виду.  
  
спор ниочем.

[[https://habrastorage.org/getpro/habr/avatars/66d/b1d/00a/66db1d00a985fbf66c10a09b55be55f8_small.jpg](https://habr.com/users/oleganza/)**oleganza**](https://habr.com/users/oleganza/)4 ноября 2008 в 15:42

**0**

Тут нет спора. Товарищ мне ответил на «т.е. недопрограммеры так и норовят загрузить util.reflect и залезть в private-поля?» тем, что да, норовят. Мне становится интересно, где же водятся такие страшные программеры. Соображения инкапсуляции тут ни при чем. И спора тоже нет. Раз есть суровые челябинские программеры, нужно защищаться, конечно. Просто это как-то сюрреалистично смотрится.

[[https://habrastorage.org/getpro/habr/avatars/ddd/ed3/93b/ddded393b6e7de1880279795856fa521_small.jpg](https://habr.com/users/iZENfire/)**iZENfire**](https://habr.com/users/iZENfire/)4 ноября 2008 в 15:41

**0**

У меня нет команды. Но направление мысли правильное — инкапсуляция подразумевает приемлемую для всех защиту кода.  
  
Если надо написать более защищённый код, обусловленный политикой компании, то используется специально разработанный для этого SecurityManager.  
Если надобность в сильной защите доступа не важна, то используется обычный механизм сокрытия полей и методов, предоставляемый языком.  
  
Язык предоставляет право выбора механизма и политики, а так же право смешивать их.

[[https://habrastorage.org/getpro/habr/avatars/66d/b1d/00a/66db1d00a985fbf66c10a09b55be55f8_small.jpg](https://habr.com/users/oleganza/)**oleganza**](https://habr.com/users/oleganza/)4 ноября 2008 в 15:44

**0**

>> более защищённый код, обусловленный политикой компании <<  
  
Вот это мне и интересно. Чем руководствуются авторы политики защиты private-переменных от инсайдеров. Наличием злобных инсайдеров?  
  
>> Язык предоставляет право выбора механизма и политики, а так же право смешивать их. <<  
  
разумно

* + - [[https://habrastorage.org/getpro/habr/avatars/66d/b1d/00a/66db1d00a985fbf66c10a09b55be55f8_small.jpg](https://habr.com/users/oleganza/)**oleganza**](https://habr.com/users/oleganza/)4 ноября 2008 в 15:02

**0**

Уважаемое хабрачмо, хватит тут минусовать коллегу! Если идет дискуссия, есть два мнения. А минусовать нужно за спам и глупости. Опускание оппонента в \_дискуссии\_ не идет на пользу никому.

* [[https://habrastorage.org/getpro/habr/avatars/e74/153/f3e/e74153f3ef3838ca9f7c0fb59afa3984_small.jpg](https://habr.com/users/gothy/)**gothy**](https://habr.com/users/gothy/)3 ноября 2008 в 21:31

**+4**

как эта «новость» попала на главную? о\_О

* [[https://habrastorage.org/getpro/habr/avatars/43e/bea/6d7/43ebea6d7f62292f6cf6a332b95ced64_small.jpg](https://habr.com/users/malkolm/)**malkolm**](https://habr.com/users/malkolm/)3 ноября 2008 в 21:54

**0**

Это азбука. Стыдно вообще то такое писать. Разработчик БДИ… БДИ… Прежде чем защищаться, надо сначала научиться писать то, что стоит защищать.

* [**srez**](https://habr.com/users/srez/)3 ноября 2008 в 22:56

**0**

Сколько вам открытий чудных предстоит. Есть еще класс Unsafe, он и не такое позволяет.

