

Difffate的技术随笔

简洁为本。ps：好记性，不如烂笔头。

RS




个人资料


 Leon-Zheng

关注

原创	粉丝	喜欢	评论
151	9	14	2

等级： 访问：12万+

积分：2594 排名：1万+

勋章：

关于

度

©1996
京ICP
经营性
网络1
中国互
北京互

最新文章

- Fiddler另存下载的文件及Chrome添加自定义header的方式
- IDEA 数据库表转 JavaBean
- 23种设计模式意图汇总
- Mysql优化常用经验总结
- Mysql 索引的几个问题

个人分类

- | | |
|----------|-----|
| Spring专题 | 15篇 |
| 设计模式专题 | 4篇 |
| Java语言特性 | 11篇 |
| Java多线程 | 5篇 |
| Java输入输出 | 2篇 |

展开

归档

- | | |
|----------|-----|
| 2018年1月 | 4篇 |
| 2017年12月 | 9篇 |
| 2017年11月 | 9篇 |
| 2017年10月 | 5篇 |
| 2017年9月 | 15篇 |

展开

阅读量：22151

Java 指定线程执行顺序（三种方式）

阅读量：11225

JMeter模拟请求发送，并带上自定义Header，参数，Cookie

阅读量：8054

IDEA Maven-Helper插件

阅读量：7466

Spring Cacheable注解不缓存null值

阅读量：4409

联系我



关于我

©1996-2025 京ICP备15080150号

经营性网站备案信息

网络110报警服务

中国互联网络信息中心

最新评论

Java 实现最大堆

josoe：leftChildIndex = maxIndex * 2 + 1; rightChildIndex = maxIndex * 2 + 2;

Spring Cacheable注解不缓存null值

buzenmedi：[reply]buzenmedi[/reply] 还好还好，都是用户一人行为。如果大量用户都出现问...

Java 指定线程执行顺序（三种方式）

wcpsony：[reply]weixin_39441971[/reply]第一种办法你看看启动了多少条线程。

Java 指定线程执行顺序（三种方式）

weixin_39441971：[reply]wcpsony[/reply] 怎么坑？

Spring JdbcTemplate

diyangxia：确实快，但是只快了一点点

原 invokevirtual、invokespecial、invokestatic、invokeinterface、invokedynamic介绍

2017年04月16日 18:19:58

阅读数：1019

如表格所示：

invokevirtual	Invoke instance method; dispatch based on class	执行一般实例方法，创建完实例对象后，obj.method()调用的
invokespecial	Invoke instance method; special handling for superclass, private, and instance initialization method invocations	实例初始化方法（构造函数）、父类的方法（super.method()方式调用）、私有方法
invokeinterface	Invoke interface method	执行接口方法
invokestatic	Invoke a class (static) method	执行静态方法
invokedynamic	Invoke dynamic method	jdk1.7新增，执行动态方法，不需要在编译时确定

举个例子：

```
[java]
1. package org.zheng.jvmcmd;
2.
3. /**
4.  * Create by zxb on 2017/4/16
5.  */
6. class SuperClass {
7.
8.     public String method() {
9.         return "from SuperClass...";
10.    }
11.
12.    public void otherMethod() {
13.        System.out.println("In SuperClass otherMethod()...");
14.    }
15. }
```

