JThink-Framework

JAVA 应用系统开发框架(Java Application System Framework)

发起者：wenjian
机构：自由思考组织(Free Think Organization - FTO）
主页：www.free-think.org
版本：1.0

主要开发人员：wenjian

目 录

前言

如果你是一个有经验的程序员，并对模式、框架有一定认识，可以跳过前言部分。

在讲 JThink-Framework 之前，完全有必要先讲述一下计算机程序设计的演变过程。我们知道，在计算机内部的程序代码，全是由一系列的二进制码来描述的。最初的编程，也就是按照特定的规则，编写出计算机能够识别的一组二进制码。
比如：我们要让电脑计算 2+3=?，在计算机内的二进制码表示为：
10111000 0000001000000000
00000101 0000001100000000
十六进制表示法：
B8 0200
05 0300
汇编语言表示法：
MOV AX,2
ADD AX,3

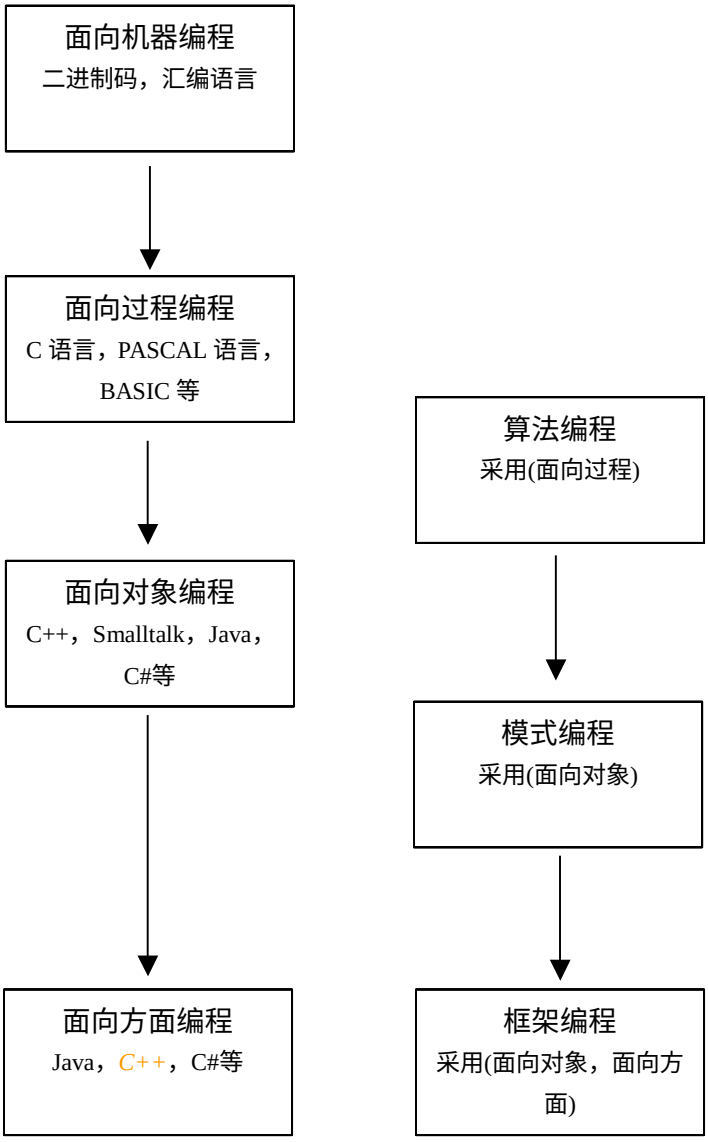
10111000是 CPU的一条复制指令，功能就是将与此指令紧挨着（后继）的两个字节的数据复制到 AX寄存器（寄存器可以理解为在 CPU里用于临时存放数据的地方）。

00000101是 CPU的一条加法指令，功能是将与此指令紧挨着（后继）的两个字节的数据与 AX寄存器中的数据相加，并将结果返回到 AX寄存器中。

同样，如果我们还想将 2+3的结果 5与其它数值进行计算，比如（2+3）*4=?，这样就还得使用乘法指令将 AX寄存器中的结果 5与 4相乘，这样就可以进行一系列的复杂运算了。然后将结果写入到显示器的显示缓冲区，让显示器将结果显示出来，写入到打印机的打印缓冲区，让打印机将结果打印出来。

