

namespace WindowsFormsApp1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

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

{

}

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

{

MessageBox.Show("Click OK to Exit");

Application.Exit();

}

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

{

bool themeChk = false;

if(rdbtnLightTheme.Checked || rdbtnLightTheme.Checked)

{

themeChk = true;

}

if(themeChk == false)

{

MessageBox.Show("Please check one of the option");

} else

{

MessageBox.Show("Correct");

Application.Exit();

}

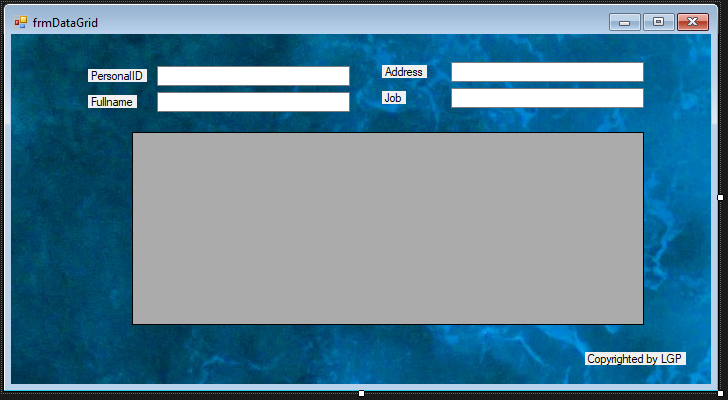
}

}

}

btnExit\_Click sẽ thoát chương trình khi được nhấ

btnLogin\_Click khi nhấn sẽ kiểm tra những radio button theme của mình được check chưa nếu không thì sẽ không cho login, nếu có thì cho login



namespace WindowsFormsApp1

{

public partial class frmDataGrid : Form

{

private List<Person> Profession { get; set; }

public frmDataGrid()

{

Profession = GetProfession();

InitializeComponent();

}

private List<Person> GetProfession()

{

var list = new List<Person>();

list.Add(new Person()

{

PersonID = 1,

Name = "Hoang Anh",

Address = "HCM City",

Job = "IT Developer"

});

list.Add(new Person()

{

PersonID = 2,

Name = "Trung Pham",

Address = "NewYork City",

Job = "IT Lecturer"

});

list.Add(new Person()

{

PersonID = 3,

Name = "Anh Thu",

Address = "Ostrava City",

Job = "IT Student"

});

list.Add(new Person()

{

PersonID = 4,

Name = "Que Chau",

Address = "Kiev City",

Job = "IT Defender"

});

return list;

}

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

{

var profession = this.Profession;

dataGridViewProfession.DataSource = profession;

}

private void dataGridViewProfession\_CellContentClick(object sender,

DataGridViewCellEventArgs e)

{

try

{

var selectRow = dataGridViewProfession.SelectedRows[0].DataBoundItem as

Person;

txtPersonID.Text = selectRow.PersonID.ToString();

txtFullName.Text = selectRow.Name.ToString();

txtAddress.Text = selectRow.Address.ToString();

txtJob.Text = selectRow.Job.ToString();

}

catch (Exception ex)

{

MessageBox.Show("Error:" + ex.Message + "-" + ex.Source);

}

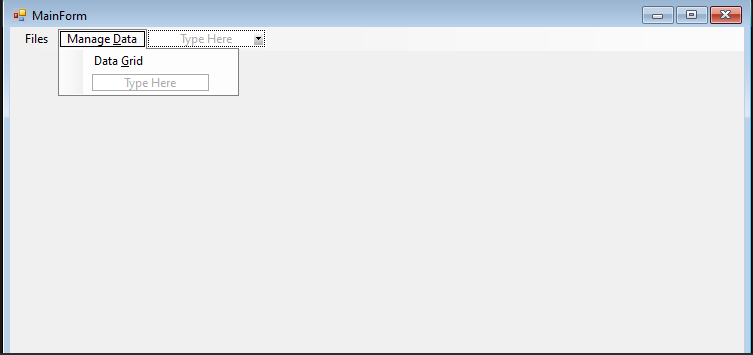
}

}

GetProfession là hàm trả về 1 list danh sách các Person (Person.class được tạo từ trước để chứa các thuộc tính) và lưu trong hàm chứa frmDataGrid.

frmDataGrid\_Load khi chúng ta khởi động chương trình thì sẽ chạy function này, function sẽ load data và đưa vào trong dataGridViewProfession.DataSource = profession; để xem

dataGridViewProfession\_CellContentClick được dùng để hiển thị data từ trong dataGrid lên trên các text box



namespace WindowsFormsApp1

{

public partial class MainForm : Form

{

public MainForm()

