



# Java EE框架 ---Hibernate

Java EE framework -环境搭建、Servlet、链接数据库、MVC

王磊

计算机工程学院

# CONTENTS



开发环境设置



导入jar包



配置文件



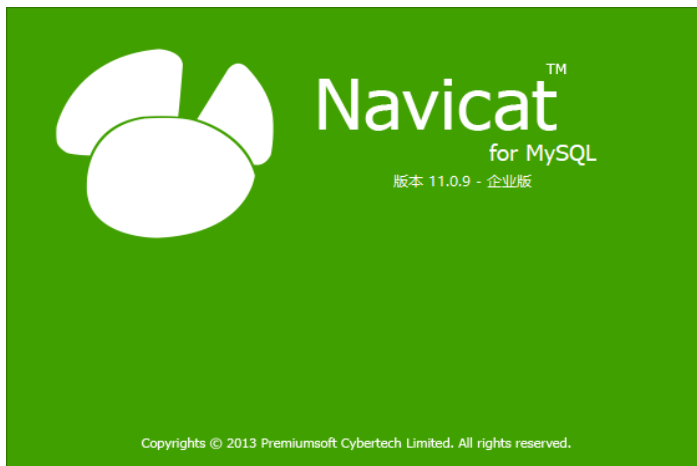
单元测试

# • 开发环境设置 •

- 1、操作系统：Windows7、Windows10
- 2、应用软件：Jdk1.8.0\_162 Eclipse Oxygen Release Milestone 5 (4.7.0 M5)、MySQL5.5、Navicat for MySQL11.0
- 3、服务器：Tomcat8.5.30

```
C:\Users\Administrator>java -version
java version "1.8.0_162"
Java(TM) SE Runtime Environment (build
Java HotSpot(TM) 64-Bit Server VM (bu
mode)
```

```
C:\Users\Administrator>
```



[Home](#) [Documentation](#) [Configuration](#) [Examples](#) [Wiki](#) [Mailing Lists](#)

Apache Tomcat/8.5.30



If you're seeing this, you've successfully installed



Recommended Reading:

[Security Considerations HOW-TO](#)

[Manager Application HOW-TO](#)

[Clustering/Session Replication HOW-TO](#)

Developer Quick Start

# 引入

模型不匹配(阻抗不匹配)

Java面向对象语言，对象模型，其主要概念有：继承、关联、多态等；数据库是关系模型，其主要概念有：表、主键、外键等。

解决办法

- 1、使用JDBC手工转换。
- 2、使用ORM(Object Relation Mapping对象关系映射)框架来解决，主流的ORM框架有Hibernate、Mybatis、OJB。

# • O/R Mapping •

ORM的全称是Object/Relation Mapping，即对象/关系映射。ORM也可理解是一种规范，具体的ORM框架可作为应用程序和数据库的桥梁。面向对象程序设计语言与关系数据库发展不同步时，需要一种中间解决方案，ORM框架就是这样的解决方案。

ORM并不是一种具体的产品，而是一类框架的总称，它概述了这类框架的基本特征：完成面向对象的程序设计语言到关系数据库的映射。基于ORM框架完成映射后，既可利用面向对象程序设计语言的简单易用性，又可利用关系数据库的技术优势。

