

曹胜欢

欢迎关注微信账号：java那些事：csh624366188.每天一篇java相关的文章

≡ 目录视图

≡ 摘要视图

RSS 订阅

★★★★交流QQ群★★★★

欢迎关注微信账号：java  
那些事：  
csh624366188.每天一  
篇java相关的文章

java交流工作群1：  
77800592（已满）  
java交流学生群2：  
234897635（已满）  
java交流工作群3：  
94507287  
java交流工作群4：  
272265434

我的邮箱：  
bzu901@163.com

PS：请勿同时加入多个  
群，一经发现，永久封  
号，谢谢！

个人资料



曹胜欢

+ 加关注 发私信



访问：3054445次

积分：30042

等级：BLOG &gt; B

排名：第128名

原创：210篇  
转载：41篇

译文：0篇  
评论：3688条

我的微博

## 原 Java程序员从笨鸟到菜鸟之（九十七）深入java虚拟机（六）——类加载的父亲委托机制

标签：java 虚拟机 classloader 扩展 class jvm

快速回复

2012-10-16 12:11 9314人阅读 评论(14) 举报

≡ 分类：

Java程序员从笨鸟到菜鸟（81） ▾ 学习专区（140） ▾ 深入jvm（7） ▾

| 版权声明：本文为博主原创文章，未经博主允许不得转载。

在前面两篇博客中我们简单介绍了类加载器的基础和类的生命周期的基础内容，今天我们来继续深入的来看一下Java的类加载器的详细内容。我们都知道。类加载器用来把类加载到java虚拟机。从JDK2.0开始，类的加载过程采用父亲委托机制。JVM的ClassLoader采用的是树形结构，除了根类加载器以外，每个ClassLoader都会有且仅有一个父类加载器，用户自定义的ClassLoader默认的父亲加载器是系统类加载器，当然你可以自己指定需要用个ClassLoader的实例，我们来看他们的父子关系：

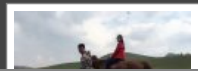


父类委托机制中，当一个java程序请求加载器loader1加载Hello类时，loader1首先委托自己的父亲加载器加载hello类，若父亲加载器能加载，则由附加器完成加载人物，否则才由加载器loader1本身加载Hello类。下面我们来再次看一下java虚拟机自带的几个加载器：

微博

曹胜欢  
+ 加关注

生活不止眼前的苟且.....还有诗和远方.....终于回来了。  
<http://t.cn/RU9K9NY>



TA 的粉丝 (1639)



李小净



希古敬



醉客



俺叫赵

## 博客专栏

深入 java  
虚拟机

深入java虚拟机

文章：8篇  
阅读：137164

jQuery

跟我学  
jquery文章：7篇  
阅读：65865

spring

细谈spring

文章：12篇  
阅读：134985

Hibernate

细谈  
Hibernate文章：18篇  
阅读：158437

Struts2

细谈struts2

文章：14篇  
阅读：138495

设计模式

大话设计模式

文章：8篇  
阅读：63427Java程序员  
从笨鸟到菜鸟Java程序员  
从笨鸟到菜鸟文章：112篇  
阅读：

Java 虚拟机自带了以下几种加载器。

- 根（Bootstrap）类加载器：该加载器没有父加载器。它负责加载虚拟机的核心类库，如 `java.lang.*` 等。例如从例程 10-4 (Sample.java) 可以看出，`java.lang.Object` 就是由根类加载器加载的。根类加载器从系统属性 `sun.boot.class.path` 所指定的目录中加载类库。根类加载器的实现依赖于底层操作系统，属于虚拟机的实现的一部分，它并没有继承 `java.lang.ClassLoader` 类。
- 扩展（Extension）类加载器：它的父加载器为根类加载器。它从 `java.ext.dirs` 系统属性所指定的目录中加载类库，或者从 JDK 的安装目录的 `jre\lib\ext` 子目录（扩展目录）下加载类库，如果把用户创建的 JAR 文件放在这个目录下，也会自动由扩展类加载器加载。扩展类加载器是纯 Java 类，是 `java.lang.ClassLoader` 类的子类。
- 系统（System）类加载器：也称为应用类加载器，它的父加载器为扩展类加载器。它从环境变量 `classpath` 或者系统属性 `java.class.path` 所指定的目录中加载类，它是用户自定义的类加载器的默认父加载器。系统类加载器是纯 Java 类，是 `java.lang.ClassLoader` 类的子类。