{

InitializeComponent();

}

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

{

frmDataGrid f = new frmDataGrid();

f.TopLevel = false;

this.Controls.Add(f);

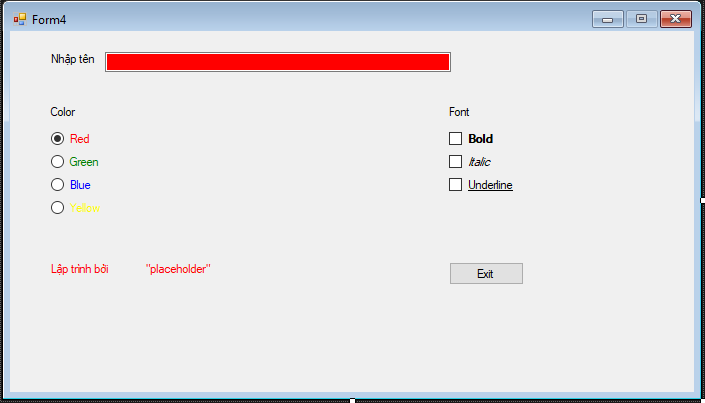
f.Show();

}

}

}

Dùng để hiển thị các form khác nhưng trong trường hợp này, chúng ta đã cố định form hiển thị trong bản thân form đó bằng cách tạo object là frmDataGrid và add Control frmDataGrid và f.Show() để hiển thị



namespace Lab01

{

public partial class Form4 : Form

{

public Form4()

{

InitializeComponent();

}

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

{

MessageBox.Show("Click OK to Exit", "Exit Box");

Application.Exit();

}

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

{

//txtbxInput.Select(0, txtbxInput.TextLength);

//txtbxInput.Select(txtbxInput.Text.Length, 0);

lblOutput.Text = String.Format(txtbxInput.Text);

}

private void rdbtnRed\_CheckedChanged(object sender, EventArgs e)

{

lblLapTrinhBoi.ForeColor = Color.Red;

txtbxInput.BackColor = Color.Red;

lblOutput.ForeColor = Color.Red;

}

private void rdbtnGreen\_CheckedChanged(object sender, EventArgs e)

{

lblLapTrinhBoi.ForeColor = Color.Green;

txtbxInput.BackColor = Color.Green;

lblOutput.ForeColor = Color.Green;

}

private void rdbtnBlue\_CheckedChanged(object sender, EventArgs e)

{

lblLapTrinhBoi.ForeColor = Color.Blue;

txtbxInput.BackColor = Color.Blue;

lblOutput.ForeColor = Color.Blue;

}

private void rdbtnYellow\_CheckedChanged(object sender, EventArgs e)

{

lblLapTrinhBoi.ForeColor = Color.Yellow;

txtbxInput.BackColor = Color.Yellow;

lblOutput.ForeColor = Color.Yellow;

}

private void chkbxBold\_CheckedChanged(object sender, EventArgs e)

{

if (chkbxBold.Checked == true)

{

txtbxInput.Font = new Font(txtbxInput.Font, txtbxInput.Font.Style ^ FontStyle.Bold);

lblOutput.Font = txtbxInput.Font;

}

if(chkbxBold.Checked != true)

{

txtbxInput.Font = new Font(txtbxInput.Font, FontStyle.Bold ^ txtbxInput.Font.Style);

lblOutput.Font = txtbxInput.Font;

}

//if(chkbxBold.Checked == true && chkbxItalic.Checked != true && chkbxUnderline.Checked != true)

//{

// txtbxInput.Font = new Font(txtbxInput.Font, FontStyle.Regular);

//} else if()

//if (chkbxBold.Checked == false && chkbxItalic.Checked == true)

//{

// //MessageBox.Show(txtbxInput.Font.Style.ToString());

// txtbxInput.Font = new Font(txtbxInput.Font,FontStyle.Regular ^ FontStyle.Italic);

// //lblOutput.Font = txtbxInput.Font;

//} else if(chkbxBold.Checked == false && chkbxUnderline.Checked == true)

//{

// txtbxInput.Font = new Font(txtbxInput.Font, FontStyle.Regular ^ FontStyle.Underline);

//}

}

private void chkbxItalic\_CheckedChanged(object sender, EventArgs e)

{

if(chkbxItalic.Checked == true)

