JavaOne℠

Sun's 2004 Worldwide Java Developer Conference℠

JDBC™ 4.0 API—新一代应用程序编程接口

Java 规范需求-221(JSR-221)

java.sun.com/javaone/sf

Jonathan Bruce
JDBC技术规范的领导者和设计者,Sun
John Goodson
产品运营副总裁, Data Direct技术。

Sun microsystems

讲座目标

“发现组成JDBC™ 4.0 API技术规范的组件”

JDBC API 已经完成——为什么还要一个Java 规范需求(JSR)?

议程

JDBC应用程序编程接口——十二个月来的回顾

开发的简易性

SQL:2003桥接到XML世界

连接增强

连接和语句池增强

大对象增强

其他特性

议程

JDBC应用程序编程接口——十二个月来的回顾

开发的简易性

SQL:2003桥接到XML世界

连接增强

连接和语句池增强

大对象增强

其他特性

JDBC API——回顾

已完成的JSR

- JDBC RowSet实现
  - JSR-114
  - 2004年4月7日最后版本
  - 2004年5月12日维护回顾存档
    - JDBC RowSet实现1.01
      - 实现修改
      - 规范和文件修改
- JDBC API连接设备配置(CDC)/基础简表
  - JSR-169
  - 2004年4月8日最后版本

JDBC API——回顾

……实现的可用性……

- Java Web服务包1.4
  - Co-Bundle包含教程、例子和最后的规范
- Sun Java Studio Creator 1.0应用开发工具
  - 正式名为Project Rave
  - http://wwws.sun.com/software/products/jscreator/
- Java 2标准版1.5.0 (Tiger)

JDBC API——回顾

为什么要新的JDBC 4.0 API规范?

- Java平台中的核心元素
  开发简易性的目标
- 有机会更新API到最新的SQL技术规范
  - SQL:2003
- 解决JDBC 3.0 API规范的突出问题
- 扩展规范以迎合企业需求

JDBC API——回顾

定位JDBC 4.0 API的听众……

- 什么是定位听众?
  - 开发的简易性是针对合作开发者
- 什么是企业特征?
  - 企业的体系结构设计师
  - 技术师

议程

JDBC应用程序编程接口——十二个月来的回顾

开发的简易性

SQL:2003桥接到XML世界

连接增强

连接和语句池增强

大对象增强

其他特性

JDBC 4.0 API——开发的简易性

最初的JDBC 4.0 API提交给Java标准制定组织(JCP™)服务

"JDBC 4.0 API规范寻求通过提供开发的简易性的重要特征和在使用工具和API水平上的改善来促进Java应用程序进入SQL数据存储。

通过使用为JSR-176 Java 2标准版1.5软件而计划的新Java语言特征，比如定义在JSR-175中的注解和定义在JSR-014中的泛型，以及提供一套JDBC工具类，SQL机智的开发者就能够在从JDBC API中获利的同时更容易地进入SQL数据源。"

JDBC 4.0 API开发的简易性

注意！

下面的内容只是提议，会改变！！

JDBC 4.0 API开发的简易性

定位开发的简易性领域

- JDBC驱动程序管理
- 连接管理
- 查询/更新机制
- 与JDBC RowSet实现(JSR-114)紧密联系

JDBC 4.0 API开发的简易性

JDBC 驱动程序管理

- 建立一个ConnectionFactory的概念
  - 负责产生一个到数据源的连接
- JDBC驱动程序配置
  - Java平台已经开始认识到JDBC驱动程序
    - 对JDBC驱动程序的生命周期管理负责
    - 自动管理类装载
  - 为JDBC API URL和DataSource提供可选择的信赖

JDBC 4.0 API开发的简易性——注解

- JSR-175——由J2SE™ 1.5.0平台提供
- 观察：经常需要样板代码——总是一样的，但输入它很糟糕：

```
Context ctx = new InitialContext();
Object objref =
        ctx.lookup("java:comp/env/ejb/SimpleFoo");
FooHome home =
    (FooHome)PortableRemoteObject.narrow(objref,
       FooHome.class);
Foo myFoo = home.create();
```

