# Техническая информация

Сборка приложения создана на целевой платформе .net framework 4.8. (нужно будет установить пакет, на пк)

Использует реляционная база данных SqLite. База данных проекта находится в PayrollPersonnelManagement\bin\Debug в виде файла PayrollPersonnelManagement.sqlite.

Для общения с базой данных, используется ORM Entity Framework. С помощью контекста и   
dbset-ов выполняются запросы к бд.

Для отображения используются модели Dto, с компонентами свойств. Из модели базы данных происходит маппинг в Dto с помощью библиотеки AutoMapper.

Справочники:

Физическое лицо

Подразделение

Должность

Журналы:

Сотрудник

Разовые начисления

# Расчет начисления

Расчёт разового начисления происходит – по выбранной дате выбирается месяц и рассчитывается количество рабочих дней (5/2 т.е. с понедельника по пятницу). Они умножаются на 8 часов  
(21 день \* 8 = 168) получится сумма рабочих часов в месяце. Оклад делится на сумму рабочих часов в месяце (30000 /168 = 178,57) получится часовая ставка. Часовую ставку умножаем на отработанных часов в месяце и умножаем на набавку которая зависит от подразделения  
( 178,57 \* 172 часа \* 1,05 = 32 249,742) получаем итоговый расчёт зарплаты без НДФЛ

# Основная формИзображение выглядит как стол Автоматически созданное описаниеИзображение выглядит как стол Автоматически созданное описание

## Код основной формы:

using System;

using System.Windows.Forms;

using PayrollPersonnelManagement.Common;

namespace PayrollPersonnelManagement

{

public partial class Main : Form

{

AddEditRead Table { get; set; }

DataSQLite DataSQLite;

string pathSQLite = System.IO.Directory.GetCurrentDirectory()+ @"\PayrollPersonnelManagement.db";

public Main()

{

InitializeComponent();

DataSQLite = new DataSQLite(pathSQLite);

Table = new EmployeeContoll(panel3, new EditForm(), DataSQLite);

}

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

{

Table = new EmployeeContoll(panel3, new EditForm(), DataSQLite);

}

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

{

Table = new PhisicalFaceControll(panel3, new EditForm(), DataSQLite);

}

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

{

Table = new PostContoll(panel3, new EditForm(), DataSQLite);

}

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

{

Table = new SubdivisionControll(panel3, new EditForm(), DataSQLite);

}

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

{

Table = new OneAccuralControll(panel3, new EditForm(), DataSQLite);

}

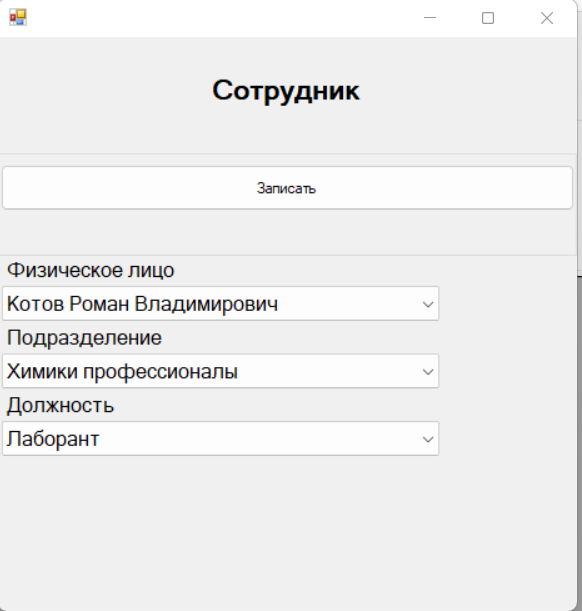
}

}

# Сотрудник:

Изображение выглядит как стол

Автоматически созданное описание



## Код сотрудника:

using System;

using System.Linq;

using System.Windows.Forms;

namespace PayrollPersonnelManagement.Common

{

class EmployeeContoll : AddEditRead

{

Label PhisicalFaceLabel = new Label();

ComboBox PhisicalFaceComboBox = new ComboBox();

Label PostLabel = new Label();

ComboBox PostComboBox = new ComboBox();

Label SubdivisionLabel = new Label();

ComboBox SubdivisionComboBox = new ComboBox();

Employee Employee { get; set; }

const string TableName = "Employee";

const string PhisicalFaceST = "PhisicalFace";

const string PostST = "Post";

const string SubdivisionST = "Subdivision";

public string SelectQuery = "SELECT " +

TableName + ".ID, " +

" PhisicalFace.\*, " +

"Post.\*, " +

"Subdivision.\* " +

"FROM " + TableName + " " +

"INNER JOIN PhisicalFace ON PhisicalFace.ИД = Employee.PhisicalFace " +

"INNER JOIN Post ON Post.ИД = Employee.Post " +

"INNER JOIN Subdivision ON Subdivision.ИД = Employee.Subdivision; ";

public EmployeeContoll(DataSQLite dataSQLite) : base(dataSQLite) { }

public EmployeeContoll(Panel panel, EditForm editForm, DataSQLite dataSQLite)

: base(panel, editForm, dataSQLite)

{

Title = "Сотрудник";

AddButton.MouseClick += AddButton\_MouseClick;

EditButton.MouseClick += EditButton\_MouseClick;

DeleteButton.MouseClick += DeleteButton\_MouseClick;

RefreshButton.MouseClick += RefreshButton\_MouseClick;

directorys.dataGridView1.MouseDoubleClick += EditButton\_MouseClick;

Record.MouseClick += Record\_MouseClick;

Refresh();

}

private void DeleteButton\_MouseClick(object sender, MouseEventArgs e)

{

Employee getItem = (Employee)GetSelectItem();

if (getItem != null)

{

int ID = getItem.ID;

string queryUpdate = "DELETE FROM " + TableName + " " +

"WHERE ID =" + ID + ";";

dataSQLite.Update(queryUpdate);

Refresh();

}

}

private void RefreshButton\_MouseClick(object sender, MouseEventArgs e)

{

Refresh();

}

private void Record\_MouseClick(object sender, MouseEventArgs e)

{

if (Employee is null)

{

string queryUpdate = "INSERT INTO " + TableName + " " +

"( " + PhisicalFaceST + ", " + PostST + ", " + SubdivisionST + ") " +

" VALUES ( '" + ((PhisicalFace)((ComboboxItem)PhisicalFaceComboBox.SelectedItem).Value).ID + "', '"