{

txtbxInput.Font = new Font(txtbxInput.Font, txtbxInput.Font.Style ^ FontStyle.Italic);

lblOutput.Font = txtbxInput.Font;

}

if(chkbxItalic.Checked != true)

{

txtbxInput.Font = new Font(txtbxInput.Font, FontStyle.Italic ^ txtbxInput.Font.Style);

lblOutput.Font = txtbxInput.Font;

}

//if (chkbxItalic.Checked == false && chkbxBold.Checked == true)

//{

// txtbxInput.Font = new Font(txtbxInput.Font, txtbxInput.Font.Style ^ FontStyle.Bold);

// //lblOutput.Font = txtbxInput.Font;

// //lblOutput.Font = new Font(lblOutput.Font, FontStyle.Regular);

//} else if(chkbxItalic.Checked == false && chkbxUnderline.Checked == true)

//{

// txtbxInput.Font = new Font(txtbxInput.Font, txtbxInput.Font.Style ^ FontStyle.Underline);

//}

}

private void chkbxUnderline\_CheckedChanged(object sender, EventArgs e)

{

if (chkbxUnderline.Checked == true)

{

txtbxInput.Font = new Font(txtbxInput.Font, txtbxInput.Font.Style ^ FontStyle.Underline);

lblOutput.Font = txtbxInput.Font;

}

if (chkbxUnderline.Checked != true)

{

txtbxInput.Font = new Font(txtbxInput.Font, FontStyle.Underline ^ txtbxInput.Font.Style);

lblOutput.Font = txtbxInput.Font;

}

//if (chkbxUnderline.Checked == true)

//{

// txtbxInput.Font = new Font(txtbxInput.Font, FontStyle.Underline);

// lblOutput.Font = new Font(lblOutput.Font, FontStyle.Underline);

//}

//if (chkbxUnderline.Checked == false)

//{

// txtbxInput.Font = new Font(txtbxInput.Font, FontStyle.Regular);

// lblOutput.Font = new Font(lblOutput.Font, FontStyle.Regular);

//}

}

private void Form4\_Load(object sender, EventArgs e)

{

this.ActiveControl = txtbxInput;

}

protected override bool ProcessCmdKey(ref Message msg, Keys keyData)

{

if(keyData == (Keys.Escape))

{

Close();

return true;

}

return base.ProcessCmdKey(ref msg, keyData);

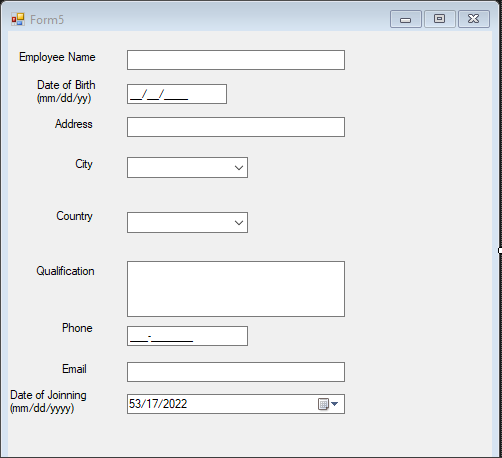
}

}

}

Các fucntion Bold, Italic, Underline hoạt động giống nhau nên chúng ta chỉ nói về 1 cái đó là Bold. Bold sẽ kiểm tra xem ô của function đã check chưa nếu check rồi thì chúng ta sẽ gán Font mới, nếu không thì chúng ta sẽ dùng Font.Regular (chúng ta dùng toán tử XOR txtbxInput.Font.Style ^ FontStyle.Bold để kiểm tra trạng thái của text area)

Còn việc thay màu sắc của Font thì chúng ta dùng những radio button, bắt sự kiện với lable và text, chúng sẽ thay đổi màu sắc tương ứng radio button được chọn



namespace Lab01

{

public partial class Form5 : Form

{

public Form5()

{

InitializeComponent();

}

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

{

this.ActiveControl = txtbxEmployeeName;

}

private void txtbxEmployeeName\_Leave(object sender, EventArgs e)

{

if(txtbxEmployeeName.Text == "")

{

MessageBox.Show("Employee Name should not be empty", "Warning Box");

this.ActiveControl = txtbxEmployeeName;

}

}

protected override bool ProcessCmdKey(ref Message msg, Keys keyData)

{

if (keyData == (Keys.Escape))

{

Close();

return true;

}

return base.ProcessCmdKey(ref msg, keyData);

