**武汉大学计算机学院**

**本科生课程设计报告**

**影院售票排片服务系统**

专 业 名 称 ：计算机科学与技术

指 导 教 师 ：王健（副教授）

课 程 名 称 ：软件设计与体系结构

学 生 学 号 ：2018300003063；

2018302050164；

2018302100039；

学 生 姓 名 ：曾文瑄；李嘉俊；周夏龙

二○二○年十月

1. **简单工厂模式**
2. 实现代码
3. Form1：主界面

using System;

using System.CodeDom.Compiler;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Runtime.Remoting.Messaging;

using System.Security.Cryptography.X509Certificates;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace ticketsysyem

{

public partial class Form1 : Form

{

public abstract class MOVIE

{

public abstract void query();

public string name;

public string data;

public double score;

public string country;

public string type;

public string intro;

}

public Form1()

{

InitializeComponent();

}

public class Resident\_Evil\_6 : MOVIE /\* 1 \*/

{

public Resident\_Evil\_6( )

{

name = "生化危机：终章";

data = "2017-1-27";

score = 6.5;

country = "美国";

type = "动作，冒险，恐怖";

intro = "爱丽丝（米拉·乔沃维奇 Milla Jovovich 饰）在华盛顿特区被威斯克背叛后身陷险境，" +

"人类几乎要失去最后的希望。作为唯一的幸存者，也是人类对抗僵尸大军的最后防线，爱丽丝必须回到噩梦开始的地方——浣熊市，" +

"才能完成拯救世界救赎人类的正义使命。回归故事发生的起点浣熊市，爱丽丝将和昔日的朋友一起对抗僵尸和最新变种怪物，" +

"与保护伞公司展开了一场惊心动魄的终极决战。";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class Let\_The\_Bullets\_Fly : MOVIE /\* 2 \*/

{

public Let\_The\_Bullets\_Fly()

{

name = "让子弹飞";

data = "2010-12-16";

score = 8.8;

country = "中国";

type = "喜剧";

intro = "民国年间，花钱捐得县长的马邦德（葛优 饰）携妻（刘嘉玲 饰）及随从走马上任。途经南国某地，遭劫匪张麻子（姜文 饰）一伙伏击，随从尽死，只夫妻二人侥幸活命。马为保命，谎称自己是县长的汤 师爷。为汤师爷许下的财富所动，张麻子摇身一变化身县长，带着手下赶赴鹅城上任。有道是天高皇帝远，鹅城地处偏僻，一方霸主黄四郎（周润发 饰）只手遮天，全然不将这个新来的县长放在眼里。张麻子痛打了黄的武教头（姜武 饰），黄则设计害死张的义子小六（张默 饰）。原本只想赚钱的马邦德，怎么也想不到竟会被卷入这场土匪和恶霸的角力之中。鹅城上空愁云密布，血雨腥风在所难免……";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class The\_Great\_Gatsby : MOVIE /\* 3 \*/

{

public The\_Great\_Gatsby()

{

name = "了不起的盖茨比";

data = "2013-08-30";

score = 7.9;

country = "美国";

type = "剧情，爱情";

intro = "1922年的春天，作家尼克(托比•马奎尔 Tobey Maguire 饰)满怀希望离开家乡，随淘金热潮来到纽约这个新兴的城市，虽然这里爵士乐流行，股票飞涨，但是贫富两极分化，人们沉沦在纸醉金迷中。尼克为了追寻美国梦，放弃写作而进入证券市场，并搬入纽约附近的海湾居住，成为了神秘富豪盖茨比（莱昂纳多•迪卡普里奥 Leonardo DiCaprio 饰）的邻居。而海湾的对岸住着尼克的表妹黛西（凯瑞•穆里根 Carey Mulligan 饰）和她的贵族丈夫汤姆（乔尔•埃哲顿 Joel Edgerton 饰），尼克不仅被邀请去赴宴，之后汤姆还带着他去找情妇寻欢，尼克渐渐迷失在这个充满魅力，以及富豪编制的假象、爱与谎言的世界中。但是只有盖茨比最特别，他是唯一让尼克感到在这个迷失城市里充满希望的人。尼克被邀请参加了盖茨比豪宅中的盛宴，还发现盖茨比心中的秘密，原来他一直深爱着黛西，即使她没等到他战后归来，转而嫁给富豪汤姆，盖茨比一直深信他们的爱情矢志不渝。尼克作为盖茨比与黛西爱情的见证者，终于在盖茨比被谋杀之后，看清了这个上流社会的虚情寡义，决心远离喧嚣、冷漠、虚假的大都市，黯然回到故乡";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class Amilie : MOVIE /\* 4 \*/

{

public Amilie()

{

name = "天使爱美丽";

data = "2001-04-25";

score = 8.7;

country = "法国";

type = "戏剧，爱情";

intro = "艾米莉（奥黛丽·塔图 Audrey Tautou 饰）有着别人看来不幸的童年——父亲给她做健康检查时，发现她心跳过快，便断定她患上心脏病，从此艾米莉与学校绝缘。随后因为一桩意外，母亲在她眼前突然死去。这一切都毫不影响艾米莉对生活的豁达乐观。1997年，戴安娜王妃的去世让她倍感人生的孤独脆弱，艾米莉从此开始了一系列助人计划，包括自闭忧郁的邻居老人，被老板刻薄的菜摊伙计、遗失了童年器物的旧房东、爱情失意的咖啡店同事。但她万万想不到，成人录象带商店店员尼诺（马修·卡索维 Mathieu Kassovitz 饰），竟成为她的棘手对象，艾米莉开始了令人哭笑不得的另类计划";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class Green\_Book : MOVIE /\* 5 \*/

{

public Green\_Book()

{

name = "绿皮书";

data = "2019-03-01";

score = 8.9;

country = "美国";

type = "剧情，喜剧，传记";

intro = "托尼（维果·莫腾森 Viggo Mortensen 饰）是一个吊儿郎当游手好闲的混混，在一家夜总会做侍者。这间夜总会因故要停业几个月，可托尼所要支付的房租和生活费不会因此取消，所以他的当务之急是去寻找另一份工作来填补这几个月的空缺。在这个节骨眼上，一位名叫唐雪莉（马赫沙拉·阿里 Mahershala Ali 饰）的黑人钢琴家提出雇佣托尼。唐雪莉即将开始为期八个星期的南下巡回演出，可是，那个时候南方对黑人的歧视非常的严重，于是托尼便成为了唐雪莉的司机兼保镖。一路上，两人迥异的性格使得他们之间产生了很多的矛盾，与此同时，唐雪莉在南方所遭受的种种不公平的对待也让托尼对种族歧视感到深恶痛绝";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class The\_Bourne\_Identity\_5 : MOVIE /\* 6 \*/