+ ((Post)((ComboboxItem)PostComboBox.SelectedItem).Value).ID + "', '"

+ ((Subdivision)((ComboboxItem)SubdivisionComboBox.SelectedItem).Value).ID + "');";

dataSQLite.Update(queryUpdate);

}

else

{

string queryUpdate = "UPDATE " + TableName + " " +

"SET " + PhisicalFaceST + " = '" + ((PhisicalFace)((ComboboxItem)PhisicalFaceComboBox.SelectedItem).Value).ID + "', " +

PostST + " = '" + ((Post)((ComboboxItem)PostComboBox.SelectedItem).Value).ID + "', " +

SubdivisionST + " = '" + ((Subdivision)((ComboboxItem)SubdivisionComboBox.SelectedItem).Value).ID + "' " +

"WHERE ID =" + Employee.ID + ";";

dataSQLite.Update(queryUpdate);

}

editForm.Close();

Refresh();

}

private void FillCombobox()

{

PhisicalFaceControll phisicalFaceControll = new PhisicalFaceControll(dataSQLite);

var phisicalFaceList = phisicalFaceControll.Select(phisicalFaceControll.SelectQuery);

ComboBoxDataSource(PhisicalFaceComboBox, phisicalFaceList);

PostContoll postControll = new PostContoll(dataSQLite);

var postList = postControll.Select(postControll.SelectQuery);

ComboBoxDataSource(PostComboBox, postList);

SubdivisionControll subdivisionControll = new SubdivisionControll(dataSQLite);

var subdivisionList = subdivisionControll.Select(subdivisionControll.SelectQuery);

ComboBoxDataSource(SubdivisionComboBox, subdivisionList);

}

private void EditButton\_MouseClick(object sender, MouseEventArgs e)

{

FillCombobox();

Employee getItem = (Employee)GetSelectItem();

if (getItem != null)

{

Employee = getItem;

try

{

PhisicalFaceComboBox.SelectedItem = PhisicalFaceComboBox.Items.Cast<ComboboxItem>().Where(a => ((PhisicalFace)a.Value).ID == Employee.\_PhisicalFace.ID).First();

PostComboBox.SelectedItem = PostComboBox.Items.Cast<ComboboxItem>().Where(a => ((Post)a.Value).ID == Employee.\_Post.ID).First();

SubdivisionComboBox.SelectedItem = SubdivisionComboBox.Items.Cast<ComboboxItem>().Where(a => ((Subdivision)a.Value).ID == Employee.\_Subdivision.ID).First();

editForm.ShowDialog();

Employee = null;

Refresh();

}

catch(Exception ex)

{

System.Windows.Forms.MessageBox.Show(ex.Message, "Ошибка",

System.Windows.Forms.MessageBoxButtons.OK, System.Windows.Forms.MessageBoxIcon.Error);

}

}

}

private void AddButton\_MouseClick(object sender, MouseEventArgs e)

{

FillCombobox();

//Clear();

editForm.ShowDialog();

}

private void Refresh()

{

directorys.dataGridView1.DataSource = Select(SelectQuery);

ColumnNameHeader();

}

private void ColumnNameHeader()

{

if (directorys.dataGridView1.Columns.Count >= 4)

{

directorys.dataGridView1.Columns[0].HeaderText = "ИД";

directorys.dataGridView1.Columns[1].HeaderText = "ФИО";

directorys.dataGridView1.Columns[2].HeaderText = "Подразделение";

directorys.dataGridView1.Columns[3].HeaderText = "Должность";

directorys.dataGridView1.Columns[4].Visible = false;

directorys.dataGridView1.Columns[5].Visible = false;

directorys.dataGridView1.Columns[6].Visible = false;

}

}

public override void LoadEditForm()

{

InitLabel(PhisicalFaceLabel, "PhisicalFace", "Физическое лицо");

InitComboBox(PhisicalFaceComboBox, "PhisicalFace");

InitLabel(SubdivisionLabel, "Subdivision", "Подразделение");

InitComboBox(SubdivisionComboBox, "Subdivision");

InitLabel(PostLabel, "Post", "Должность");

InitComboBox(PostComboBox, "Post");

}

public override object ReaderVirtual()

{

DateTime DateBirth = reader.GetValue(5).ToString() == string.Empty ? DateTime.Now : DateTime.Parse(reader.GetValue(5).ToString());

PhisicalFace phisicalFace = new PhisicalFace()

{

ID = int.Parse(reader.GetValue(1).ToString()),

Surname = reader.GetValue(2).ToString(),

Name = reader.GetValue(3).ToString(),

Patronymic = reader.GetValue(4).ToString(),

DateBirth = DateBirth

};

Post post = new Post()

{

ID = int.Parse(reader.GetValue(6).ToString()),

Name = reader.GetValue(7).ToString(),

Salary = decimal.Parse(reader.GetValue(8).ToString())

};

Subdivision subdivision = new Subdivision()

{

ID = int.Parse(reader.GetValue(9).ToString()),

Name = reader.GetValue(10).ToString(),

Allowance = decimal.Parse(reader.GetValue(11).ToString())

};

Employee employee = new Employee()

{

ID = int.Parse(reader.GetValue(0).ToString()),

\_PhisicalFace = phisicalFace,

\_Post = post,

\_Subdivision = subdivision

};

if (listCombobox)

{

ComboboxItem comboboxItem = new ComboboxItem()

{

Text = employee.FIO + " " + employee.Post,

Value = employee

};

return comboboxItem;

}

return employee;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PayrollPersonnelManagement.Common

{

class Employee

{

public int ID { get; set; }

public string FIO

{

get { return \_PhisicalFace.Surname + " " + \_PhisicalFace.Name + " " + \_PhisicalFace.Patronymic; }