## O/R Mapping的优点

- 1、提高生产效率
- 2、可维护性
- 3、更好性能
- 4、厂商独立性

# • Hibernate优势 •

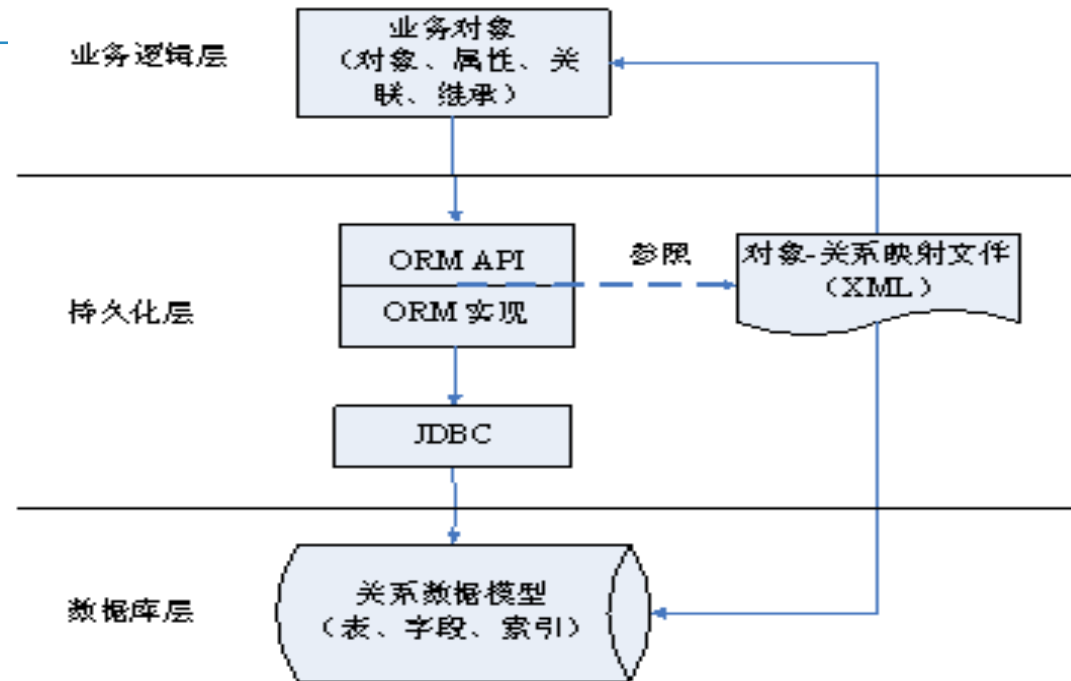
**Hibernate**是一个免费的开源**Java**包，是目前最流行的**ORM**框架，它是一个面向**Java**环境的对象/关系数据库映射工具。它使得程序与数据库的交互变得十分容易，更加符合面向对象的设计思想，像数据库中包含普通**Java**对象一样，而不必考虑如何把它们从数据库表中取出。使开发者可以专注于应用程序的对象和功能，而不必关心如何保存它们或查找这些对象。甚至在对**SQL**语句完全不了解的情况下，使用**hibernate**仍然可以开发出优秀的包含数据库访问的应用程序。

优势：

- 1、开源和免费的**License**，方便需要时研究源代码、改写源代码、进行功能定制。
- 2、轻量级封装，避免引入过多复杂的问题，调试容易，可减轻程序员负担。
- 3、具有可扩展性，**API**开放。功能不够用时，可自己编码进行扩展。
- 4、开发者活跃，产品有稳定的发展保障。**Hibernate**的工作方式  
灵巧的设计，出色的性能表现

## 持久化层含义

访问数据库代码(Dao)与业务逻辑(Service)混杂在一起带来了很多问题, 这样的程序设计严重限制了程序的可扩展性和适应性, 所以有必要要把涉及数据库操作的代码分离出来与业务逻辑分离。就形成了所谓“持久化层”的概念。持久化(Persistence), 即把数据(如内存中的对象)保存到可永久保存的存储设备中(如磁盘)。持久化的主要应用是将内存中的数据存储在关系型的数据库中, 当然也可以存储在磁盘文件中、XML数据文件中。



