What is reflection and why is it useful?

Asked 16 years, 6 months ago Modified 3 months ago Viewed 1.1m times

2571

What is reflection, and why is it useful?

I'm particularly interested in Java, but I assume the principles are the same in any language.

java reflection terminology

Share Improve this question Follow

edited Nov 27, 2022 at 7:51
Lii
12.1k 9 68 90

asked Sep 1, 2008 at 8:39
Lehane
48.7k 14 55 54

16 For me it is a way of getting class names at runtime and creating objects of that class. – Nabin Nov 3, 2016 at 5:02

108 because this is a popular question I'd like to point out that reflection (Without annotations) should be the very last tool you go to when solving a problem. I use it and love it, but it cancels out all the advantages of Java's static typing. If you do need it, isolate it to as small an area as possible (One method or one class). It's more acceptable to use it in tests than production code. With annotations it should be fine--The main point is not to specify class or method names as "Strings" if you can possibly avoid it. – Bill K Apr 4, 2017 at 21:45

2 see also: softwareengineering.stackexchange.com/questions/123956/… – givanse Apr 20, 2018 at 22:34

9 In addition to @BillK's comment: Reflection is very powerful, I'd call it magic. With great power comes great responsibility. Use it only if you know what you're doing. – MC Emperor Nov 22, 2018 at 13:02

3 @Trap I don't know, that's why I was recommending against it—it's really annoying when I run into reflection when there were other solutions available, or reflection that isn't isolated to a very small, constrained and clearly documented area of the code. But asking why programmers do what they do is beyond my ability to answer. – Bill K Jun 12, 2020 at 21:03

24 Answers

Sorted by: Highest score (default)

2032

The name reflection is used to describe code which is able to inspect other code in the same system (or itself).

For example, say you have an object of an unknown type in Java, and you would like to call a 'doSomething' method on it if one exists. Java's static typing system isn't really designed to support this unless the object conforms to a known interface, but using reflection, your code can look at the object and find out if it has a method called 'doSomething' and then call it if you want to.

So, to give you a code example of this in Java (imagine the object in question is foo) :

```
Method method = foo.getClass().getMethod("doSomething", null);
method.invoke(foo, null);
```

One very common use case in Java is the usage with annotations. JUnit 4, for example, will use reflection to look through your classes for methods tagged with the @Test annotation, and will then call them when running the unit test.

There are some good reflection examples to get you started at http://docs.oracle.com/javase/tutorial/reflect/index.html

And finally, yes, the concepts are pretty much similar in other statically typed languages which support reflection (like C#). In dynamically typed languages, the use case described above is less necessary (since the compiler will allow any method to be called on any object, failing at runtime if it does not exist), but the second case of looking for methods which are marked or work in a certain way is still common.

Update from a comment:

> The ability to inspect the code in the system and see object types is not reflection, but rather Type Introspection. Reflection is then the ability to make modifications at runtime by making use of introspection. The distinction is necessary here as some languages support introspection, but do not support reflection. One such example is C++

Share Improve this answer Follow

edited Mar 7, 2020 at 23:08
Marome
47 1 11

answered Sep 1, 2008 at 8:44
Matt Sheppard
118k 46 113 134

43 can u please explain what is the significance of that null parameter in this line Method method = foo.getClass().getMethod("doSomething", null); – Chaitanya Gudala Nov 23, 2012 at 5:59

64 The null indicates there are no parameters being passed to the foo method. See docs.oracle.com/javase/6/docs/api/java/lang/reflect/…, java.lang.Object...) for details. – Matt Sheppard Nov 27, 2012 at 1:14

904 Just to clear up since this has so many upvotes. The ability to inspect the code in the system and see object types is not reflection, but rather Type Introspection. Reflection is then the ability to make modifications at runtime by making use of introspection. The distinction is necessary here as some languages support introspection, but do not support reflection. One such example is C++. – bigtunacan Mar 12, 2013 at 3:46

