知识回顾：

1. Maven统一项目管理工具

优点：

1. 管理jar包，依赖jar包。pom.xml文件中输入jar信息（坐标），不用像javaWeb工程非要熟悉项目中要用到所有的jar包。只引入需要的核心的jar包。spring-tx事务，自动依赖bean,core。变的模糊。
2. 命令：clean清除，删除target目录（编译完的class文件）目标是验证maven环境是否安装正常。

tomcat7:run

生命周期：complier – test – war - deploy – run

Install自动完成前面命令，生成war包

Maven命令是有特点的，自动执行前面的命令，不会执行后面的命令。

大生命周期3阶段：clean清除/ surefire报告文件（不用）/deploy（重要）

[INFO] Scanning for projects... 扫描工程

[INFO]

[INFO] ------------------------------------------------------------------------

[INFO] Building CGB-JT-SYS-V1.05 0.0.1-SNAPSHOT

[INFO] ------------------------------------------------------------------------

[INFO]

[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ CGB-JT-SYS-V1.05 ---

[WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dependent!

[INFO] Copying 14 resources

[INFO]

[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ CGB-JT-SYS-V1.05 ---

[INFO] Nothing to compile - all classes are up to date

[INFO]

[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ CGB-JT-SYS-V1.05 ---

[WARNING] Using platform encoding (GBK actually) to copy filtered resources, i.e. build is platform dependent!

[INFO] Copying 0 resource

[INFO]

[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ CGB-JT-SYS-V1.05 ---

[INFO] Nothing to compile - all classes are up to date

[INFO]

[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ CGB-JT-SYS-V1.05 ---

[INFO]

[INFO] --- maven-war-plugin:2.2:war (default-war) @ CGB-JT-SYS-V1.05 ---

[INFO] Packaging webapp

[INFO] Assembling webapp [CGB-JT-SYS-V1.05] in [D:\java\ws\o\CGB-JT-SYS-V1.05\target\CGB-JT-SYS-V1.05-0.0.1-SNAPSHOT]

[INFO] Processing war project（非常慢，第三方jar包 WEB-INF/lib）

[INFO] Copying webapp resources [D:\java\ws\o\CGB-JT-SYS-V1.05\src\main\webapp]

[INFO] Webapp assembled in [15358 msecs]

[INFO] Building war: D:\java\ws\o\CGB-JT-SYS-V1.05\target\CGB-JT-SYS-V1.05-0.0.1-SNAPSHOT.war

[INFO] WEB-INF\web.xml already added, skipping

[INFO]

[INFO] --- maven-install-plugin:2.4:install (default-install) @ CGB-JT-SYS-V1.05 ---

[INFO] Installing D:\java\ws\o\CGB-JT-SYS-V1.05\target\CGB-JT-SYS-V1.05-0.0.1-SNAPSHOT.war to D:\java\env\mvn\_repo\com\jt\CGB-JT-SYS-V1.05\0.0.1-SNAPSHOT\CGB-JT-SYS-V1.05-0.0.1-SNAPSHOT.war

[INFO] Installing D:\java\ws\o\CGB-JT-SYS-V1.05\pom.xml to D:\java\env\mvn\_repo\com\jt\CGB-JT-SYS-V1.05\0.0.1-SNAPSHOT\CGB-JT-SYS-V1.05-0.0.1-SNAPSHOT.pom