ORM 工具实现持久化示意图

# • 对象 - 关系数据库的匹配 •

## 对象-关系数据库匹配

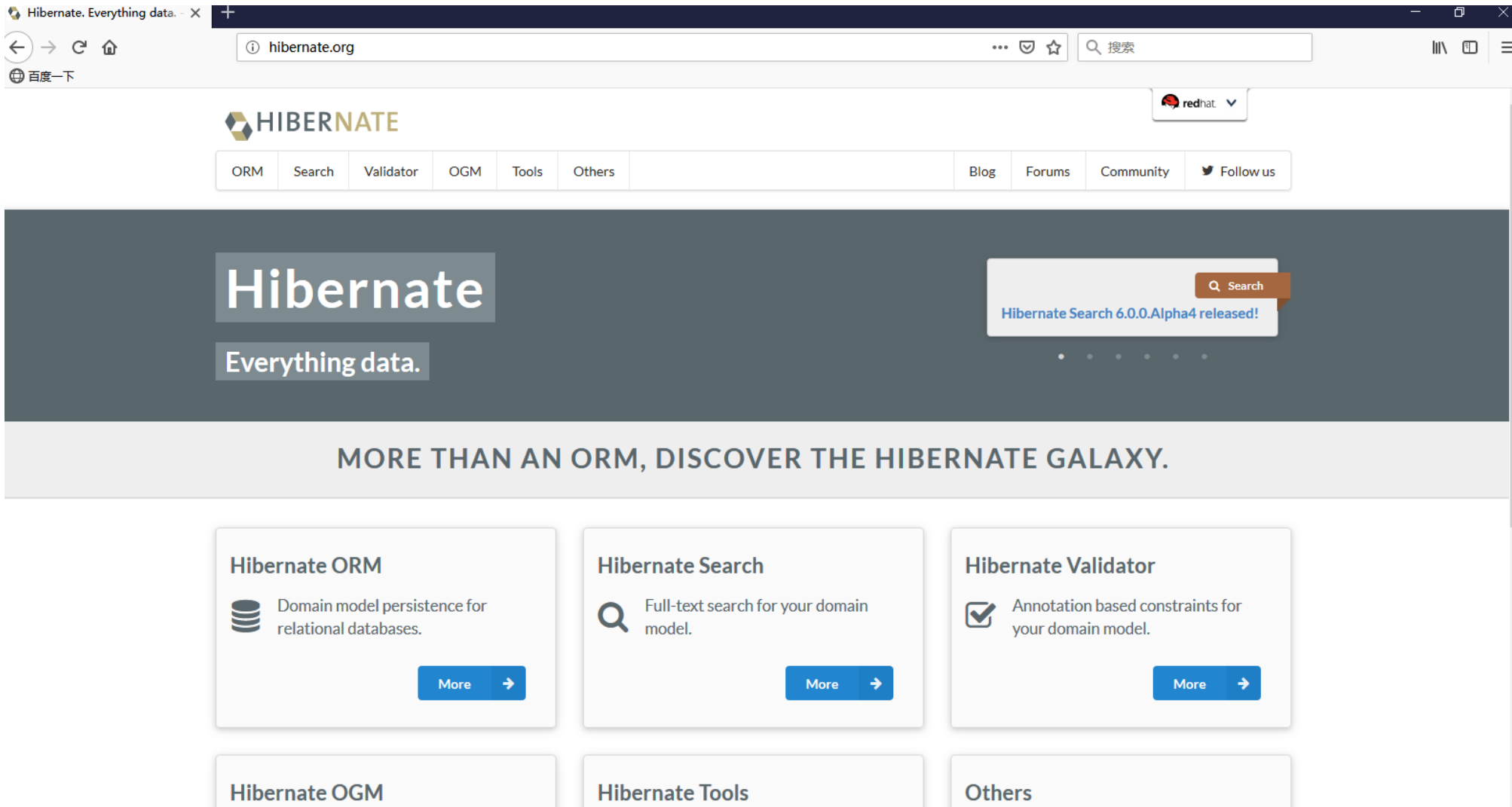
```
public class User {  
    private String name;  
    private String password;  
    private String address;  
    .....  
}  
  
create table t_user (  
    name varchar(255) not null ,  
    password varchar(255),  
    .....  
    primary key (name)  
)
```

