第 28 章　新闻发布系统的设计

本章介绍的是新闻发布系统。该系统内容包括新闻发布和管理，其中涉及到文件的读写操作，将相应新闻信息生成一个静态页面，并把对应的静态页面地址保存在数据库中。在前台，通过提供该地址来浏览相应的新闻页面。该新闻发布系统中还涉及到图片上传和管理功能。

本章要点包括以下内容：

- 新闻发布系统的介绍
- 新闻发布系统的需求分析
- 新闻发布系统的功能设计
- 新闻发布系统的数据库设计
- MyWebProject 项目的创建
- Factory 和 SqlFactory 工厂类的创建

28.1　需求分析

新闻发布系统在网络中是一个很常见（例如新浪门户网站）和实用的系统。它所有的新闻信息和格式是以静态文件的形式保存在硬盘上的，而不是存储在数据库中。采用这种模式是可以避免以下问题：

（1）每次用户点击浏览某一个新闻，都要进行数据库读取操作，然后将数据按照一定格式组成返回给客户端，这会加大服务器的负担

（2）由于每次都要进行数据库操作，所以反应速度会很慢。

（2）随着存储的新闻越来越多，容易收到数据库空间的限制。

所以这些原因，导致不能将新闻信息存储在数据库中。而是直接将每条新闻写入到一个静态文件中，并保存在服务器的硬盘中。用户在客户端点击某一条新闻链接时，直接调用服务器相对应的静态文件即可。

28.1.1　用户功能

根据实际分析，该系统中普通用户只能有浏览新闻的权限。

28.1.2　管理员功能

管理员可以进行如下操作：

（1）对新闻分类的添加、修改和删除操作；

（2）进行一般新闻和图片新闻的发布操作；

（3）对所有信息进行统一管理，其中包括新闻修改、删除、排序等操作。

（4）新闻图片的上传操作；

（5）新闻图片的统一管理，其中包括图片信息的修改、图片的删除等操作。

28.2　系统功能设计

在项目开发前期，最为重要的就是需求分析、功能和数据库设计。下面首先介绍新闻发布系统的功能设计方案。

28.2.1　功能模块设计

该新闻发布系统的功能模块如图 28.1 所示。

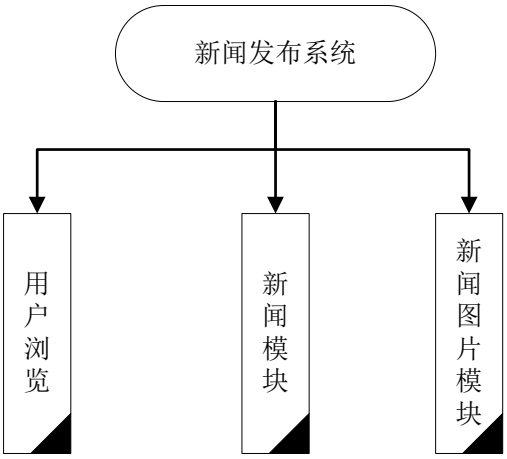

图 28.1　新闻发布系统的功能模块图

- 用户浏览：用户登陆该系统后，可以通过浏览器浏览已经发布的相关新闻信息。
- 新闻图片模块：该模块负责上传和管理新闻中所使用到的各类图片信息。
- 新闻模块：负责新闻的发布和管理，其中新闻可以发成一般新闻和图片新闻。一般新闻在客户端显示的是文字链接，而图片新闻在客户端显示的是相关图片链接。