55 I love reflection but if you have control over the code then using reflection as specified in this answer is unnessary and therefore an abuse--You should use Type Introspection (instanceof) and strong types. If there is any way but reflection to do something, that's how it should be done. Reflection causes serious heartache because you lose all advantages of using a statically typed language. If you need it you need it, however even then I'd consider a pre-packaged solution like Spring or something that completely encapsulates the reflection necessary--IE: let someone else have the headaches. – Bill K Jul 29, 2013 at 15:50

9 @bigtunacan Where did you get that information from? I see the term "reflection" used in official Java documentation from Oracle to describe not only the ability to make changes at runtime but also the ability to see the type of an object. Not to mention that most so-called "type introspection" related classes (ex: `Method`, `Constructor`, `Modifier`, `Field`, `Member`, basically apparently all except `Class`) are within the `java.lang.*reflect*` package. Perhaps the concept "reflection" comprehensively includes both "type introspection" and modification at run-time? – juung666 Apr 29, 2016 at 9:34

300

> **Reflection** is a language's ability to inspect and dynamically call classes, methods, attributes, etc. at runtime.

For example, all objects in Java have the method `getClass()`, which lets you determine the object's class even if you don't know it at compile time (e.g. if you declared it as an `Object`) - this might seem trivial, but such reflection is not possible in less dynamic languages such as `C++`. More advanced uses lets you list and call methods, constructors, etc.

Reflection is important since it lets you write programs that do not have to "know" everything at compile time, making them more dynamic, since they can be tied together at runtime. The code can be written against known interfaces, but the actual classes to be used can be instantiated using reflection from configuration files.

Lots of modern frameworks use reflection extensively for this very reason. Most other modern languages use reflection as well, and in scripting languages (such as Python) they are even more tightly integrated, since it feels more natural within the general programming model of those languages.

Share Improve this answer Follow

edited Jan 24, 2016 at 0:36
Sheharyar
75.9k 21 174 223

answered Sep 1, 2008 at 8:52
Liedman
10.3k 4 35 36

3 So in other words, you can create an instance out of it's qualified name and the compiler won't complain about it (because say you use just a String for the class name). Then, at run time, if that class is not present you get an exception. You kind of bypassed the compiler in this case. Would you give me some specific use case for this? I just can't picture when i would choose it. – Fernando Gabrieli Dec 1, 2018 at 21:40

4 @FernandoGabrieli while it is true that it is easy to create runtime errors with reflection, it is also perfectly possible to use reflection without risking runtime exceptions. As hinted in my answer, a common use of reflection is for libraries or frameworks, which explicitly *can't* know the structure of the application at compile time, since they are compiled separate from the application. Any library that uses "code by convention" is likely to use reflection, but not necessarily using magic strings. – Liedman Jan 15, 2019 at 12:25

1 `C++` does have Run-time type information. RTTI – Aykhan Hagverdili Dec 10, 2019 at 20:32

127

One of my favorite uses of reflection is the below Java dump method. It takes any object as a parameter and uses the Java reflection API to print out every field name and value.

```
import java.lang.reflect.Array;
import java.lang.reflect.Field;

public static String dump(Object o, int callCount) {
    callCount++;
    StringBuffer tabs = new StringBuffer();
    for (int k = 0; k < callCount; k++) {
        tabs.append("\t");
    }
    StringBuffer buffer = new StringBuffer();
    Class oClass = o.getClass();
    if (oClass.isArray()) {
        buffer.append("\n");
        buffer.append(tabs.toString());
        buffer.append("[");
        for (int i = 0; i < Array.getLength(o); i++) {
            if (i < 0)
                buffer.append(",");
            Object value = Array.get(o, i);
            if (value.getClass().isPrimitive() ||
                    value.getClass() == java.lang.Long.class ||
                    value.getClass() == java.lang.String.class ||
                    value.getClass() == java.lang.Integer.class ||
                    value.getClass() == java.lang.Boolean.class
                    ) {
                buffer.append(value);
            } else {
                buffer.append(dump(value, callCount));
            }
        }
        buffer.append(tabs.toString());
        buffer.append("]\n");
    } else {
        buffer.append("\n");
        buffer.append(tabs.toString());
        buffer.append("{\n");
        while (oClass != null) {
            Field[] fields = oClass.getDeclaredFields();
```