}

public string Subdivision

{

get { return \_Subdivision.Name; }

}

public string Post

{

get { return \_Post.Name; }

}

public PhisicalFace \_PhisicalFace { get; set; }

public Subdivision \_Subdivision { get; set; }

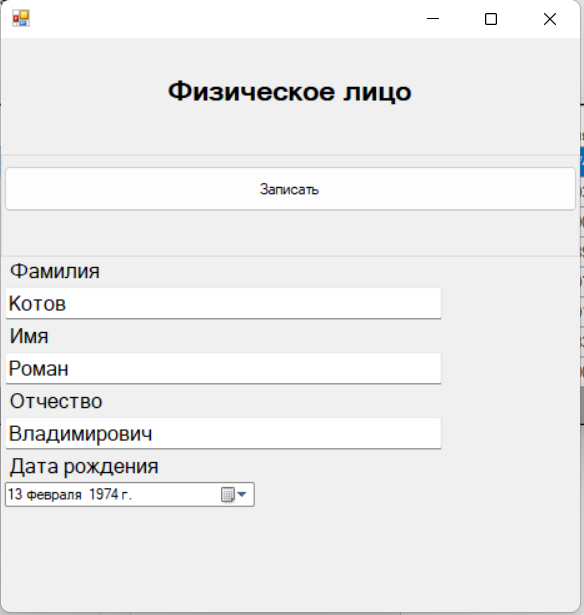
public Post \_Post { get; set; }

}

}

# Физическое лицо:

Изображение выглядит как стол

Автоматически созданное описание 

## Код физического лица:

using System;

using System.Windows.Forms;

namespace PayrollPersonnelManagement.Common

{

class PhisicalFaceControll : AddEditRead

{

Label SurnameLabel = new Label();

TextBox SurnameTextBox = new TextBox();

Label NameLabel = new Label();

TextBox NameTextBox = new TextBox();

Label PatronymicLabel = new Label();

TextBox PatronymicTextBox = new TextBox();

Label DateBirthLabel = new Label();

DateTimePicker DateBirthDateTimePicker = new DateTimePicker();

PhisicalFace PhisicalFace { get; set; }

const string TableName = "PhisicalFace";

const string Surname = "[Фамилия]";

const string Name = "[Имя]";

const string Patronymic = "[Отчество]";

const string DateBirth = "[Дата рождения]";

public string SelectQuery = "Select \* FROM " + TableName;

public PhisicalFaceControll(DataSQLite dataSQLite) : base(dataSQLite) { }

public PhisicalFaceControll(Panel panel, EditForm editForm, DataSQLite dataSQLite)

: base(panel, editForm, dataSQLite)

{

Title = "Физическое лицо";

AddButton.MouseClick += AddButton\_MouseClick;

EditButton.MouseClick += EditButton\_MouseClick;

DeleteButton.MouseClick += DeleteButton\_MouseClick;

RefreshButton.MouseClick += RefreshButton\_MouseClick;

directorys.dataGridView1.MouseDoubleClick += EditButton\_MouseClick;

Record.MouseClick += Record\_MouseClick;

Refresh();

}

private void DeleteButton\_MouseClick(object sender, MouseEventArgs e)

{

PhisicalFace getItem = (PhisicalFace) GetSelectItem();

if (getItem != null)

{

int ID = getItem.ID;

string queryUpdate = "DELETE FROM " + TableName + " " +

"WHERE ИД =" + ID + ";";

dataSQLite.Update(queryUpdate);

Refresh();

}

}

private void RefreshButton\_MouseClick(object sender, MouseEventArgs e)

{

Refresh();

}

private void Record\_MouseClick(object sender, MouseEventArgs e)

{

if(PhisicalFace is null)

{

string queryUpdate = "INSERT INTO " + TableName + " " +

"( " + Surname + ", " + Name + ", " + Patronymic + ", " + DateBirth + ") " +

" VALUES ( '" + SurnameTextBox.Text + "', '" + NameTextBox.Text + "', '" + PatronymicTextBox.Text + "', '"

+ DateBirthDateTimePicker.Text + "');";

dataSQLite.Update(queryUpdate);

}

else

{

string queryUpdate = "UPDATE " + TableName + " " +

"SET " + Surname + " = '" + SurnameTextBox.Text + "', " +

Name + " = '" + NameTextBox.Text + "', " +

Patronymic + " = '" + PatronymicTextBox.Text + "', " +

DateBirth + " = '" + DateBirthDateTimePicker.Text + "' " +

"WHERE ИД =" + PhisicalFace.ID + ";";

dataSQLite.Update(queryUpdate);

}

editForm.Close();

Refresh();

}

private void EditButton\_MouseClick(object sender, MouseEventArgs e)

{

PhisicalFace getItem = (PhisicalFace) GetSelectItem();

if(getItem != null)

{

PhisicalFace = getItem;

SurnameTextBox.Text = PhisicalFace.Surname;

NameTextBox.Text = PhisicalFace.Name;

PatronymicTextBox.Text = PhisicalFace.Patronymic;

DateBirthDateTimePicker.Value = PhisicalFace.DateBirth;

editForm.ShowDialog();

PhisicalFace = null;

Refresh();

}

}

private void Clear()

{

SurnameTextBox.Text = string.Empty;

NameTextBox.Text = string.Empty;

PatronymicTextBox.Text = string.Empty;

DateBirthDateTimePicker.Value = DateTime.Now;

}

private void AddButton\_MouseClick(object sender, MouseEventArgs e)

{

Clear();

editForm.ShowDialog();

}

private void Refresh()

{

directorys.dataGridView1.DataSource = Select(SelectQuery);

ColumnNameHeader();

}

private void ColumnNameHeader()

{

if (directorys.dataGridView1.Columns.Count >= 5)

{

directorys.dataGridView1.Columns[0].HeaderText = "ИД";

directorys.dataGridView1.Columns[1].HeaderText = "Фамилия";

directorys.dataGridView1.Columns[2].HeaderText = "Имя";

directorys.dataGridView1.Columns[3].HeaderText = "Отчество";

directorys.dataGridView1.Columns[4].HeaderText = "Дата рождения";

}

}

public override void LoadEditForm()

{

InitLabel(SurnameLabel, "Surname", "Фамилия");

InitTextbox(SurnameTextBox, "Surname");

InitLabel(NameLabel, "Name", "Имя");

InitTextbox(NameTextBox, "Name");

InitLabel(PatronymicLabel, "Patronymic", "Отчество");

InitTextbox(PatronymicTextBox, "Patronymic");

InitLabel(DateBirthLabel, "DateBirth", "Дата рождения");

InitDateTimePiker(DateBirthDateTimePicker, "DateBirth");

}

public override object ReaderVirtual()

{

DateTime DateBirth = reader.GetValue(4).ToString() == string.Empty ? DateTime.Now : DateTime.Parse(reader.GetValue(4).ToString());