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

[INFO] Total time: 28.148 s

[INFO] Finished at: 2018-06-13T09:14:43+08:00

[INFO] Final Memory: 18M/585M

[INFO] ------------------------------------------------------------------------

eclipse中配置maven怎么配置？

默认会有maven插件，这个插件功能有限，而且出现很多异常bug。

实际开发中要自己配置自己的maven插件。

1. 安装apache maven绿色解压，配置conf/settings.xml文件

默认会自己定义本地仓库位置：C:\Users\tony\.m2。仓库文件就会放入其中。

有隐患。C盘操作系统所在盘符，重装系统时就可能被格式化，文件都丢失，不要重复发生。

所以需要配置换到其他盘符。先把目录创建好，目录不能有中文目录和含空格目录。

<localRepository>D:/java/env/mvn\_repo</localRepository>

1. 默认下载jar到maven提供中央仓库。国外网站，镜像仓库（私服）

<mirror>

<id>nexus-tedu</id>

<mirrorOf>central</mirrorOf>

<name>Nexus tedu</name>

<url>http://maven.aliyun.com/nexus/content/groups/public/</url>

</mirror>

问题：

环境出错，特点：一旦解决问题，后续很少出错。

1. plugin插件报错

D:\java\env\mvn\_repo\org\apache\maven\plugins

1. archetype新建项目出错

D:\java\env\mvn\_repo\org\apache\maven\archetypes

特点：骨架很烂，少文件少目录，自己创建。动不动就坏了。（删除）

1. 导入Maven工程

推荐方式：

新建一个项目MavenProject

拷贝旧项目的src目录和pom.xml

绝招：project菜单clean命令。内部干什么事情，梳理整个的项目，校验所有的文件。把工程内容资源也重新链接（jdk,tomcat）

Maven 有时报Dynamic Web Module 2.5，有时不报。不影响代码，只是不美观，工程上有x。

今日任务：

1. 新三大框架SSM/SSI/SSH

jsp +servlet +jdbc（ResultSet,SQL）面向过程

SSH： struts2/1 +spring +hibernate ORM 面向对象，（反射，性能低）自动把数据映射到java对象

SSI： struts2 +spring +ibatis （SQL） 半ORM （性能高）

SSM: springmvc +spring +mybatis（apache/google）

Dubbo 微服务

Springboot+springCloud

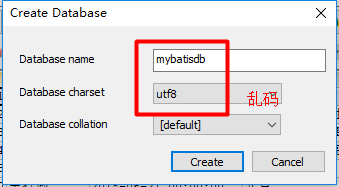
Mybatis解决什么问题？

Hiberante业界定位小型项目。Struts2+Hibernate严重Bug出现了很多次

Mybatis业界定位大中型项目。Springmvc+mybatis没有严重bug

工作原理图，核心对象

回顾jdbc



package mybatis1803;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import org.junit.Test;

import com.mysql.jdbc.Driver;

public class TestJdbc {

@Test

public void jdbc() throws SQLException{

String url = "jdbc:mysql://localhost:3306/mybatisdb?characterEncoding=utf8";

new Driver();

Connection cn = DriverManager.getConnection(url, "root", "root");

Statement stat = cn.createStatement();

String sql = "SELECT \* FROM user";

ResultSet rs = stat.executeQuery(sql);

while(rs.next()){

System.out.println(rs.getString("name"));

}

rs.close();

stat.close();

cn.close();

}

}

JDBC出“问题”？

API一堆，晦涩难记

Mybatis轻量级封装jdbc框架。直接SQL语句。

Jdbc把结果放到ResultSet对象中。

习惯使用pojo对象（普通java对象） ORM，把rs对象数据放入pojo对象中

POJO（Plain Ordinary Java Object）简单的Java对象

Java创建对象，属性，方法（业务逻辑）+

POJO对象，private属性，get获取，set设置

了解MyBatis处理ORM整个过程会用到核心对象？

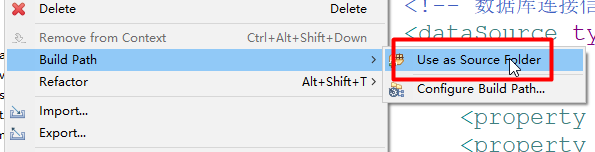


开发步骤：

1. 在pom.xml中加入依赖
2. 核心配置文件：sqlMapConfig.xml（全局、环境、事务JDBC/MANAGER、数据源POOLED/UNPOOLED/JNDI）
3. 封装业务数据对象pojo， User对象（一堆私有的属性，get/set自动产生）
4. 映射文件UserMapper.xml
5. 创建SqlSessionFactory对象，读取核心配置文件
6. 利用工厂创建SqlSession对象，执行sql语句
7. 展现数据，集合List<User>
8. 遍历集合