Share Improve this answer Follow

edited Jan 29, 2019 at 0:08
[illegible]
3,767 2 32 34

answered Sep 2, 2008 at 16:15
Ben Williams
2,297 2 19 15

8 What should Callcount be set to? – Tom Apr 29, 2010 at 13:06

9 I got a Exception in thread "AWT-EventQueue-0" java.lang.StackOverflowError when I ran this. – Tom Apr 29, 2010 at 13:13

3 @Tom `callCount` should be set to zero. It's value is used to determine how many tabs should precede each line of output: each time dump needs to dump a "subobject", the output would print as nested in the parent. That method proves useful when wrapped in another. Consider `printDump(Object obj){ System.out.println(dump(obj, 0)); }`. – fny Nov 23, 2012 at 6:09

1 The java.lang.StackOverflowError might be created in case of circular references, because of the unchecked recursion: buffer.append(dump(value, callCount)) – Arnaud P Jul 1, 2013 at 23:03

4 Can you specifically release your code to Public Domain, pretty please? – stolsvik Oct 8, 2013 at 10:06

101

Uses of Reflection

Reflection is commonly used by programs which require the ability to examine or modify the runtime behavior of applications running in the Java virtual machine. This is a relatively advanced feature and should be used only by developers who have a strong grasp of the fundamentals of the language. With that caveat in mind, reflection is a powerful technique and can enable applications to perform operations which would otherwise be impossible.

Extensibility Features

An application may make use of external, user-defined classes by creating instances of extensibility objects using their fully-qualified names. Class Browsers and Visual Development Environments A class browser needs to be able to enumerate the members of classes. Visual development environments can benefit from making use of type information available in reflection to aid the developer in writing correct code. Debuggers and Test Tools Debuggers need to be able to examine private members in classes. Test harnesses can make use of reflection to systematically call a discoverable set APIs defined on a class, to ensure a high level of code coverage in a test suite.

Drawbacks of Reflection

Reflection is powerful, but should not be used indiscriminately. If it is possible to perform an operation without using reflection, then it is preferable to avoid using it. The following concerns should be kept in mind when accessing code via reflection.

- **Performance Overhead**

Because reflection involves types that are dynamically resolved, certain Java virtual machine optimizations cannot be performed. Consequently, reflective operations have slower performance than their non-reflective counterparts and should be avoided in sections of code which are called frequently in performance-sensitive applications.

- **Security Restrictions**

Reflection requires a runtime permission which may not be present when running under a security manager. This is in an important consideration for code which has to run in a restricted security context, such as in an Applet.

- **Exposure of Internals**

Since reflection allows code to perform operations that would be illegal in non-reflective code, such as accessing private fields and methods, the use of reflection can result in unexpected side-effects, which may render code dysfunctional and may destroy portability. Reflective code breaks abstractions and therefore may change behavior with upgrades of the platform.

source: The Reflection API

51

Reflection is a key mechanism to allow an application or framework to work with code that might not have even been written yet!

Take for example your typical web.xml file. This will contain a list of servlet elements, which contain nested servlet-class elements. The servlet container will process the web.xml file, and create new a new instance of each servlet class through reflection.

Another example would be the Java API for XML Parsing (JAXP). Where an XML parser provider is 'plugged-in' via well-known system properties, which are used to construct new instances through reflection.

And finally, the most comprehensive example is Spring which uses reflection to create its beans, and for its heavy use of proxies

Share Improve this answer Follow

answered Sep 1, 2008 at 9:30
toolkit
50.3k 18 111 139

49

Not every language supports reflection, but the principles are usually the same in languages that support it.

Reflection is the ability to "reflect" on the structure of your program. Or more concrete. To look at the objects and classes you have and programmatically get back information on the methods, fields, and interfaces they implement. You can also look at things like annotations.

It's useful in a lot of situations. Everywhere you want to be able to dynamically plug in classes into your code. Lots of object relational mappers use reflection to be able to instantiate objects from databases without knowing in advance what objects they're going to use. Plug-in architectures is another place where reflection is useful. Being able to dynamically load code and determine if there are types there that implement the right interface to use as a plugin is important in those situations.

