新闻发布系统开发文档

1.1、开发背景

Internet的蓬勃发展，使新闻的传播方式发生了巨大的变化，传统的信息传播媒体电视、广播、报纸等不再是人们茶余饭后的精神甜点，人们更多的开始关注网络新闻，由于互联网所容纳的信息量大、内容丰富、信息及时、准确，大大方便了人们的阅读。为了适应市场的需要，鼎盛科技有限公司现决定组织公司技术团队开发一套以新闻发布、浏览为主的新闻发布系统。

1.2、系统分析

1.2.1、需求分析

新闻发布系统的一个重要功能模块就是实现网站的动态新闻显示、新闻内容的维护、网站的投票和友情链接等。新闻发布系统分为前台用户浏览新闻和后台管理员新闻维护两大功能权限。后台管理员可以进行新闻一级标题、新闻二级标题的添加、删除、修改，也包括对管理员的管理及密码的修改。同时本系统各还具有链接管理和人物投票管理等功能。

1.2.2、可行性分析

鼎盛科技有限公司是一家以软件开发为主的高科技公司。公司根据目前市场的需求，现委托公司的研发部门开发一个集新闻发布、浏览、检索为主的网站，项目名称定为新闻发布系统。

①、功能可行性分析

新闻发布系统网站要求准确、及时、全面地提供各行业的新闻信息，如：军事、娱乐、科技、财经、体育等方面的新闻信息。考虑到对企业及国家的影响，对一些非法、不健康的信息要及时删除。此外，应加强网站的安全性，避免有意或无意的破坏导致系统瘫痪，造成严重损失。

②、投资及效益分析

·支出：根据预算，公司计划投入8个人，为此需要支付9万元的工资及各种福利待遇；项目的安装、调试以及用户培训、员工出差等费用支出需要2万元；在项目后期维护阶段预计需要投入2万元的资金，项目累计投入需要13万元。

·收益：客户提供项目资金30万元。对于项目运行后进行的改动，采取协商原则，根据改动规模额外提供资金。因此，从投资与收益的效益比上，公司可以获得17万元的利润。

根据上面的分析，在技术上不会存在问题，因此项目延期的可能性很小。在效益上，公司投入8个人、2个月的时间获利17万元，比较可观。公司还可以借此储备网站开发的经验和资源。因此，该项目可以开发。

1.2.3、编写项目计划书

①、项目完成后应交付的成果

新闻发布系统网站是鼎盛科技有限公司的待开发项目，可为用户提供军事、娱乐、住房、旅游和体育等方面的新闻。项目开发周期为两个月。项目完成后，应向公司提交如下成果：

·编译后的新闻发布系统的资源文件、系统数据库文件和系统使用说明书。

·将开发的新闻发布系统发布到Internet上，并进行后期的维护。

②、项目开发的软硬件需求

·项目开发环境：操作系统为Windows2003，需要安装JDK1.6以上版本的开发包，选用Tomcat6.0作为服务器，采用SQL Server2005数据库系统。

·项目验收方式与依据：项目开发完成后，首先进行内部测试。验收通过后，交给客户进行验收，验收的主要依据为需求规格说明书。

③、项目团队组织

公司针对该项目组建了一个由公司副经理、项目经理、系统分析员、软件工程师、网页设计师和测试人员构成的开发团队。人员分工如下表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 姓名 | 技术水平 | 所属部门 | 工作描述 |
| 秦某 | MBA | 经理部 | 负责项目审批、决策的实施 |
| 汉某 | MBA | 项目开发部 | 负责项目的前期分析、策划、项目开发进度的跟踪、项目质量的检查 |
| 魏某 | 系统分析员 | 项目开发部 | 负责系统功能分析框架设计 |
| 唐某 | 软件工程师 | 项目开发部 | 负责软件设计与编码 |
| 宋某 | 软件工程师 | 项目开发部 | 负责软件设计与编码 |
| 元某 | 软件工程师 | 项目开发部 | 负责软件编码 |
| 明某 | 美工设计师 | 设计部 | 负责网页风格的确定、网页图片的设计 |
| 清某 | 系统测试工程师 | 项目开发部 | 对测试软件进行、编写软件测试文档 |