{

public The\_Bourne\_Identity\_5()

{

name = "谍影重重5";

data = "2016-08-23";

score = 7.4;

country = "美国";

type = "动作，悬疑，惊悚";

intro = "后斯诺登时代，中情局系统被黑客入侵， 多项特工计划可能泄露。与此同时，中情局还发现了伯恩和好搭档尼基·帕森斯的踪迹。年轻的网络专家海瑟·李自告奋勇追踪伯恩，而伯恩也在寻找着关于自己身世的惊天黑幕。中情局高官杜威是知晓一切幕后秘密的人，他、海瑟·李和伯恩之间的角力让故事充满悬念，不到最后一刻，就无法了解真正的结局。";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class Olympus\_Has\_Fallen : MOVIE /\* 7 \*/

{

public Olympus\_Has\_Fallen()

{

name = "奥林匹斯的陷落";

data = "2013-12-29";

score = 6.4;

country = "美国";

type = "动作，犯罪";

intro = "时政府迅速派遣美国突击队登陆白宫，企图营救总统，却被恐怖分子控制的白宫防御系统击退，营救计划几度失败。眼看恐怖分子的阴谋即将得逞。因为意外，班宁活了下来，他与临时政府取得联系，并孤身一人营救总统，班宁成为了整个美国的最后一丝希望。这位前特工依靠自己过硬的身手，在危机密布的白宫只身力挽狂澜，率先拯救了总统的儿子，让总统摆脱后顾之忧，恐怖分子获得地狱犬密码的阴谋被延迟。经过艰苦的鏖战和激烈的厮杀，班宁拯救了被劫持的总统，并粉碎了恐怖分子企图利用核弹毁灭美国的阴谋";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class Bohemian\_Rhapsody : MOVIE /\* 8 \*/

{

public Bohemian\_Rhapsody()

{

name = "波西米亚狂想曲";

data = "2019-03-22";

score = 8.7;

country = "英国";

type = "剧情，音乐，传记";

intro = "本片是对皇后乐队、传奇主唱弗雷迪·莫库里以及他们音乐的致敬盛宴，这是一段充满爱、痛苦、接纳和音乐的旅程。弗雷迪·莫库里（拉米·马雷克 饰）曾是希思罗机场的一名普通搬运工，对音乐满腔热血的他，与布莱恩·梅（格威利姆·李 饰）、罗杰·泰勒（本·哈迪 饰）、约 翰·迪肯（约瑟夫·梅泽罗 饰）组成皇后乐队。这个殿堂级乐队的从无到有，从疏离到重聚，从低谷到巅峰，仿佛就是弗雷迪一生的缩影。在最后的日子里，弗雷迪做了生命中最重要的决定——在音乐史上最伟大的“拯救生命”大型摇滚乐演唱会上，将所有热血付诸于歌声";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class The\_Mechanic\_2 : MOVIE /\* 9 \*/

{

public The\_Mechanic\_2()

{

name = "机械师2：复活";

data = "2016-10-21";

score = 5.7;

country = "美国";

type = "动作，犯罪";

intro = "系列电影《机械师》第一部中，假装死亡的顶级杀手亚瑟（杰森·斯坦森饰）金盆洗手，过着普通人平静的生活。在一桩看似意外的事件中，亚瑟的红颜知己梅（杨紫琼饰）请求亚瑟解救屡屡遭人毒打的神秘女子吉娜（杰西卡·阿尔芭饰），哪料，向来对爱情有免疫的亚瑟，竟与吉娜坠入了爱河，然而他平静的生活也因此被打破。五年远离是非的生活风平浪静，亚瑟以为自己已经和过去的杀手生涯告别，然而宿命难敌，竟然有人暗中盯上了他，亚瑟的身世之谜也浮出水面。原来，他年幼时和一群孩子被军火商抓去，训练他们玩枪、杀人，难以忍受的亚瑟从军火商手中逃脱，而他的一位小伙伴长大后却成了军火商的接班人，也是影片中的最大反派。这位反派找到了亚瑟，并绑架了他的挚爱吉娜，要挟他去刺杀三个世界上头号危险人物。为了他和他的爱人能够逃脱，亚瑟被迫再度从事刺客工作，他必须在有限的时间内完成一系列不可能的刺杀任务。";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class Madagascar\_3 : MOVIE /\* 10 \*/

{

public Madagascar\_3()

{

name = "马达加斯加3：欧洲通缉犯";

data = "2012-06-08";

score = 8.2;

country = "美国";

type = "动画，喜剧";

intro = "在非洲游历了一圈之后，纽约中央公园的四位明星们继续着回到纽约，回到中央公园的旅程。不过，只要旅途中有猩猩和那群企鹅的“帮忙”，旅程就没有想象中的那么顺利。在蒙特卡洛的一家赌场里，由于大家的耍宝，大闹赌场，进而招致了心狠手辣的动物管理局上尉杜波依斯（弗兰西斯·麦克多蒙德 Frances McDormand 配音）的围追堵截。为了逃过追捕，大家逃进了一个在欧洲进行巡回演出的马戏团。在马戏团里，大家延续着“马达加斯加”式的幽默，笑料百出。虽然，欧洲这一路上的旅行并不顺利，不断受到杜波依斯上尉的追捕威胁，不过大家一直怀揣着回到纽约，回到纽约中央公园的梦想。在马戏团，还帮助了东北虎维塔利（布莱恩·科兰斯顿 Bryan Cranston 配音）、美洲虎吉亚（杰西卡·查斯坦 Jessica Chastain 配音）以及海狮斯蒂芬诺（马丁·肖特 Martin Short 配音）重拾对马戏表演的信心。";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class Raiders\_of\_the\_Lost\_Ark : MOVIE /\* 11 \*/

{

public Raiders\_of\_the\_Lost\_Ark()

