链滴

笔记 _Java 语言的动态性

作者：damoluomu

原文链接：https://ld246.com/article/1491618558285

来源网站：链滴

许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

<h2 id="反射API">反射 API</h2>
使用场合
<blockquote>
要调用的方法或者要操作的域的名称按照某种规律变化的时候。
</blockquote>
典型场景 1
<blockquote>
在 Servlet 中用 HTTP 请求的参数值来填充领域对象。
</blockquote>
典型场景 2
<blockquote>
在数据库操作中，从 SQL 查询结果集中创建并填充领域对象。数据库的列名和领域对象的属性称也存在着类似的对应关系。
</blockquote>
术语
Constructor
Field
Method
构造函数
nested class 嵌套类
静态嵌套类
非静态嵌套类
<blockquote>
如果构造函数声明了长度可变的参数，在获取构造函数的时候，要使用对应的数组类型的 Class 象。这是因为长度可变的参数实际上是通过数组来实现的。
</blockquote>
field 域
<blockquote>
类 > 静态域
</blockquote>
<blockquote>
对象 > 实例域
</blockquote>
只能对类中的公开域进行操作。私有域没有办法通过反射 API 获取到，也无法进行操作。
方法
<blockquote>
与构造函数和域不同的是，通过反射 API 可以获取类中的私有方法。调用私有方法之前，需要先用 Method 类的 setAccessible 方法来设置可以访问的权限。
</blockquote>
<h2 id="动态代理-dynamic-proxy-">动态代理(dynamic proxy)</h2>
AOP 面向切面编程
cross-cutting 横切
动态代理只对接口起作用。
<h2 id="动态语言支持">动态语言支持</h2>
<blockquote>
JSR 292
Java 标准库中新的方法调用 API
Java 虚拟机规范中新的 invokedynamic 指令
</blockquote>
<blockquote>
method handle 方法句柄
</blockquote>

方法句柄的类型

方法句柄的调用
参数长度可变的方法句柄
参数绑定
获取方法句柄
方法句柄变换
特殊方法句柄
使用方法句柄实现接口
访问控制权限
交换点
使用方法句柄进行函数式编程

<blockquote>
通过方法句柄可以直接调用该句柄所引用的底层方法。
</blockquote>
<blockquote>
通过反射 API 获取方法句柄。
获取和设置数组中元素的值的方法句柄
</blockquote>
<blockquote>
Null object 空对象
MethodHandles 类中的 identity 方法和 constant 方法的作用类似于在开发中用到的“空对象 Null object）”模式的应用。
</blockquote>
<blockquote>
原始方法句柄
变换之前的方法句柄
新方法句柄
变换之后的方法句柄
</blockquote>
<blockquote>
预处理方法句柄
</blockquote>
<hr>
元方法句柄
<blockquote>
通过 invoker 方法和 exactInvoker 方法得到的方法句柄。具有调用其他方法句柄的能力。
</blockquote>
查找类
<blockquote>
在 Lookup 对象被创建的时候，会记录下当前所在的类。
只要查找类能够访问某个方法或域，就可以通过 Lookup 的方法来查找到对应的方法句柄。
</blockquote>
交换点
<blockquote>
交换点的一个重要作用是在多线程环境下使用，可以在多个线程中共享同一个交换点对象。当某线程的交换点状态改变之后，其他线程所使用的 guardWithTest 方法返回的方法句柄的调用行为就发生变化。
</blockquote>
function pointer 函数指针
在程序中，方法句柄可以在对象之间自由传递，不受访问控制权限的限制。
currying 柯里化
<blockquote>
对一个方法的参数值进行预先设置后，得到一个新的方法。
</blockquote>

<h2 id="invokedynamic-指令">invokedynamic 指令</h2>
<blockquote>
Java 虚拟机 4 种方法调用指令
invokestatic
invokespecial
invokevirtual
invokeinterface
</blockquote>
<blockquote>
对于 invokedynamic 指令来说，在 Java 源代码中是没有直接的对应产生方式的。
</blockquote>
single dispatch 单派发
dynamic call site 动态调用点
bootstrap method 启动方法
<blockquote>
对 invokedynamic 指令的调用实际上就等价于对方法句柄的调用，具体来说是被转换成对方法柄的 invoke 方法的调用。
</blockquote>
<blockquote>
Java 语言的动态性体现在 脚本语言支持 API、反射 API、动态代理和 Java7 中通过 JSR292 引的动态语言支持上。
JSR292 引入的方法句柄在灵活性上要远胜于反射 API 中的 Method 类。
</blockquote>
<hr>
书《[201205]深入理解 Java7 核心技术与最佳实践(成富 著)》