Share Improve this answer Follow

edited Oct 9, 2020 at 3:53
Pang
10.1k 146 86 124

answered Sep 1, 2008 at 8:50
Mendelt
37.5k 6 75 97

39

Reflection allows instantiation of new objects, invocation of methods, and get/set operations on class variables dynamically at run time without having prior knowledge of its implementation.

```
Class myObjectClass = MyObject.class;
Method[] method = myObjectClass.getMethods();

//Here the method takes a string parameter if there is no param, put null.
Method method = aClass.getMethod("method_name", String.class);

Object returnValue = method.invoke(null, "parameter-value1");
```

In above example the null parameter is the object you want to invoke the method on. If the method is static you supply null. If the method is not static, then while invoking you need to supply a valid MyObject instance instead of null.

Reflection also allows you to access private member/methods of a class:

```
public class A {
  private String str = null;

  public A(String str) {
    this.str = str;
  }
}
```

Then one can:

```
A obj = new A("Some value");

Field privateStringField = A.class.getDeclaredField("privateString");

//Turn off access check for this field
privateStringField.setAccessible(true);

String fieldValue = (String) privateStringField.get(obj);
System.out.println("fieldValue = " + fieldValue);
```

- For inspection of classes (also know as introspection) you don't need to import the reflection package (`java.lang.reflect`). Class metadata can be accessed through `java.lang.Class` .

Reflection is a very powerful API but it may slow down the application if used in excess, as it resolves all the types at runtime.

Share Improve this answer Follow

edited Dec 5, 2024 at 8:32
avishj
3 1 3

answered Jul 8, 2013 at 16:12
Nikhil Shekhar
566 4 4

26

Java Reflection is quite powerful and can be very useful. Java Reflection makes it possible **to inspect classes, interfaces, fields and methods at runtime,** without knowing the names of the classes, methods etc. at compile time. It is also possible to **instantiate new objects, invoke methods and get/set field values using reflection.**

A quick Java Reflection example to show you what using reflection looks like:

```
Method[] methods = MyObject.class.getMethods();

    for(Method method : methods){
        System.out.println("method = " + method.getName());
    }
```

This example obtains the Class object from the class called MyObject. Using the class object the example gets a list of the methods in that class, iterates the methods and print out their names.

Exactly how all this works is explained here

Edit: After almost 1 year I am editing this answer as while reading about reflection I got few more uses of Reflection.

- Spring uses bean configuration such as:

```
<bean id="someID" class="com.example.Foo">
    <property name="someField" value="someValue" />
</bean>
```

When the Spring context processes this < bean > element, it will use Class.forName(String) with the argument "com.example.Foo" to instantiate that Class.

It will then again use reflection to get the appropriate setter for the < property > element and set its value to the specified value.

- Junit uses Reflection especially for testing Private/Protected methods.

For Private methods,

```
Method method = targetClass.getDeclaredMethod(methodName, argClasses);
method.setAccessible(true);
return method.invoke(targetObject, argObjects);
```

For private fields,

```
Field field = targetClass.getDeclaredField(fieldName);
field.setAccessible(true);
field.set(object, value);
```

25

Example:

Take for example a remote application which gives your application an object which you obtain using their API Methods . Now based on the object you might need to perform some sort of computation .

The provider guarantees that object can be of 3 types and we need to perform computation based on what type of object .

So we might implement in 3 classes each containing a different logic .Obviously the object information is available in runtime so you cannot statically code to perform computation hence reflection is used to instantiate the object of the class that you require to perform the computation based on the object received from the provider .