28.2.2　页面基本流程设计

该新闻发布系统的基本流程如图 28.2 所示。

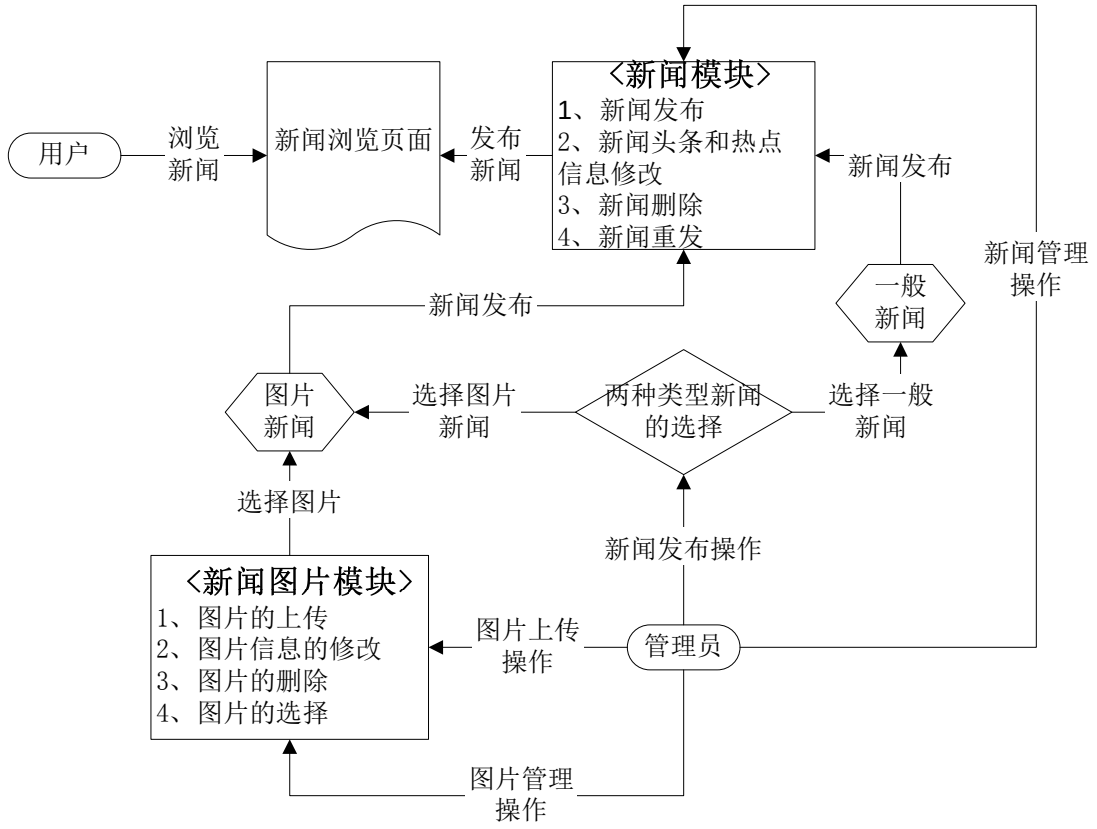

图 28.2　新闻发布系统的基本流程图

从基本流程图中可以看出：

（1）用户只可以在新闻浏览页面中进行浏览操作。除此之后，普通用户没有其他权限。

（2）管理员对新闻模块有新闻发布和新闻管理操作的权限。在进行新闻发布时，需要选择是一般新闻还是图片新闻，图片新闻需要选择一个已经上传的图片，并在浏览页面上显示的是图片链接形式。

（3）管理员还可以对新闻图片模块进行上传和管理操作。这些上传的图片将使用在新闻信息中。

28.3　数据库设计

在分析了系统的功能和基本流程之后，下面根据需要的功能来设计数据库，下面介绍的是需要创建的各类数据库表。

28.3.1　新闻表news

新闻表 news 用来存储各项新闻的基本信息，其中包括发布日期以及存放新闻静态页面的相对地址。生成改表的 SQL 语句如下：

```
DROP TABLE IF EXISTS `webshop`.`news`;
CREATE TABLE   `webshop`.`news` (
  `ID` bigint(20) unsigned NOT NULL auto_increment,
  `BoardID` int(10) unsigned NOT NULL default '0',
```

```
  `Title` varchar(100) NOT NULL default '',
  `Editor` varchar(45) NOT NULL default '',
  `FirstFrom` varchar(100) NOT NULL default '',
  `Author` varchar(45) NOT NULL default '',
  `Department` varchar(45) NOT NULL default '',
  `Keyword` varchar(100) NOT NULL default '',
  `NewsFilePath` varchar(100) NOT NULL default '',
  `Content` varchar(1024) NOT NULL default '',
  `PublishDate` bigint(20) unsigned NOT NULL default '0',
  `DCount` int(10) unsigned NOT NULL default '0',
  `IfHot` tinyint(3) unsigned NOT NULL default '0',
  `IfTop` tinyint(3) unsigned NOT NULL default '0',
  `Type` tinyint(3) unsigned NOT NULL default '0',
  `PicID` int(10) unsigned NOT NULL default '0',
  PRIMARY KEY  (`ID`)
) ENGINE=InnoDB DEFAULT CHARSET=gbk;
```