{

name = "夺宝奇兵";

data = "1981-06-12";

score = 7.9;

country = "美国";

type = "动作，冒险";

intro = "二战期间，希特勒在世界各地召集考古学家寻找“失落的约柜”——圣经中引导希伯来人与上帝交流的圣物，希特勒欲借其来护佑纳粹的战争。为了使希特勒的计划破灭，印第安纳琼斯博士（Harrison Ford饰）奔赴尼泊尔，一边奋力挖掘蛛丝马迹，一边还要与无孔不入的纳粹分子周旋。凭借着过人的智慧和异于常人的胆量及大无畏的勇气，琼斯终于在罗马找到了指示约柜位置的太阳手杖，继而发现了约柜。就在大功告成之际，纳粹军却侵吞了胜利果实，还将琼斯等人留在了蛇穴里！但是无往不胜的琼斯总是有办法死里逃生，并狠狠地还以颜色！";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class World\_War\_Z : MOVIE /\* 12 \*/

{

public World\_War\_Z()

{

name = "僵尸世界大战";

data = "2013-06-02";

score = 7.5;

country = "美国";

type = "动作，科幻，恐怖";

intro = "费城一个毫无征兆的宁静早晨，前联合国调查员盖瑞·雷恩（布拉德·皮特 Brad Pitt 饰）驾车载着妻子凯伦（米瑞·伊诺丝 Mireille Enos 饰）和两个女儿蕾切尔（Abigail Hargrove 饰）、康妮（Sterling Jerins 饰）外出。谁知原本快乐的气氛很快变为恐慌与死亡所替代，连环的爆炸，惊恐逃散的人群，无序的车祸以及啃咬他人的丧尸，繁华大都会转瞬沦陷。盖瑞费尽九牛二虎之力带着妻女逃生，最终逃到联合国副秘书长所在的航母上，进而得知包括美国总统在内的数名大国首脑均已死亡，丧尸病毒业已蔓延全球。为了阻止世界毁灭，格里无奈受命，和来自哈佛的病毒学博士法斯巴克（Elyes Gabel 饰）前往病毒的源头之地寻找解除危机的办法。";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class Blade\_Out : MOVIE /\* 13 \*/

{

public Blade\_Out()

{

name = "利刃出鞘";

data = "2019-11-29";

score = 8.2;

country = "美国";

type = "剧情，喜剧，悬疑，犯罪";

intro = "富豪小说家哈兰·斯隆比在自己85岁生日第二天，被发现在自家庄园离奇自杀，遗留了亿万遗产。久负盛名的大侦探布兰科（丹尼尔·克雷格饰）突然被匿名人士雇佣调查此案真相。同时，哈兰的孙子兰森（克里斯·埃文斯饰）也正在秘密调查此案。当布兰科和哈兰·斯隆比家族的其他人对谈时， 他发现事情并没有想象中那么简单。哈兰家族没有表面上那么和睦，每个人都有着不可告人的秘密，每个人都想得到遗产……究竟这起命案是自杀还是他杀？似乎每个人都有嫌疑。随着一位遗产继承人的意外亮相，真相谜底渐渐浮出水面……";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class Alita\_Battle\_Angel : MOVIE /\* 14 \*/

{

public Alita\_Battle\_Angel()

{

name = "阿丽塔：战斗天使";

data = "2019-02-22";

score = 7.5;

country = "美国";

type = "动作，科幻，冒险";

intro = "故事发生在遥远的26世纪，外科医生依德（克里斯托弗·瓦尔兹 Christoph Waltz 饰）在垃圾场里捡到了只剩下头部的机械少女将她带回家中，给她装上了本来为自己已故的女儿所准备的义体，并取名阿丽塔（罗莎·萨拉扎尔 Rosa Salazar 饰）。苏醒后的阿丽塔对这个五彩斑斓但却暴力而又残酷的世界产生了浓厚的兴趣，在结识了青年雨果（基恩·约翰逊 Keean Johnson 饰）后，阿丽塔开始接触名为机动球的运动，并在比赛中展现出了惊人的格斗天赋。在废铁城居民们的头顶，漂浮着巨大的浮空城市撒冷，废铁城居民们的一切劳作和付出，都是为了给撒冷提供继续运作的燃料。在大财阀维克特（马赫沙拉·阿里 Mahershala Ali 饰）所设立的机动球比赛中，最终获得冠军的人能够获得前往撒冷生活的资格，阿丽塔决定利用自己的格斗天赋参加比赛，却被卷入了一个巨大的阴谋之中。";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class The\_Wolf\_of\_Wall\_Street : MOVIE /\* 15 \*/

{

public The\_Wolf\_of\_Wall\_Street()

{

name = "华尔街之狼";

data = "2013-12-17";

score = 7.9;

country = "美国";

type = "剧情，喜剧，传记，犯罪";

intro = "在危机四伏的投资界，掌管史崔顿·奥克蒙公司的乔丹·贝尔福特（莱昂纳多·迪卡普里奥 Leonardo DiCaprio 饰）生活侈靡、实力雄厚。1987年，22岁的乔丹进入罗斯柴尔德公司，从接线员做起，在高级经纪人马克（马修·麦康纳 Matthew McConaughey 饰）指点下，进入了股票经纪人充满毒品和欲望的世界，半年后因“黑色星期一”，乔丹转投靠贩卖“便士股票”的小公司，凭借巧舌如簧获得成功，与邻居唐尼（乔纳·希尔 Jonah Hill 饰）和一班朋友另立门户，游走在灰色地带获利颇丰，获称“华尔街之狼”，然而，FBI已经开始留意他……";

}

public override void query()

{

Console.WriteLine("name: " + name);

Console.WriteLine("data: " + data);

Console.WriteLine("score: {0}", score);

Console.WriteLine("country: " + country);

Console.WriteLine("type: " + type);

Console.WriteLine("intro: " + intro);

}

}

public class Factory

{

public static MOVIE QueryMovie(string args)

{

MOVIE TEMP;

TEMP = null;

switch (args)

{

case "生化危机":

TEMP = new Resident\_Evil\_6();

break;

case "让子弹飞":

TEMP = new Let\_The\_Bullets\_Fly();

break;

case "了不起的盖茨比":

TEMP = new The\_Great\_Gatsby();

break;

case "天使爱美丽":

TEMP = new Amilie();

break;

case "绿皮书":

TEMP = new Green\_Book();

break;

case "谍影重重":

TEMP = new The\_Bourne\_Identity\_5();

break;

case "奥林匹斯陷落":