Share Improve this answer Follow

edited Aug 14, 2019 at 14:55
Ihor Patsian
1,298 2 17 27

answered Jun 22, 2012 at 15:35
human.js
1,382 13 15

1 I need something similar.. An example would help me a lot as I am new to reflection concepts.. – Atom Jun 22, 2015 at 18:42

2 I'm confused: can't you use `instanceof` to determine object type at runtime? – ndm13 Jan 11, 2018 at 2:09

25

Simple example for reflection. In a chess game, you do not know what will be moved by the user at run time. reflection can be used to call methods which are already implemented at run time:

```
public class Test {

    public void firstMoveChoice(){
        System.out.println("First Move");
    }
    public void secondMOveChoice(){
        System.out.println("Second Move");
    }
    public void thirdMoveChoice(){
        System.out.println("Third Move");
    }

    public static void main(String[] args) throws IllegalAccessException, 
IllegalArgumentException, InvocationTargetException {
        Test test = new Test();
        Method[] method = test.getClass().getMethods();
        //firstMoveChoice
        method[0].invoke(test, null);
        //secondMoveChoice
        method[1].invoke(test, null);
        //thirdMoveChoice
        method[2].invoke(test, null);
    }

}
```

Share Improve this answer Follow

edited Aug 14, 2019 at 14:56
Ihor Patsian
1,298 2 17 27

answered Nov 6, 2014 at 4:42
Isuru Jayakantha
318 3 3

19

As per my understanding:

Reflection allows programmer to access entities in program dynamically. i.e. while coding an application if programmer is unaware about a class or its methods, he can make use of such class dynamically (at run time) by using reflection.

It is frequently used in scenarios where a class name changes frequently. If such a situation arises, then it is complicated for the programmer to rewrite the application and change the name of the class again and again.

Instead, by using reflection, there is need to worry about a possibly changing class name.

Share Improve this answer Follow

edited Apr 2, 2013 at 16:30
DeadChex
4,669 1 29 36

answered Feb 6, 2012 at 5:37
pramod
191 1 2

19

Reflection is an API which is used to examine or modify the behaviour of *methods, classes, interfaces* at runtime.

1. The required classes for reflection are provided under `java.lang.reflect package` .
2. Reflection gives us information about the class to which an object belongs and also the methods of that class which can be executed by using the object.
3. Through reflection we can invoke methods at runtime irrespective of the access specifier used with them.

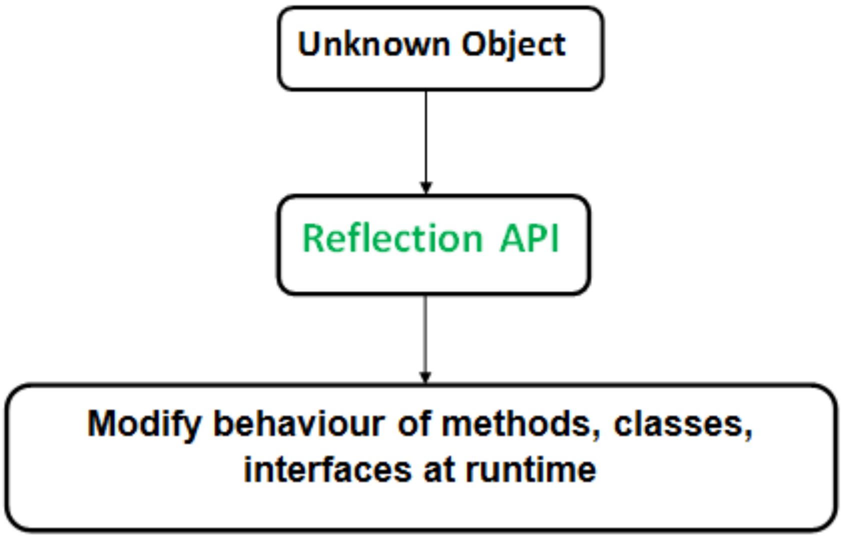
The `java.lang` and `java.lang.reflect` packages provide classes for java reflection.

Reflection can be used to get information about –

1. **Class** The `getClass()` method is used to get the name of the class to which an object belongs.
2. **Constructors** The `getConstructors()` method is used to get the public constructors of the class to which an object belongs.
3. **Methods** The `getMethods()` method is used to get the public methods of the class to which an objects belongs.

The **Reflection API** is mainly used in:

IDE (Integrated Development Environment) e.g. Eclipse, MyEclipse, NetBeans etc.
Debugger and Test Tools etc.



Advantages of Using Reflection:

Extensibility Features: An application may make use of external, user-defined classes by creating instances of extensibility objects using their fully-qualified names.

