RJ CODE
Advance

M1

RJ CODE MODERN UI

Custom themes, styles, forms and controls for WinForm + Source code

FAST GUIDE

PROGRAMMING TUTORIALS

Simple, easy, fast and fun learning

Contents

1. Introduction

Hello colleague ☺, RJ Code greets you. First of all, I want to thank you for acquiring the project, I really hope this adds a little more to your skills and that you gain new knowledge.

Well, **the project itself is NOT a template**. **The main objective of the project is to teach how to make custom forms and controls** to build modern and elegant user interfaces, which is why I categorize it as a written tutorial. There are **3 help components for this**: the documentation, the comments and the demo.

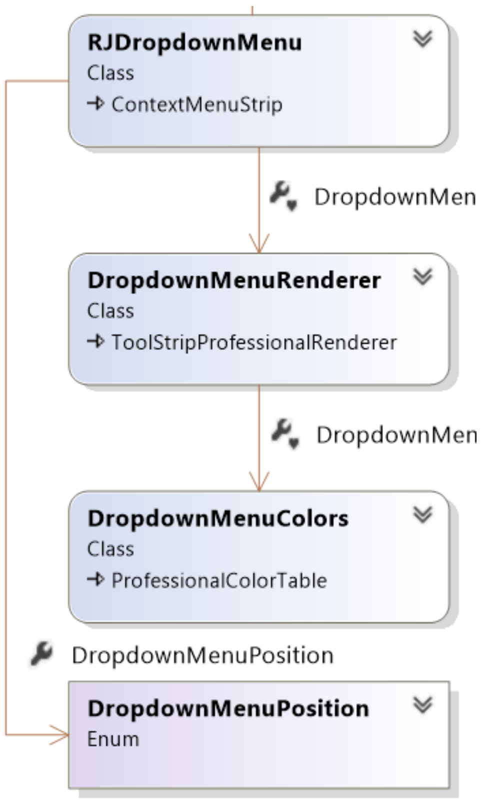
1.1. Documentation

It should be mentioned that it does not refer to the documentation of the life cycle of a software, but rather it is a simple documentation for the end user (For you).

The documentation **describes each of the components of the project**, this facilitates the interaction with the source code, so that you can know the functionality of each of the fields, properties and methods, or **be able to quickly locate a class, method or property** in a context determined, thus being able to modify or add new functions or appearance properties.

The trick is to **look at the class diagrams**, because with this you will already **know from which class a control or form inherits**, as well as knowing which fields, properties and methods it implements. **For example**, suppose we have the **dropdown menu control** (RJDropdownMenu):

- RJDropDownMenu - Collapsed Class Diagram



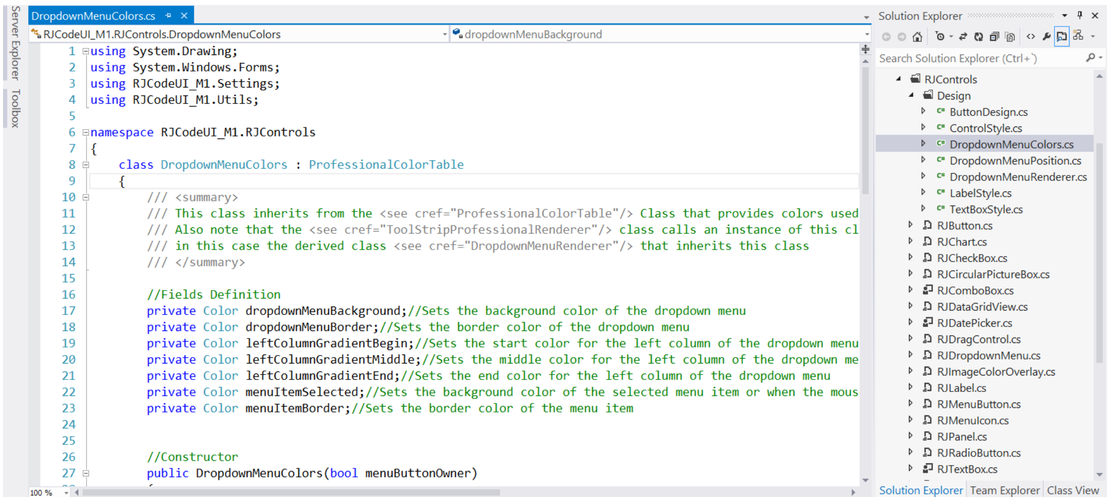
With this we can determine that:

- ✓ The **RJDropdownMenu control inherits from the ContextMenuStrip class** (existing Windorm Form control), therefore it retains all its functionalities.
- ✓ The **RJDropdownMenu control depends on the DropdownMenuRenderer and DropdownMenuColors class** to configure its appearance properties.
- ✓ The **DropdownMenuRenderer class inherits from the ToolStripProfessionalRenderer class.**
- ✓ The **DropdownMenuColors class inherits from the ProfessionalColorTable class**.

