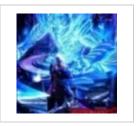
nirvana1988

永久域名 http://nirvana1988.iteye.com

◀ 精确获取JS变量类型



nirvana1988

浏览: 2810 次

性别: 💣

来自: 郑州







搜索本博客

最近访客 客

>>更多访



绿窗明月





xiaokang1582 hanfeng450

博客分类

- 全部博客 (10)
- 框架整合 (2)
- web前端 (5)
- <u>i2se基础 (2)</u>
- JVM (1)

我的留言簿 >>更多留言

其他分类

- 我的收藏 (19)
- 我的代码 (0)
- 我的书籍 (1)
- 我的论坛主题帖 (0)
- 我的所有论坛帖 (35)
- <u>我的精华良好帖</u> (0)

2011-03-14

mybatis3.0 多对多映射CRUD操作实例

博客分类: 框架整合

iBATIS XML junit Apache

mybatis3.0添加了association和collection标签专门用于对多个相关实体类数据进行级联查询,但仍不支持多个相 关实体类数据的级联保存和级联删除操作。因此在进行实体类多对多映射表设计时,需要专门建立一个关联对象类对 相关实体类的关联关系进行描述。下文将以"User"和"Group"两个实体类之间的多对多关联映射为例进行CRUD操

建立user表,对应实体类"User",建表语句如下:

<!--StartFragment-->

Sq1代码 23

```
1.
     CREATE TABLE `user` (
2.
       `id` int(11) NOT NULL auto_increment,
3.
       `name` varchar(40) collate utf8_unicode_ci default NULL,
       `password` varchar(20) collate utf8_unicode_ci default NULL,
4.
5.
       `createtime` timestamp NULL default CURRENT_TIMESTAMP,
       PRIMARY KEY ('id')
6.
7.
   )
```

建立group_info表,对应实体类"Group",建表语句如下:

23 Sql代码

```
CREATE TABLE `group_info` (
      `id` int(11) NOT NULL auto_increment,
      `name` varchar(40) collate utf8_unicode_ci default NULL,
3.
       `createdate` timestamp NULL default CURRENT_TIMESTAMP,
4.
       `state` int(1) default '0' COMMENT '0:可见;1:不可见',
6.
       PRIMARY KEY ('id')
7.
   )
```

建立user_group表,对应实体类"UserGroupLink"(该类为User和Group两个实体类之间的关系描述),建表语句如 下:

23 **Sql**代码

```
CREATE TABLE `user_group` (
1.
      `user_id` int(11) default NULL,
2.
       `group_id` int(11) default NULL,
3.
4.
       `createdate` timestamp NULL default CURRENT_TIMESTAMP,
5.
       KEY `FK_user_group_user_id` (`user_id`),
       KEY `FK user group group id` (`group id`),
7.
       CONSTRAINT `FK_user_group_id` FOREIGN KEY (`group_id`) REFERENCES `group_info` (`id`),
       CONSTRAINT `FK_user_group_user_id` FOREIGN KEY (`user_id`) REFERENCES `user` (`id`)
8.
9.
```

建立实体类User, 代码如下:

最近加入群组

■ Vim编辑器

- Jquery
- 每日一题JAVA
- 济南JavaEyer群
- 设计模式

存档

- **2011-08** (1)
- **2011-07** (1)
- **2011-06** (3)
- 更多存档...

评论排行榜

- spring与mvbatis三种整合方法
- AJAX传值中文乱码解决方法
- mybatis3.0 多对多映射CRUD操 作实例





```
Java代码 🛣
```

```
package com.xxt.ibatis.dbcp.domain;
 2.
 3.
      import java.util.Date;
 4.
      import java.util.List;
 5.
 6.
 7.
       * @describe: User实体类
 8.
       * @author:
                    Nirvana
 9.
       * @version: V1.0 2011-3-11下午06:06:49 create
       */
10.
11.
12.
      public class User {
13.
               private long id;
14.
15.
16.
               private String name;
17.
18.
               private String password;
19.
20.
               private Date createTime;
21.
               private List<Group> group;
22.
23.
24.
               public Date getCreateTime() {
25.
                       return createTime;
26.
27.
28.
               public void setCreateTime(Date createTime) {
29.
                       this.createTime = createTime;
30.
               }
31.
32.
               public long getId() {
33.
                       return id;
34.
               }
35.
               public void setId(long id) {
36.
37.
                       this.id = id;
38.
               }
39.
40.
               public String getName() {
41.
                       return name;
42.
               }
43.
44.
               public void setName(String name) {
                       this.name = name;
45.
46.
               }
47.
48.
               public String getPassword() {
49.
                       return password;
50.
               }
51.
52.
               public void setPassword(String password) {
53.
                       this.password = password;
54.
               }
55.
               public List<Group> getGroup() {
56.
57.
                       return group;
58.
               }
59.
60.
               public void setGroup(List<Group> group) {
                       this.group = group;
61.
62.
               }
63.
```