TEMP = new Olympus\_Has\_Fallen();

break;

case "波西米亚狂想曲":

TEMP = new Bohemian\_Rhapsody();

break;

case "机械师":

TEMP = new The\_Mechanic\_2();

break;

case "马达加斯加":

TEMP = new Madagascar\_3();

break;

case "夺宝奇兵":

TEMP = new Raiders\_of\_the\_Lost\_Ark();

break;

case "僵尸世界大战":

TEMP = new World\_War\_Z();

break;

case "利刃出鞘":

TEMP = new Blade\_Out();

break;

case "阿丽塔：战斗天使":

TEMP = new Alita\_Battle\_Angel();

break;

case "华尔街之狼":

TEMP = new The\_Wolf\_of\_Wall\_Street();

break;/\*为了少写一个选择界面，工厂类在展示时是这样的，实际程序中并不会\*/

}

return TEMP;

}

}

private void Form1\_Load(object sender, EventArgs e)

{

}/\*标题\*/

private void label1\_Click(object sender, EventArgs e)

{

}/\*欢迎标语\*/

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}/\*查询输入\*/

private void button1\_Click\_1(object sender, EventArgs e)

{

string args = textBox1.Text;

if(Factory.QueryMovie(args) == null)

{

MessageBox.Show("无相应查询结果","Alert",MessageBoxButtons.OK,MessageBoxIcon.Warning);

return;

}

else

{

Form5 mainForm = new Form5(Factory.QueryMovie(args));

mainForm.Show();

}

}

private void button2\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}

(2)Form5：查询结果界面  
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace ticketsysyem

{

public partial class Form5 : Form

{

public Form5(ticketsysyem.Form1.MOVIE x)

{

InitializeComponent();

label1.Text = x.name;

label2.Text = "发行日期： " + x.data;

label3.Text = "网络评分： " + x.score;

label4.Text = "拍摄国家： " + x.country;

label5.Text = "影片类型： " + x.type;

label6.Text = "电影简介： " + x.intro;

}

private void Form5\_Load(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

this.Close();

}

private void label2\_Click(object sender, EventArgs e)

{

}

private void label4\_Click(object sender, EventArgs e)

{

}

private void label3\_Click(object sender, EventArgs e)

{

}

private void button2\_Click(object sender, EventArgs e)

{

Form6 mainForm = new Form6();

mainForm.Show();

}

}

}

(3)Form6：购票登陆界面及订票结果  
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace ticketsysyem

{

public partial class Form6 : Form

{

public Form6()

{

InitializeComponent();

}

private void label2\_Click(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

if(textBox1.Text=="naviman44" && textBox2.Text=="hashi006")

{

MessageBox.Show("your cdkey: T6S552DAZSR35S4865SAWDSW0", "subscribe comfirmed", MessageBoxButtons.OK);

}

else

{

MessageBox.Show("用户名或密码无效", "subscribe failed", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

}

1. 设计思路分析

这一部分实现了影院排片订票系统中面向用户的功能部分，即查询当前正在上映的电影，在看到电影的详细信息后可以选择订票，验证信息无误后可以订票，系统会返回给用户一个唯一的序列识别码用于验证购买情况。

由于系统存储影片的相关属性时方法基本一致，因而将电影本身设计成一个抽象类，这个抽象类MOVIE包含了影片的基本属性（名字name，发行日期data，网络评分score，发行国家country，影片类型type，影片简介intro），和一个能够修改相关属性的函数alter（）。当需要添加新的电影到数据库中是，电影作为一个继承类，继承基类MOVIE的相关属性，并且根据自身的情况实现函数alter（）。这里的工厂Factory（）是用来在查询时将相关影片实例化从而返回相关数据的。例如对于电影“马达加斯加“，会有一个继承了MOVIE的类Madagascar发，当用户查询时，通过工厂将Madagascar实例化从而能够在后续的界面访问其内部元素。

修改操作函数本身需要一个验证码string[] access\_code ，由于没有使用服务器，暂时将这个验证码写死在底层中，只有当给函数正确的验证码时，函数才会执行后面部分，执行时打开新的窗口，接收用户输入的数据并对数据库中的内容进行相应的更改。

接下来说订票系统，用户在看到电影的详细信息后，可以在对应的界面点击“订票“按钮来预定当前影片的演出票。订票系统对应的是另一个数据库（未在上面展示），该数据库中有若干二维数组，对应标签site1-n，其中存有另一个列表，对应site1-n和电影的对应关系，这主要是便于修改。用户点击订票时，如果当前电影无剩余作为，则返回”当前电影无余量“，而如果不满，则将作为情况以一个二维图标阵列的形式显示给用户，可用座位显示为蓝色，不可用的则显示为红色，用户可以选择其中的一个座位（直接点击）并进入下一步，也可以取消返回。

当用户选择继续且所选座位为可用时，进入登陆验证界面，这一步需要用户输入用户名和密码进行验证，当验证成功时，返回“订票成功“，并给出唯一的验证序列号，序列号包含了用户的编号，场次，电影编号和座位号，用户即可凭此号入场。

