# BCIT

**Comp 3951 Selected Topics in Application Development**

**Technical Programming Option**

# Option Head Mirela Gutica

**Winter 2022**

Mark: \_\_\_\_\_\_\_\_\_ /100

Lab GUI, Dialog Boxes, Menus, MDI C#

This is an **individual assignment**. **No late assignments will be accepted**.

1. The code should be documented (comments).
2. Any redundant code will be marked down.
3. You have to find solutions using the most appropriate code and functions.
4. The application should include the functionality of MDI.exe.

**Readings:**

MDI: <https://docs.microsoft.com/en-us/dotnet/desktop/winforms/advanced/multiple-document-interface-mdi-applications?view=netframeworkdesktop-4.8>

Dialog Box: <https://docs.microsoft.com/en-us/dotnet/desktop/wpf/app-development/dialog-boxes-overview?view=netframeworkdesktop-4.8>

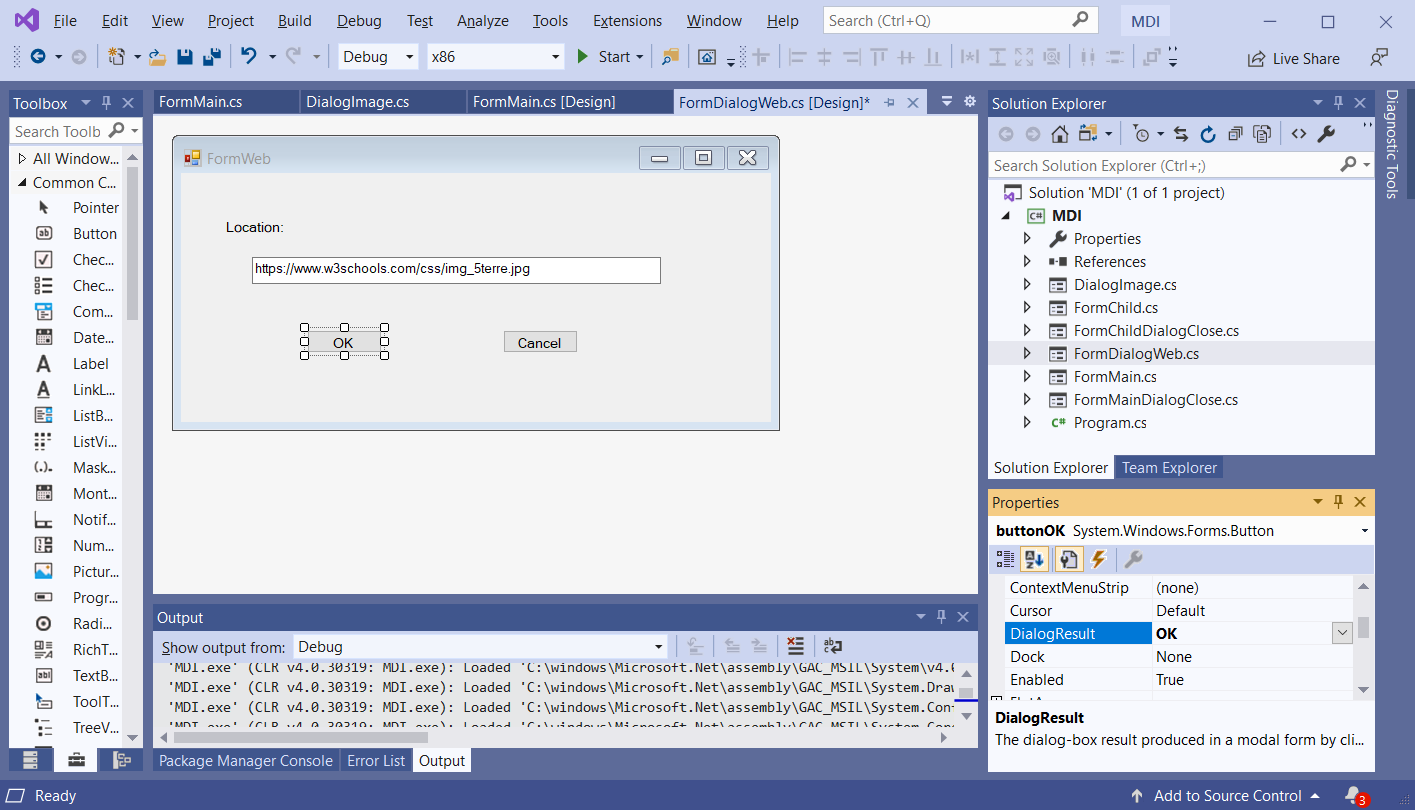
Save/Open: <https://docs.microsoft.com/en-us/dotnet/desktop/winforms/controls/how-to-save-files-using-the-savefiledialog-component?view=netframeworkdesktop-4.8>

**Attacking the Problem:**

1. MDI stand for Multiple Document Interface.
2. Start with a Window application.
3. Observe the properties that are exposed in the “Properties” window.
4. Create the GUI using the Properties (you can do it in code, too).
   1. Set your application as an MDI application.
   2. Add a Main Menu. Add two menus: File and Windows
   3. Add a child form.
   4. Add dialog forms.
   5. Add the existing OpenFile and SaveFile dialogs.
5. Add functionality to your code.