开发步骤：

1. 在pom.xml中加入依赖
2. 核心配置文件：sqlMapConfig.xml



<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE configuration

PUBLIC "-//mybatis.org//DTD Config 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-config.dtd">

<configuration>

<!-- 全局配置 -->

<!-- 環境 -->

<environments default="test">

<environment id="test">

<!-- 配置数据库事务：JDBC/MANAGER自己管理事务 -->

<transactionManager type="JDBC"/>

<!-- 数据库连接信息（数据源）：POOLED池/UNPOOLED/JNDI目录检索（DNS) -->

<dataSource type="POOLED">

<property name="driver" value="com.mysql.jdbc.Driver"/>

<property name="url" value="jdbc:mysql://localhost:3306/mybatisdb?characterEncoding=utf8"/>

<property name="username" value="root"/>

<property name="password" value="root"/>

</dataSource>

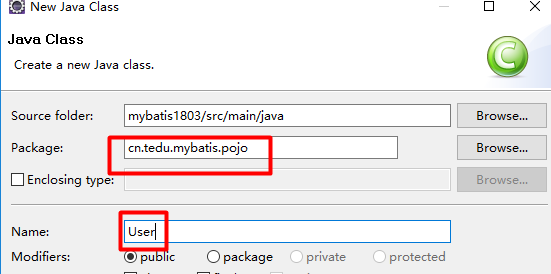
</environment>

</environments>

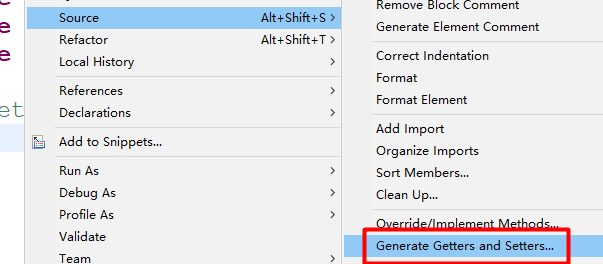
<!-- 引入映射文件 -->

</configuration>

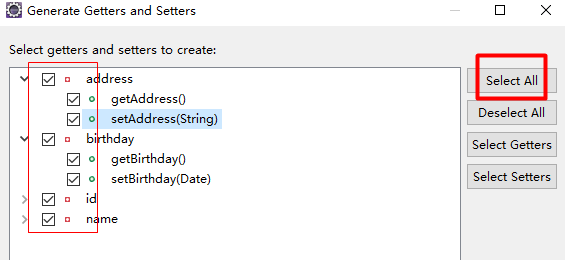
1. 封装业务数据对象pojo， User对象



Pojo类的名称User和数据库表对应USER表。



Eclipse自动生成get和set方法



package cn.tedu.mybatis.pojo;

import java.util.Date; //必须使用java.util.Date，mybatis封装对象，不能java.sql.Date

//类名和数据库表名称对应USER表

public class User {

//很多私有属性，属性和数据库表一一对应。规则：驼峰规则

private Integer id;

private String name;

private Date birthday;

private String address;

//很多get方法和set方法

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Date getBirthday() {

return birthday;

}

public void setBirthday(Date birthday) {

this.birthday = birthday;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

}

1. 映射文件UserMapper.xml

在映射文件中，Mybatis对应集合不写集合类型，它结合类型中的元素类型List<User>

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper

PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<!-- namespace命名空间 -->

<mapper namespace="cn.tedu.mybatis.pojo.UserMapper">

<!-- 查询所有，resultType返回值，标识返回类型List<> -->

<select id="find" resultType="cn.tedu.mybatis.pojo.User">

select \* from user

</select>

</mapper>

1. 创建SqlSessionFactory对象，读取核心配置文件
2. 利用工厂创建SqlSession对象，执行sql语句
3. 展现数据，集合List<User>
4. 遍历集合