There are also the **expanded class diagrams** to know the fields, properties and methods of a class. So if you want to modify the border color, just locate the ColorBorder property or something related to it.

1.2. Comments

Comments **describe or specify the function of a field, property, method, or class**. As seen in the following screenshot of the **DropdownMenuColors** class.



```
using System.Drawing;
using System.Windows.Forms;
using RJCodeUI_M1.Settings;
using RJCodeUI_M1.Utils;

namespace RJCodeUI_M1.RJControls
{
    class DropdownMenuColors : ProfessionalColorTable
    {
        /// <summary>
        /// This class inherits from the <see cref="ProfessionalColorTable"/> Class that provides colors used
        /// Also note that the <see cref="ToolStripProfessionalRenderer"/> class calls an instance of this cl
        /// in this case the derived class <see cref="DropdownMenuRenderer"/> that inherits this class
        /// </summary>

        //Fields Definition
        private Color dropdownMenuBackground;//Sets the background color of the dropdown menu
        private Color dropdownMenuBorder;//Sets the border color of the dropdown menu
        private Color leftColumnGradientBegin;//Sets the start color for the left column of the dropdown menu
        private Color leftColumnGradientMiddle;//Sets the middle color for the left column of the dropdown me
        private Color leftColumnGradientEnd;//Sets the end color for the left column of the dropdown menu
        private Color menuItemSelected;//Sets the background color of the selected menu item or when the mous
        private Color menuItemBorder;//Sets the border color of the menu item


        //Constructor
        public DropdownMenuColors(bool menuButtonOwner)
```

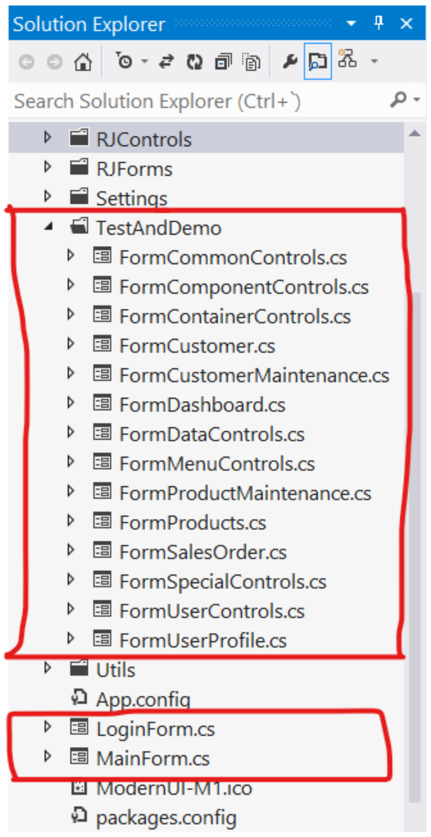
Therefore, **it is not so necessary to read the description of the fields, properties and methods in the documentation, as they are often similar.**

1.3. Demonstration

The demo basically consists of all the forms already designed in the project that test and demonstrate how to use the controls and custom forms (Most of these forms group and describe a specific control).

The LoginForm, MainForm, and all forms in the TestAndDemo folder are demo samples.