1.3、系统设计

新闻发布系统分为前台和后台两部分，系统前台提供了新闻内容的分类、浏览、查询，以大量的新闻信息展示了网站的主题。新闻按不同类别进行分类显示，同时新闻按照发布的时间依次显示，并进行分页显示。系统前台首页显示各栏目的最新3条新闻，同时能够在不同栏目中按照新闻的标题进行模糊查询。

系统后台的有效管理是网站的重要功能需要，也是动态性能的保证。后台的登录权限分为普通管理员权限和超级管理员权限两种，保证了系统的安全性。超级管理员具有普通管理员的所有权限并能对普通管理员进行管理设置。普通管理员在后台能完成对各新闻类别、新闻内容的添加、修改、删除操作，完成新闻人物的信息添加、修改和删除操作。

1.3.1、系统目标

根据需求分析以及与开发人员的沟通，新闻发布系统需要达到以下目标：

·界面设计友好、美观。

·能够实现站内新闻搜索，如模糊查询

·对用户输入的数据，进行严格的数据检验。

·具有操作方便、功能强大的后台新闻管理功能

·具有易维护性和易操作性

1.3.2、系统功能结构

新闻发布系统分为前台、后台两部分。前台功能结构图如下所示：

**新闻发布系统前台**

**新闻搜索**

**新闻显示**

**模糊搜索**

**列表显示**

**详细显示**

**首页新闻列表显示**

**类别下新闻列表显示**

**搜索结果列表显示**

后台功能结构图如下所示：

**新闻发布系统后台**

**退出登录**

**管理员管理**

**新闻管理**

**新闻显示**

**修改密码**

**新**

**闻**

**添**

**加**

**新**

**闻**

**删**

**除**

**新**

**闻**

**修**

**改**

**详**

**细**

**显**

**示**

**列**

**表**

**显**

**示**

1.3.3、系统流程图

新闻发布系统的前台系统流程图如下所示：

**用**

**户**

访问

**前台**

**首页**

**查看某类别下的新闻**

**显示**

**查看新闻的详细内容**

**站内搜索**

**人物投票**

**友情链接**

新闻发布系统的后台系统流程图如下所示：

登录

**后台**

**首页**

**管理员**

**修改**

**新闻管理**

**管理员管理**

**添加**

**新闻人物管理**

**修**

**改**

**密**

**码**

**删除**

**新闻人物管理**

1.3.4、系统预览

供求信息网中包含多个页面，现列出几个典型页面的预览：

新闻发布系统后台首页如下所示：



新闻发布页面如下所示：



用户登录页面如下所示：



1.3.5、构建开发环境

在开发供求信息网时，需要具备以下开发环境：

服务器端：

·操作系统：Windows2003

·Web服务器：Tomcat6.0

·Java开发包：JDK1.5以上

·数据库：SQL Server2005

·浏览器：IE6.0

·分辨率：最佳效果为1024×768像素

客户端：

·浏览器：IE6.0及以上版本

·分辨率：最佳效果为1024×768像素

1.3.6、文件夹组织结构

在编写代码之前，可以把系统总可能用到的文件夹先创建出来，这样不但可以方便以后得开发工作，也可以规范网站的整体架构。

news文件夹：系统根文件夹

servlet文件夹：保存自定义的servlet类

dao文件夹：保存数据库操作类

model文件夹：保存JavaBean组件

tools文件夹：保存工具类

css文件夹：保存CSS外部样式表文件

images文件夹：保存网站中应用的图片

js文件夹：保存网站中应用的JavaScript文件

1.4、数据库设计

1.4.1、数据库分析

考虑到开发成本、客户需求和项目功能等问题，决定采用SQL Server 2005作为项目中的数据库。该数据库具有很强的数据完整性、可伸缩性、可管理性以及较高的性价比。同时也提供了JDBC编程接口，可以非常方便地应用Java来操作。