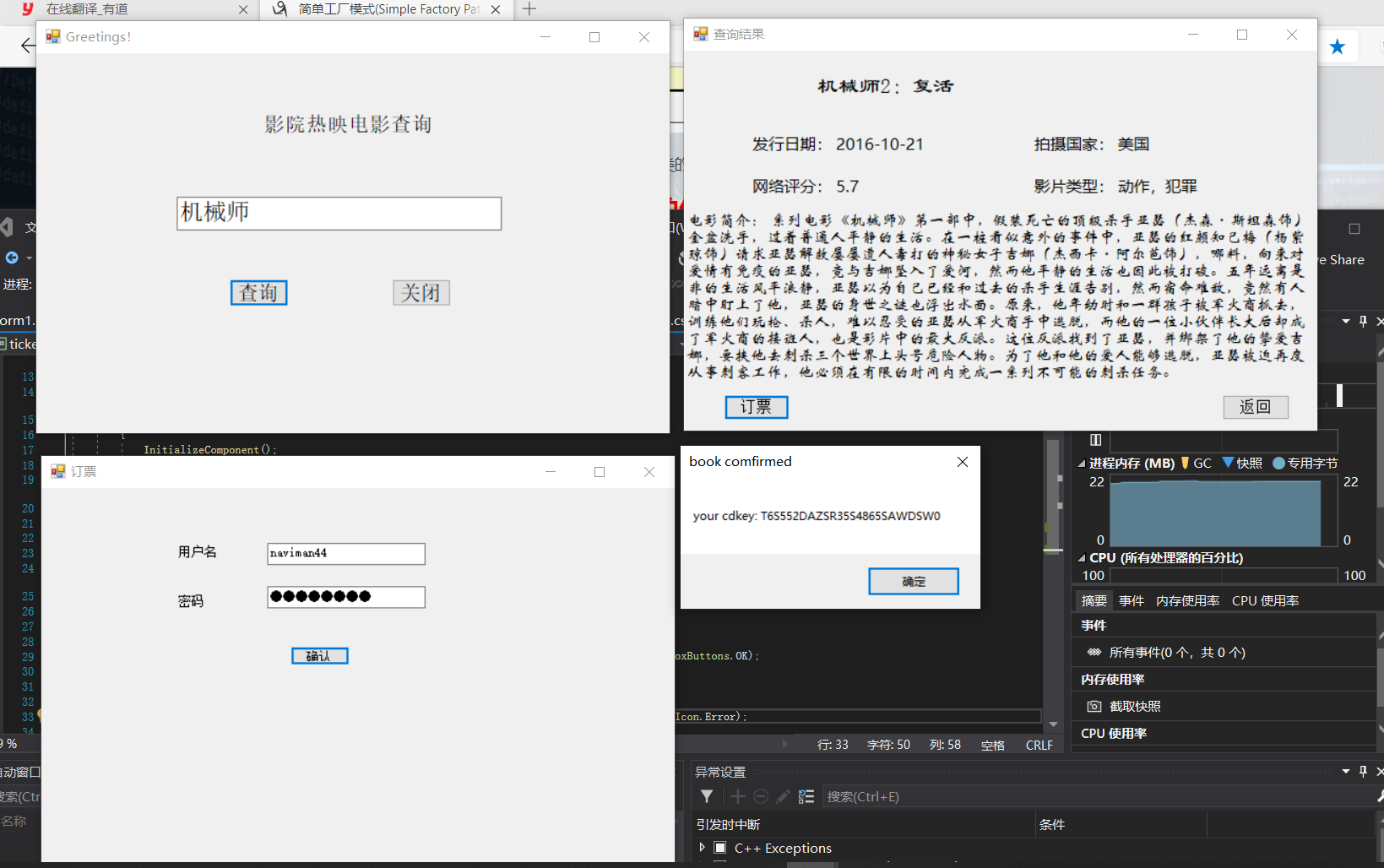
1. 简单工厂模式的优点

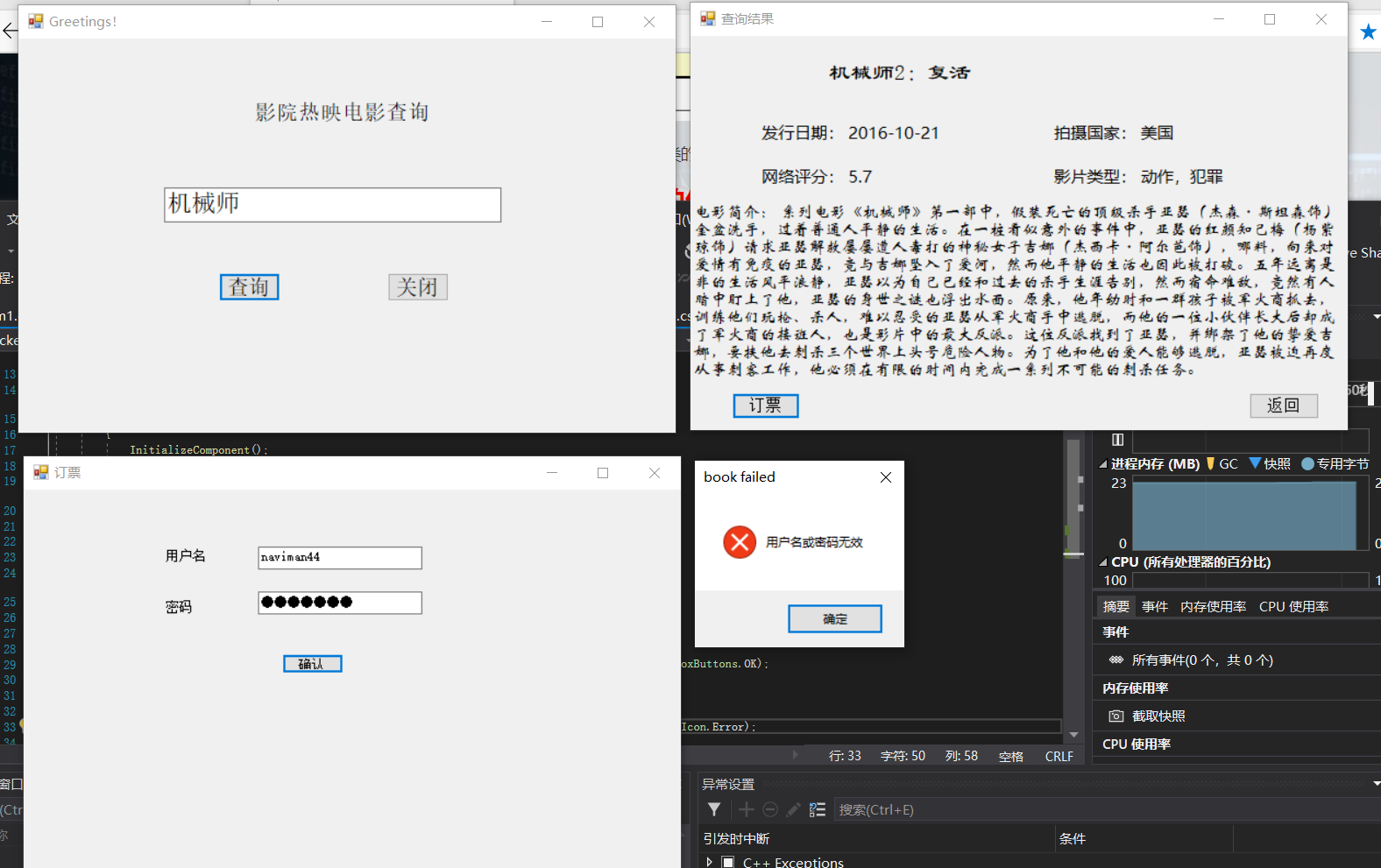
首先是方便用户的使用。用户只需要知道他想要查询的电影的名称既可以获得所有的信息，不需要提供更多的数据。