package mybatis1803;

import java.io.IOException;

import java.io.InputStream;

import java.util.List;

import org.apache.ibatis.io.Resources;

import org.apache.ibatis.session.SqlSession;

import org.apache.ibatis.session.SqlSessionFactory;

import org.apache.ibatis.session.SqlSessionFactoryBuilder;

import org.junit.Test;

import cn.tedu.mybatis.pojo.User;

public class TestMybatis {

@Test //查询所有记录

public void find() throws IOException{

//读取核心配置文件

InputStream is = Resources.getResourceAsStream("sqlMapConfig.xml");

//获取到工厂对象

SqlSessionFactory factory =

new SqlSessionFactoryBuilder().build(is);

//创建SqlSession对象

SqlSession session = factory.openSession();

//查询所有的记录，namespace+"."+id

String statement = "cn.tedu.mybatis.pojo.UserMapper.find";

//selectList查询，找到映射文件，找到其中一个方法

List<User> userList = session.selectList(statement);

//遍历对象

for(User u:userList){

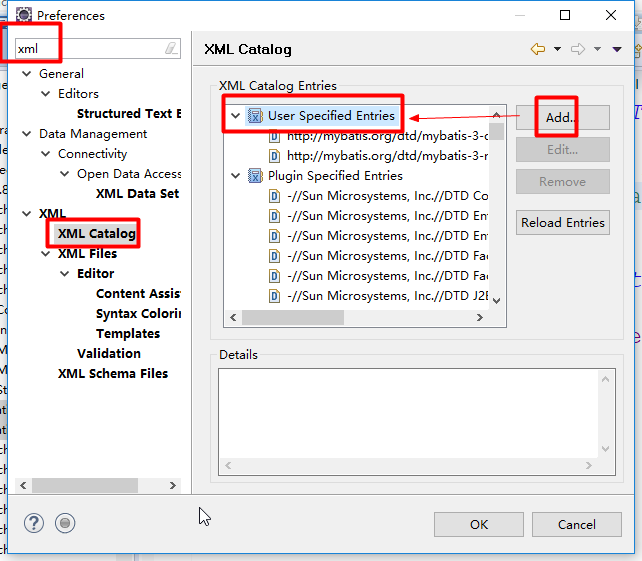
System.out.println(u);

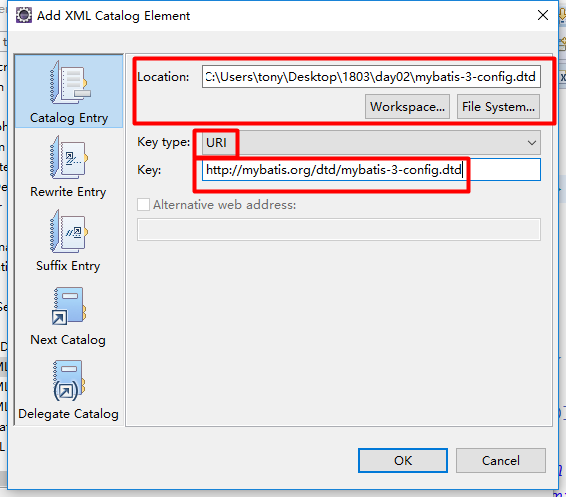
}

}

}

怎么支持xml种提示dtd约束





以后不用上网，它优先检查本地文件

通过xml的http链接，先去找eclipse配置，然后就定位本地文件

失败：config dtd配置



新增、修改、删除、总数、获取一条

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper

PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<!-- namespace命名空间 -->

<mapper namespace="cn.tedu.mybatis.pojo.UserMapper">

<!-- 查询所有，resultType返回值，标识返回类型List<> -->

<select id="find" resultType="cn.tedu.mybatis.pojo.User">

select \* from user

</select>

<!-- 只返回一个值，返回一个整数值 -->

<select id="count" resultType="integer">

SELECT COUNT(\*) FROM user

</select>

<!-- 获取一个用户的信息（一条记录） -->

<select id="get" resultType="cn.tedu.mybatis.pojo.User">

SELECT \* FROM USER WHERE id=5

</select>

<!-- 新增 -->

<insert id="insert">

INSERT INTO USER

(id,NAME,birthday,address)