* [[https://habrastorage.org/getpro/habr/avatars/7d6/3e2/18b/7d63e218b2e05a19d378d1a30ffbf023_small.png](https://habr.com/users/umputun/)**umputun**](https://habr.com/users/umputun/)3 ноября 2008 в 23:09

**+2**

Ваш пример гораздо уязвимей при использования ординарного наследованния. Т.е. гораздо вероятнее сценарий предопределения test(). Мне кажется, до того как защищаться от умышленных и маловероятных на практике атак через рефелкшин, стоит принять элементарные меры, типа запрещение расширения класса и final на неизменяемые переменные.

* + [**nikolaikopernik**](https://habr.com/users/nikolaikopernik/)3 ноября 2008 в 23:27

**–2**

какие атаки, при чем тут защищаться? Я говорю об подходе к программированию (который в народе именуется ООП) и одним из его «китов» — инкапсуляция, и о том, что надеятся на инкапсуляцию без дополнительных телодвижений не стоит.

* + - [[https://habrastorage.org/getpro/habr/avatars/7d6/3e2/18b/7d63e218b2e05a19d378d1a30ffbf023_small.png](https://habr.com/users/umputun/)**umputun**](https://habr.com/users/umputun/)3 ноября 2008 в 23:32

**+1**

этот кит уязвим по определению. Без всяких технических изысков, без рефлешена. Надеятся на мистику не надо, но надо понимать ограничения которые накладывает парадигма.  
  
Что касается атаки, то это такой термин, часто применяющийся в литературе при описании подбных вещей. К хакерам он отншение не имеет.

* [[https://habrastorage.org/getpro/habr/avatars/c25/697/6a9/c256976a9deabface17ba02c30ef62d8_small.jpg](https://habr.com/users/Regis/)**Regis**](https://habr.com/users/Regis/)3 ноября 2008 в 23:57

**0**

Reflection — очень полезный инструмент. Используется во многих библиотеках, связанных с обработкой метаданных (Hibernate например использует рефлекш по полной). Но важно понимать, что это всего лишь инструмент. Когда вы используете этот инструмент — вы должны представлять возможные последствия (вроде нарушения инкапсуляции). Сетовать на то, что инструмент де плох — так это уже проблема «плохого танцора»: вы с тем же успехом можете «выстрелить себе в ногу» и говорить, что пистолет вам дали плохой.  
  
Что касается «защиты» с помощью SecurityManager — то это глупость. С одной стороны, масса библиотек просто перестанет работать, а с другой стороны, если кому-то *надо* будет залезть в каш класс, то он все равно залезеть, хотя бы путём подмены байткода, включающего SecurityManager.

* + [[https://habrastorage.org/getpro/habr/avatars/c25/697/6a9/c256976a9deabface17ba02c30ef62d8_small.jpg](https://habr.com/users/Regis/)**Regis**](https://habr.com/users/Regis/)3 ноября 2008 в 23:58

**0**

\* рефлекшн  
\* все равно сможет залезть

* + [[https://habrastorage.org/getpro/habr/avatars/291/cbc/fba/291cbcfbaa68b613af0fffd1f79b98de_small.jpg](https://habr.com/users/Alien/)**Alien**](https://habr.com/users/Alien/)4 ноября 2008 в 01:22

**+1**

Кажется при проектировании Java разработчики стремились создать как можно более простой и надежный язык… это выкинем, это скроем… будет меньше ошибок. Мои знакомые программисты на Java говорят, что программы на C++ подвержены ошибкам и вообще C++ очень сложный. А я всегда говорил:

Сетовать на то, что инструмент де плох — так это уже проблема «плохого танцора»: вы с тем же успехом можете «выстрелить себе в ногу» и говорить, что пистолет вам дали плохой.

:)

* + - [[https://habrastorage.org/getpro/habr/avatars/c25/697/6a9/c256976a9deabface17ba02c30ef62d8_small.jpg](https://habr.com/users/Regis/)**Regis**](https://habr.com/users/Regis/)4 ноября 2008 в 03:06

**0**

Боюсь склонен согласиться с вашими знакомыми. C++ не то чтобы сложен — просто есть набор дополнительных (по сравнению с Java) нюансов, которые всё время нужно держать в голове. С другой стороны в Java есть эти самые отражения и несколько своих нюансов. Но их-то обычно как раз-таки не нужно учитывать.   
  
