DERBY源码结构

在通过官网获取DERBY源码包中，可以获得整个DERBY体系的结构，从

org.apache.derby.drda中作为启动DERBY数据库的入口，下面将从入口开始进入DB的世界，看看一个关系数据库从中的结构

在ORG.APACHE.DERBY.DRDA包下的NetworkServerControl这个文件作为启动的入口，分为了以下方法构成，从这些方法中可以进行深入解构

方法

方法

解释

getCodeSourcePrefix( NetworkServerControlImpl server )

获取作为DERBY主要JAR包的DERBYNET和DERBY在LIB的URL，进行假定存在策略

getCurrentProperties()

获取当前的网络服务器属性

getHostNameForSocketPermission( NetworkServerControlImpl server )

获取本地地址与默认服务器进行匹配

getMaxThreads()

用于返回正在运行网络服务器的最大线程

getPolicyFileURL()

获取政策文件的URL

getRuntimeInfo()

用于返回在SESSION下的运行信息

getSysinfo()

用于返回DERBY工具下的信息详情

getTimeSlice()

返回当前时间片的运行网络服务器设置

hostnamesEqual( String left, String right )

返回两个等价主机名

installSecurityManager( NetworkServerControlImpl server )

用于安装安全管理政策文件

isIPV6Address( String hostname )

验证IPV6地址的合法性

logConnections(boolean on)

打开或关闭日志记录连接

main(String args[])

从方法为整个文件的入口，下面会详细介绍此MAIN方法

needsSecurityManager( NetworkServerControlImpl server, int command )

验证是否需要安装安全管理政策

ping()

检测网络服务器是否启动

setClientLocale(String locale)

设置服务器集

setMaxThreads(int max)

写入最大线程数

setTimeSlice(int timeslice)

设置网络服务器连接时间片参数。

setTraceDirectory(String traceDirectory)

设置跟踪文件目录

getSysinfo()

获取系统信息

shutdown()

关闭

start(PrintWriter consoleWriter)

用于启动网络服务器，将启动单独的线程启动。

trace(boolean on)

在网络服务器中为指定的连接打开或关闭

trace(int connNum, boolean on)

打开或关闭网络服务器上所有连接的跟踪。

verifySecurityState( NetworkServerControlImpl server )

该方法用于验证安全管理的合法性

这些作为启动DERBY的主要方法，下面将详细深入方法，此外，文件也设置了一些产量用于对路径的访问，分别是

常量

属性

解释

DEFAULT\_PORTNUMBER = 1527;

作为DERBY的默认端口号，方便进行连接

DERBYNET\_JAR = "derbynet.jar";

获取主要JAR包的位置,必须精确地获取改路径，否则将导致整个程序无法启动！

POLICY\_FILENAME = "server.policy";

用于获取政策文件属性

DERBY\_HOSTNAME\_WILDCARD

用于获取DERBY本地的通配符

IPV6\_HOSTNAME\_WILDCARD = "::";

IPV6本机的通配符

SOCKET\_PERMISSION\_HOSTNAME\_WILDCARD = "\*";

SOCKET本机验证通配符

private NetworkServerControlImpl serverImpl;

以及调用控制网络服务器的数据访问层

下面将详细的进行深入，首先通过MAIN方法

程序的入口-MAIN

代码块

Public static void main(String args[]){

NetworkServerControlImpl server = null;

Boolean printErrors =true;

Try{

Server = new NetworkServerControlImpl();

Int command = server.parseArgs(args);

If(needSecurityManager(server,command)){

verifySecurityState(server);

InstallSecurityManager(server);

}

printErrors =false;

server.executeWork(command);

}

Catch(Exception e){  
 if((e.getMessage()==null|| !e.getMessage().equals(NetworkServerControlImpl.printErrors))){

If(server!=null)

server.consoleExceptionPrint(e);

Else

e.printStackTrace();

}

System.exit(1);

}

System.exit(0);

此为入口的例程，首先通过NetworkServerControl控制类进行实例，通过此类进行初始化，在进入INIT()这个方法时，

DERBY通过一个ModuleFactory接口去监视器为系统中的所有模块提供中央注册表.，并管理加载，启动和查找它们。

为了验证初始服务器的合法性，DERBY通过一个ProductVersionHolder类去获取版本的合法性以确保安全管理下机制的生成。

当确定好后，会进入 getPropertyInfo()方法去系统属性初始化字段，

DERBY通过建立一个工厂类PropertyUtil去声明。

此段通过调用访问类来进行数据访问，

此方法用于解析命令行参数并执行参数，通过executework()方法去调用参数并返回结果COMMAND；

在DERBY中需要一个安全管理机制去管理数据库，为了数据库本身的安全性和对数据本身的保护机制，需要通过验证来启动安全管理机制

此方法在启动网络服务器时通过方法，验证在本机是否存在安全管理机制，如果存在进行验证，否则采取安装

通过以上运行时，在确保无差错后，程序将会自动生成日志文件去监控程序的进行时，一切就绪后将在系统目录中生成以下信息