- 现在你能只注解，然后让工具来工作：

```
private @Javax.ejb.create Foo myFoo;
```

JDBC 4.0 API开发的简易性——泛型

- JSR-014——由J2SE 1.5.0平台提供
- 一些类能在许多类型上工作
  - **HashMap**是一个一般目的的哈希表
  - 能从任何东西映射到任何东西
  - 但所有的东西都以“对象”来对待
- 程序员都知道明确的类型
  - “这个HashMap输入字符串，却返回Mammals”
- 泛型让我们表达那个知识
  - HashMap<String,Mammal>
  - 这让编译器检查类型
  - 避免对明确类型转换的需求

JDBC 4.0 API开发的简易性——泛型（II）

- 定义一个`DataSet`接口,它提供了一个简单的`RowSet`泛型视图

```
interface DataSet<T>
```

- 一个`DataSet`作为一个集合被定义
  - 通过泛型化的DataSet进行集合样式迭代

JDBC 4.0 API开发的简易性

JDBC注解

- 建立一套JDBC注解以提高开发者使用JDBC API的经验
  - 建立一个连接
    `@Database`
  - 查询处理
    `@Query`
  - 更新/修改
    `@Update`
  - 元数据
    `@MetaData`
- 更易于结果处理的用户定义类绑定

JDBC 4.0 API开发的简易性

开发的简易性——事件序列（I）

- 定义一个可保存结果数据的用户类

```
class MammalsResults {
    public String name
    public String address;
}
```

- 使用JDBC @Query注解定义数据库连接性

```
@Database(type="MYSQL", host="jmb.sfbay"
  port="123", user="scott",  password="tiger");
```

JDBC 4.0 API开发的简易性

开发的简易性——事件序列（II）

- 在一个接口中使用@Query 和 @Update注解定义查询和更新
- 例子：

```
interface ZebadeeQueries {
   @Query(SQL="SELECT *.* FROM MAMMALS",
    writeable=true, disconnected=true);
   MammalsResults getAllMammals();
   @Query(SQL="SELECT *.* FROM MAMMALS WHERE “+
        “WEIGHT > {weight}")
   MammalsResults getBigMammals(int weight);
   @Update(SQL="UPDATE WOMBATS SET WEIGHT = 5” +
    “WHERE WEIGHT > {weight}")
   int shrinkBigMammals(int weight);
}
```

JDBC 4.0 API开发的简易性

开发的简易性——事件序列（III）

- 从Factory中获取一个处理程序实例

```
MyQueries mq = QueryFactory.create
(MyQueries.class);
```

  - QueryFactory机制提供：
    - MyQueries接口的实现
    - 定义了@Query和@Update注解的进程
- 使用处理程序进入数据库
  - 注意：DataSet提供了一个JDBC RowSet的泛型化视图

```
DataSet<MammalsResults> rows = mq.getAllNames();
```

JDBC 4.0 API开发的简易性

开发的简易性——事件序列（IV）

- 作为用户类实例处理数据

```
for (MammalResults elem: rows) {
    System.out.println(elem.name);
  System.out.println(elem.address);
}
```

- 现在插入一个新行.....

```
MammmalResults newRow = new MammalResults();
newRow.name = “Homer Simpson”;
newRow.address = “Springfield”;
rows.insert(row);
```

JDBC 4.0 API开发的简易性——总结

回顾事件序列

1. 定义使用JDBC注解的数据库连接性
2. 在接口定义使用JDBC注解的查询/更新
3. 定义可保存数据的用户类
4. 从工厂中获取处理程序实例
5. 使用处理程序进入数据库
6. 作为用户类的实例处理数据

JDBC 4.0 API开发的简易性——总结

- @Query和@Update注解是完全可配置的
  - Writable, isolation, mapping (1:1或1:n，n>1)
  - 分离的或连接的

议程

JDBC应用程序编程接口——十二个月来的回顾

开发的简易性

SQL:2003桥接到XML世界

连接增强

连接和语句池增强

大对象增强

其他特性

SQL 2003

- 增加
  - XML数据类型
  - SQL/XML语言扩展

SQL 2003——XML 数据类型

- 定义在SQL标准中的X010特性就是增加了一个“XML类型”
- JDBC API已经扩展到允许使用XML结果列和XML参数绑定

SQL 2003——XML数据类型

- **String sqlStr= new String** (“select XMLColumn from T1 where ...”);

```
Statement stat=con.createStatement();
ResultSet rs=stat.executeQuery(sqlStr);
while(rs.next())
    {
    XMLType xmlC= (XMLType) rs.getObject(1);
    org.w3c.dom.Document doc=xmlC.getDOM();
    ...
    }
```