}

private void msktxtbxDoB\_Leave(object sender, EventArgs e)

{

if (msktxtbxDoB.MaskCompleted != true)

{

MessageBox.Show("Employee's DoB should not be empty", "Warning Box");

this.ActiveControl = msktxtbxDoB;

msktxtbxDoB.Select(0, 0);

}

}

private void txtbxAddress\_Leave(object sender, EventArgs e)

{

if(txtbxAddress.Text == "")

{

MessageBox.Show("Employee Address should not be empty", "Warning Box");

this.ActiveControl = txtbxAddress;

}

}

private void cbbxCity\_Leave(object sender, EventArgs e)

{

String temp = cbbxCity.Text;

if(temp == "")

{

MessageBox.Show("Can not leave City empty", "Warning Box");

this.ActiveControl = cbbxCity;

}

}

private void cbbxCountry\_Leave(object sender, EventArgs e)

{

String temp = cbbxCountry.Text;

if (temp == "")

{

MessageBox.Show("Can not leave Country empty", "Warning Box");

this.ActiveControl = cbbxCountry;

}

}

private void txtbxQualification\_Leave(object sender, EventArgs e)

{

if(txtbxQualification.Text == "")

{

MessageBox.Show("Can not leave Qualification Empty", "Warning Box");

this.ActiveControl = txtbxQualification;

}

}

private void msktxtbxPhone\_Leave(object sender, EventArgs e)

{

if(msktxtbxPhone.MaskCompleted != true)

{

MessageBox.Show("Can not leave Phone Number empty", "Warning Box");

this.ActiveControl= msktxtbxPhone;

msktxtbxPhone.Select(0, 0);

}

}

private void txtbxEmail\_Leave(object sender, EventArgs e)

{

if(txtbxEmail.Text == "")

{

MessageBox.Show("Can not leave Email Empty", "Warning Box");

this.ActiveControl = txtbxEmail;

} else

{

if (txtbxEmail.Text.Contains("@") == false)

{

MessageBox.Show("There must be @ in email", "Missing character in Email");

this.ActiveControl = txtbxEmail;

}

}

}

private void linklblVNExpress\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

try

{

VisitLink();

}

catch (Exception ex)

{

MessageBox.Show("Unable to open link that was clicked.");

}

}

private void VisitLink()

{

// Change the color of the link text by setting LinkVisited

// to true.

linklblVNExpress.LinkVisited = true;

//Call the Process.Start method to open the default browser

//with a URL:

System.Diagnostics.Process.Start("https://www.youtube.com/watch?v=dQw4w9WgXcQ");

}

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

{

String name = txtbxEmployeeName.Text.ToString();

String dob = msktxtbxDoB.Text.ToString();

String address = txtbxAddress.Text.ToString();

String city = cbbxCity.Text.ToString();

String country = cbbxCountry.Text.ToString();

String qualification = txtbxQualification.Text.ToString();

String phone = msktxtbxPhone.Text.ToString();

String email = txtbxEmail.Text.ToString();

String date = dttmpkrDateJoining.Text.ToString();

String temp = name + "\n" + dob + "\n" + address + "\n" + city + "\n" + country + "\n" + qualification + "\n" + phone + "\n" + email + "\n" + date + "";

MessageBox.Show(temp, "Information Box");

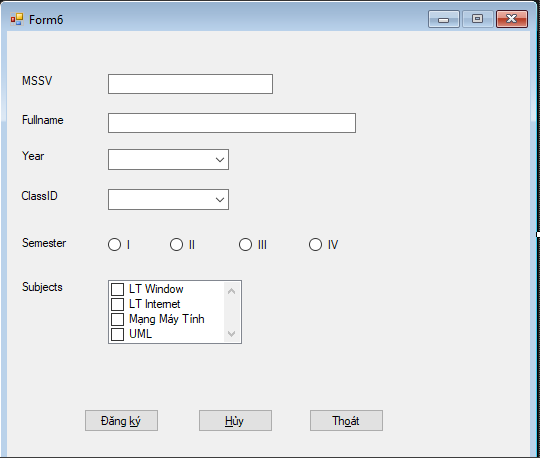
}

}

}

Đối với các text area có masked thì chúng ta chọn mask text và trong mục properties chọn mask và đưa kiểu mask vào, kiểm tra bằng cách dùng Mask.MaskCompleted