我们可以想象世界第一台电子计算机研发出来后，那些人是如何编写程序并将程序输入到计算机里面的。在没有象现在的 C语言， Java语言，以至于那个时候连汇编语言都还没有开发出来的情况下，进行程序设计难度是可想而知的。

计算机程序的先驱们，为了提高程序编写的速度、减少在编写程序过程中出现的问题，开发出来了一系列的编程语言。从最初的低级语言汇编语言，到现在的高级语言 C、C++、Java等。编程语言在发展，同样，随着编程语言的发展，程序设计方法也在发展。以下描述了编程语言和程序设计方法的发展过程：



上图说明了编程语言和程序设计方法的进化过程，可以看出，程序设计方法与编程语言是有一定关系的，我们要进行面向对象的程序设计，就必须得采用支持面向对象的编程语言，比如：C++、Java等。越到后来，编程语言的进步已经变得非常缓慢，而程序设计方法确在不段的进步。在过程化编程时代，人们在编程过程中，为了提高开发效率，对于解决一些在程序中经常会遇到的问题，总结出了一些有效的方法，就叫做算法。算法的定义是：在有限步骤内求解某一问题所使用的一组定义明确的规则。算法已经形成了一些通用标准，比如：比较常用到的数据排序算法，递归算法，与树的操作相关的算法，与图操作相关的算法等。后来，在面向对象编程时代，人们总结出了面向对象编程的常用方法，即模式。模式的定义是：描述在我们周围不断重复发生的、性质比较相近似的问题，以及该问题的解决方案。模式也已经形成了一套标准，比如：比较常用的对象创建型模式，抽象工厂（Abstract Factory），

生成器 (Builder),单件 (Singleton)；行为模式，命令 (Command),迭代器 (Iterator)；结构型模式，适配器 (Adapter),代理 (Proxy)等。现在，人们为了能快速的解决特定领域的问题，总结出了更能提高程序开发效率的方法，框架。框架的定义：构成一类特定应用软件可复用设计的一组相互协作的类，以及一套采用此框架进行编程的规范。比如：解决数据库的访问层面的，有 Hibernate,JDO, J2EE中的数据持久层；解决 WEB应用显示层面的，有 Struts 等；解决业务逻辑层面的，有 Spring,还有就是我们接下来要讲的 JThink-Framework。

算法，模式，框架之间是相互关系的，模式中会用到算法，框架中会用到模式和算法。我们可以理解为点、面、全的一种关系。算法是解决某一个问题点；模式是解决一方面的问题，而这方面的问题的真正实现会用到大量的算法；框架是用来解决一类集体的问题，当然，它里面可能会用到多种模式。

另外，算法与模式是不会局限于某种编程语言的。算法的描述可以采用具体的编程语言，也可以采用一种自定义的伪语言,但它们都可以非常方便的转换为具体的某种编程语言来加以实现。模式主要采用面向对象程序设计思想的图形化，并加上必要文字来加以描述，描述模式时不会关心采用什么语言来实现，但它可以用任何支持面向对象的编程语言来实现，在实现过程中还可以根据问题的具体情况对模式加以改进以适应当前问题。

框架则与具体的问题具体的编程语言相关，比如 Hibernate 它的职责就是解决数据存取问题，并且必须由 Java语言来开发，因为框架本身就已经提供了解决问题的具体类。框架规定了应用的体系结构。它定义了整体结构，类和对象的关系，各部分的主要职责，类和对象怎么协作，以及控制流程。框架预定义了一些设计参数，以便于应用设计者或实现者能集中精力于应用本身的细节。框架记录了其应用领域的共同设计决策,因而框架更强调设计复用。

JThink-Framework 就是为了解决 JAVA 应用系统在开发过程中的一系列问题所发起的一个框架。她的主要目的是用于解决 JAVA应用系统中业务逻辑层面中反复遇到的问题。比如事务处理，数据访问，日志处理等。