Debugging and testing tools: Debuggers use the property of reflection to examine private members on classes.

Drawbacks:

Performance Overhead: Reflective operations have slower performance than their non-reflective counterparts, and should be avoided in sections of code which are called frequently in performance-sensitive applications.

Exposure of Internals: Reflective code breaks abstractions and therefore may change behaviour with upgrades of the platform.

Ref: [Java Reflection](#) [javarevisited.blogspot.in](#)

Share Improve this answer Follow

answered Mar 22, 2017 at 7:27
roottraveller
8,252 8 64 70

4 I would add to the drawbacks "It breaks refactoring". For me that's the main reason to avoid reflection as much as possible. – SantiBailors Dec 20, 2017 at 11:27

So it allows us (for example), to inspect the classes we have (whether we have instances of them or not), correct? By this i mean, get their methods or constructors and use them to create new instances/invoke them. Why do we say "changing the program behavior" if the behavior is already there but with different code? Why is it called "reflection"? Thanks – Fernando Gabrieli Dec 1, 2018 at 21:28

18

Reflection is a set of functions which allows you to access the runtime information of your program and modify it behavior (with some limitations).

It's useful because it allows you to change the runtime behavior depending on the meta information of your program, that is, you can check the return type of a function and change the way you handle the situation.

In C# for example you can load an assembly (a .dll) in runtime an examine it, navigating through the classes and taking actions according to what you found. It also let you create an instance of a class on runtime, invoke its method, etc.

Where can it be useful? Is not useful every time but for concrete situations. For example you can use it to get the name of the class for logging purposes, to dynamically create handlers for events according to what's specified on a configuration file and so on...

Share Improve this answer Follow

edited Mar 27, 2020 at 20:37
Aurelio
25.8k 9 61 64

answered Sep 1, 2008 at 8:50
Jorge Córdoba
52.1k 11 82 130

14

I just want to add some points to all that was listed.

With **Reflection API** you can write a universal `toString()` method for any object.

It could be useful for debugging.

Here is some example:

```
class ObjectAnalyzer {

   private ArrayList<Object> visited = new ArrayList<Object>();

   /**
    * Converts an object to a string representation that lists all fields.
    * @param obj an object
    * @return a string with the object's class name and all field names and
    * values
    */
   public String toString(Object obj) {
      if (obj == null) return "null";
      if (visited.contains(obj)) return "...";
      visited.add(obj);
      Class cl = obj.getClass();
      if (cl == String.class) return (String) obj;
      if (cl.isArray()) {
         String r = cl.getComponentType() + "[]{";
         for (int i = 0; i < Array.getLength(obj); i++) {
            if (i > 0) r += ",";
            Object val = Array.get(obj, i);
            if (cl.getComponentType().isPrimitive()) r += val;
            else r += toString(val);
         }
         return r + "}";
      }

      String r = cl.getName();
      // inspect the fields of this class and all superclasses
      do {
         r += "[";
         Field[] fields = cl.getDeclaredFields();
         AccessibleObject.setAccessible(fields, true);
         // get the names and values of all fields
         for (Field f : fields) {
            if (!Modifier.isStatic(f.getModifiers())) {
               if (!r.endsWith("[")) r += ",";
```

Share Improve this answer Follow

edited Jul 16, 2022 at 18:20

answered Sep 20, 2014 at 22:22

catch32
18.6k 45 150 225

12

Reflection is to let object to see their appearance. This argument seems nothing to do with reflection. In fact, this is the "self-identify" ability.

Reflection itself is a word for such languages that lack the capability of self-knowledge and self-sensing as Java and C#. Because they do not have the capability of self-knowledge, when we want to observe how it looks like, we must have another thing to reflect on how it looks like. Excellent dynamic languages such as Ruby and Python can perceive the reflection of their own without the help of other individuals. We can say that the object of Java cannot perceive how it looks like without a mirror, which is an object of the reflection class, but an object in Python can perceive it without a mirror. So that's why we need reflection in Java.