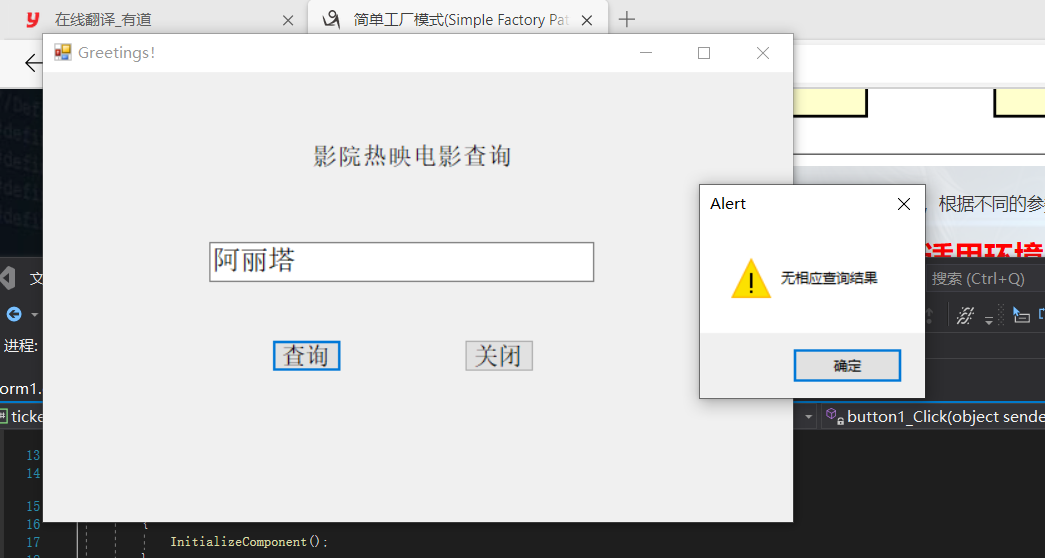
其次是方便维护和修改。当我们需要添加，修改或者下架相关影片时，不需要对底层进行增删。（当前的产品类设计并不合理，主要是方便展示功能，在实际创建产品类时，应该按照电影的分类去创建产品类，这些类继承MOVIE，如果像上述代码中将各个影片单独建类，那么增加和删除电影将十分麻烦，有时候只能对影片的信息进行覆写。）这里的工厂实际上是提供了一个用户（顾客或者维护人员）和数据库交互的接口。

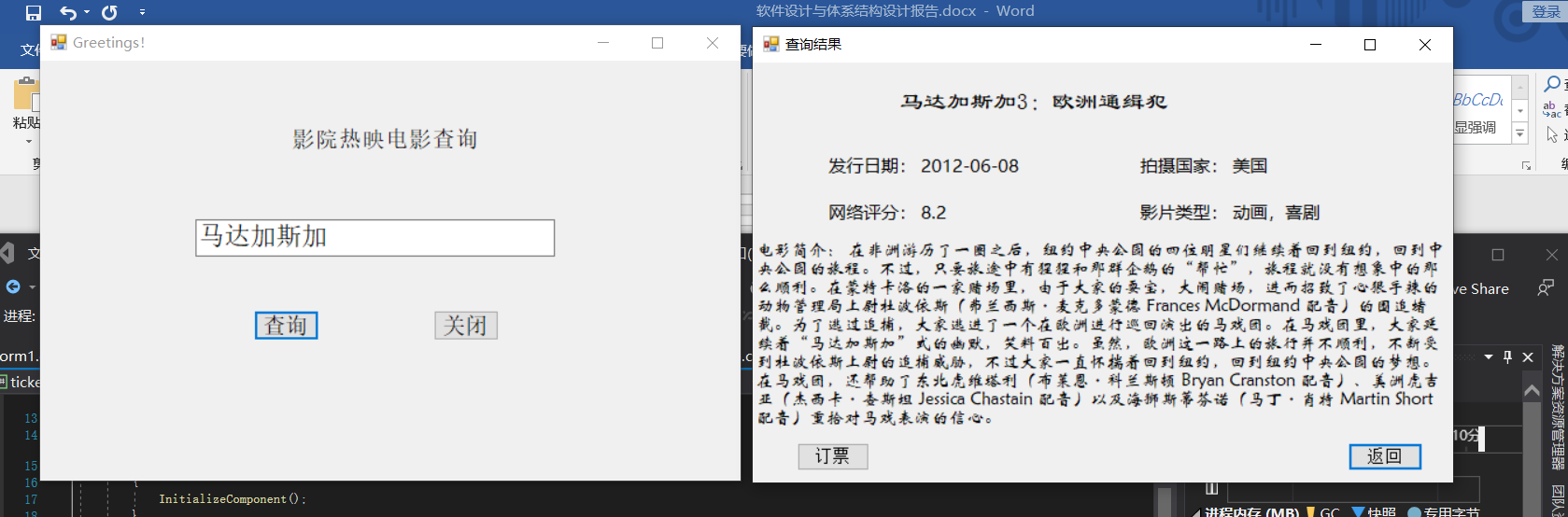
但是当数据量变大之后，工厂进行检索和创建的速度会逐渐变慢，当前还没有向工厂类中加入提升检索速度的算法（主要是目前数据库中数据很少）。

1. 展示效果图（非完全体）









1. 观察者**模式**
2. 实现代码

（1）FrmMain：主界面

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace TicketSystem

{

public interface IObserver

{

void Update(ISubject subject);

}

public interface ISubject

{

void 切换登录ToolStripMenuItem\_Click\_1(IObserver observer);

void 用户管理ToolStripMenuItem\_Click(IObserver observer);

void 营业情况ToolStripMenuItem\_Click(IObserver observer);

void 退出ToolStripMenuItem\_Click(IObserver observer);

void Notify();

}

public class Subject : ISubject

{

private List<IObserver> \_observers = new List<IObserver>();

private void 用户管理ToolStripMenuItem\_Click(object sender, EventArgs e)

{

FrmUserInfo userInfo = new FrmUserInfo();

userInfo.Show();