要求

为了能良好的学习和使用 JThink-Framework，要求至少对以下内容有一定认识和理解。

JAVA	你应该是一个 Java程序开发人员。如果你是一个非 Java程序员，且正准备转向用 Java语言来开发，建议先用 Java开发几个例子应用后，再采用 JThink-Framework 来开发。如果你还不是一个程序员，也不打算将来做程序员，那还是放弃吧。
SERVLET/JSP	如果你准备用 JThink-Framework 来开发 J2EE/WEB应用，你应该开发过至少 1个 WEB应用。
EJB	如果你准备用 JThink-Framework 来开发 J2EE/EJB应用，你应该对 EJB规范有一定了解，并开发过至少 1个 EJB应用。
JDBC/SQL	如果你准备用 JThink-Framework 来操作数据库，你应该至少对标准 SQL比较熟悉，能通过 JDBC连接到至少 2种数据库，并对其进行数据操作。
面向对象	在你以前开发的应用系统中，已经良好的运用了面向对象编辑相思，能深入理解接口编程。
模式	能良好的应用现有的标准设计模式，能开发专用模式更好。

框架　　有其它框架使用经验者。比如：Struts，Spring，Hibernate等。

职责

1. 规范开发过程

规范开发过程在团队协作开发中非常重要，可以想象如果一个团队没有规范开发过程，每个程序员都按照自己的经验习惯编码，最后将程序汇集到一起后会是一个什么样的效果。JThink-Framework 中除了提供一系列 API以外，还规范了应用系统的开发过程，虽然它不会强制开发者按照这种方式来编程，但它能对使用好 JThink-Framework 提供最好的建议。

2. 促进团队协作

大中型应用系统的开发，往往会是由一个或多个团队来协同工作，合理的工作安排是项目良性发展的基础。此外，还必须要有一种良好的机制让各团队成员之间的工作能协调进行，不至于出现一部分人忙得很，而另一部分人闲得很。JThink-Framework 中 JThink/EJB规范了企业级应用的整个开发过程。它能方便的协调系统各层面 (展示层、控制层、逻辑层)同时进行开发，最大限度的提高开发效率。

按照 JThink-Framework 建议的标准来开发。统一了系统结构和编码风格，程序员之间的工作可以相系切换， 一个新人也能非常快的接手前人的工作。在当今程序员流动非常频繁情况下，这是非常有意义的。

3. 提高代码可复用性

框架本身就是一组可高度复用的类的集合。另外，通过 JThink-Framework 开发的应用模块，更佳具有可复用特性，特别是在 JThink/EJB规范下开发的 EJB组件。

4. 提高开发效率

框架设计的目录之一，就是为了最大限度的提高开发效率。规范开发过程，促进团队协作，提高代码可复性等，这些最终都可表现为对开发效率的提高。

5. 增强产品的环境适应能力

JThink-Framework 中提供的功能大多具有非常强的可配制的特性，通过简单的配制就可可以适应不同的运行环境。比如，在不修改任何程序代码的情况，就可以可以将遵循 JThink/EJB规范开发的 EJB组件或 J2EE应用，发布到支持 J2EE/EJB规范的应用服务器 (比如：JBoss，Weblogic，Websphere)中，也可以将其发布到不支持 J2EE/EJB规范的应用服务器 (比如：Tomcat)中。

JThink-Framework 采用开放性的设计，也就是说，如果你对 JThink-Framework 中的具体设计和实现不满意或不能达到应用要求，你完全可以对其进行重新设计和实现。比如日志管理，JThink-Framework 定义了一套日志管理的接口，提供了一套最基本日志管理的实现，但如果这样的日志管理的实现不能满足应用要求，你完全可以对日志管理接口重

新实现得来使用。比如实现来支持 LOG4J等日志管理中间件。

6. 其它

面向对象的程序设计最顶层，也就是面向应用的层面，实际上大多是面向接口编程。比如传统的数据库编程中使用的 JDBC，就是一套由 SUN定义的访问数据库的标准接口，不同的数据库提供商，需要对 JDBC接口进行单独实现。还有就是大家应该都比较熟悉的 J2EE/EJB组件编程等。同样，JThink-Framework 也采用了这种设计方法，为相应的功能模块定义了一套接口。比如有：请求处理，事务处理，数据源连接处理，数据访问，EJB组件开发，EJB组件访问，日志处理，资源管理等。

