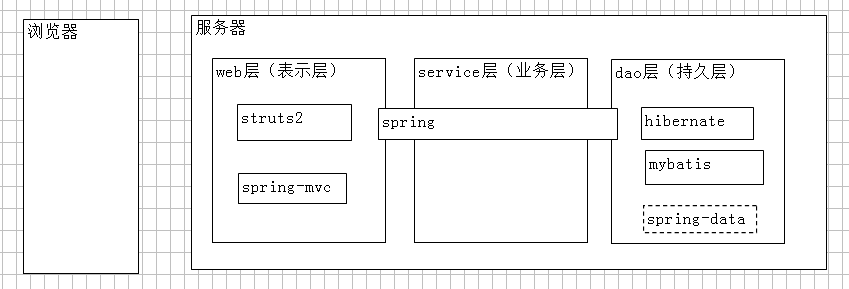
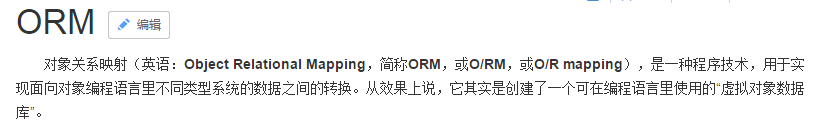
# 框架体系结构



# hibernate入门

## ORM框架

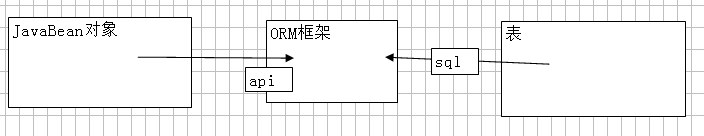


Hibernate是一个数据持久化层的ORM框架.

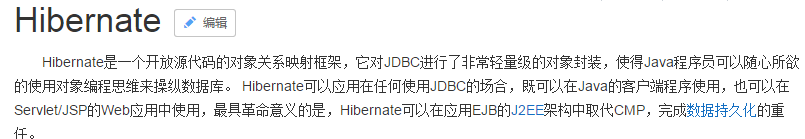
Object：对象，java对象，此处特指JavaBean

Relational：关系，二维表，数据库中的表。

映射|映射元数据：对象中属性，与表的字段，存在对应关系。



## 什么是hibernate



* Hibernate是轻量级JavaEE应用的持久层解决方案，是一个关系数据库ORM框架
  + ORM 就是通过将Java对象映射到数据库表，通过操作Java对象，就可以完成对数据表的操作
* Hibernate提供了对关系型数据库增删改成操作

## 主流的ORM框架

* + JPA Java Persistence API.JPA通过JDK 5.0注解或XML描述对象－关系表的映射关系（只有接口规范）
  + Hibernate 最流行ORM框架，通过对象-关系映射配置，可以完全脱离底层SQL
  + MyBatis 本是apache的一个开源项目 iBatis，支持普通 SQL查询，存储过程和高级映射的优秀持久层框架
  + Apache DBUtils 、Spring JDBCTemplate

## 优点

|  |
| --- |
| * Hibernate对JDBC访问数据库的代码做了封装，大大简化了数据访问层繁琐的重复性代码 * Hibernate是一个基于jdbc的主流持久化框架，是一个优秀的orm实现，它很大程度的简化了dao层编码工作 session.save(User); * Hibernate使用java的反射机制 * Hibernate的性能非常好，因为它是一个轻量级框架。映射的灵活性很出色。它支持很多关系型数据库，从一对一到多对多的各种复杂关系 |

# 入门案例【掌握】

## 编写流程

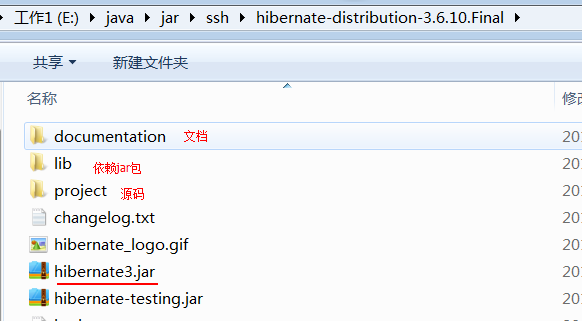
* 1. 导入jar包
* 2. 创建数据库和表
* 3. 编写核心配置文件（hibernate.cfg.xml）--> 配置获得链接等参数
* 4. 编写映射文件 hibernate mapping(\*.hbm.xml)
* 5 使用api测试