Рефлекшн не нарушает инкапсуляция в том смысле, что они не нарушают сокрытие реализации. Автору не нравится то, что рефлекшн позволяет получить доступ к данным в private-поле. Но ведь и в том же C++ при желании можно через указатель на объект добраться до приватных полей объекта. Часто это учитывают? Да почти никогда, за исключением кода, предполагающего обфускацию (проверялки леценций, активации и т.п.).  
  
Имхо, автор предлагает решение для несуществующей проблемы.

* + - * [**nikolaikopernik**](https://habr.com/users/nikolaikopernik/)4 ноября 2008 в 12:35

**+1**

давайте не будем устраивать холиваров. Есть С++ и есть Java — и они разные и предусмотрены для разных вещей.

* + - [[https://habrastorage.org/getpro/habr/avatars/c25/697/6a9/c256976a9deabface17ba02c30ef62d8_small.jpg](https://habr.com/users/Regis/)**Regis**](https://habr.com/users/Regis/)4 ноября 2008 в 03:07

**0**

И вот [еще](http://habrahabr.ru/blogs/java/43859/#comment_1093476) :)

* [[https://habrastorage.org/getpro/habr/avatars/04f/385/143/04f385143eacf606d27b850d88ea5461_small.jpg](https://habr.com/users/Goodkat/)**Goodkat**](https://habr.com/users/Goodkat/)4 ноября 2008 в 01:55

**0**

это вы ещё не программировали на ABAP :)  
там загрузить в память нужную чужую программу, изменить в ней приватные поля, сгенерировать кусок кода, выполнить, извлечь данные из приватных переменных, писать данные в таблицы доступные только для чтения и т.п. — в порядке вещей :)

* [[https://habrastorage.org/getpro/habr/avatars/6a9/e91/b51/6a9e91b51c9199fbd2a9d056882cfb9f_small.png](https://habr.com/users/Nulldevice/)**Nulldevice**](https://habr.com/users/Nulldevice/)4 ноября 2008 в 02:11

**+7**

Будтье осторожны! Розетка — очень опасное устройство! Раньше я тоже думал, что она безопасна. Но вчера я сделал следующее. Взял отвертку и аккуратно открытил крышку, чтобы залезть внутрь. Потом лизнул ро… Бац! Помните! Розетка — очень опасное устройство!

* + [[https://habrastorage.org/getpro/habr/avatars/916/c12/21a/916c1221ab340daef32732c85d7f513d_small.jpg](https://habr.com/users/alexbig/)**alexbig**](https://habr.com/users/alexbig/)4 ноября 2008 в 11:52

**+1**

отличное сравнение! :-)

* [[https://habrastorage.org/getpro/habr/avatars/6a9/e91/b51/6a9e91b51c9199fbd2a9d056882cfb9f_small.png](https://habr.com/users/Nulldevice/)**Nulldevice**](https://habr.com/users/Nulldevice/)4 ноября 2008 в 03:46

**+1**

Поясню свой минус за статью. Расставлять проверки там, где проверять не надо — это антипаттерн, это замусоривание кода. Не стоит воспринимать других разработчиков, как людей, которые хотят встроить экcплоит в разработку соседа — так коллективной разработки не получится…

* + [**nikolaikopernik**](https://habr.com/users/nikolaikopernik/)4 ноября 2008 в 12:29

**0**

Отличное сравнение с розетками! Но тут вы не правы. Я не агеритую пользоваться джава секюрити, а просто напоминаю, что такая возможность есть, т.к. многие программеры просто это не учитывают, а еще больше прогеров вообще об этом не знают (как я например до недавнего момента — знал про розетку, но что там такое напряжение — не знал )). Так что "*Розетка — очень опасное устройство!*" :)

* [**methlab**](https://habr.com/users/methlab/)4 ноября 2008 в 08:59

**0**

>Поэтому у нас есть все основания считатать, что метод test() возвращает всегда число 33. Более того, мы бы   
>могли использовать это предположение в своем коде, завязав на этом какую-либо логику.  
  