采用模式驱动，JThink-Framework 中的每一个功能模块，都有其详细的模式与之匹配。这对框架的发展，学习，应用起到了非常好的促进作用。

实现

1. JThink-Framework 中的产品开发方式

1) J2EE快速应用开发

简单 WEB应用开发，比如普通网站，小型企业应用，简单订单管理等。
开发结构： JSP+JavaBean+DB或 JSP+Servlet/JavaBean+DB。
中间件： JThink或 Struts + JThink等。
应用服务器：一般采用 Tomcat等小型 WEB应用服务器。

2) J2EE企业应用开发

大中型 WEB应用开发，基于 B/S的企业级产品应用，比如：OA系统，ERP系统，ITIL/ITSM服务管理系统，Project项目管理系统等。
开发结构： JSP+JavaBean+EJB+DB或 JSP+Servlet/JavaBean+EJB+DB。
中间件： JThink 或 Struts + JThink + Hibernate 或 Struts + JThink + JThink/J2EE/EJB或 Struts + JThink + JThink/J2EE/EJB + Hibernate等。
应用服务器：JBoss, Weblogic, Websphere 等。

3) 桌面、Applet、客户/服务器(C/S)应用开发

这种应用实际上要根据应用本身的情况来选择一种合理的开发方式，但还是建议采用以下方式。
开发结构： GUI+EJB或 GUI+EJB+DB
中间件：JThink或 JThink+Hibernate或 JThink+JThink/EJB或 JThink+JThink/EJB + Hibernate等。
应用服务器：无

2. JThink-Framework 实现的功能

JThink-Framework 实现的功能有：资源管理，请求处理，事务处理，连接数据源，数据访问，EJB组件开发，EJB组件访问，日志处理等。当前版本主要就支持这些功能，在以后的版本中将会融入更多特性，比如：国际化处理，面向方面编程的支持等。

1) 资源管理

我们可以将应用系统中的所有事物想象为是一个个的资源，而资源管理器的职责是负责创建，管理，释放资源，在需要的时候可以直接从资源管理器中获取资源。资源管理相关的类型有：

ResourceManager	类，资源管理器，负责对资源的增、删、查功能。
ResourceContainer	接口，资源容器， 存储资源的地方，资源的生命周期直接与资源容器的生命周期相关。在 JThink中的资源容器有：Request,Transaction,JDBCTransaction, ApplicationContext, DefaultJDBCTransaction, DefaultRequest, EJBApplicationContext, HttpPageContext, HttpSession, WEBApplicationContext
ResourceFactory	接口，资源工厂，负责创建资源 ,可以实现此工厂来创建资源。

了解资源管理更多信息，链接到： JThink-Framework模式 -资源 .doc

2) 请求处理

请求是应用系统中整个事务处理过程中的开始部分，将请求数据和请求的处理规范化非常重要。由 Client 端通过鼠标、键盘或在 WEB 应用中的提交表单等来触发一个请求，由具体的请求对象封装请求数据，然后依次传递给请求处理机。一个请求可由多个请求处理机处理，由请求处理总线来决定是否终止请求的传递。与请求处理相关的类型有：

Request	接口，用于描述一个请求，根据应用实际情况做具体实现。可以将此请求接口的具体实现作为资源容器(ResourceContainer)使用，因为此接口扩展了资源容器接口。在 JThink中请求的实现有 DefaultRequest，GUIRequest，HttpRequest。
DefaultRequest	类，*Request* 的默认实现。
GUIRequest	类，*Request* 的具体实现，主要用于描述桌面应用中的窗口数据信息，它可以自动的将指定窗口中的大部分用户输入的数据信息收集到 GUIRequest 中。
HttpRequest	类，*Request* 的具体实现，Http 请求，相当于 javax.servlet.http.HttpServletRequest ，只是在 HttpServletRequest 的基础上加入了些新特性。可以适应不同类型的请求，比如流数据方式的请求(上传文件的情况), HttpRequest 能自动收集请求字段；可以再次向 HttpRequest 中加入请求参数值；可以从 HttpRequest 中返回文件上传对象 FileUpload。
RequestProcessBus	类，请求处理总线，用于传递请求给具体的请求处理机。
RequestProcessor	接口，请求处理机，用于处理请求信息。