}

private void 营业情况ToolStripMenuItem\_Click(object sender, EventArgs e)

{

FrmSelling sell = new FrmSelling();

sell.Show();

}

private void 切换登录ToolStripMenuItem\_Click\_1(object sender, EventArgs e)

{

this.Close();

}

private void 退出ToolStripMenuItem\_Click(object sender, EventArgs e)

{

Application.Exit();

}

public void Notify()

{

foreach (var observer in \_observers)

{

observer.Update(this);

}

}

}

class ConcreteObserverA : IObserver

{

private void 影片管理ToolStripMenuItem\_Click(object sender, EventArgs e)

{

FrmMovieAdmin admin = new FrmMovieAdmin();

admin.Show();

}

}

}（2）FrmUserInfo：增删改查功能：

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Runtime.Serialization.Formatters.Binary;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace TicketSystem

{

public partial class FrmUserInfo : Form

{

public FrmUserInfo()

{

InitializeComponent();

}

//加载

private void FrmUserInfo\_Load(object sender, EventArgs e)

{

RoleManage.InitialRole();

BindData();

}

/// <summary>

/// 临时集合

/// </summary>

Dictionary<string, Role> dirRole = new Dictionary<string, Role>();

/// <summary>

/// 绑定数据

/// </summary>

private void BindData()

{

dataGridView1.AutoGenerateColumns = false;

dataGridView1.DataSource = new BindingList<Role>(RoleManage.dirRole.Values.ToList());

}

#region 增删改查

//查询

private void button4\_Click(object sender, EventArgs e)

{

dirRole.Clear();

if (textBox4.Text.Trim().Length < 1)

{

BindData();

}

string name = textBox4.Text.Trim();

foreach (Role item in RoleManage.dirRole.Values)

{

if (item.Name.IndexOf(name) != -1)

{

dirRole.Add(item.Name,item);

}

}

dataGridView1.AutoGenerateColumns = false;

dataGridView1.DataSource = new BindingList<Role>(dirRole.Values.ToList());

}

//增加

private void button1\_Click(object sender, EventArgs e)

{

panel1.Visible = true;

txtName.Text = "";

txtPwd.Text = "";

button8.Text = "添加";

comboBox2.SelectedIndex = 0;

}

//修改

string roleName = null;

private void button2\_Click(object sender, EventArgs e)

{

panel1.Visible = true;

button8.Text = "修改";

roleName = dataGridView1.CurrentRow.Cells[0].Value.ToString();

txtName.Text = RoleManage.dirRole[roleName].Name;

txtPwd.Text = RoleManage.dirRole[roleName].Pwd;

}

//删除

private void button3\_Click(object sender, EventArgs e)

{

roleName = dataGridView1.CurrentRow.Cells[0].Value.ToString();

DialogResult r = MessageBox.Show("确定永久删除用户名为《"+roleName+"》的账号吗？","提示：",

MessageBoxButtons.OKCancel,MessageBoxIcon.Information);

if (r != DialogResult.OK)

return;

RoleManage.dirRole.Remove(roleName);

BindData();

}

#endregion

#region 确定

//确定

private void button8\_Click(object sender, EventArgs e)

{

if (txtName.Text.Trim().Length < 1 || txtPwd.Text.Trim().Length < 1)

{

MessageBox.Show("请填写完整信息！", "提示：");

return;

}

string name = txtName.Text.Trim();

string pwd = txtPwd.Text.Trim();

if (button8.Text == "修改")

{

RoleManage.dirRole[roleName].Name = name;

RoleManage.dirRole[roleName].Pwd = pwd;

MessageBox.Show("修改成功！", "提示：");

panel1.Visible = false;

return;

}

Role role;

if (comboBox2.SelectedIndex == 0)

{

role = new User(name, pwd);

}

else

{

role = new Admin(name, pwd);

}

try

{

RoleManage.dirRole.Add(role.Name, role);

MessageBox.Show("添加成功！","提示：");

panel1.Visible = false;

}

catch

{

MessageBox.Show("该用户名已存在！","提示：");

}

BindData();

}

#endregion

#region 取消

//取消

private void button5\_Click(object sender, EventArgs e)

{

panel1.Visible = false;

}

//设置密码栏显示

private void dataGridView1\_CellFormatting(object sender, DataGridViewCellFormattingEventArgs e)

{

// 把第2列显示\*号，\*号的个数和实际数据的长度相同

if (e.ColumnIndex == 1)

{

if (e.Value != null && e.Value.ToString().Length > 0)

{

e.Value = new string('\*', e.Value.ToString().Length);

}

}

}

#endregion

//窗体关闭

private void FrmUserInfo\_FormClosing(object sender, FormClosingEventArgs e)

{

RoleManage.SerializeRole();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

}

}

（3）FrmMoviePlay：影片信息修改功能：using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace MyMovieSystem

{

public partial class FrmMoviePlay : Form

{

public FrmMoviePlay()

{

InitializeComponent();

}

/// <summary>

/// 影片对象

/// </summary>

public Movie movie = new Movie();

/// <summary>

/// 临时数据集

/// </summary>

List<Hall> listHall = new List<Hall>();

/// <summary>

/// 临时存储影厅信息

/// </summary>

Hall hall = new Hall();

#region 加载事件

//加载事件

private void FrmMoviePlay\_Load(object sender, EventArgs e)

{

textBox1.Text = movie.MovieName;

dateTimePicker1.MinDate = DateTime.Now.Date;

dateTimePicker2.MinDate = DateTime.Now.Date;

panel1.BringToFront();

BindDgvAll();

BindCmb();

}

#endregion

#region 绑定数据

/// <summary>

/// 绑定dgv

/// </summary>

private void BindData()

{

listHall.Clear();

label3.Visible = false;

DateTime time = dateTimePicker1.Value;

foreach (Hall item in movie.HallInfo)

{

if (item.ShowData.ToShortDateString() == time.ToShortDateString())

{

listHall.Add(item);

}

}

dataGridView1.AutoGenerateColumns = false;

dataGridView1.DataSource = new BindingList<Hall>(listHall);

if (dataGridView1.RowCount < 1)

label3.Visible = true;

}

/// <summary>

/// 绑定所有数据

/// </summary>

private void BindDgvAll()