1.4.2、数据库概念设计

根据对项目做的需求分析及系统设计，规划出本系统涉及的数据库实体有：新闻实体、新闻类别实体、管理员实体以及新闻人物实体。

①、新闻实体

新闻实体包括新闻编号、所属类别、新闻标题、新闻内容、发布者人、发布时间，新闻实体E-R图如下所示：

**所属类别**

**新闻编号**

**新闻标题**

**发布时间**

**新闻内容**

**发布人**

**新闻实体**

②、新闻类别实体

新闻类别实体包括：类别编号、类别标识、类别名称，其E-R图如下所示

**类别名称**

**类别编号**

**类别标识**

**新闻类别**

③、管理员实体

管理员实体包括编号、用户名和密码属性，其E-R图如下所示：

**管理级别**

**ID编号**

**管理员**

**用户名**

**密码**

④、新闻人物实体

新闻人物实体包括ID编号、人物姓名、人物出生地、人物年龄、所从事的工作、个人简介、得票数，新闻人物的E-R图如下所示：

**发布时间**

**得票数**

**个人简介**

**ID编号**

**从事的工作**

**管理员**

**姓名**

**年龄**

**出生地**

1.4.3、数据库逻辑结构

根据数据库概念设计，需要创建与实体对应的数据表，它们分别为数据表tb\_info、tb\_type和tb\_user分别对应供求信息实体、信息类别实体和管理员实体。其中数据表tb\_info与tb\_type之间相互关联。

数据库所包含的数据表结构图如下所示：

db\_news

tb\_news(新闻实体表)

tb\_type(新闻类别表)

tb\_user(管理员表)

tb\_person(新闻人物表)

根据数据库的概念设计，本系统的数据表的结构如下所示：

①、tb\_news(新闻实体表)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **是否为空** | **默认值** | **描述** |
| **id** | **int** | **no** |  | **ID自动编号** |
| **info\_type** | **int** | **yes** | **null** | **新闻类别** |
| **info\_title** | **varchar(50)** | **yes** | **null** | **新闻标题** |
| **info\_content** | **varchar(3000)** | **yes** | **null** | **新闻内容** |
| **info\_creator** | **varchar(50)** | **yes** | **null** | **创建者** |
| **info\_date** | **datetime** | **yes** | **null** | **发布时间** |

②、tb\_type(新闻类别表)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **是否为空** | **默认值** | **描述** |
| **id** | **int** | **no** |  | **ID自动编号** |
| **type\_name** | **varchar(50)** | **yes** | **null** | **类别名称** |

③、tb\_user(管理员表)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **是否为空** | **默认值** | **描述** |
| **id** | **int** | **no** |  | **ID自动编号** |
| **user\_name** | **varchar(20)** | **yes** | **NULL** | **管理员名称** |
| **user\_password** | **varchar(20)** | **yes** | **NULL** | **密码** |
| **user\_type** | **varchar(2)** | **yes** | **NULL** | **管理员级别** |

④、tb\_person(新闻人物表)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **是否为空** | **默认值** | **描述** |
| **id** | **int** | **no** |  | **ID自动编号** |
| **person\_name** | **varchar(20)** | **yes** | **NULL** | **姓名** |
| **person\_address** | **varchar(50)** | **yes** | **NULL** | **地址** |
| **person\_age** | **int** | **yes** |  | **年龄** |
| **person\_job** | **varchar(50)** | **yes** | **NULL** | **工作** |
| **person\_remark** | **varchar(500)** | **yes** | **NULL** | **个人简介** |
| **person\_number** | **int** | **yes** | **0** | **得票数** |
| **person\_time** | **datetime** | **yes** | **NULL** | **发布时间** |

1.5、公共类设计

在开发过程中，经常会遇到在不同的方法中进行相同处理的情况，为了避免重复编码，可将这些处理封装到单独的类中，通常称这些类为公共类或工具类。本系统中用到的公共类包括：数据库连接及操作类、业务处理类、分页类和字符串处理类。