SQL 2003——SQL/XML

- XML数据类型
- 映射规则(概要和实例)
- 扩展到select语句语法来详细说明怎样从一个(表格式的)选择结果集构造XML
  - xmlelement
  - xmlattributes
  - xmlconcat
  - xmlforest
  - xmlagg

SQL 2003——SQL/XML

CustId	Name	Address
1	Woodworks	Baltimore
2	Software Solutions	Boston
3	Food Supplies	New York
4	Hardware Store	Washington
5	Books Inc.	New Orleans
6	Photo Shop	Dallas
7	Software Solutions	Sant Jose
8	Computer Supplies	Chicago
9	Eastern Publishers	New York
10	Car Supplies	Los Angeles
11	Health Insurances	Seattle
12	Bank Lucerne	San Francisco
13	Marlington Hotels	New York
14	US Bakeries	Columbia
15	Entertainment Inc	Orlando

```
<Customers>
  <row>
    <CustId>1</CustId>
    <Name>Woodworks</Name>
    <Address>Baltimore</Address>
  </row>
  <row>
    <CustId>2</CustId>
    <Name>Software
Solutions</Name>
    <Address>Boston</Address>
  </row>
 <row>
    <CustId>3</CustId>
    <Name>Food Supplies</Name>
    <Address>New York</Address>
  </row>
  ...
</Customers>
```

SQL 2003——SQL/XML

```
select
  c.CustId,
  xmlelement(name customer,
    xmlelement(name name,c.Name),
    xmlelement(name address,c.Address)) as
   CustInfo
from Customers c
```

CustId	CustInfo
1	<customer> <name>Woodworks</name> <address>Baltimore</address> </customer>
2	<customer> <name>Software Solutions</name> <address>Boston</address> </customer>
…	…

议程

JDBC应用程序编程接口——十二个月来的回顾

开发的简易性

SQL:2003桥接到XML世界

连接增强

连接和语句池增强

大对象增强

其他特性

连接增强

- 使用连接池，难以跟踪记录下哪些应用程序正在使用数据库资源
  - 增加一个方法以在连接之后设置/获取客户信息/名称
- 你怎样才知道一个连接仍然“可以”使用?
  - Connection.isClosed()只告诉我们连接对象是否已关闭
  - 增加新方法，如Connection.isValid()可以决定连接是否可用

议程

JDBC应用程序编程接口——十二个月来的回顾

开发的简易性

SQL:2003桥接到XML世界

连接增强

连接和语句池增强

大对象增强

其他特性

连接池增强

- 连接划分
  - 使用名称标识在一个池中的连接以使它们在商业应用中更加灵活
    - 例如：一个池可用以下名称“划分”：“OrderEntry”、”Inventory“和”CustomerInformation” 。一个应用程序可以递交一个“OrderEntry”连接

议程

JDBC应用程序编程接口——十二个月来的回顾

开发的简易性

SQL:2003桥接到XML世界

连接增强

连接和语句池增强

大对象增强

其他特性

大对象增强

- 关注给程序员的使用简易性和灵活性
  - 创建一个大对象的能力
  - 关闭一个大对象的能力

议程

JDBC应用程序编程接口——十二个月来的回顾

开发的简易性

SQL:2003桥接到XML世界

连接增强

连接和语句池增强

大对象增强

其他特性

更多信息

- 列表
  - DataDirect升级
  - 用于Java的XQuery API讨论会
  - 兴趣小组讨论会

讲座总结

- JDBC RowSet实现
- JDBC 4.0应用编程接口(API)
  - 开发的简易性
  - 连接和池的管理
  - 与XML和SQL:2003的关系
  - Blob管理进展

行动起来——参与进来!

- 网络在线
  - Java.sun.com JDBC社区
  - DataDirect开发者论坛
- 兴趣小组讨论会
  - 星期三
- JavaOne℠大会
- JDBC技术主页
  - http://java.sun.com

问与答

JavaOne℠

Sun's 2004 Worldwide Java Developer Conference℠

JDBC™ 4.0 API—新一代应用编程接口

JSR-221

java.sun.com/javaone/sf

Jonathan Bruce
JDBC技术规范的领导者和设计者，Sun
John Goodson
产品运营副总裁，Data Direct 技术。