0

收藏

评论

微信

微博

QQ

联系我



关于度

©1999

京ICP

经营性

网络1

中国互

北京互

```
17.
18. class SubClass extends SuperClass {
19.
20.     public String method() {
21.         return "from SubClass...";
22.     }
23.
24.     public void subMethod() {
25.         System.out.println("SubClass calls super.method(): " + super.method());    //invokespecial
26.         privateMethod();    //invokespecial
27.     }
28.
29.     private void privateMethod() {
30.         System.out.println("This is private Method From SubClass");
31.     }
32. }
33.
34. interface ITest {
35.
36.     void print();
37. }
38.
39. class TestImpl implements ITest {
40.
41.     public void print() {
42.         System.out.println("print in TestImpl...");
43.     }
44. }
45.
46. public class InvokeTest {
47.
48.     public static void staticMethod() {
49.         System.out.println("I am static method..");
50.     }
51.
52.     public static void main(String args[]) {
53.         staticMethod();    //invokestatic
54.
55.         SubClass b = new SubClass();    //invokespecial 初始化
56.         SuperClass supper = b;          //向上转型引用
57.         System.out.println(supper.method());    //invokevirtual, 当前引用的对象是b
58.
59.         b.subMethod();    //invokevirtual
60.         b.otherMethod();    //invokevirtual
61.
62.         ITest iTTest = new TestImpl();    //invokespecial 初始化
63.         iTTest.print();    //invokeinterface
64.     }
65. }
```

使用命令javac -encoding UTF-8 InvokeTest.java

使用命令javap -p -v InvokeTest.class

加入CSDN，享受更精准的内容推荐，与500万程序员共同成长！

登录

注册

×



```
public static void main(java.lang.String[]);
  flags: ACC_PUBLIC, ACC_STATIC
  Code:
    stack=2, locals=4, args_size=1
    0: invokestatic #5          // Method staticMethod:()V
    3: new           #6          // class org/zheng/jvmcmd/SubClass
    6: dup
    7: invokespecial #7          // Method org/zheng/jvmcmd/SubClass."<init>":()V
    10: astore_1
    11: aload_1
    12: astore_2
    13: getstatic     #2          // Field java/lang/System.out:Ljava/io/PrintStream;
    16: aload_2
    17: invokevirtual #8          // Method org/zheng/jvmcmd/SuperClass.method:()Ljava/lang/String;
    20: invokevirtual #4          // Method java/io/PrintStream.println:(Ljava/lang/String;)V
    23: aload_1
    24: invokevirtual #9          // Method org/zheng/jvmcmd/SubClass.subMethod:()V
    27: aload_1
    28: invokevirtual #10         // Method org/zheng/jvmcmd/SubClass.otherMethod:()V
    31: new           #11         // class org/zheng/jvmcmd/TestImpl
    34: dup
    35: invokespecial #12         // Method org/zheng/jvmcmd/TestImpl."<init>":()V
    38: astore_3
    39: aload_3
    40: invokeinterface #13, 1    // InterfaceMethod org/zheng/jvmcmd/ITest.print:()V
    45: return
  lineNumberTable:
    line 53: 0
    line 55: 3
    line 56: 11
    line 57: 13
    line 59: 23
    line 60: 27
    line 62: 31
    line 63: 39
    line 64: 45
```

<http://blog.csdn.net/diffate>

使用命令javap -p -v SuperClass.class

```
{
  org.zheng.jvmcmd.SuperClass();
  flags:
  Code:
    stack=1, locals=1, args_size=1
    0: aload_0
    1: invokespecial #1          // Method java/lang/Object."<init>":()V
    4: return
  lineNumberTable:
    line 6: 0

  public java.lang.String method();
  flags: ACC_PUBLIC
  Code:
    stack=1, locals=1, args_size=1
    0: ldc           #2          // String from SuperClass...
    2: areturn
  lineNumberTable:
    line 9: 0

  public void otherMethod();
  flags: ACC_PUBLIC
  Code:
    stack=3, locals=1, args_size=1
    0: getstatic     #3          // Field java/lang/System.out:Ljava/io/PrintStream;
    3: ldc           #4          // String In SuperClass otherMethod()...
    5: invokevirtual #5          // Method java/io/PrintStream.println:(Ljava/lang/String;)V
    8: getstatic     #3          // Field java/lang/System.out:Ljava/io/PrintStream;
    11: new           #6          // class java/lang/StringBuilder
    14: dup
    15: invokespecial #7          // Method java/lang/StringBuilder."<init>":()V
    18: ldc           #8          // String SuperClass otherMethod() calls method():
    20: invokevirtual #9          // Method java/lang/StringBuilder.append:(Ljava/lang/String;)Ljava/lang/StringBuilder;
    23: aload_0
    24: invokevirtual #10         // Method method:()Ljava/lang/String;
    27: invokevirtual #9          // Method java/lang/StringBuilder.append:(Ljava/lang/String;)Ljava/lang/StringBuilder;
    30: invokevirtual #11         // Method java/lang/StringBuilder.toString:()Ljava/lang/String;
    33: invokevirtual #5          // Method java/io/PrintStream.println:(Ljava/lang/String;)V
    36: return
  lineNumberTable:
    line 13: 0
    line 14: 8
    line 15: 36
}
```