建立实体类Group,代码如下:

Java代码 😭

```
1.
      package com.xxt.ibatis.dbcp.domain;
 2.
      import java.util.Date;
 3.
      import java.util.List;
 4.
 5.
 6.
 7.
      * @describe: Group实体类
 8.
       * @author: Nirvana
 9.
       * @version: V1.0 2011-3-7下午08:10:29 create
10.
11.
      public class Group {
12.
13.
              private long id;
14.
15.
              private String name; // 组名
16.
              private Date createTime;
17.
18.
19.
              private int state; // 0可见状态 1不可见状态
20.
21.
              private List<User> user;
22.
23.
              public Date getCreateTime() {
                       return createTime;
24.
25.
              }
26.
27.
              public void setCreateTime(Date createTime) {
                       this.createTime = createTime;
28.
29.
30.
              public long getId() {
31.
32.
                       return id;
33.
34.
              public void setId(long id) {
35.
                      this.id = id;
36.
37.
38.
39.
              public String getName() {
40.
                       return name;
41.
42.
43.
              public void setName(String name) {
44.
                       this.name = name;
45.
46.
47.
              public int getState() {
48.
                       return state;
49.
50.
51.
              public void setState(int state) {
52.
                      this.state = state;
53.
54.
55.
              public List<User> getUser() {
56.
                      return user;
57.
              }
58.
59.
              public void setUser(List<User> user) {
```

```
60. this.user = user;
61. }
62. 63. }
```

建立实体类UserGroupLink,用于描述User和Group之间的对应关系,代码如下:

Java代码 😭

```
1.
      package com.xxt.ibatis.dbcp.domain;
 2.
 3.
      import java.util.Date;
 4.
 5.
       * @describe: 描述User和Group之间的映射关系
 6.
       * @author: Nirvana
       * @version: V1.0 2011-3-11下午02:57:52 create
 8.
 9.
10.
      public class UserGroupLink {
11.
12.
               private User user;
13.
14.
               private Group group;
15.
16.
               private Date createTime;
17.
18.
               public Date getCreateTime() {
19.
                       return createTime;
20.
               }
21.
22.
               public void setCreateTime(Date createTime) {
23.
                       this.createTime = createTime;
24.
               }
25.
26.
               public Group getGroup() {
27.
                       return group;
               }
28.
29.
30.
               public void setGroup(Group group) {
31.
                       this.group = group;
32.
               }
33.
34.
               public User getUser() {
35.
                       return user;
36.
               }
37.
38.
               public void setUser(User user) {
39.
                       this.user = user;
40.
               }
41.
```

建立user实体类的映射文件user.map.xml,代码如下:

Xm1代码 **∑**

```
<?xml version="1.0" encoding="UTF-8" ?>
 1.
      <!DOCTYPE mapper
 2.
          PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
 3.
 4.
          "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
 5.
      <mapper namespace="com.xxt.ibatis.dbcp.domain.User">
               <resultMap type="User" id="userMap">
 6.
                       <id property="id" column="id" />
 7.
 8.
                       <result property="name" column="name" />
 9.
                       <result property="password" column="password" />
10.
                       <result property="createTime" column="createtime" />
11.
               </resultMap>
```

```
12.
13.
               <resultMap type="User" id="userGroupMap" extends="userMap">
                       <collection property="groups" ofType="Group">
14
15.
                                <id property="id" column="goupId" />
                                <result property="name" column="groupName" />
16.
17.
                                <result property="state" column="state" />
                                <result property="createTime" column="groupCreateTime" />
18.
19.
                       </collection>
20.
               </resultMap>
21.
22.
               <!-- 根据user表中的id查询用户信息 -->
23.
24.
               <select id="selectUser" parameterType="long" resultMap="userMap">
                       select * from user where id = #{id}
25.
26.
               </select>
27.
               <!-- 根据user表中的id查询用户和组信息 -->
28.
               <select id="selectUserGroup" parameterType="long"</pre>
29.
30.
                       resultMap="userGroupMap">
31.
                       select u.id,u.name,u.password,u.createtime, gi.id as
32.
                       goupId,gi.name as groupName, gi.createdate as groupCreateTime,
33.
                       gi.state from user u left join user_group ug on u.id=ug.user_id
34.
                       left join group_info gi on ug.group_id=gi.id where u.id = #{id}
35.
               </select>
36.
               <!-- 插入用户信息 -->
37.
38.
               <insert id="saveUser" parameterType="User" keyProperty="id"</pre>
                       useGeneratedKeys="true">
39.
40.
                       insert into user(name,password) values(#{name},#{password})
41.
              </insert>
42.
               <!-- 保存用户和组之间的关系信息 -->
43.
44.
               <insert id="saveRelativity" parameterType="UserGroupLink">
45.
                       insert into user group(user id,group id)
46.
                       values(#{user.id},#{group.id})
47.
               </insert>
48.
49.
               <select id="selectAllUser" resultMap="userMap">
50.
                       select * from user
51.
               </select>
52.
53.
      </mapper>
```

建立group实体类的映射文件group.map.xml,代码如下:

Xml代码 ☆

```
<?xml version="1.0" encoding="UTF-8"?>
 1.
 2.
      <!DOCTYPE mapper
 3.
          PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
 4.
          "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
 5.
      <mapper namespace="com.xxt.ibatis.dbcp.domain.Group">
 6.
               <resultMap type="Group" id="groupMap">
                       <id property="id" column="id" />
 8.
                       <result property="name" column="name" />
 9.
                       <result property="createTime" column="createdate" />
10.
               </resultMap>
11.
12.
               <resultMap type="Group" id="groupUserMap" extends="groupMap">
13.
                       <collection property="users" ofType="User">
14.
                                <id property="id" column="userId" />
15.
                                <result property="name" column="userName" />
16.
                                <result property="password" column="password" />
```

```
<result property="createTime" column="userCreateTime" />
18.
                       </collection>
19
               </resultMap>
20.
21.
               <!-- 根据Group表中的id或name查询组信息和组内用户信息 -->
22.
               <select id="selectGroupUser" parameterType="Group"</pre>
23.
24.
                       resultMap="groupUserMap">
25.
                       select u.id as userId, u.name as userName,
26.
                       u.password,u.createtime as userCreateTime,
27.
                       gi.id,gi.name,gi.createdate,gi.state from group_info gi left
28.
                       join user_group ug on gi.id=ug.group_id left join user u on
29.
                       uug.user_id=u.id
                       <where>
30.
                               <!--当id为初始值0,不再使用id作为查询条件 -->
31.
32.
                               <if test="id != 0 ">gi.id=#{id}</if>
                               <!-- 当name为空或为空串时,不再使用name作为查询条件 -->
33.
34.
                               <if test="name != null and name != ''">
35.
                                       or gi.name = #{name}
                               </if>
36.
37.
                       </where>
38.
               </select>
39.
               <!-- 根据id查询group组信息 -->
40.
41.
               <select id="selectGroup" parameterType="Date"</pre>
42.
                       resultMap="groupMap">
43.
                       select * from group_info where id=#{group_id}
44
              </select>
45.
46.
               <!--根据name查询group组信息 -->
               <select id="getGroupByName" parameterType="String"</pre>
47.
48.
                       resultMap="groupMap">
49.
                       select * from group_info where name=#{name}
50.
              </select>
51.
52.
              <!-- 插入组信息 -->
              <insert id="saveGroup" parameterType="Group" keyProperty="id"</pre>
53.
54
                       useGeneratedKeys="true">
55.
                       insert into group_info(name) values(#{name})
56.
              </insert>
57.
               <!-- 删除组与组内成员之间的对应关系 -->
58.
59.
              <delete id="deleteGroupUser" parameterType="UserGroupLink">
60.
                       delete from user_group
                       <where>
61.
62.
                               <if test="user.id != 0">user_id = #{user.id}</if>
                               <if test="group.id != 0">and group id = #{group.id}</if>
63.
64.
                       </where>
               </delete>
65.
66.
               <!--根据组id或者组name删除组信息 -->
               <delete id="deleteGroup" parameterType="Group">
68.
69.
                       delete from group_info
70.
71.
                               <if test="id!=0">id=#{id}</if>
                               <if test="name!=null || name!=''">and name=#{name}</if>
72
73.
                       </where>
74.
               </delete>
75.
               <!-- 更新根据组id或者组name更新组状态 -->
76.
77.
               <update id="updateGroupState" parameterType="Group">
78.
                       update group_info set state=#{state}
79.
                       <where>
                               <if test="id!=0">id=#{id}</if>
80.
```

```
81. <if test="name!=null || name!=''">and name=#{name}</if>
82. </where>
83. </update>
84. </mapper>
```