PhisicalFace phisicalFace = new PhisicalFace()

{

ID = int.Parse(reader.GetValue(0).ToString()),

Surname = reader.GetValue(1).ToString(),

Name = reader.GetValue(2).ToString(),

Patronymic = reader.GetValue(3).ToString(),

DateBirth = DateBirth

};

if (listCombobox)

{

ComboboxItem comboboxItem = new ComboboxItem()

{

Text = phisicalFace.Surname +" "+ phisicalFace.Name + " " + phisicalFace.Patronymic,

Value = phisicalFace

};

return comboboxItem;

}

return phisicalFace;

}

}

}

using System;

namespace PayrollPersonnelManagement.Common

{

class PhisicalFace

{

public int ID { get; set; }

public string Surname { get; set; }

public string Name { get; set; }

public string Patronymic { get; set; }

public DateTime DateBirth { get; set; }

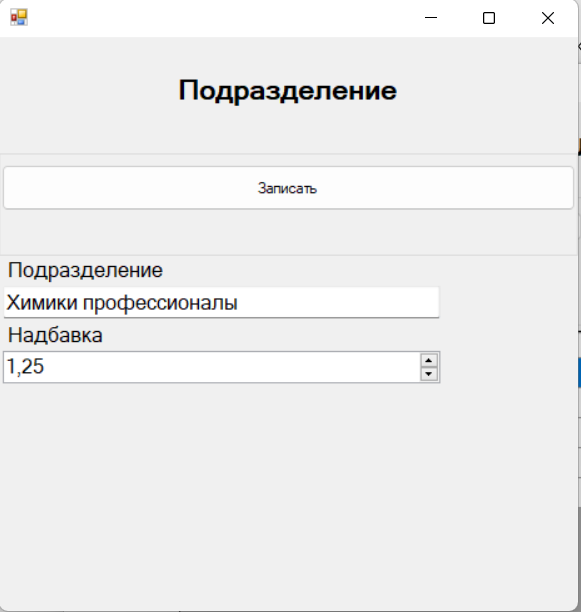
}

}

# Подразделение:

Изображение выглядит как стол

Автоматически созданное описание



## Код подразделения:

using System.Windows.Forms;

namespace PayrollPersonnelManagement.Common

{

class SubdivisionControll : AddEditRead

{

Label NameLabel = new Label();

TextBox NameTextBox = new TextBox();

Label AllowanceLabel = new Label();

NumericUpDown AllowanceNumericUpDown = new NumericUpDown();

Subdivision Subdivision { get; set; }

const string TableName = "Subdivision";

const string Name = "[Подразделение]";

const string Allowance = "[Надбавка]";

public string SelectQuery = "Select \* FROM " + TableName;

public SubdivisionControll(DataSQLite dataSQLite) : base(dataSQLite) { }

public SubdivisionControll(Panel panel, EditForm editForm, DataSQLite dataSQLite)

: base(panel, editForm, dataSQLite)

{

Title = "Подразделение";

AddButton.MouseClick += AddButton\_MouseClick;

EditButton.MouseClick += EditButton\_MouseClick;

DeleteButton.MouseClick += DeleteButton\_MouseClick;

RefreshButton.MouseClick += RefreshButton\_MouseClick;

directorys.dataGridView1.MouseDoubleClick += EditButton\_MouseClick;

Record.MouseClick += Record\_MouseClick;

Refresh();

}

private void DeleteButton\_MouseClick(object sender, MouseEventArgs e)

{

PhisicalFace getItem = (PhisicalFace)GetSelectItem();

if (getItem != null)

{

int ID = getItem.ID;

string queryUpdate = "DELETE FROM " + TableName + " " +

"WHERE ИД =" + ID + ";";

dataSQLite.Update(queryUpdate);

Refresh();

}

}

private void RefreshButton\_MouseClick(object sender, MouseEventArgs e)

{

Refresh();

}

private void Record\_MouseClick(object sender, MouseEventArgs e)

{

if (Subdivision is null)

{

string queryUpdate = "INSERT INTO " + TableName + " " +

"( " + Name + ", " + Allowance + ") " +

" VALUES ( '" + NameTextBox.Text + "', '" + AllowanceNumericUpDown.Value + "');";

dataSQLite.Update(queryUpdate);

}

else

{

string queryUpdate = "UPDATE " + TableName + " " +

"SET " + Name + " = '" + NameTextBox.Text + "', " +

Allowance + " = '" + AllowanceNumericUpDown.Value + "' " +

"WHERE ИД =" + Subdivision.ID + ";";

dataSQLite.Update(queryUpdate);

}

editForm.Close();

Refresh();

}

private void EditButton\_MouseClick(object sender, MouseEventArgs e)

{

Subdivision getItem = (Subdivision)GetSelectItem();

if (getItem != null)

{

Subdivision = getItem;

NameTextBox.Text = Subdivision.Name;

AllowanceNumericUpDown.Value = Subdivision.Allowance;

editForm.ShowDialog();

Subdivision = null;

Refresh();

}

}

private void Clear()

{

NameTextBox.Text = string.Empty;

AllowanceNumericUpDown.Value = decimal.Zero;

}

private void AddButton\_MouseClick(object sender, MouseEventArgs e)

{

Clear();

editForm.ShowDialog();

}

private void Refresh()

{

directorys.dataGridView1.DataSource = Select(SelectQuery);

ColumnNameHeader();

}

private void ColumnNameHeader()

{

if(directorys.dataGridView1.Columns.Count >= 3)

{

directorys.dataGridView1.Columns[0].HeaderText = "ИД";

directorys.dataGridView1.Columns[1].HeaderText = "Подразделение";

directorys.dataGridView1.Columns[2].HeaderText = "Надбавка";

}

}

public override void LoadEditForm()

{

InitLabel(NameLabel, "Name", "Подразделение");

InitTextbox(NameTextBox, "Name");

InitLabel(AllowanceLabel, "Allowance", "Надбавка");

InitNumericUpDown(AllowanceNumericUpDown, "Allowance", 2);

}

public override object ReaderVirtual()

{

Subdivision subdivision = new Subdivision()

{

ID = int.Parse(reader.GetValue(0).ToString()),

Name = reader.GetValue(1).ToString(),

Allowance = decimal.Parse(reader.GetValue(2).ToString())

};

if (listCombobox)

{

ComboboxItem comboboxItem = new ComboboxItem()