<http://blog.csdn.net/diffate>

使用命令javap -p -v SubClass.class

```
[
  org.zheng.jvmcmd.SubClass();
  flags:
  Code:
    stack=1, locals=1, args_size=1
    0: aload_0
    1: invokespecial #1           // Method org/zheng/jvmcmd/SuperClass.<init>:()V
    4: return
  LineNumberTable:
    line 18: 0

public java.lang.String method();
  flags: ACC_PUBLIC
  Code:
    stack=1, locals=1, args_size=1
    0: ldc           #2           // String from SubClass...
    2: areturn
  LineNumberTable:
    line 21: 0

public void subMethod();
  flags: ACC_PUBLIC
  Code:
    stack=3, locals=1, args_size=1
    0: getstatic     #3           // Field java/lang/System.out:Ljava/io/PrintStream;
    3: new           #4           // class java/lang/StringBuilder
    6: dup
    7: invokespecial #5           // Method java/lang/StringBuilder.<init>:()V
    10: ldc           #6           // String SubClass calls super.method():
    12: invokevirtual #7           // Method java/lang/StringBuilder.append:(Ljava/lang/String;)Ljava/lang/StringBuilder;
    15: aload_0
    16: invokespecial #8           // Method org/zheng/jvmcmd/SuperClass.method:()Ljava/lang/String;
    19: invokevirtual #7           // Method java/lang/StringBuilder.append:(Ljava/lang/String;)Ljava/lang/StringBuilder;
    22: invokevirtual #9           // Method java/lang/StringBuilder.toString:()Ljava/lang/String;
    25: invokevirtual #10          // Method java/io/PrintStream.println:(Ljava/lang/String;)V
    28: aload_0
    29: invokespecial #11          // Method privateMethod:()V
    32: return
  LineNumberTable:
    line 25: 0
    line 26: 28
    line 27: 32

private void privateMethod();
  flags: ACC_PRIVATE
  Code:
    stack=2, locals=1, args_size=1
    0: getstatic     #3           // Field java/lang/System.out:Ljava/io/PrintStream;
    3: ldc           #12          // String This is private Method From SubClass
    5: invokevirtual #10          // Method java/io/PrintStream.println:(Ljava/lang/String;)V
    8: return
  LineNumberTable:
    line 30: 0
    line 31: 8
}
```

联系我



关于度

©1999京ICP

经营性网络1

中国互北京互

版权声明：本文为博主原创文章，欢迎转载，转载请注明作者、原文超链接，博主地址：http://blog.csdn.net/diffate。https://blog.csdn.net/diffate/article/details/70196210

文章标签：jvm java

个人分类：Java虚拟机

想对作者说点什么？

我来说一句

Nginx开发介绍

本课程是Nginx的入门课程，主要介绍Nginx服务的简介，快速安装，定制安装，Ngnix配置以及简单的模块开发流程。

学院 2017年12月24日 18:26

JVM规范学习：invokespecial

本文译自：invokespecial 操作 调用实例初始化，父类初始化和私有方法。 格式 invokespecial indexbyte1 indexbyte2 编码 invoke...

 lvvista 2014-08-10 12:56:02 阅读数：2131

多态在JVM中的实现——invokevirtual与invokespecial

先给一段程序public class BaseClass{ void supermethod() { System.out.println("superMethod"); } vo...

 qc_liu 2015-01-10 16:49:54 阅读数：2415

Java中各个方法：getstatic invokespecial invokeinterface invokevirtual invokestatic

invokespecial和invokevirtual两种指令

1.invokespecial只能调用三类方法：方法；private方法；super.method()。因为这三类方法的调用对象在编译时就可以确定。 2.invokevirtual是一种动态分派的调...