1.5.1、数据库连接及操作类----DB类

DB类主要是针对数据库的操作，如连接、关闭数据库、执行SQL语句操作数据库。每一种操作均对应一个方法。具体实现过程如下所示：

**package** com.news.dao;

**import** java.sql.\*;

//import com.microsoft.sqlserver.jdbc.SQLServerDriver;

**public** **class** DB {

//加载数据库驱动程序( 静态常量)

**private** **static** **final** String *DRIVER\_CLASS* = "com.microsoft.sqlserver.jdbc.SQLServerDriver";

//连接数据库驱动程序的URL

**private** **static** **final** String *DRIVER\_URL* = "jdbc:sqlserver://localhost:1433;DatabaseName=db\_news";

//数据库用户名和密码

**private** **static** **final** String *DRIVER\_USERNAME* = "sa";

**private** **static** **final** String *DRIVER\_PASSWORD* = "";

**private** Connection conn = **null**;//连接数据库对象

**private** PreparedStatement pStm = **null**;//操作数据库对象

**private** ResultSet res = **null**;//结果集对象

//注册数据库驱动程序(只能在某个方法体里)

**public** DB(){

**try** {

Class.*forName*(*DRIVER\_CLASS*);

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

}

}

/\*\*连接数据库的方法\*/

**private** Connection getConn(){

**try** {

conn = DriverManager.*getConnection*(*DRIVER\_URL*, *DRIVER\_USERNAME*, *DRIVER\_PASSWORD*);

} **catch** (SQLException e) {

e.printStackTrace();

System.*out*.println(e.getMessage());

}

**return** conn;

}

/\*\*数据库增删改查的方法\*/

**public** **void** doPstm(String sql,Object[] params){

**if**(sql != **null** && !sql.equals("")){

//sql命令中没有占位符时，把数组的长度设为0，因为不缺定会传哪种sql命令

**if**(params == **null**){

params = **new** Object[0];

}

//获取数据库连接对象

conn = **this**.getConn();

**if**(conn != **null**){//连接了数据库，就获取数据库操作对象pStm

**try** {

pStm = conn.prepareStatement(sql,ResultSet.*TYPE\_SCROLL\_INSENSITIVE*,ResultSet.*CONCUR\_READ\_ONLY*);

//每循环一次为占位符赋一次值

**for**(**int** i=0;i<params.length;i++){

pStm.setObject(i+1, params[i]);

}

pStm.execute();//判断数据是否插入成功返回true/false

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

}

/\*\*获取结果集对象的方法\*/

**public** ResultSet getRs(){

**try** {

res = pStm.getResultSet();//获取查询的结果集保存到res中

} **catch** (SQLException e) {

e.printStackTrace();

System.*out*.println(e.getMessage());

}

**return** res;

}

/\*\*获取影响的记录数的方法\*/

**public** **int** getCounts(){

**int** count = -1;

**try** {

count = pStm.getUpdateCount();//获取被修改的记录的数量，返回int

} **catch** (SQLException e) {

e.printStackTrace();

System.*out*.println(e.getMessage());

}

**return** count;

}

}

DB类的测试类：测试数据库是否连接成功，具体实现过程如下：

package com.news.test;

import java.sql.Connection;

import com.news.dao.\*;

public class DBTest {

public static void main(String[] args) {

DB db = new DB();

System.out.println("数据库驱动程序注册成功！");

/\*Connection conn = db.getConn();

if(conn != null){

System.out.println("数据库连接成功！");

}\*/

String sql = "insert into tb\_type(type\_name) values(?)";

Object [] params = {"财经"};

db.doPstm(sql, params);

int count = db.getCounts();

if(count > 0){

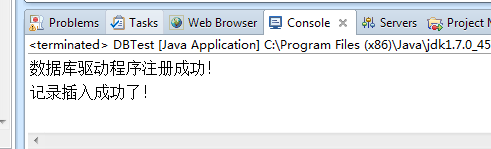
System.out.println("记录插入成功了！");

