
도서관리
Progame
김남우

목차 A table of contents

1

소개 및 특징

도서관리 program

1 소개 및 특징

도서관리 프로그램을 XML 파일로 저장 관리

도서 및 회원관리 목적

1

Window Form 자체 클래스
와 람다 식으로도 만듬

2

외부에 저장파일 관리
(XML파일형식)

3

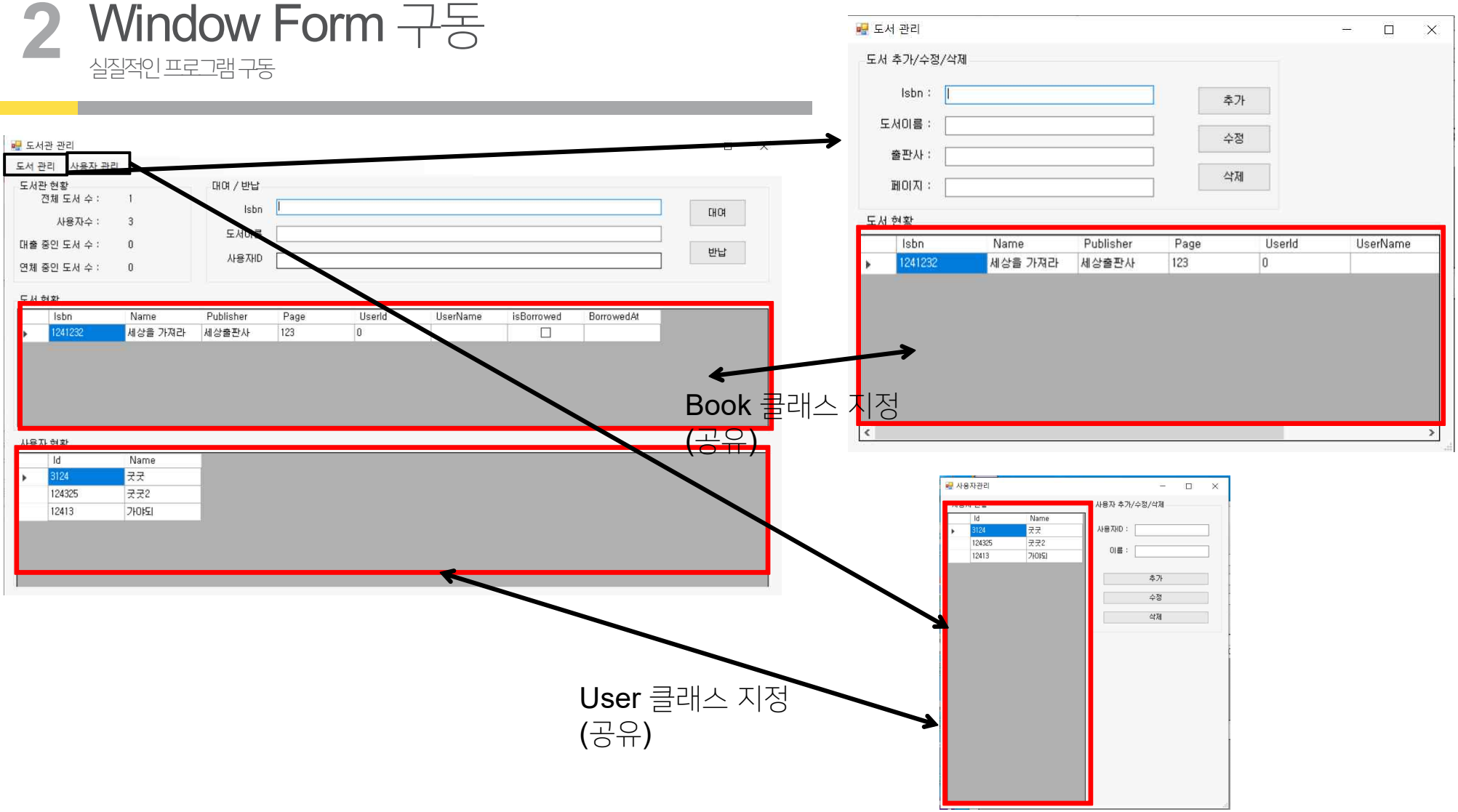
Window Form 구동

실질적인 프로그램 구동

2

