[Hibernate 入门教程(纯Eclipse版)](http://blog.csdn.net/cpp_lzth/article/details/5418995)

2010-03-26 11:45 8281人阅读 [评论](http://blog.csdn.net/cpp_lzth/article/details/5418995#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/cpp_lzth/article/details/5418995#report)

[hibernate](http://www.csdn.net/tag/hibernate)[eclipse](http://www.csdn.net/tag/eclipse)[date](http://www.csdn.net/tag/date)[数据库驱动程序](http://www.csdn.net/tag/%e6%95%b0%e6%8d%ae%e5%ba%93%e9%a9%b1%e5%8a%a8%e7%a8%8b%e5%ba%8f)[class](http://www.csdn.net/tag/class)[string](http://www.csdn.net/tag/string)

**[java]** [view plaincopy](http://blog.csdn.net/cpp_lzth/article/details/5418995)

1. Hibernate是一个不依赖其他平台的轻量级的中间件，现在到处充斥着通过各种工具使用Hibernate进行开发的教程，如 MyEclipse, Ant, Maven等等，但是去掉这些工具，事实上，Hibernate仍然可以使用的。下面的讲解就是使用这种方法，让大家认识Hibernate。需要的软件有：Eclipse + MySQL + Hibernate。
3. 1. 准备jar包
4. 首先作为准备，我们去Hibernate官方网站下载Hibernate jar包。本教程用的是hibernate-distribution-3.3.2.GA-dist， 解压，我们使用的主要有hibernate3.jar 和lib/required里的包，其他的包在需要的时候再进行导入即可。需要特别注意的是，我在这些包里没有找到slf4j-nop-1.5.2.jar这个包，需要自己去下载，不然在使用本教程运行时会抛出找不到一些类的异常。下载的地址可以在baidu或者google里搜索。另外需要数据库驱动程序, 本教程使用的是MySQL数据库,使用的jar包为mysql-connector-java-5.1.0-bin.jar,你可根据自己的mysql版本等信息选择合适的jar包.
6. 2. 建立工程
7. 为了方便，我们使用Eclipse作为开发平台，注意，这里没有使用其他的插件，从这个意义上来说，还是比较纯粹的，呵呵。本贴原创,转载请注明来自historycreator.com

10. 2.0 在MySQL中建立数据库,名为event.
11. 建一表,名为events,包含字段有EVENT\_ID,title,EVENT\_DATE,类型分别是整型自动增长主键,varchar,timestamp.
13. 2.1 打开eclipse，建立一个Java Project。导入相关类,包括hibernate3.jar和/lib/required里的所有jar包,加上slf4j-nop-1.5.2.jar和mysql-connector-java-5.1.0-bin.jar.
15. 2.2 新建一个实体类Event
16. **package** com.historycreator.hibernate;
18. **import** java.util.Date;
20. **public** **class** Event {
21. **private** Long id;
23. **private** String title;
24. **private** Date date;
26. **public** Event() {}
28. **public** Long getId() {
29. **return** id;
30. }
32. **private** **void** setId(Long id) {
33. **this**.id = id;
34. }
36. **public** Date getDate() {
37. **return** date;
38. }
40. **public** **void** setDate(Date date) {
41. **this**.date = date;
42. }
44. **public** String getTitle() {
45. **return** title;
46. }
48. **public** **void** setTitle(String title) {
49. **this**.title = title;
50. }
51. }
53. 2.3 在com.historycreator.hibernate下建立配置文件Event.hbm.xml,内容如下
54. <?xml version="1.0" encoding="UTF-8"?>
55. <!DOCTYPE hibernate-mapping PUBLIC
56. "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
57. "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
59. <hibernate-mapping **package**="com.historycreator.hibernate">
61. <**class** name="Event" table="EVENTS">
62. <id name="id" column="EVENT\_ID">
63. <generator **class**="native" />
64. </id>
65. <property name="date" type="timestamp" column="EVENT\_DATE" />
66. <property name="title" />
67. </**class**>
69. </hibernate-mapping>
71. 2.4 在src文件夹,也就是在com同级目录下,建立配置文件hibernate.cfg.xml,内容如下:
72. <?xml version="1.0" encoding="UTF-8"?>
73. <!DOCTYPE hibernate-configuration PUBLIC
74. "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
75. "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
77. <hibernate-configuration>
79. <session-factory>
81. <!-- Database connection settings -->
82. <property name="connection.driver\_class">org.gjt.mm.mysql.Driver</property>
83. <property name="connection.url">jdbc:mysql://localhost/event?useUnicode=true&characterEncoding=gbk</property>
84. <property name="connection.username">root</property>
85. <property name="connection.password">test</property>
87. <!-- JDBC connection pool (use the built-in) -->
88. <property name="connection.pool\_size">1</property>
90. <!-- SQL dialect -->
91. <property name="dialect">org.hibernate.dialect.MySQL5Dialect</property>
93. <!-- Enable Hibernate's automatic session context management -->
94. <property name="current\_session\_context\_class">thread</property>
96. <!-- Disable the second-level cache  -->
97. <property name="cache.provider\_class">org.hibernate.cache.NoCacheProvider</property>
99. <!-- Echo all executed SQL to stdout -->
100. <property name="show\_sql">**true**</property>
102. <!-- Drop and re-create the database schema on startup -->
103. <property name="hbm2ddl.auto">update</property>
105. <mapping resource="com/historycreator/hibernate/Event.hbm.xml"/>
107. </session-factory>
109. </hibernate-configuration>
111. 2.5 建工厂类HibernateUtil
112. **package** com.historycreator.hibernate;
114. **import** org.hibernate.SessionFactory;
115. **import** org.hibernate.cfg.Configuration;
117. **public** **class** HibernateUtil {
119. **private** **static** **final** SessionFactory sessionFactory = buildSessionFactory();
121. **private** **static** SessionFactory buildSessionFactory() {
122. **try** {
123. // Create the SessionFactory from hibernate.cfg.xml
124. **return** **new** Configuration().configure().buildSessionFactory();
125. }
126. **catch** (Throwable ex) {
127. // Make sure you log the exception, as it might be swallowed
128. System.err.println("Initial SessionFactory creation failed." + ex);
129. **throw** **new** ExceptionInInitializerError(ex);
130. }
131. }
133. **public** **static** SessionFactory getSessionFactory() {
134. **return** sessionFactory;
135. }
137. }
139. 2.6  建类EventManager
140. **package** com.historycreator.hibernate;
142. **import** java.util.Date;
144. **import** org.hibernate.classic.Session;
146. **public** **class** EventManager {
148. **public** **static** **void** main(String[] args) {
149. EventManager mgr = **new** EventManager();
151. mgr.createAndStoreEvent("My Event", **new** Date());
153. HibernateUtil.getSessionFactory().close();
154. }
156. **private** **void** createAndStoreEvent(String title, Date theDate) {
157. Session session = HibernateUtil.getSessionFactory().getCurrentSession();
158. session.beginTransaction();
160. Event theEvent = **new** Event();
161. theEvent.setTitle(title);
162. theEvent.setDate(theDate);
163. session.save(theEvent);
165. session.getTransaction().commit();
167. }
169. }
171. 运行即可.效果就是往数据库中插入了一条记录.