}

}

}

运行结果如图：



1.5.2、字符串操作类----DoString类

DoString类主要是字符串的操作，比如将日期型数据转换为字符型数据，解决中文乱码，通过修改字符编码格式的方法，过滤“%”，“，”等危险字符等。每一种操作均对应一种方法。具体实现过程如下所示：

**package** com.news.tools;

**import** java.util.Date;

**import** java.text.SimpleDateFormat;

//将日期型数据转换成String型数据的方法

**public** **class** DoString {

**public** **static** **final** String toStrTime(Date date){

**if**(date == **null**){

date = **new** Date();//当前系统时间

}

SimpleDateFormat hTime = **new** SimpleDateFormat("yyyy年MM月dd日 HH:mm:ss");//设置日期的显示格式

String time = hTime.format(date); //将日期转换为字符型

**return** time;

}

//修改字符编码格式的方法

**public** **static** **final** String toGBK(String str){

**if**(str == **null**){

str = "";

}

**try** {

str = **new** String(str.getBytes("ISO-8859-1"),"GBK");

str = str.trim();

} **catch** (Exception e) {

str = "";

}

**return** str;

}

//过滤危险字符的方法

**public** **static** **final** String filterStr(String str){

**if**(str == **null**){

str = "";

}

//危险字符

str = str.replaceAll("\\s", "");

str = str.replaceAll(";", "");

str = str.replaceAll("&", "");

str = str.replaceAll("<", "");

str = str.replaceAll(">", "");

str = str.replaceAll("'", "");

str = str.replaceAll("-", "");

str = str.replaceAll("/", "");

str = str.replaceAll("%", "");

str = str.replaceAll(",", "");

**return** str;

}

}

DoString类的测试类：测试字符型数据的操作是否成功，具体实现过程如下：

**package** com.news.test;

**import** com.news.tools.\*;

**public** **final** **class** DoStringTest {

**public** **static** **void** main(String[] args) {

String str = " kskdij\* jjss% f js// jdk@#$123";

System.*out*.println("过滤前："+str);

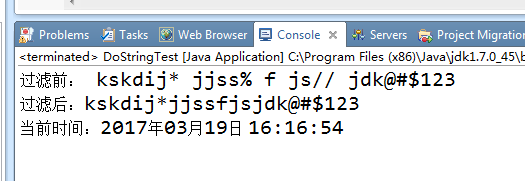
System.*out*.println("过滤后："+DoString.*filterStr*(str));

System.*out*.println("当前时间："+DoString.*toStrTime*(**null**));

}

}

运行结果如图所示：



1.6.1、新闻发布系统的实体类开发

根据需求分析，本系统涉及的实体类包括：新闻实体、新闻类别实体、管理员实体、新闻人物实体。这些实体类都放置在com.news.model包中，具体实现过程如下所示：