JVM规范学习：invokestatic

invokestatic 操作：调用一个类(static)方法。 格式： invokestatic indexbyte1 indexbyte2 编码： invokestatic = ...

码农怎能不懂英语？！试试这个数学公式

老司机教你一个数学公式秒懂天下英语



深入理解JVM内幕

每个Java开发者都知道Java字节码是执行在JRE(（Java Runtime Environment Java运行时环境）)上的。JRE中最重要的部分是Java虚拟机（JVM），JVM负责分析和执行...

JVM规范学习：invokeinterface

Invokeinterface 操作 调用接口方法 格式 invokeinterface indexbyte1 indexbyte2 count 0 编码 invokeinterf...

Java调用重载方法(invokevirtual)和接口方法(invokeinterface)的解析

先开坑

Java中的@interface以及method.invoke()

前言 这几天在看暑假要去实习的公司的框架源码，该框架十分轻量级，上手贼快。 Web容器是Jetty，数据库是MySQL，使用该框架形式为插件开发，只需要按照该框架可以识别的规则开发，再将项目放...

方法调用指令invoke...

1、方法调用（分派、执行过程）： JVM调用方法有五条指令，分别是（1）invokestatic,用来调用static方法（类方法）（2）invokespecial,用来调用需要特殊处理...

smali语法总结

smali语法总结

invokevirtual

先抄一段，见《深入理解java虚拟机 JVM高级特性与最佳实践》p254由于 invokevirtual 指令执行的第一步就是在运行期确定接收者的实际类型，所以两次调用中的invokevirtual ...

JVM规范学习：invokevirtual

invokevirtual 操作：调用实例方法，基于类进行分派 格式： invokevirtual indexbyte1 indexbyte2 编码： invokevirtual =...

50万码农评论：英语对于程序员有多重要？





自制Java虚拟机（五）实现继承、多态、invokevirtual

自制Java虚拟机（五）实现继承、多态、invokevirtual本篇文章将研究如何实现面向对象的继承和多态特性，同时实现invokevirtual。一、实例属性的继承继承实现了数据与方法的复用。类...

 chunyuan314 2017-06-07 19:12:27 阅读数：801

APK反编译之一：基础知识

APK反编译之一：基础知识 本人接触不久，有错误望请各位神牛不吝赐教，仅仅希望把自己这段时间研究的东西分享一下，如果可以帮助到有需要的童鞋万感荣幸。欢迎评论转载，但请加上转载来源谢谢！请尊重开发者劳...

 lpohvbe 2012-09-15 10:19:24 阅读数：31209

smali语法积累(1)

ZZ: <http://www.miui.com/thread-409543-1-1.html> 转载这篇文章的目的是学习内部类在smali的表现。感谢原创作者，感谢MIUI。 1. 为什么使用...

 crazyjiang 2013-04-15 18:00:09 阅读数：19434

Java 虚拟调用(virtual invoke)分析

此文章来分析下，Java 的虚拟调用 When we say Java language has virtual method calling we mean that in java applic...

 u014088294 2015-11-13 09:21:46 阅读数：643

invokevirtual , invokespecial , invokestatic , invokeinterface

invokevirtual , invokespecial , invokestatic , invokeinterface 博客分类： Java JavaJVM invokev...

 csluqiang 2014-05-05 10:53:24 阅读数：846

自制Java虚拟机（四）-对象、new、invokespecial

自制Java虚拟机（四）-对象、new、invokespecial一、对象的表示刚开始学Java的时候，图书馆各种教程，书名往往都是“Java面向对象高级编程”，通常作者都会与C++做个比较，列出的优...

 chunyuan314 2017-06-04 09:32:19 阅读数：1584