Chúng ta sẽ nói chủ yếu đến Focus và error box khi chúng ta không hoàn thành 1 text area. Chúng ta sẽ kiểm tra xem Text của text area trả về là null hoặc không, nếu là null thì chúng ta chạy error và this.ActiveControl = text area box để đưa về text area box đó



namespace Lab01

{

public partial class Form6 : Form

{

public Form6()

{

InitializeComponent();

}

private void Form6\_Load(object sender, EventArgs e)

{

this.ActiveControl = txtMSSV;

}

protected override bool ProcessCmdKey(ref Message msg, Keys keyData)

{

if (keyData == (Keys.Escape))

{

Close();

return true;

}

return base.ProcessCmdKey(ref msg, keyData);

}

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

{

MessageBox.Show("Click OK to Leave", "Exit Message Box");

Application.Exit();

}

private void txtMSSV\_Leave(object sender, EventArgs e)

{

if(txtMSSV.Text == "")

{

MessageBox.Show("Can not leave MSSV empty", "Warning Box");

this.ActiveControl = txtMSSV;

}

}

private void txtFullname\_Leave(object sender, EventArgs e)

{

if(txtFullname.Text == "")

{

MessageBox.Show("Can not leave Fullname empty", "Warning Box");

this.ActiveControl = txtFullname;

}

}

private void cbbxYear\_Leave(object sender, EventArgs e)

{

if(cbbxYear.Text == "")

{

MessageBox.Show("Please Choose a Year", "Warning Box");

this.ActiveControl = cbbxYear;

}

}

private void cbbxClassID\_Leave(object sender, EventArgs e)

{

if(cbbxClassID.Text == "")

{

MessageBox.Show("Please Choose Your Class ID", "Warning Box");

this.ActiveControl = cbbxClassID;

}

}

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

{

String s1 = "";

bool radioToken = false;

List<RadioButton> rdlist = new List<RadioButton>();

rdlist.Add(rdbtnSemester1);

rdlist.Add(rdbtnSemester2);

rdlist.Add(rdbtnSemester3);

rdlist.Add(rdbtnSemester4);

foreach(RadioButton rd in rdlist)

{

if(rd.Checked == true)

{

s1 = rd.Text;

radioToken = true;

break;

}

}

if(radioToken == false)

{

MessageBox.Show("Choose a Semester", "Warning Box");

}

string s = "";

if (chkListbxSubject.CheckedItems.Count == 0)

{

MessageBox.Show("Please Choose a Subject", "Warning Box");

}

else

{

for (int x = 0; x < chkListbxSubject.CheckedItems.Count; x++)

{

s = s + "\tChecked Item " + (x + 1).ToString() + " = " + chkListbxSubject.CheckedItems[x].ToString() + "\n";

}

}

String returnResult = "Sinh viên: " + txtFullname.Text + "\nMSSV: " + txtMSSV.Text + "\nYear: " + cbbxYear.Text + "\nSemester: " + s1 + "\nTotal Subjects:\n" + s + "";

MessageBox.Show(returnResult);

}

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

{

txtMSSV.Clear();

txtFullname.Text = String.Empty;

cbbxYear.Items.Clear();

cbbxClassID.Items.Clear();

rdbtnSemester1.Checked = false;

rdbtnSemester2.Checked = false;

rdbtnSemester3.Checked = false;

rdbtnSemester4.Checked = false;

cbbxYear.Items.Clear();

cbbxClassID.Items.Clear();

for (int x = 0; x < chkListbxSubject.Items.Count; x++)

{

chkListbxSubject.SetItemChecked(x, false);

}

}

}

}

Tất cả mọi thứ đều giống ở trên, trừ 1 vài thành phần như các chúng ta lấy được dữ liệu từ radio button

Chúng ta lấy dữ liệu của radio button khi click vào Login, khi đó chúng ta sẽ tạo 1 List<RadioButton> và add tất cả vào, sau đó chúng ta chạy vòng lặp để kiểm tra xem button nào được click (nếu không button nào được click thì sẽ báo lỗi và bắt chúng ta chọn) khi có button được click phát hiện thì chúng ta sẽ dừng vòng lặp với break và trả giá trị về. Đối với check list thì chúng ta sẽ đếm số lượng phần tử được check bằng cách dùng chkListbxSubject.CheckedItems.Count, sau đó đi đến những phần tử đó bằng cách dùng chkListbxSubject.CheckedItems[x].ToString()

Chúng ta Clear các phần tử bằng dùng câu lệnh Clear() là đủ (có 1 phần tử dùng câu lệnh String.Empty() đây chỉ là thể hiện rằng Clear không phải là cách duy nhất)