对象	关系数据库
类	表
类的属性(基本类型)	表的列
一对多，多对一	外键
多对多	关联表
继承	单表继承、具体表继承、 类表继承



# • 下载Hibernate •

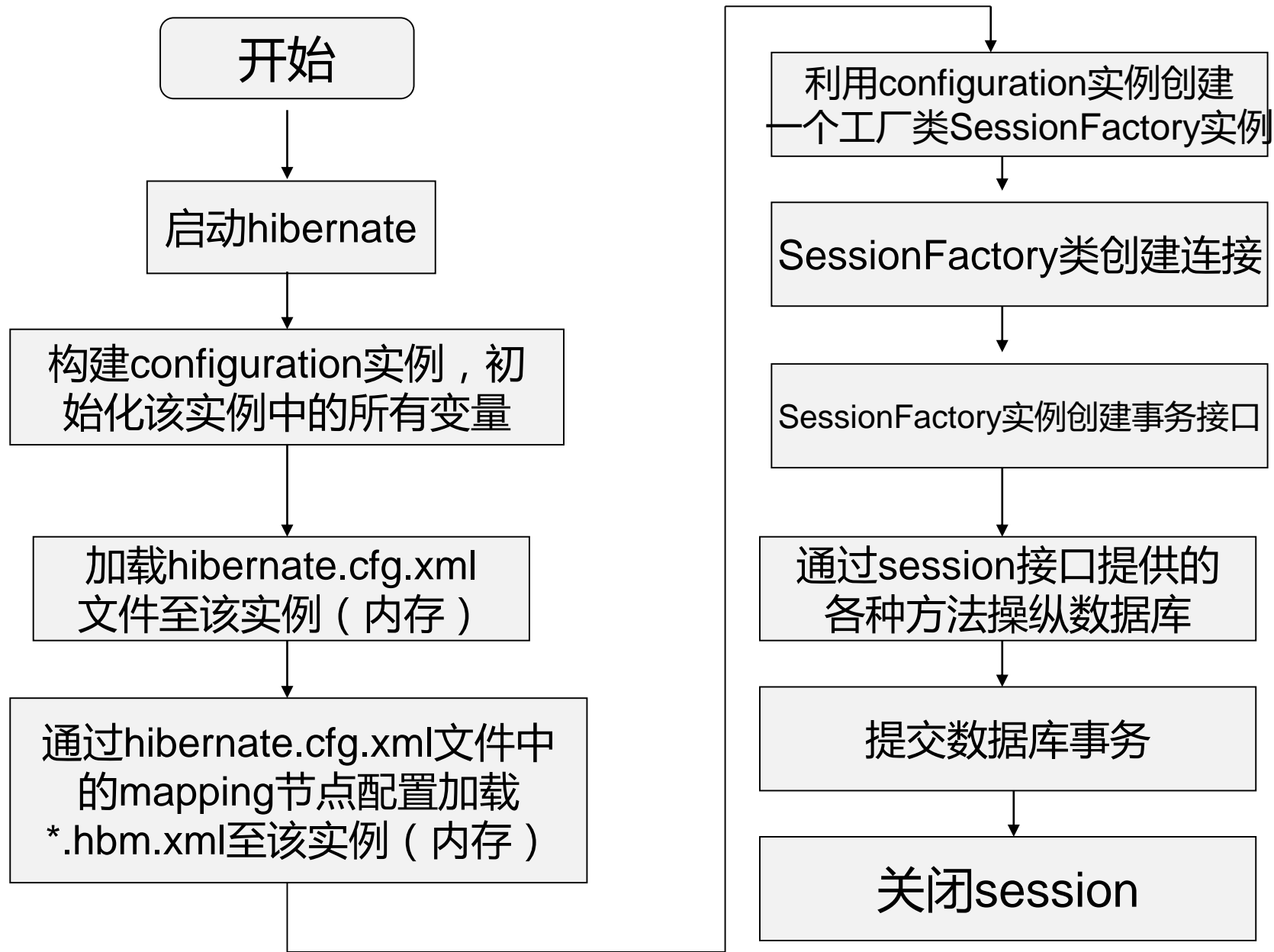
下载地址<http://www.hibernate.org>



## • Hibernate工作原理 •

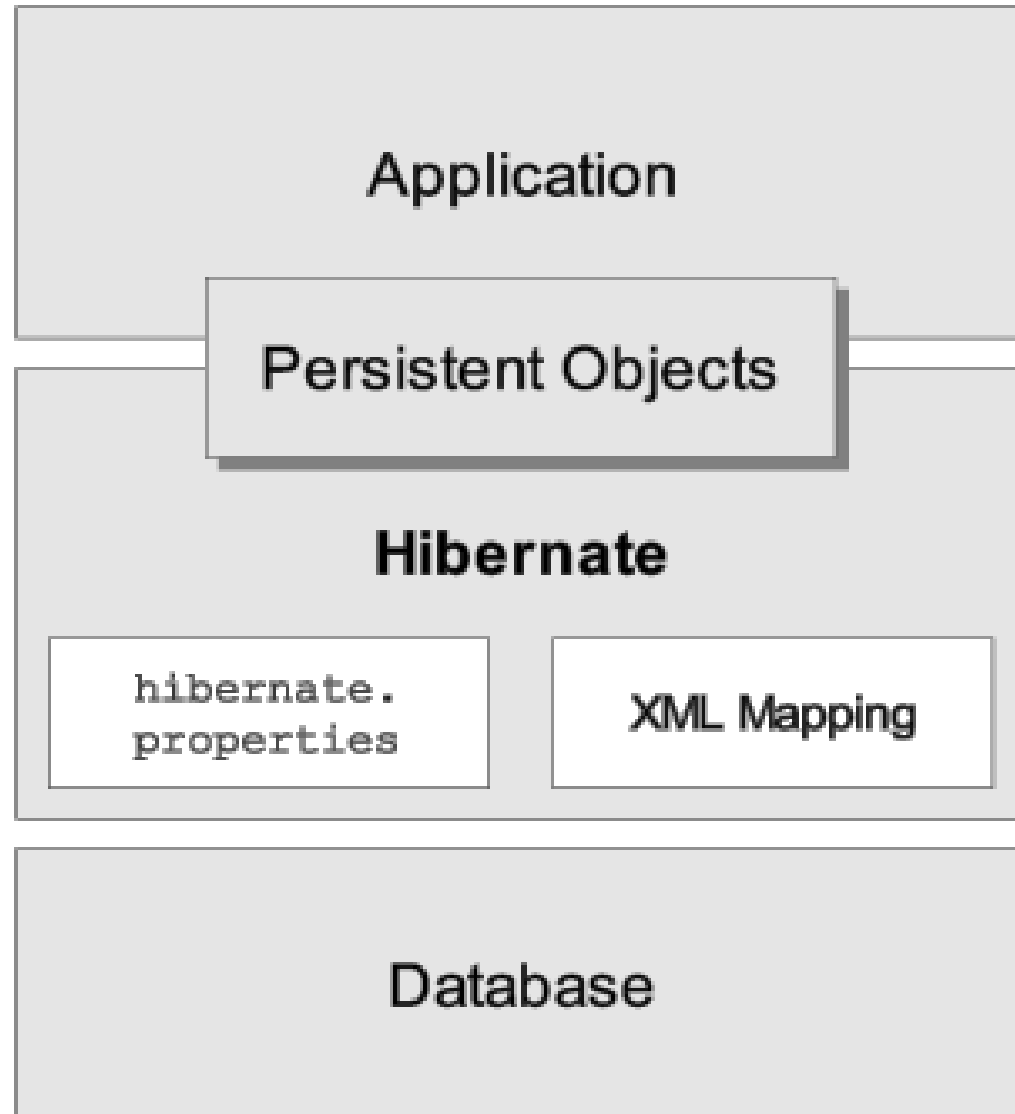
- 1、读取并解析配置文件
- 2、读取并解析映射信息，创建SessionFactory
- 3、打开Session
- 4、创建事务Transaction
- 5、持久化操作
- 6、提交事务
- 7、关闭Session
- 8、关闭SessionFactory

# • Hibernate工作原理 •



## • Hibernate开发步骤

- 1、持久化类的设计
- 2、持久化类和关系数据库的映射
- 3、应用的开发



## 持久化Java类必须遵循的原则



















- 1、为类的持久化类字段声明访问方法（ get/set ）。Hibernate对JavaBeans风格的属性实行持久化。
- 2、实现一个默认的构造方法（ constructor ）。Hibernate就可以使用 `Constructor.newInstance()` 来实例化它们。
- 3、如果是集合类型的属性，它的类型必须定义为集合的接口。  
例如：List、Set。
- 4、提供一个标识属性（ identifier property ）。如果没有该属性，一些功能不起作用，比如：级联更新（ Cascaded updates ） `Session.saveOrUpdate()`。