{

Text = subdivision.Name,

Value = subdivision

};

return comboboxItem;

}

return subdivision;

}

}

}

namespace PayrollPersonnelManagement.Common

{

class Subdivision

{

public int ID { get; set; }

public string Name { get; set; }

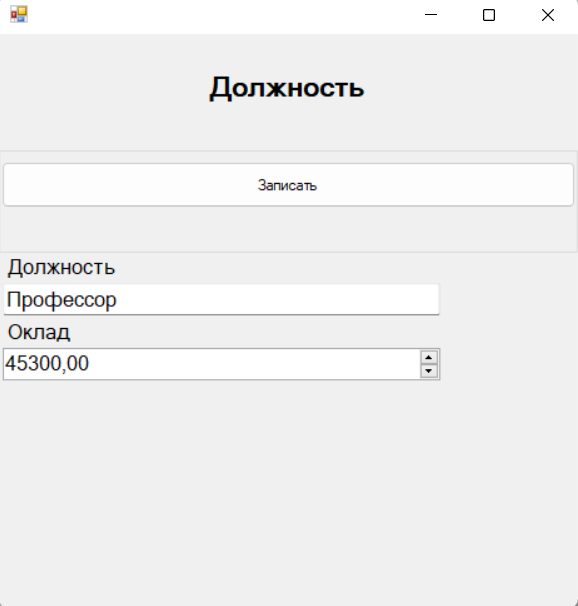
public decimal Allowance { get; set; }

}

}

# Должность:

Изображение выглядит как стол

Автоматически созданное описание 

## Код должности:

using System.Windows.Forms;

namespace PayrollPersonnelManagement.Common

{

class PostContoll : AddEditRead

{

Label NameLabel = new Label();

TextBox NameTextBox = new TextBox();

Label SalaryLabel = new Label();

NumericUpDown SalaryNumericUpDown = new NumericUpDown();

Post Post { get; set; }

const string TableName = "Post";

const string Name = "[Должность]";

const string Salary = "[Оклад]";

public string SelectQuery = "Select \* FROM " + TableName;

public PostContoll(DataSQLite dataSQLite) : base(dataSQLite) { }

public PostContoll(Panel panel, EditForm editForm, DataSQLite dataSQLite)

: base(panel, editForm, dataSQLite)

{

Title = "Должность";

AddButton.MouseClick += AddButton\_MouseClick;

EditButton.MouseClick += EditButton\_MouseClick;

DeleteButton.MouseClick += DeleteButton\_MouseClick;

RefreshButton.MouseClick += RefreshButton\_MouseClick;

directorys.dataGridView1.MouseDoubleClick += EditButton\_MouseClick;

Record.MouseClick += Record\_MouseClick;

Refresh();

}

private void DeleteButton\_MouseClick(object sender, MouseEventArgs e)

{

Post getItem = (Post) GetSelectItem();

if (getItem != null)

{

int ID = getItem.ID;

string queryUpdate = "DELETE FROM " + TableName + " " +

"WHERE ИД =" + ID + ";";

dataSQLite.Update(queryUpdate);

Refresh();

}

}

private void RefreshButton\_MouseClick(object sender, MouseEventArgs e)

{

Refresh();

}

private void Record\_MouseClick(object sender, MouseEventArgs e)

{

if (Post is null)

{

string queryUpdate = "INSERT INTO " + TableName + " " +

"( " + Name + ", " + Salary + ") " +

" VALUES ( '" + NameTextBox.Text + "', '" + SalaryNumericUpDown.Value.ToString() + "');";

dataSQLite.Update(queryUpdate);

}

else

{

string queryUpdate = "UPDATE " + TableName + " " +

"SET " + Name + " = '" + NameTextBox.Text + "', " +

Salary + " = '" + SalaryNumericUpDown.Value.ToString() + "' " +

"WHERE ИД =" + Post.ID + ";";

dataSQLite.Update(queryUpdate);

}

editForm.Close();

Refresh();

}

private void EditButton\_MouseClick(object sender, MouseEventArgs e)

{

Post getItem = (Post) GetSelectItem();

if (getItem != null)

{

Post = getItem;

NameTextBox.Text = Post.Name;

SalaryNumericUpDown.Value = Post.Salary;

editForm.ShowDialog();

Post = null;

Refresh();

}

}

private void Clear()

{

NameTextBox.Text = string.Empty;

SalaryNumericUpDown.Value = decimal.Zero;

}

private void AddButton\_MouseClick(object sender, MouseEventArgs e)

{

Clear();

editForm.ShowDialog();

}

private void Refresh()

{

directorys.dataGridView1.DataSource = Select(SelectQuery);

ColumnNameHeader();

}

private void ColumnNameHeader()

{

if (directorys.dataGridView1.Columns.Count >= 3)

{

directorys.dataGridView1.Columns[0].HeaderText = "ИД";

directorys.dataGridView1.Columns[1].HeaderText = "Должность";

directorys.dataGridView1.Columns[2].HeaderText = "Оклад";

}

}

public override void LoadEditForm()

{

InitLabel(NameLabel, "Name", "Должность");

InitTextbox(NameTextBox, "Name");

InitLabel(SalaryLabel, "Salary", "Оклад");

InitNumericUpDown(SalaryNumericUpDown, "Salary", 2);

}

public override object ReaderVirtual()

{

Post post = new Post()

{

ID = int.Parse(reader.GetValue(0).ToString()),

Name = reader.GetValue(1).ToString(),

Salary = decimal.Parse(reader.GetValue(2).ToString())

};

if (listCombobox)

{

ComboboxItem comboboxItem = new ComboboxItem()

{

Text = post.Name,

Value = post

};

return comboboxItem;

}

return post;