## 数据库和表

|  |
| --- |
| create database h\_day01\_db;  use h\_day01\_db;  create table t\_user(  id int auto\_increment primary key,  username varchar(50),  password varchar(30)  ); |

## 导入jar包

* 版本：3.6.10 --> hibernate 4 建议注解开发，hibernate 4 对 3 不兼容。
* 目录结构



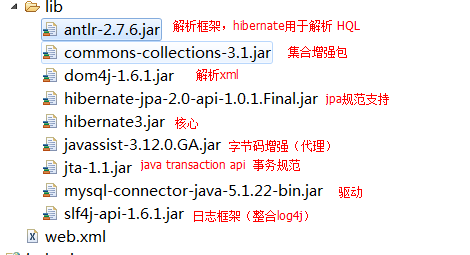
* jar介绍

核心：

必须：\lib\required

jpa规范：lib\jpa

mysql驱动： 



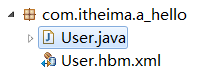
## 编写JavaBean + 映射文件

* 文件位置：javabean同包
* 文件名称：javabean同名
* 扩展名：\*.hbm.xml
* 内容：

添加约束







|  |
| --- |
| **public** **class** User {    /\*  \* create table t\_user(  id int auto\_increment primary key,  username varchar(50),  password varchar(30)  );  \*/  **private** Integer uid;  **private** String username;  **private** String password; |
| <!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.itheima.a\_hello.User"* table=*"t\_user"*>  <!-- 主键 -->  <id name=*"uid"*>  <!-- 固定值：主键生成策略 -->  <generator class=*"native"*></generator>  </id>    <!-- 普通属性 -->  <property name=*"username"*></property>  <property name=*"password"*></property>    </class>  </hibernate-mapping> |

## 编写核心配置文件

* 位置：类路径（classpath、src）-->WEB-INF/classes
* 名称：hibernate.cfg.xml
* 内容：

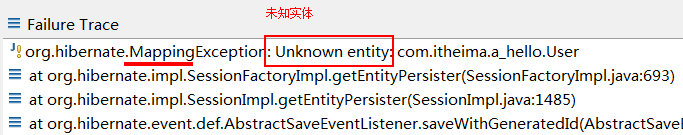
添加约束

|  |
| --- |
| <!DOCTYPE hibernate-configuration PUBLIC  "-//Hibernate/Hibernate Configuration DTD 3.0//EN"  "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">  <hibernate-configuration>  <!-- SessionFactory，相当于之前学习连接池配置 -->  <session-factory>  <!-- 1 基本4项 -->  <property name=*"hibernate.connection.driver\_class"*>com.mysql.jdbc.Driver</property>  <property name=*"hibernate.connection.url"*>jdbc:mysql:///h\_day01\_db</property>  <property name=*"hibernate.connection.username"*>root</property>  <property name=*"hibernate.connection.password"*>1234</property>    <!-- 添加映射文件 -->  <mapping resource=*"com/px/bean/User.hbm.xml"*/>  </session-factory>  </hibernate-configuration> |

## 测试

|  |
| --- |
| @Test  **public** **void** demo01(){  User user = **new** User();  user.setUsername("伟哥哥");  user.setPassword("1234");      //1 加载配置文件获得核心配置对象  Configuration config = **new** Configuration().configure();  //2 获得工厂 SessionFactory，相当于连接池  SessionFactory factory = config.buildSessionFactory();  //3获得会话session，相当于链接Connection  Session session = factory.openSession();  //4 开启事务  Transaction transaction = session.beginTransaction();    //操作  session.save(user);    //5 提交事务 | 回滚事务  transaction.commit();  //6 释放资源--关闭session  session.close();  //7 释放资源--关闭工厂factory  factory.close();  } |

## 常见异常



解决方案：

将映射文件添加到核心配置文件中 hbm.xml --> hibernate.cfg.xml