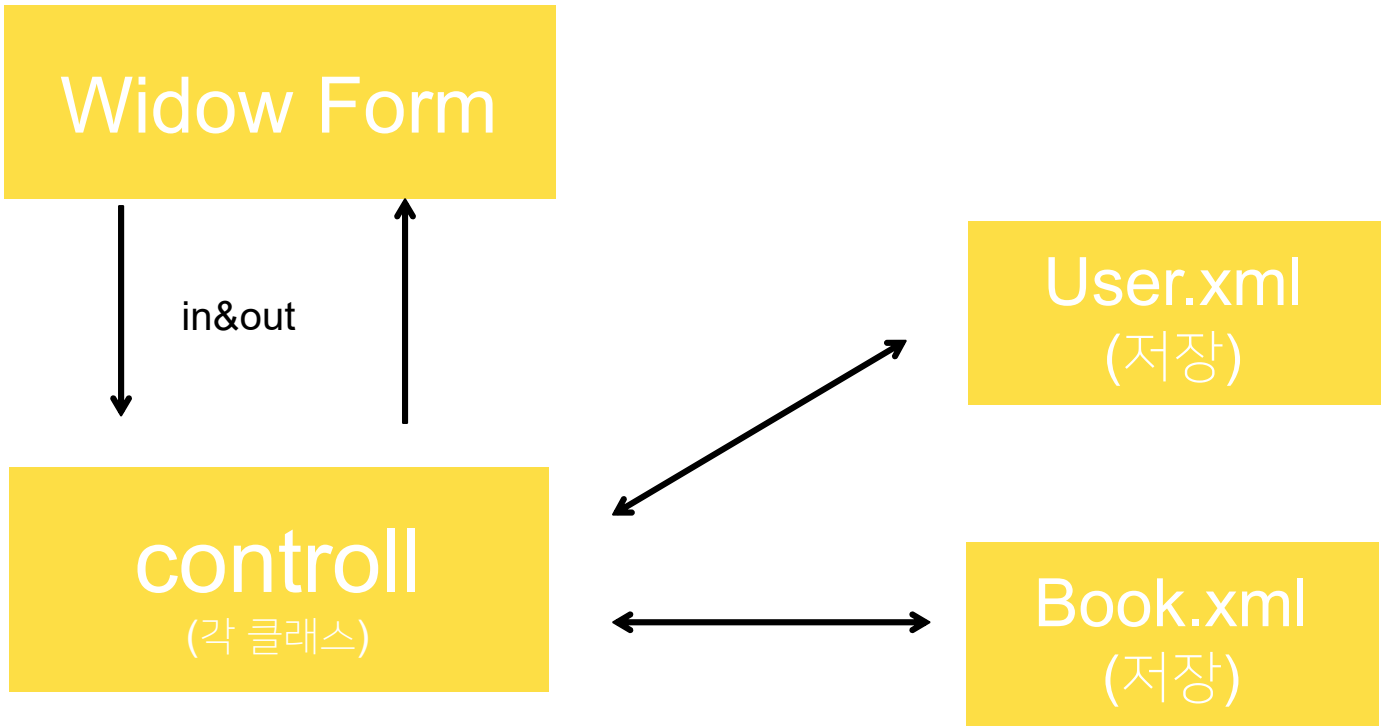
2 Window Form 구동

실질적인 프로그램 구동



2 Window Form 구동

실질적인 프로그램 구동



3

사용 코드

xml형식 및 내용

3 사용 코드

XML파일로 저장

```
public static List<Book> Books = new List<Book>();
public static List<User> Users = new List<User>();
```