}

}

}

namespace PayrollPersonnelManagement.Common

{

class Post

{

public int ID { get; set; }

public string Name { get; set; }

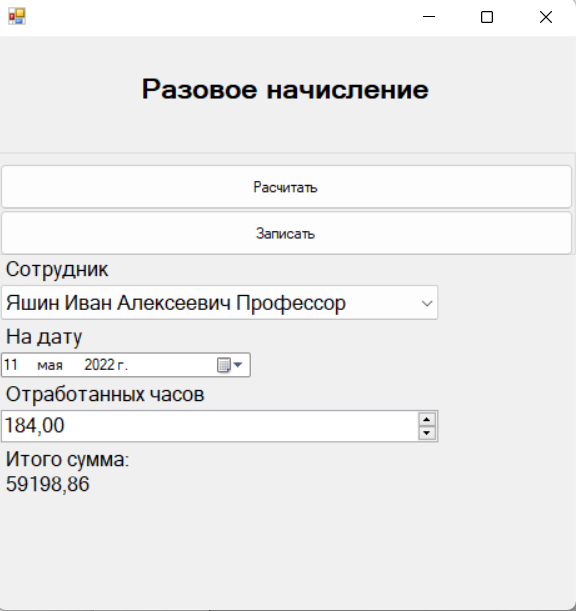
public decimal Salary { get; set; }

}

}

# Разовые начисления:

Изображение выглядит как стол

Автоматически созданное описание 

## Код разовых начислений:

using System;

using System.Linq;

using System.Windows.Forms;

namespace PayrollPersonnelManagement.Common

{

class OneAccuralControll : AddEditRead

{

Label EmployeeLabel = new Label();

ComboBox EmployeeComboBox = new ComboBox();

Label InDateLabel = new Label();

DateTimePicker InDateDateTimePicker = new DateTimePicker();

Label HoursMonthLabel = new Label();

NumericUpDown HoursMonthNumericUpDown = new NumericUpDown();

Label ViewSumLabel = new Label();

Label SumLabel = new Label();

Button AccuralButton = new Button();

OneAccrual OneAccrual { get; set; }

const string TableName = "OneAccrual";

const string EmployeeST = "Employee";

const string InDateST = "InDate";

const string HoursMonthST = "HoursMonth";

const string SumST = "Sum";

public string SelectQuery = "SELECT " +

"Employee.ID, " +

"PhisicalFace.\*, " +

"Post.\*, " +

"Subdivision.\*, " +

TableName + ".ID, " +

"OneAccrual.InDate, " +

"OneAccrual.HoursMonth, " +

"OneAccrual.Sum " +

"FROM " + TableName + " " +

"INNER JOIN Employee ON Employee.ID = " + TableName + ".Employee " +

"INNER JOIN PhisicalFace ON PhisicalFace.ИД = Employee.PhisicalFace " +

"INNER JOIN Post ON Post.ИД = Employee.Post " +

"INNER JOIN Subdivision ON Subdivision.ИД = Employee.Subdivision; ";

public OneAccuralControll(DataSQLite dataSQLite) : base(dataSQLite) { }

public OneAccuralControll(Panel panel, EditForm editForm, DataSQLite dataSQLite)

: base(panel, editForm, dataSQLite)

{

Title = "Разовое начисление";

AddButton.MouseClick += AddButton\_MouseClick;

EditButton.MouseClick += EditButton\_MouseClick;

DeleteButton.MouseClick += DeleteButton\_MouseClick;

RefreshButton.MouseClick += RefreshButton\_MouseClick;

directorys.dataGridView1.MouseDoubleClick += EditButton\_MouseClick;

Record.MouseClick += Record\_MouseClick;

AccuralButton.MouseClick += AccuralButton\_MouseClick;

Refresh();

}

private void AccuralButton\_MouseClick(object sender, MouseEventArgs e)

{

try

{

DateTime now = InDateDateTimePicker.Value;

DateTime first = new DateTime(now.Year, now.Month, 1);

DateTime last = new DateTime(now.Year, now.Month + 1, 1).AddDays(-1);

decimal daysWork = decimal.Parse(GetBusinessDays(first, last).ToString());

decimal hoursWork = daysWork \* 8;

decimal hourlyRate = ((Employee)((ComboboxItem)EmployeeComboBox.SelectedItem).Value).\_Post.Salary / hoursWork;

decimal sum = HoursMonthNumericUpDown.Value \* hourlyRate \*

((Employee)((ComboboxItem)EmployeeComboBox.SelectedItem).Value).\_Subdivision.Allowance;

SumLabel.Text = System.Math.Round(sum,2).ToString();

}

catch (Exception ex)

{

System.Windows.Forms.MessageBox.Show(ex.Message, "Ошибка",

System.Windows.Forms.MessageBoxButtons.OK, System.Windows.Forms.MessageBoxIcon.Error);

}

}

private double GetBusinessDays(DateTime startD, DateTime endD)

{

double calcBusinessDays =

1 + ((endD - startD).TotalDays \* 5 -

(startD.DayOfWeek - endD.DayOfWeek) \* 2) / 7;

if (endD.DayOfWeek == DayOfWeek.Saturday) calcBusinessDays--;

if (startD.DayOfWeek == DayOfWeek.Sunday) calcBusinessDays--;

return calcBusinessDays;

}

private void DeleteButton\_MouseClick(object sender, MouseEventArgs e)

{

OneAccrual getItem = (OneAccrual)GetSelectItem();

if (getItem != null)

{

int ID = getItem.ID;

string queryUpdate = "DELETE FROM " + TableName + " " +

"WHERE ID =" + ID + ";";

dataSQLite.Update(queryUpdate);

Refresh();

}

}

private void RefreshButton\_MouseClick(object sender, MouseEventArgs e)

{

Refresh();

}

private void Record\_MouseClick(object sender, MouseEventArgs e)

{

decimal res = 0;

if(!decimal.TryParse(SumLabel.Text, out res))

{

System.Windows.Forms.MessageBox.Show("Расчитайте начисление", "Информация",

System.Windows.Forms.MessageBoxButtons.OK, System.Windows.Forms.MessageBoxIcon.Information);

return;

}

if(!(res > 0))

{

System.Windows.Forms.MessageBox.Show("Расчитайте начисление", "Информация",

System.Windows.Forms.MessageBoxButtons.OK, System.Windows.Forms.MessageBoxIcon.Information);

return;

}

if (OneAccrual is null)

{

string queryUpdate = "INSERT INTO " + TableName + " " +

"( " + EmployeeST + ", " + InDateST + ", " +

HoursMonthST+ ", " + SumST + ") " +

" VALUES ( '" + ((Employee)((ComboboxItem)EmployeeComboBox.SelectedItem).Value).ID + "', '"

+ InDateDateTimePicker.Text + "', '" + HoursMonthNumericUpDown.Value.ToString() + "', '" +

res.ToString() + "');";

dataSQLite.Update(queryUpdate);