Share Improve this answer Follow

edited Jul 24, 2018 at 17:22
Yoon5oo
514 6 12

answered Dec 8, 2015 at 7:02
Marcus Thornton
6,203 8 51 54

type(), isinstance(), callable(), dir() and getattr(). .... these are pythonic reflection calls
– AnthonyJClink Jan 30, 2018 at 16:14

9

From java documentation page

> `java.lang.reflect` package provides classes and interfaces for obtaining reflective information about classes and objects. Reflection allows programmatic access to information about the fields, methods and constructors of loaded classes, and the use of reflected fields, methods, and constructors to operate on their underlying counterparts, within security restrictions.

`AccessibleObject` allows suppression of access checks if the necessary `ReflectPermission` is available.

Classes in this package, along with `java.lang.Class` accommodate applications such as debuggers, interpreters, object inspectors, class browsers, and services such as `Object Serialization` and `JavaBeans` that need access to either the public members of a target object (based on its runtime class) or the members declared by a given class

It includes following functionality.

1. Obtaining Class objects,
2. Examining properties of a class (fields, methods, constructors),
3. Setting and getting field values,
4. Invoking methods,
5. Creating new instances of objects.

Have a look at this documentation link for the methods exposed by ***`Class`*** class.

From this article (by Dennis Sosnoski, President, Sosnoski Software Solutions, Inc) and this article (security-explorations pdf):

I can see considerable drawbacks than uses of using Reflection

User of Reflection:

1. It provides very versatile way of dynamically linking program components
2. It is useful for creating libraries that work with objects in very general ways

Drawbacks of Reflection:

1. Reflection is much slower than direct code when used for field and method access.
2. It can obscure what's actually going on inside your code
3. It bypasses the source code can create maintenance problems
4. Reflection code is also more complex than the corresponding direct code
5. It allows violation of key Java security constraints such as data access protection and type safety

General abuses:

1. Loading of restricted classes,
2. Obtaining references to constructors, methods or fields of a restricted class,
3. Creation of new object instances, methods invocation, getting or setting field values of a restricted class.

Have a look at this SE question regarding abuse of reflection feature:

[How do I read a private field in Java?]

Summary:

*Insecure use of its functions conducted from within a system code can also easily lead to the compromise of a Java security mode*l. ***So use this feature sparingly***

Share Improve this answer Follow

edited May 23, 2017 at 12:34
Community Bot
1 1

answered Feb 13, 2016
Ravindra babu
39k 11 256 220

A way to avoid the performance problems of Reflection in some cases is to have a class `Woozle` examine other classes on startup to see which ones have a static `RegisterAsWoozleHelper()` method, and invoke all such methods it finds with a callback they can use to tell `Woozle` about themselves, avoiding need to use Reflection while e.g. deserializing data. – supercat Feb 1, 2020 at 18:53

9

As name itself suggest it reflects what it holds for example class method,etc apart from providing feature to invoke method creating instance dynamically at runtime.

It is used by many frameworks and application under the wood to invoke services without actually knowing the code.

Share Improve this answer Follow

answered Apr 9, 2017 at 2:29
Mohammed Sarfaraz
460 6 10

7

Reflection gives you the ability to write more generic code. It allows you to create an object at runtime and call its method at runtime. Hence the program can be made highly parameterized. It also allows introspecting the object and class to detect its variables and method exposed to the outer world.

Share Improve this answer Follow

edited Jul 24, 2018 at 17:22
Yoon5oo
514 6 12

answered Apr 26, 2016 at 22:54
saumik gupta
167 2 10

6

`Reflection` has many **uses**. The one I am more familiar with, is to be able to create code on the fly.

> IE: dynamic classes, functions, constructors - based on any data (xml/array/sql results/hardcoded/etc..)