建立ibatis总配置文件salMapConfig.xml,代码如下:

```
Xml代码 ☆
```

```
<?xml version="1.0" encoding="UTF-8"?>
 1.
 2.
      <!DOCTYPE configuration
 3.
          PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
 4.
          "http://mybatis.org/dtd/mybatis-3-config.dtd">
      <configuration>
 5.
               resource="jdbc.properties" />
 6.
 7.
               <typeAliases>
 8.
 9.
                       <typeAlias type="com.xxt.ibatis.dbcp.domain.User" alias="User" />
                       <typeAlias type="com.xxt.ibatis.dbcp.domain.Group"
10.
11.
                                alias="Group" />
                       <typeAlias type="com.xxt.ibatis.dbcp.domain.UserGroupLink"
12.
                                alias="UserGroupLink" />
13.
14.
               </typeAliases>
15.
16.
               <environments default="development">
17.
                       <environment id="development">
                                <transactionManager type="JDBC" />
18.
19.
                                <dataSource type="UNPOOLED">
20.
                                        roperty name="driver" value="${driver}" />
21.
                                        cproperty name="url" value="${url}" />
                                        cproperty name="username" value="${username}" />
22.
                                        cproperty name="password" value="${password}" />
23.
24.
                                </dataSource>
25.
                       </environment>
               </environments>
26.
27.
28.
               <mappers>
29.
                       <mapper resource="com/xxt/ibatis/dbcp/domain/user.map.xml" />
30.
                       <mapper resource="com/xxt/ibatis/dbcp/domain/group.map.xml" />
31.
               </mappers>
      </configuration>
32.
```

测试用例, 代码如下:

Java代码 😭

```
package com.xxt.ibatis.dbcp.test;
 1.
 2.
 3.
      import java.io.Reader;
 4.
      import java.text.SimpleDateFormat;
      import java.util.List;
 5.
 6.
 7.
      import org.junit.AfterClass;
 8.
      import org.junit.BeforeClass;
9.
      import org.junit.Test;
10.
11.
      import org.apache.ibatis.io.Resources;
12.
      import org.apache.ibatis.session.SqlSession;
      import org.apache.ibatis.session.SqlSessionFactory;
13.
14.
      import org.apache.ibatis.session.SqlSessionFactoryBuilder;
15.
16.
      import com.xxt.ibatis.dbcp.domain.Group;
17.
      import com.xxt.ibatis.dbcp.domain.User;
18.
      import com.xxt.ibatis.dbcp.domain.UserGroupLink;
19.
```

```
* @describe: 测试用例
21.
                   Nirvana
22.
       * @author:
       * @version: V1.0 2011-3-4下午03:21:17 create
23.
24.
       */
25.
      public class IbatisUserTest {
26.
27.
          private final static String IBATIS_CONFIG_XML = "sqlMapConfig.xml";
28.
          private static SqlSession session;
29.
          private SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd k:mm:ss");
30.
31.
              @BeforeClass
              public static void setUpBeforeClass() throws Exception {
32.
33.
                       Reader reader = Resources.getResourceAsReader(IBATIS_CONFIG_XML); //读取ibatis
      配置文件
                       SqlSessionFactory sqlMapper = new SqlSessionFactoryBuilder().build(reader);
34.
35.
                        session = sqlMapper.openSession(true);
36.
                        session.commit(false); //将默认提交事务属性设置为否
              }
37.
38.
39.
              //保存用户信息
40.
              @Test
41.
              public void saveUserTest(){
42.
                      User user = new User();
43.
                      user.setName("张三");
44.
                      user.setPassword("123456");
45.
                      session.insert("com.xxt.ibatis.dbcp.domain.User.saveUser", user);
46.
47.
              }
48.
49.
              //获取用户信息
50.
              @Test
51.
              public void getUserTest(){
52.
                      User user = (User) session.selectOne("com.xxt.ibatis.dbcp.domain.User.selectUse
      r", 1L);
                      System.out.println("用户名: "+user.getName());
53.
                      System.out.println("用户密码: "+user.getPassword());
54.
55.
                      System.out.println("用户创建日期: "+sdf.format(user.getCreateTime()));
56.
              }
57.
              //获取用户和用户所在组信息
58.
59.
60.
              public void getUserAndGroupTest(){
                       User user = (User)session.selectOne("com.xxt.ibatis.dbcp.domain.User.selectU
61.
      serGroup", 4L);
62.
63.
                       System.out.println(user.getName() + "所属组信息:");
64.
                      for(Group group : user.getGroups() ){
                              System.out.println("
                                                      组名:"+group.getName()+", 组创建时间:"+sdf.forma
65.
      t(group.getCreateTime()));
66.
68.
69.
              }
70.
              //保存用户和用户所在组信息
71.
          //当用户所在组不存在时, 创建该组, 并生成映射关系
72.
73.
74.
              public void saveUserAndGroupTest(){
75.
              User user = new User();
                  user.setName("无常鬼");
76.
77.
                      user.setPassword("wuchang");
78.
                      session.insert("com.xxt.ibatis.dbcp.domain.User.saveUser", user);
79.
```

```
Group groupImpl = (Group)session.selectOne("com.xxt.ibatis.dbcp.domain.Group.get
      GroupByName","用户组4");//获取组实例
                      UserGroupLink ugl = new UserGroupLink();//声明User和Group实体间映射关系实例
 81.
 82.
 83.
                      //查询到的组实例为空时的逻辑处理
 84.
                      if(groupImpl == null){
                               Group group = new Group();
 85.
                               group.setName("用户组4");
 86.
                              session.insert("com.xxt.ibatis.dbcp.domain.Group.saveGroup", group);//
 87.
      持久化创建好的组实例
 88.
 89.
                              //设置映射关系实例相关的属性
 90.
                              ugl.setUser(user);
 91.
                              ugl.setGroup(group);
 92.
                  session.insert("com.xxt.ibatis.dbcp.domain.User.saveRelativity",ugl);//持久化映射关系
      实力
 93.
 94.
                      }else{
                          ugl.setGroup(groupImpl);
 95.
 96.
                              ugl.setUser(user);
97.
                              session.insert("com.xxt.ibatis.dbcp.domain.User.saveRelativity",ugl);
 98.
                      }
              }
99.
100.
101.
               //删除组信息的同时,取消组内所有的成员与该组的关联关系
102.
103
              @Test
104.
              public void deleteGroupTest(){
105.
                      Group group = new Group();
106.
                      //group.setId(1L); //以组id作为查询条件
107.
                      group.setName("用户组1"); //以组name作为查询条件
108.
                      Group groupImpl = (Group)session.selectOne("com.xxt.ibatis.dbcp.domain.Group.sel
      ectGroupUser",group);//获取组实例
109.
110.
                      //组实例存在时
111.
                  if(groupImpl != null){
112.
113.
                      List<User> users = groupImpl.getUsers();
114
                //查看用户组1中是否存在用户
115.
116.
                      if(users != null && users.size() > 0){
117.
                              //存在用户时, 先删除组与用户的对应关系
118.
119.
                              UserGroupLink ugl = new UserGroupLink();
120.
                              for(User user : users){
121.
                                 ugl.setUser(user);
122.
                                 ugl.setGroup(groupImpl);
123.
                                 session.delete("com.xxt.ibatis.dbcp.domain.Group.deleteGroupUser",ug
      1);
                              }
124.
125.
                      }
                      //删除组信息
126.
127.
                      session.delete("com.xxt.ibatis.dbcp.domain.Group.deleteGroup", groupImpl);
128.
129.
                  }else{
130.
                      throw new RuntimeException("查询的组不存在!!");
131.
132.
133.
134.
              }
135.
136.
              //更新组状态, 当组状态由可见状态变成不可见时, 取消该组下的用户与该组的映射关系
137.
138.
```

```
public void updateGroupStateTest(){
140.
                       Group group = new Group();
141.
                       group.setName("用户组2");
142.
                       Group groupImpl = (Group) session.selectOne("com.xxt.ibatis.dbcp.domain.Group.s
       electGroupUser",group);
143.
144.
                       if(groupImpl != null){
145.
                               int state = groupImpl.getState() == 1 ? 0 : 1;
146.
147.
                               //组状态由0变成1时,即由可见变为不可见
148.
149.
                               if(state == 1){
150.
                                        List<User> users = groupImpl.getUsers();
                  //查看用户组2中是否存在用户
151.
152.
                               if(users != null && users.size() > 0){
153.
                                        //存在用户时,删除组与用户的对应关系
154.
155.
                                        UserGroupLink ugl = new UserGroupLink();
156.
                                        for(User user : users){
157.
                                           ugl.setUser(user);
158.
                                           ugl.setGroup(groupImpl);
159.
                                           session.delete("com.xxt.ibatis.dbcp.domain.Group.deleteGroup
       User",ugl );
160.
                                        }
161.
                               }
162.
                               }
163.
164.
                               //更新组状态
165.
                               groupImpl.setState(state);
166.
                       }
167.
                       else{
168.
                               throw new RuntimeException("查询的组不存在!!");
169.
                       }
170.
171.
               }
172.
173.
               @AfterClass
174.
               public static void tearDownAfterClass() throws Exception {
175.
                 if(session != null){
176.
                        session.commit(); //提交事务
177.
                        session.close(); //关闭连接
178.
                 }
179.
               }
180.
```