}

else

{

string queryUpdate = "UPDATE " + TableName + " " +

"SET " + EmployeeST + " = '" + ((Employee)((ComboboxItem)EmployeeComboBox.SelectedItem).Value).ID + "', " +

InDateST + " = '" + InDateDateTimePicker.Text + "', " +

HoursMonthST + " = '" + HoursMonthNumericUpDown.Value.ToString() + "', " +

SumST + " = '" + res.ToString() + "' " +

"WHERE ID =" + OneAccrual.ID + ";";

dataSQLite.Update(queryUpdate);

}

editForm.Close();

Refresh();

}

private void FillCombobox()

{

EmployeeContoll employeeContoll = new EmployeeContoll(dataSQLite);

var employeeList = employeeContoll.Select(employeeContoll.SelectQuery);

ComboBoxDataSource(EmployeeComboBox, employeeList);

}

private void EditButton\_MouseClick(object sender, MouseEventArgs e)

{

FillCombobox();

OneAccrual getItem = (OneAccrual)GetSelectItem();

if (getItem != null)

{

OneAccrual = getItem;

try

{

EmployeeComboBox.SelectedItem = EmployeeComboBox.Items.Cast<ComboboxItem>().Where(a => ((Employee)a.Value).ID == OneAccrual.\_Employee.ID).First();

InDateDateTimePicker.Value = OneAccrual.InDate;

HoursMonthNumericUpDown.Value = OneAccrual.HoursMonth;

SumLabel.Text = OneAccrual.Sum.ToString();

editForm.ShowDialog();

OneAccrual = null;

Refresh();

}

catch (Exception ex)

{

System.Windows.Forms.MessageBox.Show(ex.Message, "Ошибка",

System.Windows.Forms.MessageBoxButtons.OK, System.Windows.Forms.MessageBoxIcon.Error);

}

}

}

private void AddButton\_MouseClick(object sender, MouseEventArgs e)

{

FillCombobox();

Clear();

editForm.ShowDialog();

}

private void Clear()

{

InDateDateTimePicker.Value = DateTime.Now;

HoursMonthNumericUpDown.Value = decimal.Zero;

SumLabel.Text = decimal.Zero.ToString();

}

private void Refresh()

{

directorys.dataGridView1.DataSource = Select(SelectQuery);

ColumnNameHeader();

}

private void ColumnNameHeader()

{

if (directorys.dataGridView1.Columns.Count >= 4)

{

directorys.dataGridView1.Columns[0].HeaderText = "ИД";

directorys.dataGridView1.Columns[1].HeaderText = "ФИО должность";

directorys.dataGridView1.Columns[2].HeaderText = "На дату";

directorys.dataGridView1.Columns[3].HeaderText = "Количество часов";

directorys.dataGridView1.Columns[4].HeaderText = "Сумма";

directorys.dataGridView1.Columns[5].Visible = false;

}

}

public override void LoadEditForm()

{

InitLabel(EmployeeLabel, "Employee", "Сотрудник");

InitComboBox(EmployeeComboBox, "Employee");

InitLabel(InDateLabel, "InDate", "На дату");

InitDateTimePiker(InDateDateTimePicker, "InDateDate");

InitLabel(HoursMonthLabel, "HoursMonth", "Отработанных часов");

InitNumericUpDown(HoursMonthNumericUpDown, "HoursMonth", 2);

InitLabel(ViewSumLabel, "ViewSum", "Итого сумма:");

InitLabel(SumLabel, "SumLabel", decimal.Zero.ToString());

InitEditDefaultButton(AccuralButton, "AccuralButton", "Расчитать");

}

public override object ReaderVirtual()

{

DateTime DateBirth = reader.GetValue(5).ToString() == string.Empty ? DateTime.Now : DateTime.Parse(reader.GetValue(5).ToString());

DateTime inDate = reader.GetValue(13).ToString() == string.Empty ? DateTime.Now : DateTime.Parse(reader.GetValue(13).ToString());

PhisicalFace phisicalFace = new PhisicalFace()

{

ID = int.Parse(reader.GetValue(1).ToString()),

Surname = reader.GetValue(2).ToString(),

Name = reader.GetValue(3).ToString(),

Patronymic = reader.GetValue(4).ToString(),

DateBirth = DateBirth

};

Post post = new Post()

{

ID = int.Parse(reader.GetValue(6).ToString()),

Name = reader.GetValue(7).ToString(),

Salary = decimal.Parse(reader.GetValue(8).ToString())

};

Subdivision subdivision = new Subdivision()

{

ID = int.Parse(reader.GetValue(9).ToString()),

Name = reader.GetValue(10).ToString(),

Allowance = decimal.Parse(reader.GetValue(11).ToString())

};

Employee employee = new Employee()

{

ID = int.Parse(reader.GetValue(0).ToString()),

\_PhisicalFace = phisicalFace,

\_Post = post,

\_Subdivision = subdivision

};

OneAccrual oneAccrual = new OneAccrual()

{

\_Employee = employee,

ID = int.Parse(reader.GetValue(12).ToString()),

InDate = inDate,

HoursMonth = decimal.Parse(reader.GetValue(14).ToString()),

Sum = decimal.Parse(reader.GetValue(15).ToString())

};

return oneAccrual;

}

}

}

using System;

namespace PayrollPersonnelManagement.Common

{

class OneAccrual

{

public int ID { get; set; }

public string FIOPost { get { return \_Employee.FIO + " " + \_Employee.Post; } }

public DateTime InDate { get; set; }

public decimal HoursMonth { get; set; }

public decimal Sum { get; set; }

public Employee \_Employee { get; set; }

}

}

# Работа с базой данных SQLite:

using System;

using System.Data.SQLite;

namespace PayrollPersonnelManagement.Common

{

class DataSQLite

{

internal SQLiteConnection SQLiteConnection;

public DataSQLite(string pathSQLite)

{

try

{

if(!System.IO.File.Exists(pathSQLite))

{

System.Windows.Forms.MessageBox.Show("База данных не найдена", "Ошибка",

System.Windows.Forms.MessageBoxButtons.OK, System.Windows.Forms.MessageBoxIcon.Error);

}

else

{

SQLiteConnection = new SQLiteConnection("data source="+pathSQLite);

SQLiteConnection.Open();

}

}

catch(Exception ex)