# • Hibernate开发实例 •


- 1、创建Dynamic Web Project项目，命名为hibernate
- 2、导入开发hibernate必要jar包
- 3、加载MySQL驱动jar包

 hibernate-release-5.4.0.Final.zip

( 1 )

 antlr-2.7.7.jar  
 byte-buddy-1.9.5.jar  
 classmate-1.3.4.jar  
 dom4j-2.1.1.jar  
 FastInfoset-1.2.15.jar  
 hibernate-commons-annotations-5.1....  
 hibernate-core-5.4.0.Final.jar  
 istack-commons-runtime-3.0.7.jar  
 jandex-2.0.5.Final.jar  
 javassist-3.24.0-GA.jar  
 javax.activation-api-1.2.0.jar  
 javax.persistence-api-2.2.jar  
 jaxb-api-2.3.1.jar  
 jaxb-runtime-2.3.1.jar  
 jboss-logging-3.3.2.Final.jar  
 jboss-transaction-api\_1.2\_spec-1.1.1....  
 stax-ex-1.8.jar  
 txw2-2.3.1.jar

( 2 )

 mysql-connector-java-5.1.6-bin.jar

( 3 )

# • Hibernate开发实例 •

## 4、创建数据库，准备表

```
SET FOREIGN_KEY_CHECKS=0
```

```
DROP TABLE IF EXISTS `tuserlogin`;
```

```
CREATE TABLE `tuserlogin` (
```

```
  `id` int(11) NOT NULL auto_increment,
```

```
  `username` varchar(20) NOT NULL,
```

```
  `password` varchar(20) default NULL,
```

```
  `grade` int(11) default '1',
```

```
  `email` varchar(50) default NULL,
```

```
  PRIMARY KEY (`id`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
insert into tuserlogin values(0,'admin','123456',3,'adminl@126.com');
```

```
insert into tuserlogin values(0,'虚拟用户','123456',1,'test@126.com');
```

```
insert into tuserlogin values(0,'测试用户1','123456',1,'testl@126.com');
```

```
insert into tuserlogin values(0,'测试用户2','123456',1,'abc@126.com');
```