Истинное нарушение инкапсуляции — в коде, вызывающем метод test предполагать, что он вернет 33.

* + [**shuron**](https://habr.com/users/shuron/)20 марта 2009 в 20:28

**0**

+1

* [[https://habrastorage.org/getpro/habr/avatars/e63/f6c/331/e63f6c331dbfdd8ffeba28e96e79493b_small.jpg](https://habr.com/users/zencd/)**zencd**](https://habr.com/users/zencd/)4 ноября 2008 в 15:04

**0**

*/\* считал, что java — это 100% ОО язык \*/*  
  
Смешная шутка, про 100%  
  
*/\* Единственное, для чего я написал этот пост — предупредить всех явистов \*/*  
  
И, таки, вы в это сами верите? :-) Вот имхо полезное упражнение: найти «не менее пяти причин почему я опубликовался на Хабре».

* + [[https://habrastorage.org/getpro/habr/avatars/54f/db1/35c/54fdb135c6f606aba94ccf4532e581dc_small.jpg](https://habr.com/users/Nuru/)**Nuru**](https://habr.com/users/Nuru/)9 ноября 2008 в 05:50

**0**

До тех пор, пока в яве сушествуют атомарные типы данных, 100% ОО языком его щитать нельзя. Только вот возникает вопрос: настолько ли это плохо, как думает автор?

* [**alsor**](https://habr.com/users/alsor/)4 ноября 2008 в 20:47

**0**

На это можно взглянуть и с другой точки зрения — не все что написано в программе для машины, скорее это для другого человека. Если ты увидел private, значит не надо завязываться на тип или имя этого поля или вообще учитывать его существование — и твоя программа будет устойчива. А компилятор тебе в этом поможет. Ну а если уж решил таки что-то с этим полем сделать — вот тебе способ.

* [**xkesha**](https://habr.com/users/xkesha/)4 ноября 2008 в 22:12

**0**

Давайте всё же различать этап компиляции от этапа исполнения. Традиционно модификаторы доступа пришли от языка C++, в котором они существуют только «на листе» (в бинарном виде нет отличия). Инкапсуляция это свойство языка программирования (а не виртуальной машины) синтаксис которого проверяет компилятор. Поэтому полагаю, что свойство инкапсуляции языка Java данный пример не нарушает.

* + [**shuron**](https://habr.com/users/shuron/)20 марта 2009 в 20:30

**0**

Именно так! Инкапсуляция вобще должна упростить жизнь при разроботке. И в рантайме это слово роли не играет.

* [**pleax**](https://habr.com/users/pleax/)5 ноября 2008 в 09:46

**–2**

ппц… при чем тут инкапсуляция?..

* [**ofigenn**](https://habr.com/users/ofigenn/)5 ноября 2008 в 11:25

**0**

А если написать private final int x = 33;?

* + [**nikolaikopernik**](https://habr.com/users/nikolaikopernik/)5 ноября 2008 в 18:36

**0**

т.е.? вы хотите поломать еще и константы?

* + - [**ofigenn**](https://habr.com/users/ofigenn/)5 ноября 2008 в 20:27

**0**

А разве сломается?

* + [[https://habrastorage.org/getpro/habr/avatars/54f/db1/35c/54fdb135c6f606aba94ccf4532e581dc_small.jpg](https://habr.com/users/Nuru/)**Nuru**](https://habr.com/users/Nuru/)9 ноября 2008 в 05:49

**0**

Если я конечно не ошибаюсь, то решлекшн позволяет и модификаторы полей менять.