리스트로 메모리 생성(getter setter클래스)

```
참조 4개
public static void Load()
{
    try
    {
        string booksOutput = File.ReadAllText(@"./Books.xml");
        XElement booksEXlement = XElement.Parse(booksOutput);
        Books = (from item in booksEXlement.Descendants("book")
                 select new Book()
                 {
                     Isbn = item.Element("isbn").Value,
                     Name = item.Element("name").Value,
                     Publisher = item.Element("publisher").Value,
                     Page = int.Parse(item.Element("page").Value),
                     BorrowedAt = DateTime.Parse(item.Element("borrowedAt").Value),
                     isBorrowed = item.Element("isBorrowed").Value != "0" ? true : false,
                     UserId = int.Parse(item.Element("userId").Value),
                     UserName = item.Element("userName").Value
                 }).ToList<Book>();

        string userOutput = File.ReadAllText(@"./Users.xml");
        XElement usersEXlement = XElement.Parse(userOutput);
        Users = (from item in usersEXlement.Descendants("user")
                 select new User()
                 {
                     Id = int.Parse(item.Element("id").Value),
                     Name = item.Element("name").Value
                 }).ToList<User>();
    }
    catch (Exception)
    {
        System.Windows.Forms.MessageBox.Show("파일이 누락되었습니다.");
        Save();
        Load();
    }


}
```

외부에 저장 파일만들기(XML형식)
각 요소를 알맞은 형태로 불러오기

```
public static void Save()
{
    string booksOutput = "";
    booksOutput += "<books>\n";

    foreach(var item in Books)
    {
        booksOutput += "<book>\n";
        booksOutput = booksOutput + "<isbn>" + item.Isbn + "</isbn>\n";
        booksOutput += "<name>"+item.Name+"</name>\n";
        booksOutput += "<publisher>"+item.Publisher+"</publisher>\n";
        booksOutput += "<page>"+item.Page+"</page>\n";
        booksOutput += "<borrowedAt>"+item.BorrowedAt+"</borrowedAt>\n";
        booksOutput += "<isBorrowed>"+(item.isBorrowed ? 1 :0)+"</isBorrowed>\n";
        booksOutput += "<userId>"+item.UserId+"</userId>\n";
        booksOutput += "<userName>"+item.UserName+"</userName>\n";
        booksOutput += "</book>\n";

    }
    booksOutput += "</books>";

    string usersOutput = "";
    usersOutput += "<users>\n";
    foreach (var item in Users)
    {
        usersOutput += "<user>\n";
        usersOutput += " <id>" + item.Id + "</id>\n";
        usersOutput += " <name>" + item.Name + "</name>\n";
        usersOutput += "</user>\n";
    }
    usersOutput += "</users>";

    File.WriteAllText(@"./Books.xml",booksOutput);
    File.WriteAllText(@"./Users.xml",usersOutput);
}
```

외부에 저장하기(XML형식)
각 요소를 알맞은 형태로 저장

3 사용 코드

Form과 DB에 바인딩 후 DB내역 확인, 연결 저장

```
public Form1()
{
    InitializeComponent();
    Text = "도서관 관리";

    //전체 도서 수
    label_AllBookCount.Text = Datacontrol.Books.Count.ToString();
    //사용자 수
    label_AllUserCount.Text = Datacontrol.Users.Count.ToString();
    //대출중인 도서의 수
    label_AllBorrowedBook.Text = Datacontrol.Books.Where((x) => x.isBorrowed).Count().ToString();
    //연체중인 도서의 수
    label_AllDelayedBook.Text = Datacontrol.Books.Where((x) =>
    {
        return x.isBorrowed && x.BorrowedAt.AddDays(7) < DateTime.Now;
    }).Count().ToString();

    dataGridView_Bookbinding.DataSource = Datacontrol.Books;
    dataGridView_Userbinding.DataSource = Datacontrol.Users;
}
```