除了java虚拟机自带的加载器之外，我们用户自己也可以自定义自己的类加载器，根据自己的需要。。Java提供了抽象类`java.lang.ClassLoader`，所有用户自定义的类加载器都要继承这个`ClassLoader`类。

**注：**加载器之间的父子关系实际上指的是加载器对象之间的包装关系，而不是类之间的继承关系。一对父子加载器可能是同一个加载器类的两个实例，也可能不是。在子加载器对象中包装了一个父加载器对象。当生成一个自定义的类加载器实例时，如果没有指定它的父加载器，那么系统类加载器将成为该类加载器的父加载器。如果在构造方法中指定父类加载器那么父类加载器就是指定的加载器。证明如下：

```
protected ClassLoader()
```

使用方法 `getSystemClassLoader()` 返回的 `ClassLoader` 创建一个新的类加载器。将该加载器作为父类加载器。

如果存在安全管理器，则调用其 `checkCreateClassLoader` 方法。这可能导致安全性异常。

抛出：

`SecurityException` - 如果存在安全管理器并且其 `checkCreateClassLoader` 方法不允许创建新的类加载器。

## 方法详细信息

```
loadClass
```

```
public Class<?> loadClass(String name)  
    throws ClassNotFoundException
```

使用指定的二进制名称来加载类。此方法使用与 `loadClass(String, boolean)` 方法相同的方式搜索类。Java 虚拟机调用它来分析类引用。调用此方法等效于调用 `loadClass(name, false)`。

参数：

`name` - 类的二进制名称

返回：得到的 `Class` 对象

抛出：`ClassNotFoundException` - 如果没有找到类

```
[java] view plain copy print ?
```

```
01. ClassLoader loader1 = new MyClassLoader();  
02. //参数loader1将作为loader2的父加载器  
03. ClassLoader loader2 = new MyClassLoader(loader1);
```

JAVA

当Java虚拟机要加载一个类时，到底该派哪个类加载器去加载呢 ？我们看下图：

1376762



java一系列  
文章：0篇  
阅读：0

文章分类

- 框架Struts2 (18)
- java Web (12)
- java SE (11)
- IDE (1)
- 学习专区 (141)
- Lucene专区 (5)
- 杂谈 (39)
- 框架Hibernate (31)
- c# (4)
- 设计模式 (12)
- ASP.NET (4)
- Java程序员从笨鸟到菜鸟 (82)
- 数据结构 (2)
- jquery (10)
- 深入jvm (8)
- operate sys (1)
- 经典转载 (13)
- webservice (4)
- oracle (6)
- linux (5)

文章存档

- 2015年07月 (1)
- 2014年11月 (1)
- 2014年05月 (1)
- 2014年01月 (1)
- 2013年12月 (8)

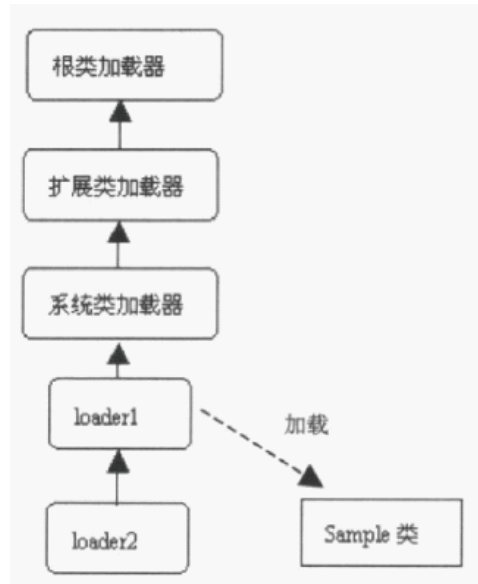
↓展开

阅读排行

- Entity Framework : (123411)
- Java程序员从笨鸟: (71736)
- Java程序员从笨鸟: (71302)
- Lucene教程详解 (64641)
- java程序员从笨鸟: (63664)
- 利用java实现的一 (60624)
- Java程序员从笨鸟: (58321)
- Java面试宝典2013 (57638)
- mvc与三层结构终 (55826)
- 我是怎么利用微信 (50712)