{

label3.Visible = false;

dataGridView1.AutoGenerateColumns = false;

dataGridView1.DataSource = new BindingList<Hall>(movie.HallInfo);

if (dataGridView1.RowCount < 1)

label3.Visible = true;

}

/// <summary>

/// 绑定combobox

/// </summary>

private void BindCmb()

{

comboBox1.DataSource = new BindingList<Hall>(MovieManage.dirHall.Values.ToList());

comboBox1.ValueMember = "H\_ID";

comboBox1.DisplayMember = "H\_Name";

}

//修改日期控件值发生

private void dateTimePicker1\_ValueChanged(object sender, EventArgs e)

{

BindData();

}

//全部场次

private void button2\_Click(object sender, EventArgs e)

{

BindDgvAll();

}

#endregion

#region 新增

//新增

private void AddScheduleButton\_Click(object sender, EventArgs e)

{

hall = new Hall();

panel1.Visible = true;

label9.Text = "新增影厅";

button4.Text = "新增";

comboBox1.SelectedIndex = 0;

comboBox4.SelectedIndex = 0;

dateTimePicker2.Value = DateTime.Now;

comboBox2.Text = "00";

comboBox3.Text = "00";

}

#endregion

#region 修改

//修改

private void ModefyButton\_Click(object sender, EventArgs e)

{

if (dataGridView1.RowCount < 1)

{

MessageBox.Show("请选择有效的数据行！","提示：");

return;

}

string id = dataGridView1.CurrentRow.Cells[0].Value.ToString();

string time = dataGridView1.CurrentRow.Cells[4].Value.ToString();

DateTime date = Convert.ToDateTime(dataGridView1.CurrentRow.Cells[3].Value);

panel1.Visible = true;

label9.Text = "修改影厅";

button4.Text = "修改";

foreach (Hall item in movie.HallInfo)

{

if (item.H\_ID == id && item.ShowData == date &&

item.ShowTime == time)

{

comboBox1.Text = item.H\_Name;

comboBox4.Text = item.H\_Type;

dateTimePicker2.Value = item.ShowData;

comboBox2.Text=time.Substring(0,time.IndexOf(':'));

comboBox3.Text = time.Substring(time.IndexOf(':')+1);

hall = item;

break;

}

}

}

#endregion

#region 新增/修改

//新增/修改

private void button4\_Click(object sender, EventArgs e)

{

Hall hall01 = new Hall();

ObjectAssign(hall01);

foreach (Hall item in movie.HallInfo)

{

if (item.H\_ID == hall01.H\_ID && item.ShowData == hall01.ShowData

&& item.ShowTime == hall01.ShowTime)

{

MessageBox.Show("该时间段场次已存在！", "提示：");

return;

}

}

if (button4.Text == "新增")

{

movie.HallInfo.Add(hall01);

MessageBox.Show("添加成功！", "提示：");

}

else if (button4.Text == "修改")

{

ObjectAssign(hall);

MessageBox.Show("修改成功！", "提示：");

}

panel1.Visible = false;

BindDgvAll();

}

#endregion

#region 取出控件文本赋值给对象

/// <summary>

/// 取出控件文本赋值给对象

/// </summary>

private void ObjectAssign(Hall hall)

{

hall.H\_ID = comboBox1.SelectedValue.ToString();

hall.H\_Name = comboBox1.Text;

hall.H\_Type = comboBox4.Text;

hall.ShowData = dateTimePicker2.Value.Date;

hall.ShowTime = comboBox2.Text + ":" + comboBox3.Text;

hall.Price = movie.MoviePrice;

hall.SeatInfo = MovieManage.dirHall[hall.H\_ID].SeatInfo;

}

#endregion

#region 删除

//删除

private void DelButton\_Click(object sender, EventArgs e)

{

string id = dataGridView1.CurrentRow.Cells[0].Value.ToString();

DateTime date = Convert.ToDateTime(dataGridView1.CurrentRow.Cells[3].Value);

string time = dataGridView1.CurrentRow.Cells[4].Value.ToString();

foreach (Hall item in movie.HallInfo)

{

if (item.H\_ID == id && item.ShowData == date&&item.ShowTime == time)

{

DialogResult r = MessageBox.Show("确定将该场次永久删除吗？","提示：",MessageBoxButtons.OKCancel,MessageBoxIcon.Information);

if (r != DialogResult.OK)

return;

movie.HallInfo.Remove(item);

break;

}

}

BindDgvAll();

}

#endregion

//取消

private void button3\_Click(object sender, EventArgs e)

{

panel1.Visible = false;

}

//关闭

private void button1\_Click(object sender, EventArgs e)

{

this.Close();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

}

}

1. 设计思路分析

这一部分实现了影院排片订票系统中面后台部分：可对影片进行增删改查，可对影片进行场次的增删改查，可对用户进行增删改查，可查看营业情况。

观察者模式也叫发布-订阅模式，其定义如下：定义对象间一种一对多的依赖关系，使得当该对象状态改变时，所有依赖于它的对象都会得到通知，并被自动更新。

后台实现相对简单，后台发布发布者基类包含订阅管理代码和通知方法。即影片管理相关信息。之后发布者基类中扩展出该类，应用程序可在运行时配置发布者和订阅者，即用户管理，营业情况等，这些类会对影片信息所发出的更新消息做出响应。整体形成一个影片信息->营业情况->信息改变的触发链。

1. 观察者模式的优点

观察者模式的通知方式可以通过直接调用等同步方式实现（如函数调用，HTTP接口调用等），也可以通过消息队列异步调用，简单来说，就是在软件设计中维护一个依赖列表，当任何状态发生改变自动通知他们。在c#中可以直接利用.net中IObservable<out T>和IObserver<in T>实现观察者模式。

观察者模式的优点在于可以实现表示层和数据逻辑层的分离， 在观察目标和观察者之间建立一个抽象的耦合。修改影片信息时整个系统都会收到改变，同时做出响应。其次观察者模式支持广播通信，简化了一对多系统设计的难度。

最后也是最方便的它符合开闭原则，增加新的具体观察者无须修改原有系统代码，在具体观察者与观察目标之间不存在关联关系的情况下，增加新的观察目标也很方便。本系统中，可以在后台直接加入查看评价，管理员信息修改，甚至加入修改影片评价（系统未实现），直接对加入的影片作出响应即可。缺点是响应的速度不够即时，适用环境不广，需要先建立触发链。

4.展示效果图