DB기반으로 내역확인

GridView에 DB연결

```
private void dataGridView_Userbinding_CellClick(object sender, DataGridViewCellEventArgs e)
{
    try
    {
        User user = dataGridView_Userbinding.CurrentRow.DataBoundItem as User;
        textBox_userId.Text = user.Id.ToString();
    }
    catch (Exception)
    {
    }
}

참조 1개
private void dataGridView_Bookbinding_CellClick(object sender, DataGridViewCellEventArgs e)
{
    try
    {
        Book book = dataGridView_Bookbinding.CurrentRow.DataBoundItem as Book;

        textBox_Isbn.Text = book.Isbn;
        textBox_bookName.Text = book.Name;
    }
    catch (Exception)
    {
    }
}
```

셀 클릭시
(bind한 DB source에서
특정 내역확인)

```
참조 1개
private void button1_Click(object sender, EventArgs e)
{
    if(textBox_Isbn.Text.Trim() == "")
    {
        MessageBox.Show("Isnb를 입력해주세요");
    }else if(textBox_userId.Text.Trim()=="){
        MessageBox.Show("사용자 Id를 입력해주세요");
    }
    else
    {
        try
        {
            Book book = Datacontrol.Books.Single((x) => x.Isbn == textBox_Isbn.Text);
            if (book.isBorrowed)
            {
                MessageBox.Show("대여 중인 도서입니다.");
            }
            else
            {
                User user = Datacontrol.Users.Single((x) => x.Id.ToString() == textBox_userId.Text);
                book.UserId = user.Id;
                book.UserName = user.Name;
                book.isBorrowed = true;
                book.BorrowedAt = DateTime.Now;

                dataGridView_Bookbinding.DataSource = null;
                dataGridView_Bookbinding.DataSource = Datacontrol.Books;
                Datacontrol.Save();

                MessageBox.Show("'" + book.Name + "'이/가 " + user.Name + "'님께 대여되었습니다.");
            }
        }
        catch (Exception)
        {
            MessageBox.Show("존재하지 않는 도서 또는 사용자입니다.");
        }
    }
}
```

도서 대여 (리스트를 통해 만들어
진 정보 저장)
- 기간은 DB 불러올때 자동 지정

3 사용 코드

회원 저장 및 자동 파일 저장(외부로), 도서 저장 쿼리

```
참조 1개
private void button1_Click(object sender, EventArgs e)
{
    if(textBox_isbn.Text.Trim() == "")
    {
        MessageBox.Show("Isnb를 입력해주세요");
    }else if(textBox_userId.Text.Trim()==""){
        MessageBox.Show("사용자 Id를 입력해주세요");
    }
    else
    {
        try
        {
            Book book = Datacontrol.Books.Single((x) => x.Isbn == textBox_isbn.Text);
            if (book.IsBorrowed)
            {
                MessageBox.Show("대여 중인 도서입니다.");
            }
            else
            {
                User user = Datacontrol.Users.Single((x) => x.Id.ToString() == textBox_userId.Text);
                book.UserId = user.Id;
                book.UserName = user.Name;
                book.IsBorrowed = true;
                book.BorrowedAt = DateTime.Now;

                dataGridView_Bookbinding.DataSource = null;
                dataGridView_Bookbinding.DataSource = Datacontrol.Books;
                Datacontrol.Save();

                MessageBox.Show("'" + book.Name + "'이/가" + user.Name + "'님께 대여되었습니다.");
            }
        }
        catch (Exception)
        {
            MessageBox.Show("존재하지 않는 도서 또는 사용자입니다.");
        }
    }
}
```

도서 대여 정보 저장형식

```
private void button_add_Click(object sender, EventArgs e)
{
    try
    {
        if(Datacontrol.Books.Exists((x)=>x.Isbn == textBox_isbn.Tex
        {
            MessageBox.Show("이미 등록된 도서입니다.");
        }
        else
        {
            Book book = new Book()
            {
                Isbn = textBox_isbn.Text,
                Name = textBox_bookname.Text,
                Publisher = textBox_publisher.Text,
                Page = int.Parse(textBox_page.Text)
            };
            Datacontrol.Books.Add(book);

            dataGridView1.DataSource = null;
            dataGridView1.DataSource = Datacontrol.Books;
            Datacontrol.Save();
        }
    }
    catch (Exception)
    {

    }
}
```