# • Java链接数据库 •

## 5、创建com.bbu.model包，创建User类

```
package com.bbu.model;

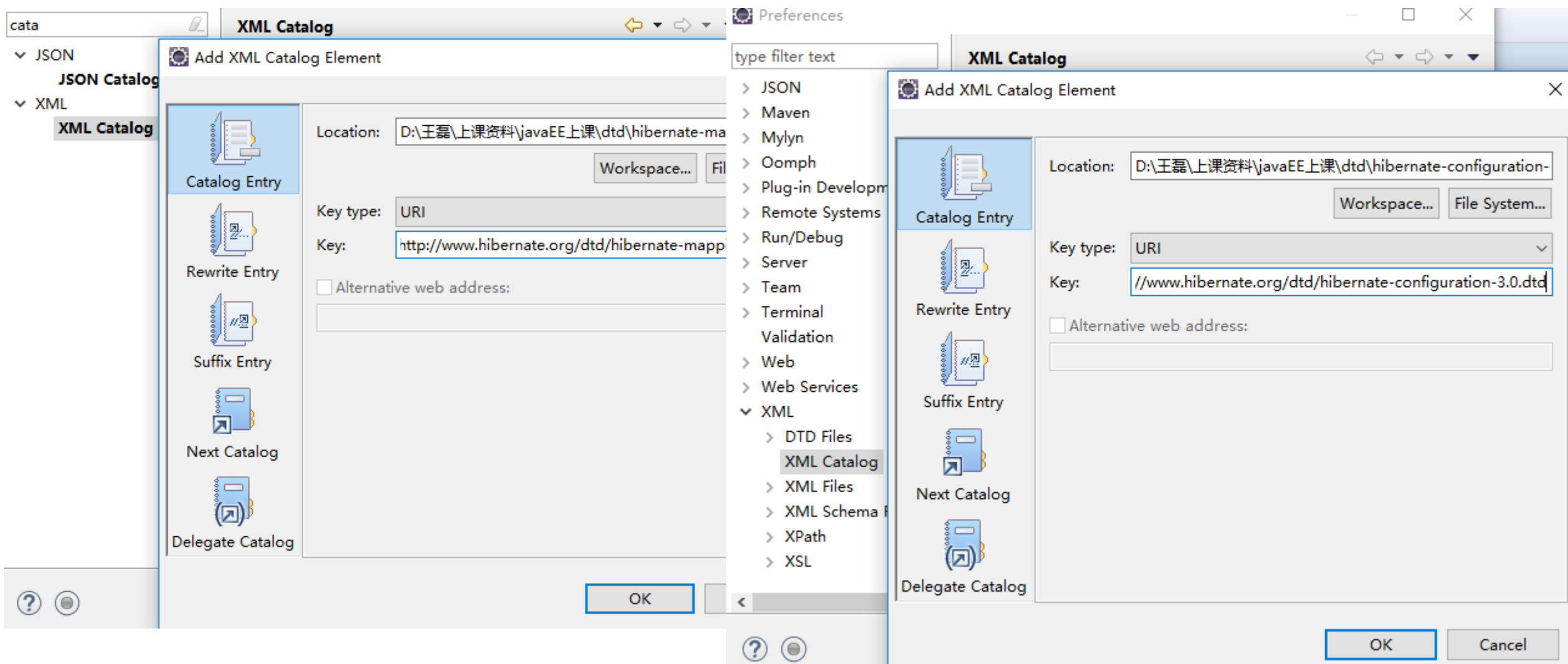
public class User {

    private Integer id;
    private String username;
    private String password;
    private Integer grade;
    private String email;
    public Integer getId() {
        return id;
    }
    public void setId(Integer id) {
        this.id = id;
    }
    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
}
```



# • 导入本地dtd约束 •

## 6、导入约束dtd



# • 书写对象与表的映射配置文件 •

## 7、在com.bbu.model包下创建User.hbm.xml配置文件

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
    "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
<class name="com.bbu.model.User" table="tuserlogin">
<id name="id" column="id">
<generator class="native"></generator>
</id>
<property name="username" column="username"></property>
<property name="password" column="password"></property>
<property name="grade" column="grade"></property>
<property name="email" column="email"></property>
</class>
</hibernate-mapping>
```

# • 创建hibernate.cfg.xml主配置文件 •

## 8、在src下创建hibernate.cfg.xml配置文件

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
<session-factory>
<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
<property
name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
<property name="hibernate.connection.url">jdbc:mysql:///usermanager</property>
<property name="hibernate.connection.username">root</property>
<property name="hibernate.connection.password">root</property>
<property name="hibernate.show_sql">>true</property>
<property name="hibernate.format_sql">>true</property>
<property name="hibernate.hbm2ddl.auto">update</property>
<mapping resource="com/bbu/model/User.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

# • 书写代码 •

9、创建com.bbu.junit包，创建Demo测试类

```
public class Demo {  
    @Test  
    public void crud() {  
        Configuration conf = new Configuration().configure();  
        SessionFactory sessionFactory = conf.buildSessionFactory();  
        Session session = sessionFactory.openSession();  
        Transaction tx = session.beginTransaction();  
        User user = new User();  
        user.setUsername("baidu");  
        user.setPassword("123");  
        user.setGrade(1);  
        session.save(user);  
        tx.commit();  
        session.close();  
        sessionFactory.close();  
    }  
}
```

# 测试结果

## 10、测试结果

```
Hibernate:
insert
into
    tuserlogin
    (username, password, grade, email)
values
    (?, ?, ?, ?)
```

95	oracle6	123456	1	oracle@163.com
96	mysql7	123456	1	mysql@126.com
97	baidu	123	1	(Null)

## • 作业 •

- 1、实现hibernate对数据库的crud基本操作。
- 2、熟悉框架开发流程。