**Step 1：**News实体类的代码如下所示：

**package** com.news.model;

**import** java.util.Date;

**public** **class** News {

**private** **int** id;//ID编号

**private** **int** info\_type;//所属类别ID

**private** String info\_title;//新闻标题

**private** String info\_content;//新闻内容

**private** String info\_creator;//发布者

**private** Date info\_date;//发布时间

**...省略getter/setter方法...**

/\*\*无参构造器\*/

**public** News(){}

/\*\*带参构造器\*/

**public** News(**int** info\_type,String info\_title

,String info\_content,String info\_creator){

**this**.info\_type = info\_type;

**this**.info\_title = info\_title;

**this**.info\_content = info\_content;

**this**.info\_creator = info\_creator;

}

}

**Step 2：**Person实体类的代码如下所示：

**package** com.news.model;

**import** java.util.Date;

**public** **class** Person {

**private** **int** id;//ID编号

**private** String person\_name;//姓名

**private** String person\_sex;//性别

**private** **int** person\_age;//年龄

**private** String person\_address;//地址

**private** String person\_job;//工作

**private** **int** person\_number;//得票数

**private** String person\_remark;//简介

**private** Date person\_time;//发布时间

**...省略getter/setter方法...**

**public** Person(){}

**public** Person(String person\_name,String person\_sex

,**int** person\_age,String person\_address

,String person\_job,**int** person\_number

,String person\_remark){

**this**.person\_name = person\_name;

**this**.person\_sex = person\_sex;

**this**.person\_age = person\_age;

**this**.person\_address = person\_address;

**this**.person\_job = person\_job;

**this**.person\_number = person\_number;

**this**.person\_remark = person\_remark;

}

}

**Step 3：**Type实体类的代码如下所示：

**package** com.news.model;

**public** **class** Type {

**private** **int** id;//ID编号

**private** String type\_name;//类别名称

**...省略getter/setter方法...**

**public** Type(){}

**public** Type(String type\_name){

**this**.type\_name = type\_name;

}

}

**Step 4：**User实体类的代码如下所示：

**package** com.news.model;

**public** **class** User {

**private** **int** id;//ID编号

**private** String user\_name;//账号

**private** String user\_password;//密码

/\*\*级别,"0"表示普通用户,"1"表示超级用户,默认是普通用户\*/

**private** **int** user\_type;

**...省略getter/setter方法...**

**public** User(){}

**public** User(String user\_name,String user\_password){

**this**.user\_name = user\_name;

**this**.user\_password = user\_password;

}

}

1.6.2、开发业务逻辑类----OpDB

在OpDB业务处理类中，包含了多个方法，这些方法对应了本系统中用户提出的所有业务的操作，如新闻显示、新闻发布、管理员登录等。在这些方法中，通过调用DB类提供的doPstm()方法实现对底层数据库的操作。OpDB类贯穿了整个项目的开发，随着开发的逐步深入，该类的方法将不断地增加。OpDB类放置在项目的com.news.dao包中，初始时该类中没有任何方法。

①、在OpDB类中创建OpAddObject()方法，该方法包含一个Object类型的参数，用于实现对象的添加操作，包括新闻类别的添加、新闻的添加、账号的添加以及新闻人物的添加。具体代码如下所示：

**public** **boolean** OpAddObject(Object obj){

**boolean** flag = **false**;

String sql = "";

Object [] params = **new** Object[0];

**int** count = -1;

**if**(obj != **null**){

**if**(obj **instanceof** Type){

Type type = (Type)obj;

sql = "insert into tb\_type(type\_name) values(?)";

params = **new** Object[]{type.getType\_name()};

}**else** **if**(obj **instanceof** News){

News news = (News)obj;

sql = "insert into tb\_news(info\_type,info\_title

,info\_content,info\_creator) values(?,?,?,?)";

params = **new** Object[]{news.getInfo\_type()

,news.getInfo\_title(),news.getInfo\_content()

,news.getInfo\_creator()};

}**else** **if**(obj **instanceof** User){

User user = (User)obj;

sql = "insert into tb\_user(user\_name,user\_password

,user\_type) values(?,?,?)";

params = **new** Object[]{user.getUser\_name()

,user.getUser\_password(),user.getUser\_type()};

}**else** **if**(obj **instanceof** Person){

Person person = (Person)obj;

sql = "insert into tb\_person(person\_name,person\_sex

,person\_age,person\_address,person\_job,person\_number

,person\_remark) values(?,?,?,?,?,?,?)";

params = **new** Object[]{person.getPerson\_name()

,person.getPerson\_sex(),person.getPerson\_age() ,person.getPerson\_address(),person.getPerson\_job()

,person.getPerson\_number()

,person.getPerson\_remark()};

}

**if**(sql != **null** && !sql.equals("")){

DB db = **new** DB();

db.doPstm(sql, params);

count = db.getCounts();

**if**(count > 0){

flag = **true**;