{

System.Windows.Forms.MessageBox.Show(ex.Message, "Ошибка",

System.Windows.Forms.MessageBoxButtons.OK, System.Windows.Forms.MessageBoxIcon.Error);

}

}

public void Update(string query)

{

try

{

SQLiteCommand cmd = new SQLiteCommand(query, SQLiteConnection);

cmd.ExecuteNonQuery();

}

catch(Exception ex)

{

System.Windows.Forms.MessageBox.Show(ex.Message, "Ошибка",

System.Windows.Forms.MessageBoxButtons.OK, System.Windows.Forms.MessageBoxIcon.Error);

}

}

}

}

Данные для составления списков Combobox:

namespace PayrollPersonnelManagement.Common

{

public class ComboboxItem

{

public string Text { get; set; }

public object Value { get; set; }

public override string ToString()

{

return Text;

}

}

}

# Основной абстрактный класс наследования :

using System;

using System.Collections.Generic;

using System.Data.SQLite;

using System.Windows.Forms;

namespace PayrollPersonnelManagement.Common

{

abstract class AddEditRead

{

private string \_title = string.Empty;

protected string Title

{

get

{ return \_title; }

set

{

\_title = value;

directorys.Title.Text = \_title;

EditDefault.Title.Text = \_title;

}

}

protected bool listCombobox = false;

protected Panel panelMain { get; set; }

protected EditForm editForm { get; set; }

protected Directorys directorys {get; set;}

protected EditDefault EditDefault { get; set; }

protected DataSQLite dataSQLite { get; set; }

protected SQLiteDataReader reader;

protected Button AddButton = new Button();

protected Button EditButton = new Button();

protected Button DeleteButton = new Button();

protected Button RefreshButton = new Button();

protected Button Record = new Button();

public AddEditRead(DataSQLite dataSQLite) { this.dataSQLite = dataSQLite; listCombobox = true; }

public AddEditRead(Panel panel, EditForm editForm, DataSQLite dataSQLite)

{

panelMain = panel;

this.editForm = editForm;

this.dataSQLite = dataSQLite;

panelMain.Controls.Clear();

directorys = new Directorys();

panelMain.Controls.Add(directorys);

directorys.Dock = DockStyle.Fill;

EditDefault = new EditDefault();

editForm.panel1.Controls.Add(EditDefault);

EditDefault.Dock = DockStyle.Fill;

directorys.flowLayoutPanel1.Controls.Add(AddButton);

directorys.flowLayoutPanel1.Controls.Add(EditButton);

directorys.flowLayoutPanel1.Controls.Add(DeleteButton);

directorys.flowLayoutPanel1.Controls.Add(RefreshButton);

AddButton.Text = "Добавить";

EditButton.Text = "Изменить";

DeleteButton.Text = "Удалить";

RefreshButton.Text = "Обновить";

InitEditDefaultButton(Record, "Record", "Записать");

LoadEditForm();

}

protected System.Drawing.Font GetFont()

{

return new System.Drawing.Font("Microsoft Sans Serif", 12F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

}

public abstract void LoadEditForm();

protected void InitLabel(Label label, string name, string text)

{

label.AutoSize = true;

label.Font = GetFont();

label.Name = name + "Label";

label.Text = text;

EditDefault.flowLayoutPanel1.Controls.Add(label);

}

protected void InitTextbox(TextBox textBox, string name)

{

textBox.Font = GetFont();

textBox.MaxLength = 50;

textBox.Name = name + "TextBox";

textBox.Size = new System.Drawing.Size(350, 30);

EditDefault.flowLayoutPanel1.Controls.Add(textBox);

}

protected void InitDateTimePiker(DateTimePicker dateTimePicker, string name)

{

dateTimePicker.Name = name + "DateTimePicker";

dateTimePicker.Size = new System.Drawing.Size(200, 22);

EditDefault.flowLayoutPanel1.Controls.Add(dateTimePicker);

}

protected void InitEditDefaultButton(Button button, string name, string text)

{

button.Dock = System.Windows.Forms.DockStyle.Top;

button.Location = new System.Drawing.Point(3, 18);

button.Name = name;

button.Size = new System.Drawing.Size(408, 37);

button.TabIndex = 0;

button.Text = text;

button.UseVisualStyleBackColor = true;

EditDefault.groupBox1.Controls.Add(button);

}

public List<object> Select(string query)

{

try

{

List<object> listobj = new List<object>();

SQLiteCommand command = new SQLiteCommand(query, dataSQLite.SQLiteConnection);

using (reader = command.ExecuteReader())

{

if (reader.HasRows) // если есть данные

{

while (reader.Read()) // построчно считываем данные

{

object obj = ReaderVirtual();

listobj.Add(obj);

}

}

}

return listobj;

}

catch (Exception ex)

{

System.Windows.Forms.MessageBox.Show(ex.Message, "Ошибка",

System.Windows.Forms.MessageBoxButtons.OK, System.Windows.Forms.MessageBoxIcon.Error);

return null;

}

}

public abstract object ReaderVirtual();

protected void InitNumericUpDown(NumericUpDown numericUpDown, string name, int decimalPlaces)

{

numericUpDown.DecimalPlaces = decimalPlaces;

numericUpDown.Font = GetFont();

numericUpDown.Maximum = new decimal(new int[] {

1410065407,

2,

0,

0});

numericUpDown.Name = name+"NumericUpDown";

numericUpDown.Size = new System.Drawing.Size(350, 30);

EditDefault.flowLayoutPanel1.Controls.Add(numericUpDown);

}

protected void ComboBoxDataSource(ComboBox comboBox, List<object> listobj)

{

comboBox.DataSource = listobj;

comboBox.DisplayMember = "Text";

comboBox.ValueMember = "Value";

}

protected void InitComboBox(ComboBox comboBox, string name)

{

comboBox.Font = GetFont();

comboBox.FormattingEnabled = true;

comboBox.Name = name + "ComboBox";

comboBox.Size = new System.Drawing.Size(350, 30);

comboBox.DropDownStyle = ComboBoxStyle.DropDownList;

EditDefault.flowLayoutPanel1.Controls.Add(comboBox);

}

protected object GetSelectItem()

{

object getItem = null;

DataGridViewSelectedRowCollection getRows = directorys.dataGridView1.SelectedRows;

if (getRows.Count > 0 && getRows[0].DataBoundItem != null)

{

getItem = getRows[0].DataBoundItem;

}

return getItem;

}

}

}