JThink-Framework 的请求处理为我们解决实际应用系统中的问题提供了一种模式和建议。了解请求处理更多信息，链接到： JThink-Framework模式 -请求 .doc

3) 事务处理

JThink-Framework 提供了唯一的事务管理抽象，它能够在各种底层事务管理技术，例如 JTA 或者 JDBC 事务，以及 EJB 容器管理的事务，提供一个一致的编程接口。与事务相关的类型有：

TransactionManager	接口，事务管理器。
DefaultTransactionManager	类，事务管理器接口的默认实现。
Transaction	接口，事务。在 JThink 中实现了些接口的事务有：DefaultJDBCTransaction，JTATransaction ，EJBContainerTransaction。另外 *JDBCTransaction* 扩展了此事务接口。
TransactionFactory	接口，事务工厂，主要用于创建事务。在 JThink 中实现了些接口的事务工厂有：DefaultJDBCTransactionFactory, JTATransactionFactory,EJBContainerTransactionFactory。
TransactionListener	接口，事务监听器。
TransactionEvent	类，事务事件对象

了解事务处理更多信息，链接到：JThink-Framework模式 -事务 .doc

4) 连接数据源

建立数据库连接的方式方法有多种多样，但最终目的都是要能正确返回数据库连接(Connection)。这里讲的数据库连接是指 JDBC 中的数据库连接接口(java.sql.Connection)。JThink已经提供了多种建立数据库连接的方法，我们可以使用统一的界面来创建和管理连接，可以在应用系统运行期间动态改变建立数据库连接方式。

ConnectionFactory	接口，连接工厂，可以根据实际情况实现此接口，以便能采用多种方式创建数据库连接。在 JThink中已经实现了常用的创建数据库连接的方式：DirectConnectionFactory。JNDIConnectionFactory。
DirectConnectionFactory	类，连接工厂，通过 JDBC驱动直接连接数据源并建立数据库连接。当采用此连接工厂时，建议配制使用连接池管理连接，在配制文件中设置。
JNDIConnectionFactory	类，连接工厂，通过 JNDI查找数据源并建立数据库连接。此种方式连接数据源的具体信息在应用服务器中配制，当前大多数应用服务器已经支持数据源配制。比如：Jboss,Webshpere,Weblogic等 ,Tomcat也已经支持数据源配制。由于在应用服务器的数据源配制中已经支持池连接管理，所以没有必要 (不推荐)在 JThink的配制文件中再使用连接池。
ConnectionPool	类，连接池，管理数据库连接 (java.sql.Connection)。
JDBCTransaction	接口， JDBC事务接口。此接口中提供了建立并返回数据库连接的方法： getConnection(String connId)，在具体应用系统中不再关心数据库连接 (java.sql.Connection)的创建和管理，这些都交由 JThink-Framework 来处理，但它又能充分利用应用服务器的特性。此接口的具体实现有：DefaultJDBCTransaction，JTATransaction ，