}

}

}

**return** flag;

}

测试类代码如下所示：

**public** **class** OpDBTest {

**public** **static** **void** main(String[] args) {

OpDB opdb = **new** OpDB();

Type type = **new** Type("军事");

**boolean** flag = opdb.OpAddObject(type);

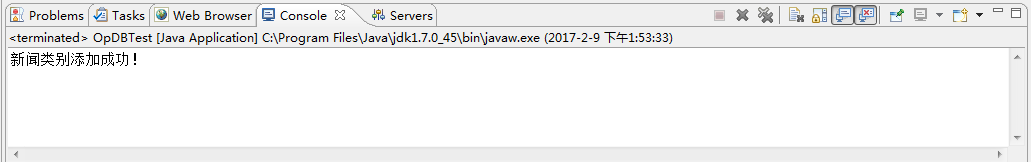
**if**(flag){

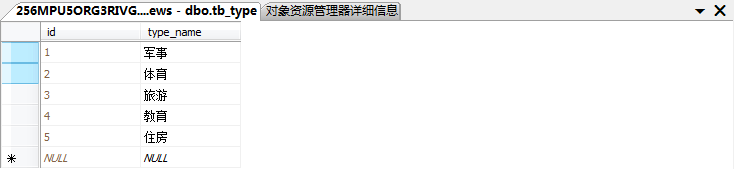
System.*out*.println("新闻类别添加成功！");

}

}

}





②、在OpDB类中添加OpGetTypeName()方法，该方法用于根据指定的id获取相应的新闻类别名称，具体代码如下所示：

**public** String OpGetTypeName(**int** id){

Type type = new Type();

String sql = "select \* from tb\_type where id="+id;

DB db = **new** DB();

db.doPstm(sql, **null**);

ResultSet res = db.getRes();

**if**(res != **null**){

**try**{

**while**(res.next()){

type.setId(res.getInt("id"));

type.setType\_name

(res.getString("type\_name"));

}

}**catch** (Exception e) {

e.printStackTrace();

type = **null**;

}

}

**return** type.getType\_name();

}

测试类代码如下所示：

**public** **class** OpDBTest {

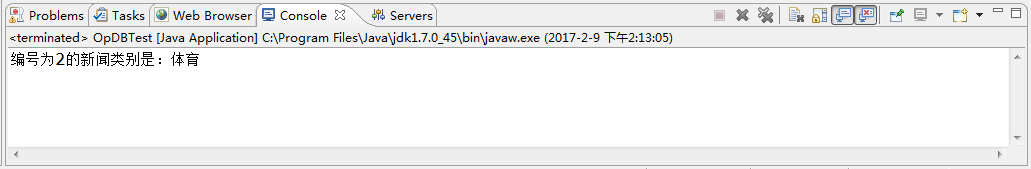
**public** **static** **void** main(String[] args) {

OpDB opdb = **new** OpDB();

System.*out*.println("类别："+ opdb.OpGetTypeName(2););

}

}



③、在OpDB类中添加OpGetTypeId()方法，该方法用于根据指定的新闻类别名称获取相应的类别ID，具体代码如下所示：

**public** **int** OpGetTypeId(String name){

**int** id = -1;

**if**(name != **null** && !name.equals("")){

String sql = "select \* from tb\_type where type\_name=?";

Object[]params = {name};

DB db = **new** DB();

db.doPstm(sql, params);

ResultSet res = db.getRes();

**if**(res != **null**){

**try**{

**while**(res.next()){

Type type = **new** Type();

type.setId(res.getInt("id"));

type.setType\_name

(res.getString("type\_name"));

id = type.getId();

}

}**catch** (Exception e) {

e.printStackTrace();

id = -1;

}

}

}

**return** id;

}

测试类的代码如下所示：

**public** **class** OpDBTest {

**public** **static** **void** main(String[] args) {

OpDB opdb = **new** OpDB();

**int** id = opdb.OpGetTypeId("住房");

**if**(id > 0){

System.*out*.println("住房的类别ID是："+id);

}

}

}