도서 등록 저장형식

```
private void button_modify_Click(object sender, EventArgs e)
{
    try
    {
        Book book = Datacontrol.Books.Single((x) => x.Isbn == textBox_isbn.Text);
        book.Name = textBox_bookname.Text;
        book.Publisher = textBox_publisher.Text;
        book.Page = int.Parse(textBox_page.Text);

        dataGridView1.DataSource = null;
        dataGridView1.DataSource = Datacontrol.Books;
        Datacontrol.Save();
    }
    catch (Exception)
    {
        MessageBox.Show("없는 도서 입니다.");
    }
}

참조 1개
private void button_delete_Click(object sender, EventArgs e)
{
    try
    {
        Book book = Datacontrol.Books.Single((x) => x.Isbn == textBox_isbn.Text);
        Datacontrol.Books.Remove(book);

        dataGridView1.DataSource = null;
        dataGridView1.DataSource = Datacontrol.Books;
        Datacontrol.Save();
    }
    catch (Exception)
    {
        MessageBox.Show("없는 도서 입니다.");
    }
}
```

도서 수정 저장형식

1 소개 및 특징

도서관리 프로그램을 XML 파일로 저장 관리

```
InitializeComponent();
Text = "사용자관리";
dataGridView1.DataSource = Datacontrol.Users;

button_add.Click += (sendser, e) =>
{
    try
    {
        if (Datacontrol.Users.Exists((x) => x.Id == int.Parse(textBox_userId.Text)))
        {
            MessageBox.Show("이 사용자 ID는 이미 존재합니다.");
        }
        else
        {
            User user = new User()
            {
                Id = int.Parse(textBox_userId.Text),
                Name = textBox_name.Text
            };
            Datacontrol.Users.Add(user);

            dataGridView1.DataSource = null;
            dataGridView1.DataSource = Datacontrol.Users;
            Datacontrol.Save();
        }
    }
    catch (Exception)
    {

    }
};
```

다른 클래스와 달리 람디식으로 메소드 생성 없이 구성

3 사용 코드

XML과 컴바인하여 저장 및 백업 마무리

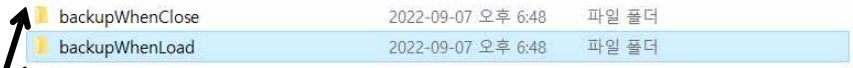
```
참조 2개
private void dataFileBackup(string folderName)
{
    DirectoryInfo di = new DirectoryInfo(folderName);
    if (!di.Exists)
    {
        di.Create();
    }
    string fileName_book = "Books.xml";
    string fileName_user = "Users.xml";
    string sourcPath = @".";
    string targetPath = @".\" + folderName;

    string sourceFile = Path.Combine(sourcPath, fileName_book);
    string desteFile = Path.Combine(targetPath, fileName_book);
    File.Copy(sourceFile, desteFile, true);

    sourceFile = Path.Combine(sourcPath, fileName_user);
    desteFile = Path.Combine(targetPath, fileName_user);
    File.Copy(sourceFile, desteFile, true );
}

참조 1개
private void Form1_Load(object sender, EventArgs e)
{
    dataFileBackup("backupWhenLoad");
}

참조 1개
private void Form1_FormClosed(object sender, FormClosedEventArgs e)
{
    dataFileBackup("backupWhenClose");
}
```



각 폴더에 전 후 저장

4

마무리

4 마무리

코딩후 미진하거나 부족한점 또는 추가해야할 기능

001 >> **보안적인 문제가 있다.**

XML파일 인이상 공공 자료(Book)의 경우 오히려 공공성이나 자료 공유가 편할 수 있으나 보안이 필요한 경우 다른 보안을 위한 장치가 필요하

002 >> **코드생성시 오타에민감**

XML 형식으로 저장시 HTMl형식에 어긋나거나 오타가 생기면 오류가 발생한다.

003 >> **추가적인 기능필요**

관리자만이 사용할 수 있도록 관리할 로그인 기능을 추가할 필요가 있다.-XML이외 다른 방식으로 저장되어야한다.


#
Thanks!