Share Improve this answer Follow

edited Jun 23, 2015 at 5:47
Rohil_PHPBeginner
6,080 2 23 32

answered May 6, 2013 at 20:16
Ess Kay
598 4 20

1 This answer would be a lot better if you gave just one unusual example of generated code either from a SQL result or XML file etc. – ThisClark Feb 15, 2017 at 1:53

No problem. I used reflection in a windows application that dynamically generates the interface based on XML taken from a database. – Ess Kay Feb 16, 2017 at 18:41

So basically, there is a class I created which shows the user a report. This report has parameters such as Date(from to) id, or whatever else. This info is stored in xml. So first we have a report selection. Based on the report selected, the form gets the xml. Once the xml is retrieved, it uses reflection to create a class with fields based on reflected types. Once you change to a different report, the slate is wiped clean and new fields are generated based on the xml. So its essentially a dynamic form based on reflection. I also used in other ways but this should be suffecient Hope that helps – Ess Kay Feb 16, 2017 at 18:45

3

I want to answer this question by example. First of all `Hibernate` project uses `Reflection API` to generate `CRUD` statements to bridge the chasm between the running application and the persistence store. When things change in the domain, the `Hibernate` has to know about them to persist them to the data store and vice versa.

Alternatively works `Lombok Project`. It just injects code at compile time, result in code being inserted into your domain classes. (I think it is OK for getters and setters)

`Hibernate` chose `reflection` because it has minimal impact on the build process for an application.

And from Java 7 we have `MethodHandles`, which works as `Reflection API`. In projects, to work with loggers we just copy-paste the next code:

```
Logger LOGGER =
Logger.getLogger(MethodHandles.lookup().lookupClass().getName());
```

Because it is hard to make typo-error in this case.

Share Improve this answer Follow

answered Dec 5, 2018 at 6:35
BSeitkazin
3,059 26 41

3

As I find it best to explain by example and none of the answers seem to do that...

A practical example of using reflections would be a Java Language Server written in Java or a PHP Language Server written in PHP, etc. Language Server gives your IDE abilities like autocomplete, jump to definition, context help, hinting types and more. In order to have all tag names (words that can be autocompleted) to show all the possible matches as you type the Language Server has to inspect everything about the class including doc blocks and private members. For that it needs a reflection of said class.

A different example would be a unit-test of a private method. One way to do so is to create a reflection and change the method's scope to public in the test's set-up phase. Of course one can argue private methods shouldn't be tested directly but that's not the point.

Share Improve this answer Follow

edited Jun 24, 2019 at 12:51

answered Apr 26, 2019 at 9:34
cprn
1,710 1 20 25

1

I am using reflection to create an object based on class name(class name in String) and call the method of that class

```
Object obj = Class.forName(config.getClassPath())
                    .getDeclaredConstructor()
                    .newInstance();
Method method = obj.getClass().getMethod("getCustomer", SearchObject.class,
ObjectConfig.class,
                HttpServletRequest.class);
method.invoke(obj, searchObject, config, request);
```

But one major problem is that if you Autowired something on that class that will **re-initialized to null**

Share Improve this answer Follow

answered Dec 27, 2021 at 10:52

Bhushan
106 1 9

0

If you want to have a sneek peek of the values in an Object, here's a fun implementation

```
        Class<?> classes = exampleObject.getClass();
        Field[] fields = classes.getDeclaredFields();
        for (Field field: fields){
            field.setAccessible(true);
            Object value;
            try {
                value = field.get(exampleObject);
            } catch (IllegalAccessException e){
                throw new RuntimeException(e);
            }
            logger.info("exampleObject: "+field.getName() + " value: "+value);
        }
```

Don't forget to import this `import java.lang.reflect.Field;` package.

Share Improve this answer Follow

answered Nov 6, 2023 at 12:48

hardMaster
21 4

-3

IMPORTANT

Starting from Java 9 you can no longer use reflection, unless the package-info.java **opens** the module to reflection access.

By default, "reflection" access is denied to all packages in the module.

See Understanding Java 9 Modules

Share Improve this answer Follow

edited Nov 16, 2021 at 7:48
nitin angadi
506 1 7 13

answered Nov 23, 2020 at 23:01
RSG
420 5 7

2 This is plain wrong. You still can use reflection. You just can't make stuff accessible (`ref.setAccessible(true);`) if the package is open to you. – Johannes Kuhn Nov 16, 2021 at 10:50

Highly active question. Earn 10 reputation (not counting the association bonus) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.

Start asking to get answers

Find the answer to your question by asking.

Ask question

Explore related questions

java reflection terminology

See similar questions with these tags.