学习博客

【酷壳】



Loader1和loader2是我们自己定义的两个类加载器，loader1和loader2是父子关系。现在我们想让loader2这个类加载器加载我们自己写的一个Sample类：loader2.loadclass（“sample”），我们来分析一下看到底应该用哪一个类加载器去加载。当这段代码被执行时，loader2首先到自己的命名空间去查找Sample类是否已经被加载，如果被加载就直接返回这个类的class对象的引用。如果Sample类还没有被加载，loader2首先请求loader1代为加载，loader1再请求系统类加载器代为加载，系统类加载器再请求扩展类加载器，扩展类加载器再请求根类加载器，若根类加载器和扩展类加载器都不能加载，则系统类加载器尝试加载，若能加载，则将Sample类所对应的Class对象的引用返回给loader1，loader1在将引用返回给loader2，从而成功将Sample类加载到虚拟机。若系统类加载器不能加载Sample类，则loader1尝试加载Sample了哦，若loader1不能加载，则loader2尝试，若所有的类加载都不能加载，则抛出ClassNotFoundException异常。

**定义类加载器**：如果某个类加载器能够加载一个类，那么该类加载器就称作：定义类加载器；定义类加载器及其所有子加载器都称作：**初始类加载器**

父委托机制的优点就是能够提高软件系统的安全性。因为在词机制下，用户自定义的类加载器不可能加载本应该由父加载器加载的可靠类，从而防止不可靠的恶意代码代替由父类加载器加载的可靠类，从而防止不可靠的甚至恶意的代码代替由父类加载器加载的可靠代码。如，java.lang.Object类总是由根类加载器加载的，其他任何用户自定义的类加载器都不可能加载含有恶意代码的java.lang.Object类。

命名空间，其实这里所说的命名空间就是我们java中常用的package，每个类加载器都有自己的命名空间，命名空间由该加载器及所有父加载器所加载的类的组成。在同一个命名空间中，不会出现类的完整名字（包括类的包名）相同的两个雷；在不同的命名空间中，有可能会出现类的完整名字（包括类的包名）相同的两个类。

由同一类加载器加载的属于相同包的类组成了运行时包。决定两个类是不是属于同一个运行时包，不仅要看他们的包名称是否相同，还要看定义类加载器是否相同。只有属于同一运行时包的类之间才能相互访问可见（默认访问级别）的类和成员。假设用户自定义了一个类java.lang.TestCase并由用于自定义的类加载器加载，由于java.lang.TestCase和核心类库java.lang.\*由不同的类加载器加载，他们属于不同的运行时包，所以java.lang.TestCase不能访问核心库java.lang包中的包可见成员。

参考资料：[北京张龙老师免费培训视频《类加载器的父亲委托机制深度详解》](#)