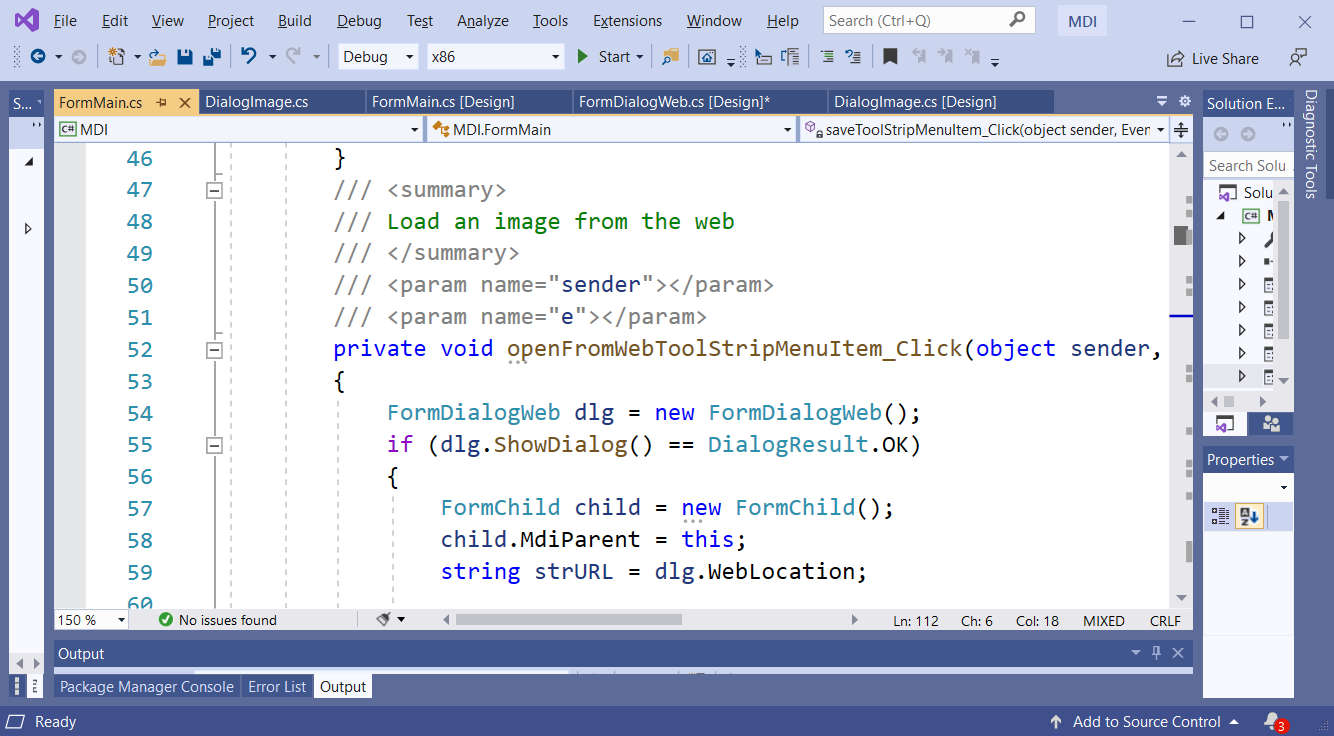
**Hints:**

1. To create an MDI application in C# is very convenient: a form is set up to be an MDI container by a property of the form. You need other forms to be the child forms. Read more about MDI applications in your text book: Chapter 14.2. (Important: creating an MDI application in C/Win32 is **very** time consuming.)
2. Dialog Boxes are created with buttons that represent possible dialog responses (note the “OK” button:

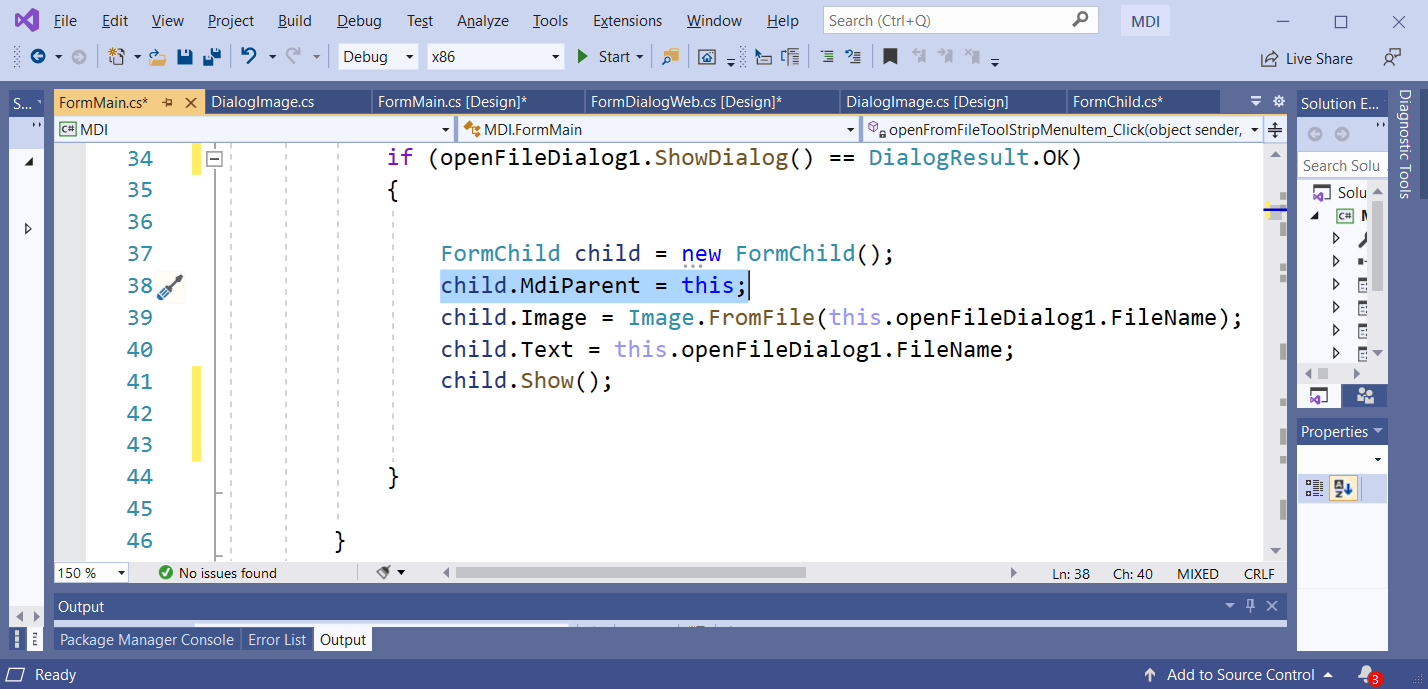


This allows to check in the main form the return from the dialog box (OK, Cancel, etc.)

In the main form, create an object of the dialog form class:



1. The child form needs the MdiParent property to connect to the MDI form:



1. In order to access the image, in the child form, add a private myImage data member (field) and a Property Image both of type Image. Define for the property public accessors: set and get. Read more about properties in textbook Chapters 4.7.
2. Add auto scrolling to the image in the set method:

this.AutoScrollMinSize=myImage.Size;

1. Create a “Paint”event handler for the child window (find the Paint event in the Properties Window under Events). This event handler will be responsible with drawing the image (see method DrawImage from the Graphics namespace). Position the scroll bars correctly in the Paint event (see Graphisc.DrawImage)

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace MDI

{

/// <summary>

/// Child form

/// </summary>

public partial class FormChild : Form

{

public FormChild()

{

InitializeComponent();

}

// field Image

private Image myImage;

/// <summary>

/// property Image

/// </summary>

public Image Image

{

set

{

myImage = value;

this.AutoScrollMinSize = myImage.Size;

}

get

{

return myImage;

}

}

/// <summary>

/// Paint event handler for child window

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void FormChild\_Paint(object sender, PaintEventArgs e)

{

e.Graphics.DrawImage(myImage, this.AutoScrollPosition.X,

this.AutoScrollPosition.Y, myImage.Width, myImage.Height);

}

}

}

1. Disable the Save and SaveAs menu items if no child exists. Use the “DropDownOpening” handler of the File menu (see events in the Properties Window).
2. For Open from Web you need the namespace System.Net.
3. For files and streams you need System.IO;
4. WebRequest and WebResponse are useful classes for loading from the Web.
5. “Save” and “Save As” should work properly. Test if Save is working by looking at the properties of the image (modified date and time).
6. The Exit menu item is implemented: the user will be asked if to save or not.
7. Pay attention to when saving a file on an existing file.
8. Use filters for the OpenFile and SaveFile dialogs. Consider jpg, jpeg, bmp and gif formats.
9. Very important: add **Exceptions**!
10. Do not forget to close and dispose resources:
    1. Example:

formChild.Close();

formChild.Dispose();

Or for image:

formChild.Image.Dispose();