该 news 表的字段定义如表 28.1 所示：

表 28.1　news表各字段定义

字段	名称	类型	描述
ID（P）	新闻编号	bigint(20)	关键字，自动编号
BoardID（M）	板块编号	int(10)	外部关键字，关联newsBoard新闻板块表
Title	新闻标题	varchar(100)	新闻标题
Editor	新闻编辑者	Editor	新闻编辑者
FirstFrom	原创转载	FirstFrom	原创转载
Author	原创作者	varchar(45)	原创作者
Department	编辑者所在部门	varchar(45)	编辑者所在部门
Keyword	新闻关键字	varchar(100)	新闻关键字
NewsFilePath	新闻文件存放地址	varchar(100)	生成的新闻静态文件所存放的位置
Content	新闻摘要	varchar(1024)	新闻摘要
PublishDate	发布日期	bigint(20)	发布日期
DCount	点击数	int(10)	点击数
IfHot	是否为热点新闻	tinyint(3)	是否热点新闻
IfTop	是否为头新闻	tinyint(3)	是否为头新闻
Type	新闻类型	tinyint(3)	分为文字新闻或者各种图片新闻
PicID（M）	图片编号	int(10)	外部关键字，关联pic表

28.3.2　新闻分类表newsBoard

新闻分类表 newsBoard 用来存储新闻各分类信息，其中包括分类名称以及简短说明信息。生成该表的 SQL 语句如下：

```
DROP TABLE IF EXISTS `webshop`.`newsboard`;
CREATE TABLE  `webshop`.`newsboard` (
  `ID` int(10) unsigned NOT NULL auto_increment,
  `Name` varchar(100) NOT NULL default '',
  `Introduce` varchar(1024) NOT NULL default '',
  PRIMARY KEY  (`ID`)
) ENGINE=InnoDB DEFAULT CHARSET=gbk;
```

该 newsboard 表的字段定义如表 28.2 所示：

表 28.2　newsboard表各字段定义

字段	名称	类型	描述
ID（P）	新闻分类编号	bigint(20)	关键字，自动编号
Name	新闻板块名称	varchar(100)	新闻板块名称
Introduce	简介	varchar(1024)	简介

28.3.3　图片表pic

图片表 pic 用来存放上传图片的信息，其中包括上传日期和图片存放的相对地址等信息。生成改表的 SQL 语句如下：

```
DROP TABLE IF EXISTS `webshop`.`pic`;
CREATE TABLE   `webshop`.`pic` (
  `PicID` int(10) unsigned NOT NULL auto_increment,
  `PicName` varchar(45) NOT NULL default '',
  `Width` smallint(5) unsigned NOT NULL default '100',
  `Height` smallint(5) unsigned NOT NULL default '100',
  `Border` smallint(5) unsigned NOT NULL default '0',
  `Alt` varchar(100) NOT NULL default '选择图片',
  `Align` varchar(45) NOT NULL default 'center',
  `Valign` varchar(45) NOT NULL default 'top',
  `Link` varchar(200) NOT NULL default '',
  `Type` tinyint(3) unsigned NOT NULL default '0',
  `BoardID` int(10) unsigned NOT NULL default '1',
  `PublishDate` bigint(20) unsigned NOT NULL default '0',
  `FileName` varchar(100) NOT NULL default '',
  PRIMARY KEY   (`PicID`)
) ENGINE=InnoDB DEFAULT CHARSET=gbk;
```