EJBContainerTransaction。

了解更多信息，链接到：JThink-Framework模式 连接数据源 .doc

5) 数据访问

JThink-Framework 提供了一套访问数据库数据的方法，它能让程序充分利用不同数据库本身的特性，但又能良好的实现将同一应用系统连接到不同的数据库。

SQLBuilder	类，用于构建 SQL。提供有常用的 SQL语句构建方法，例如：constructSQLForInsert()、constructSQLForUpdate()、constructSQLForDelete()、constructSQLForSelect()、constructSQLForCount()等。通过扩展 SQLBuilder类型，可以构建包含具体数据库特性的 SQL语句。为可能要用到的数据库编写 SQLBuilder的扩展类型，从而实现同一应用连接到不同的数据库。在 JThink中扩展此类型的类有：MssqlSQLBuilder、MysqlSQLBuilder。
SQLBuilderFactory	接口，用于建立 SQLBuilder的具体实例。在具体应用中要根据不同的数据库系统实现此接口来创建 SQLBuilder的扩展类型。
DefaultSQLBuilderFactory	类，默认 SQLBuilder工厂。实现了接口：SQLBuilderFactory
SQLExecutor	类，执行由 SQLBuilder构建的 SQL语句。要实现在不同数据库间移植应用，也要为具体数据库扩展 SQLExecutor类型，以便复盖在 SQLExecutor中没能实现的方法。在 JThink中 SQLExecutor的扩展类型有：MssqlSQLExecutor、MysqlSQLExecutor。
SQLExecutorFactory	接口，用于创建 SQLExecutor的实例。如果扩展了SQLExecutor类型，同样也要为此扩展类型实现SQLExecutorFactory接口。在 JThink中SQLExecutorFactory的实现有：DefaultSQLExecutorFactory、MssqlSQLExecutorFactory、MysqlSQLExecutorFactory。
SQLExecutorListener	接口，SQLExecutor监听器。在执行 SQL语句之前将调用监听器的 executeSQLCommand()方法。
SQLExecutorEvent	类， SQLExecutor监听器事件。
ResultMaker	接口，用于生成 SQLExecutor执行 SQL语句后的结果，可以实现此接口来生成不同形态的结果集。当前在 JThink中实现了用于生成 org.jdom.Element XML结果集的ElementResultMaker类型。
SQL	类，描述一个 SQL语句的相关信息。
Column	类，描述 SQL语句中被选择的列。
Condition	类，描述 SQL语句中的条件。
ConditionItem	类，描述 SQL语句中的条件中的一个条件项。

了解更多信息，链接到：JThink-Framework模式 数据访问 .doc

6) JThink/EJB组件开发

JThink的 EJB组件开发同样遵循 Sun J2EE/EJB 2.0的标准规范。并且，为了让应用系统更佳具有环境适应能力， JThink在 Sun J2EE/EJB标准基础上，加入了一些新特性。遵 JThink/EJB规范开发的 EJB组件或应用系统，可以不做任何程序代码的修改就可以部署到支持或不支持 Sun J2EE/EJB的企业级应用服务器上。比如：JBoss,Websphere或 Tomcat。

EJBHome	接口， JThink/EJB的 Home主类的超接口，所有遵循 JThink/EJB规范的 EJB主 Home类必须实现此接口。注意，在 J2EE/EJB中的主 Home是定义为接口，而在 JThink/EJB中定义为类。
EJBObject	接口，*JThink/EJB* 的 *Bean* 对象接口的超接口。
EJBApplicationContext	类,应用程序 EJB端上下文资源容器，此容器中的资源在一定范围内是共享的，这要根据 EJB模块 ID(ejbModuleId)来决定，因为在 EJB容器端里没有象在 WEB容器端的那种 ServletContext WEB端全局资源容器，所以只能自定义一个 EJB端的资源容器来用，但必须要将不同的 EJB端资源容器区别开，不然容易引起资源的冲突。
EJBContainerTransaction	类,容器管理的 EJB事务。如果是容器管理的 J2EE/EJB组件，推荐使用此事务，虽然所有的 J2EE/EJB应用服务器都支持 JTA事务。
EJBContainerTransactionFactory	类， EJBContainerTransaction工厂。
JTATransaction	类，JTA事务。如果 J2EE应用服务器的 WEB容器端支持 JTA事务，推荐使用此事务。
JTATransactionFactory	类， JTATransaction工厂。

了解更多信息，链接到：JThink-Framework模式 -EJB组件开发 .doc

7) JThink/EJB组件访问

Sun J2EE/EJB 2.0中的 EJB组件可以部署到远程应用服务器或本地应用服务器上，并且要为 EJB组件的访问提供远程接口和本地接口。同样，如果是遵循 JThink/EJB规范，还必须提供 JThink/EJB Bean接口。由于 Sun J2EE/EJB 2.0中 EJB组件的远程接口和本地接口是不同的类型，在客户端无法用统一的界面来访问 EJB组件，所以,在 JThink/EJB中，EJB组件的远程接口和本地接口必须扩展 JThink/EJB Bean接口。这样 EJB组件的远程接口和本地接口以及 JThink/EJB Bean都具有了相同界面，从可以实现将 EJB组件部署到不同的应用服务器，并且可以在远程与本地以及非支持 Sun J2EE/EJB的应用服务器之间自由切换。