VALUES(6,'tony','2018-06-19','北京')

</insert>

<!-- 修改 -->

<update id="update">

UPDATE USER

SET NAME='tony' WHERE id=6

</update>

<!-- 删除 -->

<delete id="delete">

DELETE FROM USER WHERE id=6

</delete>

</mapper>

package mybatis1803;

import java.io.IOException;

import java.io.InputStream;

import java.util.List;

import org.apache.ibatis.io.Resources;

import org.apache.ibatis.session.SqlSession;

import org.apache.ibatis.session.SqlSessionFactory;

import org.apache.ibatis.session.SqlSessionFactoryBuilder;

import org.junit.Before;

import org.junit.Test;

import cn.tedu.mybatis.pojo.User;

public class TestMybatis {

//成员变量，线程安全问题：

private SqlSessionFactory factory; //线程安全

@Before //junit，在其他方法执行前执行

public void init() throws IOException{

InputStream is = Resources.getResourceAsStream("sqlMapConfig.xml");

factory = new SqlSessionFactoryBuilder().build(is);

}

@Test

public void delete(){

//SqlSession不是一个线程安全类

SqlSession session = factory.openSession();

String statement = "cn.tedu.mybatis.pojo.UserMapper.delete";

session.delete(statement);

session.commit();

}

@Test //查询所有记录

public void find() throws IOException{

//读取核心配置文件，路径要求：

InputStream is = Resources.getResourceAsStream("sqlMapConfig.xml");

//获取到工厂对象

SqlSessionFactory factory =

new SqlSessionFactoryBuilder().build(is);

//创建SqlSession对象

SqlSession session = factory.openSession();

//查询所有的记录，namespace+"."+id

String statement = "cn.tedu.mybatis.pojo.UserMapper.find";

//selectList查询，找到映射文件，找到其中一个方法

List<User> userList = session.selectList(statement);

//遍历对象

for(User u:userList){

System.out.println(u);

}

}

@Test

public void count(){

SqlSession session = factory.openSession();

//返回一条记录，而必须返回一条记录

String statement = "cn.tedu.mybatis.pojo.UserMapper.count";

Integer i = session.selectOne(statement);

System.out.println("有"+i+"条记录。");

}

@Test

public void get(){

SqlSession session = factory.openSession();

String statement = "cn.tedu.mybatis.pojo.UserMapper.get";

User user = session.selectOne(statement);

System.out.println(user);

}

@Test

public void insert() throws IOException{

InputStream is = Resources.getResourceAsStream("sqlMapConfig.xml");

SqlSessionFactory factory = new SqlSessionFactoryBuilder().build(is);

SqlSession session = factory.openSession();

String statement = "cn.tedu.mybatis.pojo.UserMapper.insert";

session.insert(statement);

session.commit(); //mybatis事务，默认不是自动提交，手动提交

}

@Test

public void update() throws IOException{

InputStream is = Resources.getResourceAsStream("sqlMapConfig.xml");

SqlSessionFactory factory = new SqlSessionFactoryBuilder().build(is);

SqlSession session = factory.openSession();

String statement = "cn.tedu.mybatis.pojo.UserMapper.update";

session.update(statement);

session.commit();

}

}

怎么给sql语句传递动态值（参数）？

Jdbc中语句statement，statement，preparestatement预编译快，防SQL注入参数占位符id=?，proceduce存储过程

Mybatis默认使用的是prepareStatement，天然就防sql注入

Mybatis接收参数两种方式：

1）#{id} 防SQL注入（推荐）

2）${id} 直接拼接SQL语句，有SQL注入问题

引入log4j日志，就可以打印出SQL语句

小结：