该 pic 表的各字段定义如下表 28.3 所示：

表 28.3　pic表各字段定义

字段	名称	类型	描述
PicID（P）	图片编号	int(10)	关键字，自动编号
PicName	图片名称	varchar(45)	图片名称
Width	图片宽度	smallint(5)	图片宽度
Height	图片长度	smallint(5)	图片长度
Border	边框大小	smallint(5)	边框大小
Alt	备注	varchar(100)	备注
Align	水平位置	varchar(45)	水平位置
Valign	上校位置	varchar(45)	上校位置
Link	链接地址	varchar(200)	链接地址
Type	图片使用类型	tinyint(3)	图片使用类型
BoardID（M）	所在板块编号	int(10)	外部关键字，关联board表
PublishDate	发布日期	bigint(20)	发布日期
FileName	图片文件名称	varchar(100)	图片文件名称

28.3.4　用户表userinfo

用户表 userinfo 用来存放用户基本信息，这里很多字段在该系统中是用不着的，但是为了后期的扩

展，这里创建的用户信息还是尽量保持完整。生成该表的 SQL 语句如下：

```
DROP TABLE IF EXISTS `webshop`.`userinfo`;
CREATE TABLE   `webshop`.`userinfo` (
  `ID` varchar(50) NOT NULL default '',
  `Password` varchar(50) NOT NULL default '',
  `Question` varchar(100) default NULL,
  `Answer` varchar(100) default NULL,
  `Name` varchar(50) default NULL,
  `Sex` char(10) NOT NULL default '男',
  `School` varchar(50) default NULL,
  `Address` varchar(200) default NULL,
  `PostAddress` varchar(200) default NULL,
  `Post` char(10) default NULL,
  `Tel` varchar(20) default NULL,
  `Mobile` varchar(20) default NULL,
  `Email` varchar(100) default NULL,
  `QQ` char(15) default NULL,
  `MSN` varchar(100) default NULL,
  `BeiZhu` varchar(1024) default NULL,
  `IfAdmi` tinyint(4) NOT NULL default '0',
  PRIMARY KEY  (`ID`)
) ENGINE=InnoDB DEFAULT CHARSET=gbk;
```

该 userinfo 表的各字段定义如下表 28.4 所示：

表 28.4　userinfo表各字段定义

字段	名称	类型	描述
ID（P）	用户号	varchar(50)	关键字，自动编号
Password	密码	varchar(50)	密码
Question	取回遗忘密码问题	varchar(100)	取回遗忘密码问题
Answer	问题答案	varchar(100)	问题答案
Name	用户姓名	varchar(50)	用户姓名
Sex	用户性别	char(10)	用户性别
School	所在学校	varchar(50)	所在学校
Address	送货地址	varchar(200)	送货地址
PostAddress	邮寄地址	varchar(200)	邮寄地址
Post	邮政编码	char(10)	邮政编码
Tel	固定电话	int(10)	固定电话
Mobile	移动电话	varchar(20)	移动电话
Email	邮件地址	varchar(100)	邮件地址
QQ	QQ号	char(15)	QQ号
MSN	MSN号	varchar(100)	MSN号
BeiZhu	备注	varchar(1024)	备注
IfAdmi	用户类别	tinyint(4)	0表示临时用户，1表示普通用户，2表示管理员

28.4　创建J2EE项目

同样使用 Eclipse＋Lomboz 工具来构架该实例项目，详细的 Lomboz 工具使用请查看第九章内容。

28.4.1　项目开发环境

该实例使用的开发环境、工具以及数据库如下：

- 操作系统：Windows XP Professional
- JDK1.4.2_04 版本
- MySQL5.0.18 版本
- Web 容器 Tomcat5.0.30 版本
- Eclipse3.1.0 版本，以及相应版本的 Lomboz 插件
- CVS 版本 2.5.03.2151，具体安装和配置介绍见本章的后面小节。

28.4.2　项目创建

这里创建的新项目名为 MyWebProject，Web 模块名为 News，使用 j2src 目录作为源文件目录。生成的目录结构如图 28.3 所示。其中使用 CVS 作为该项目的版本控制工具。