19:52 <u>评论</u> / 浏览 (1 / 364) 分类:<u>企业架构</u> <u>相关推荐</u> <u>MORE</u>

```
评论

1 楼 <u>LanderRooter</u> 2011-08-22

楼主,您好,
public class UserGroupLink {
    private User user;
    private Group group;
```

```
private Date createTime;
}
  Java代码
         <resultMap type="User" id="userMap">
    1.
    2.
             <id property="id" column="id" />
   3.
             <result property="name" column="name" />
             <result property="password" column="password" />
    4.
    5.
             <result property="createTime" column="createtime" />
    6.
         </resultMap>
    7.
    8.
         <resultMap type="User" id="userGroupMap" extends="userMap">
   9.
             <collection property="groups" ofType="Group">
   10.
                 <id property="id" column="goupId" />
  11.
                 <result property="name" column="groupName" />
                 <result property="state" column="state" />
  12.
  13.
                 <result property="createTime" column="[b]groupCreateTime[/b]" />
             </collection>
  14.
  15.
         </resultMap>
```

文章中如上的映射,并没有将UserGroupLink的createDate映射到UserGroupLink,而是把groupCreateTime映射到Group的createTime属性

Java代码 😭

```
1. <select id="selectUserGroup" parameterType="long"
2.    resultMap="userGroupMap">
3.    select u.id,u.name,u.password,u.createtime, gi.id as
4.    goupId,gi.name as groupName, gi.createdate as [b]groupCreateTime[/b],
5.    gi.state from user u left join user_group ug on u.id=ug.user_id
6.    left join group_info gi on ug.group_id=gi.id where u.id = #{id}
7.    </select>
```

关联对象类UserGroupLink中的createTime属性该如何映射呢,在此例中并没有把UserGroupLink的createTime映射出来,如果此关联对象类还有boolean enable等属性,可不可以直接将结果映射成UserGroupLink呢?,如何可以的话,该如何映射呢,谢谢。小弟最近在项目中要Mybatis,也遇到类似的case,不是很清楚如何解决,谢谢。

发表评论



您还没有登录,请您登录后再发表评论

声明: ITeye文章版权属于作者,受法律保护。没有作者书面许可不得转载。若作者同意转载,必须以超链接形式标明文章原始出处和作者。 © 2003-2011 ITeye.com. All rights reserved. [京ICP证110151号]