EJBCaller	类，用于查找 EJB的主接口。 具体应用中建议扩展此类型来查找 EJB主接口，这样方便在不同 EJB容器环境中自动切换。
EJBHomeFactory	接口， EJBHome工厂，通过实现 EJBHomeFactory来返回 J2EE/EJBHome接口或 JThink/EJBHome类。
EJBLocalHomeFactory	类， Sun J2EE/EJB组件本地 Home接口工厂。
EJBRemoteHomeFactory	类， Sun J2EE/EJB组件远程 Home接口工厂。
JThinkEJBHomeFactory	类， JThink/EJB组件 Home类工厂。

了解更多信息，链接到：JThink-Framework模式 -EJB组件访问 .doc

8）日志处理

在 JThink-Framework 中定义了一套日志处理接口以及它的默认实现。如果有必要，还可以重新实现来使用其它日志管理组件，如 Log4j 等。

LogManager	类，日志管理器。
Logger	接口，日志处理主接口，提供写日志相关的方法。
LogFactory	接口，日志工厂。
Priority	类，描述写日志的优先级别。
DefaultLogger	类，默认日志处理接口的实现。
DefaultLogFactory	类，默认日志工厂实现。

了解更多信息，链接到：JThink-Framework模式 -日志 .doc

平台

J2SE 1.3/1.4/1.5 JThink-Framework 需要在 J2SE 1.3 及其后继版本中才能正常运行。

J2EE 1.3/1.4 如果开发 J2EE 应用，需要 J2EE 1.3 (Servlet 2.3, JSP 1.2, JTA 1.0, EJB 2.0) 的支持或后继版本的支持。

组件

在 JThink-Framework中使用到的第三方库：

必须组件		
名称	版本	描述
jdom.jar	1.0	用于处理 XML 文档的一个开源项目。 在经历了 10 个版本的 BETA 版后，于 2004 年 9 月发布了正式版本 1.0 版。提供者：jdom.org
xerces.jar	2.6.0	遵循了 W3C 标准的一套处理 XML 的 API。JDOM 须要此组件支持。提供者：Sun Microsystems Inc.，Apache Software Foundation，World Wide Web Consortium。
jakarta-regexp-1.3.jar	1.3	一个处理正则表达式的开源项目。在 J2SE 1.4 级其以后版本中已经提供了对正则表达式的支持，但 JThink-Framework 设计来要支持 J2SE 1.3.1/J2EE 1.3.1 版，所以引入了第三方库。提供者：Apache Software Foundation
可选组件		

名称	版本	描述
mail.jar	1.3	处理邮件的一套API。提供者：Sun Microsystems, Inc.
mysql-connector-java-3.0.16-ga-bin.jar 或 **mysql-connector-java-3.1.10-bin.jar**	3.0.16/ 3.1.10	用于连接到 MYSQL 数据库的 JDBC 驱动，3.0.16 版本可以运行在 J2SE 1.31 上，但不支持访问 Mysql 5.0 的存储过程，3.1.10 版本只能运行在 J2SE 1.4 或后继版本上，但支持访问 Mysql 5.0 的存储过程。提供者：MySQL AB。
msbase.jar **mssqlserver.jar** **msutil.jar**	1.2.2	用于连接到 MSSQL 数据库的 JDBC 驱动。 提供者：Microsoft。
其它		其它组件，在实际应用开发过程中，会用到的其它组件，比如用于访问 DB2 数据库的 JDBC 驱动等。

结束语

JThink-Framework 是一个年青的有活力的 Java应用系统开发框架，还有待持续的发展。非常欢迎您的积极参与和支持，提出你的宝贵意见和建议，为 JThink-Framework 的发展注入新的思想和血液，共同努力打造一个完美的、特色的 JThink-Framework。

提交 BUG，意见建议：jthink_tasks@yahoo.com.cn