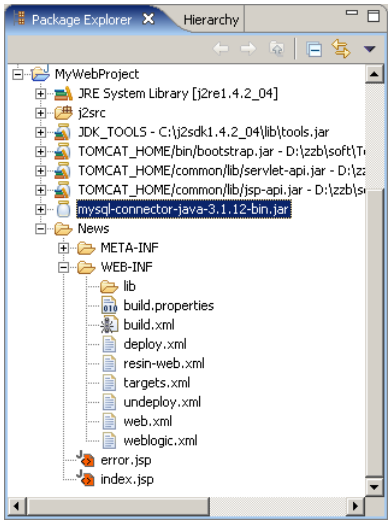

图 28.3　MyWebProject 项目的目录结构

其中需要将数据库的 JDBC 连接包 mysql-connector-java-3.1.12-bin.jar 添加到 Eclipse 库中。

28.5　创建工厂类Factory和SqlFactory

首先在 MyWebProject 项目中创建工厂类 Factory 和 SqlFactory，它们来对其他所有类进行统一初始化和管理。很形象地把它们称为工厂类，因为 Factory 和 SqlFactory 类使用特定方法得到某个类的实例对象（进行了初始化，就像工厂进行商品加工一样）。

28.5.1　Factory抽象类的创建

Factory 为抽象类，详细源代码如下：

```
package cn.com.zzb.news;
public abstract class Factory {
```

```
    private static Factory factory = null;
    public static Factory getInstance(){                    //初始化自身一个实例对象
        if(factory == null)
            try{
                                                            //使用 Class.forName（）方法进行类初始化
                Class facotryClass = Class.forName("com.stuhouse.qqNews.SqlFactory");
                factory = (Factory)facotryClass.newInstance();
            }
            catch(Exception exception) {
                exception.printStackTrace();
            }
        return factory;
    }
    //调用下面的方法对各个类进行初始化
    public abstract News getNews();                         //得到 News 类的实例对象
    public abstract NewsFactory getNewsFactory();           //得到 NewsFactory 类的实例对象
    public abstract Pic getPic();                           //得到 Pic 类的实例对象
    public abstract PicFactory getPicFactory();             //得到 PicFactory 类的实例对象
    public abstract NewsBoard getNewsBoard();               //得到 NewsBoard 类的实例对象
    public abstract NewsBoardFactory getNewsBoardFactory(); //得到 NewsBoardFactory 类的实例对象
}
```

程序说明：该工厂类对新闻发布系统中所使用到的所有类进行统一的初始化管理，这里需要初始化的各 JavaBean 类将在下面两个章节介绍的模块中提及。另外，读者也可以根据需要，在该工厂类中动态地添加需要初始 JavaBean 类的方法。

28.5.2　SqlFactory类的创建

Factory 抽象类的继承类 SqlFactory 如下所示：

```
package cn.com.zzb.news;
public class SqlFactory extends Factory {
    private NewsFactory newsfactory = null;
    private PicFactory picfactory = null;
    private NewsBoardFactory newsboardfactory = null;
    public News getNews(){                                  //实现初始化 News 类的方法
        return new SqlNews();
    }
    public NewsFactory getNewsFactory(){                    //实现初始化 NewsFactory 类的方法
        if(newsfactory == null)
            newsfactory = new SqlNewsFactory();
        return newsfactory;
    }
    public Pic getPic(){                                    //实现初始化 Pic 类的方法
        return new SqlPic();
    }
    public PicFactory getPicFactory(){                      //实现初始化 PicFactory 类的方法
        if(picfactory == null)
            picfactory = new SqlPicFactory();
        return picfactory;
    }
```

```
    public NewsBoard getNewsBoard(){                          //实现初始化 NewsBoard 类的方法
          return new SqlNewsBoard();
    }
    public NewsBoardFactory getNewsBoardFactory(){            //实现初始化 NewsBoardFactory 类的方法
        if(newsboardfactory == null)
            newsboardfactory = new SqlNewsBoardFactory();
        return newsboardfactory;
    }
}
```

程序说明：该类实现了初始化各类的方法，Factory 类和 SqlFactory 类将在各个类初始化的时候使用。这里创建的两个工厂类将在下面章节创建的 JavaBean 类中使用，用来进行各类的初始化工作。

28.6　本章小结

本章作为新闻发布系统的需求分析、功能和数据库设计部分。最后，讲解了在项目中创建 Factory 和 SqlFactory 工厂类，这这两个工厂类来对其他所有类进行通过初始化管理。下面两个章节将重点介绍新闻发布系统如何实现。