《Java程序员由笨鸟到菜鸟》电子版书正式发布，欢迎大家下载



【陈勇老师】

【李晨光老师】

【纪争光】

【李守宏】

【畅之部落格博客】

【高爽Java】

【陈建秋】

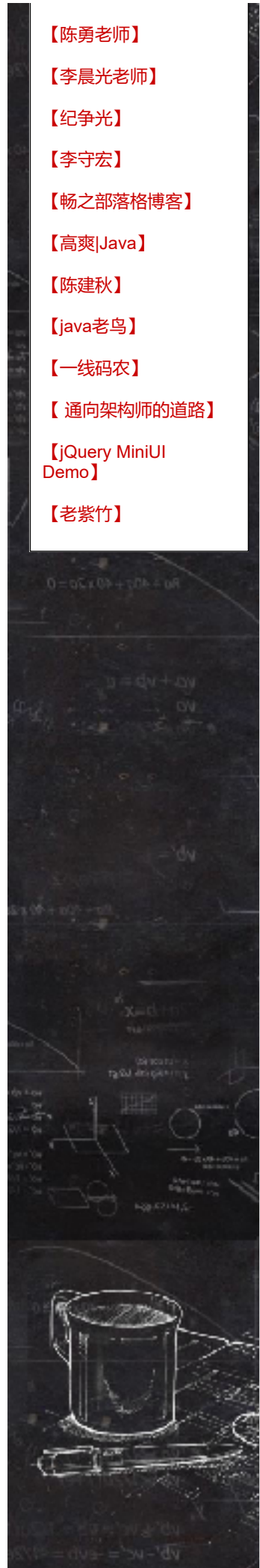
【java老鸟】

【一线码农】

【通向架构师的道路】

【jQuery MiniUI Demo】

【老紫竹】



<http://blog.csdn.net/csh624366188/article/details/7999247>

顶11

踩8

上一篇Java面试宝典2013版（超长版）

下一篇

Java程序员从笨鸟到菜鸟之（九十八）深入java虚拟机（七）深入源码看java类加载器ClassLoader

我的同类文章

Java程序员从笨鸟到菜鸟（81）

学习专区（140）

深入jvm（7）

猜你在找

查看评论

8楼YOYO3047 2012-10-22 08:17发表

我是笨鸟，努力向菜鸟靠近！

7楼Sean、萌萌 2012-10-20 14:17发表

你叫老曹，我也叫老曹啊！

Re: 曹胜欢 2012-10-20 15:09发表

回复Sean、萌萌：呵呵。。一家人

6楼ZYF902835 2012-10-18 23:21发表

老曹，我来瞧瞧啦，哈哈。

Re: 曹胜欢 2012-10-19 16:24发表

回复ZYF902835：呀呀。。。稀客啊。。峰哥啊。。

5楼li371605357 2012-10-18 18:18发表

如果你真是2011年才开始涉及到软件编程，能有这样的水平，写出这些东西来，那我就真的佩服你了。我自认为是望尘莫及了。

Re: 曹胜欢 2012-10-18 18:31发表

回复li371605357：2010年9月开始接触计算机，以前只会开机，浪费了半年大一生活后，2011年开始自学。。

4楼wangboshichina 2012-10-16 20:18发表

好

3楼郑明 2012-10-16 15:53发表

好哦，学习下哦。

Re: 曹胜欢 2012-10-16 16:57发表

回复郑明：共同学习。。

2楼我愿为一朵桔梗花 2012-10-16 15:46发表

好



C

Re: 曹胜欢 2012-10-16 16:57发表

 回复我愿为一朵桔梗花：共同学习。。

1楼 曹胜欢 2012-10-16 13:05发表

 发现越来越难写了。。。

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

\* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题

Hadoop

AWS

移动游戏

Java

Android

iOS

Swift

智能硬件

Docker

OpenStack

VPN

Spark

ERP

IE10

Eclipse

CRM

JavaScript

数据库

Ubuntu

NFC

WAP

jQuery

BI

HTML5

Spring

Apache

.NET

API

HTML

SDK

IIS

Fedora

XML

LBS

Unity

Splashtop

UML

components

Windows Mobile

Rails

QEMU

KDE

Cassandra

CloudStack

FTC

coremail

OPhone

CouchBase

云计算

iOS6

Rackspace

Web App

SpringSide

Maemo

Compuware

大数据

aptech

Perl

Tornado

Ruby

Hibernate

ThinkPHP

HBase

Pure

Solr

Angular

Cloud Foundry

Redis

Scala

Django

Bootstrap

公司简介 | 招贤纳士 | 广告服务 | 银行汇款帐号 | 联系方式 | 版权声明 | 法律顾问 | 问题报告 | 合作伙伴 | 论坛反馈

 网站客服

 杂志客服

 微博客服

 [webmaster@csdn.net](mailto:webmaster@csdn.net)

 400-600-2320 | 北京创新乐知信息技术有限公司 版权所有 |

江苏知之为计算机有限公司 | 江苏乐知网络技术有限公司

京 ICP 证 09002463 号 | Copyright © 1999-2016, CSDN.NET, All Rights Reserved 