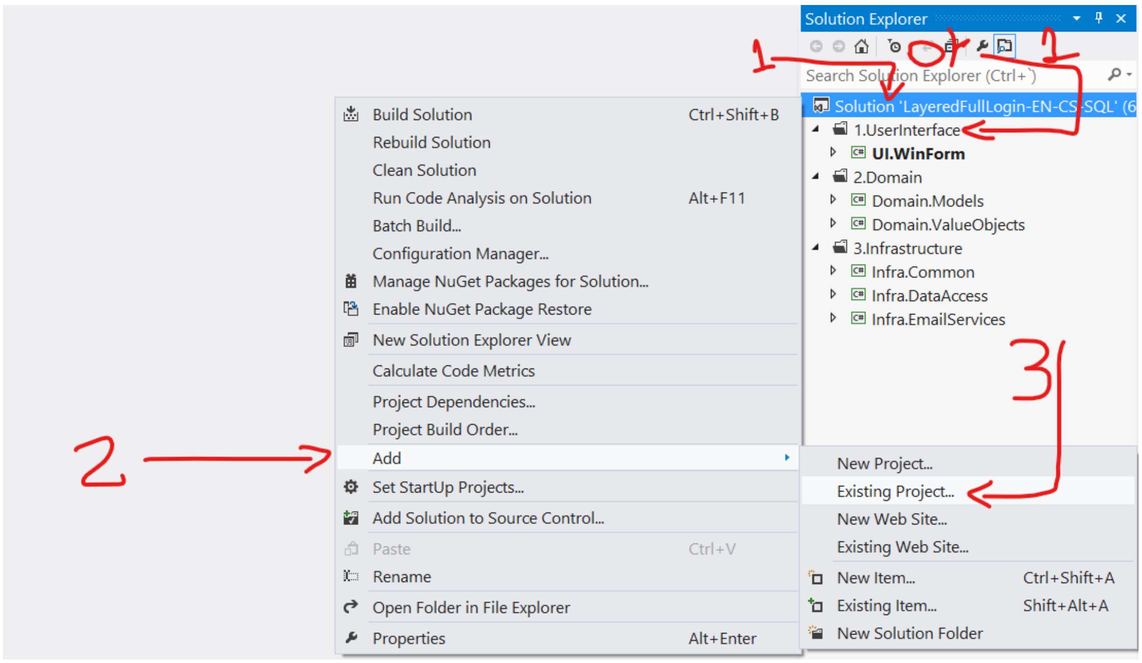
- ✓ The **LoginForm** demonstrates the use of the base class **RJBaseForm**.
- ✓ The **MainForm** demonstrates the use of the **RJMainForm** base class.
- ✓ The forms in the **TestAndDemo** folder demonstrate the use of the **RJChildForm** base class.
- ✓ **All the above forms add custom controls** to demonstrate usage of these.



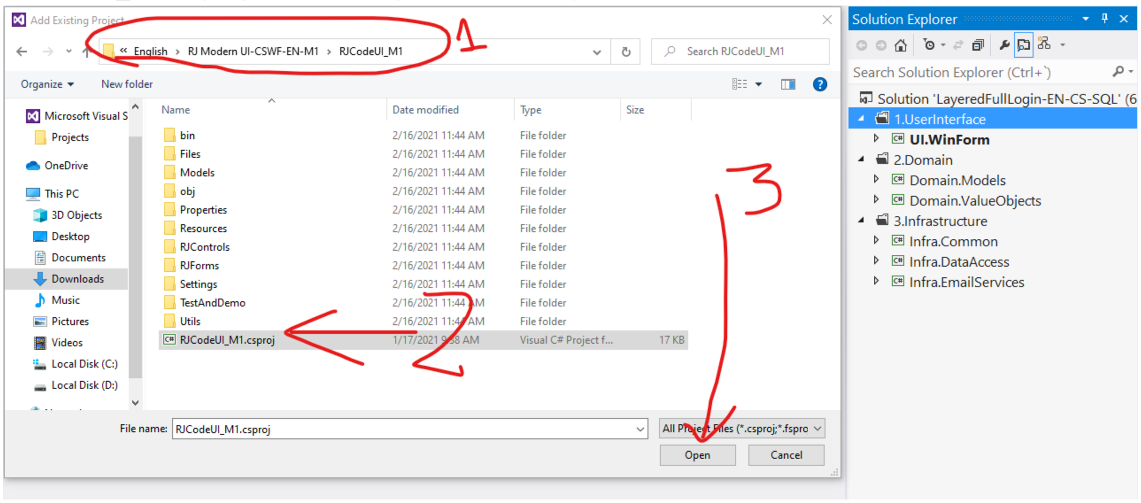
2. How to implement the project in my existing project?

You can easily add the RJ Code Moden UI-M1 project to your existing project by following the steps below:

1. **Right click** on the **solution or solution folder** of your Visual Studio project -> Hover over the **Add** option -> Select the **Existing Project** option.

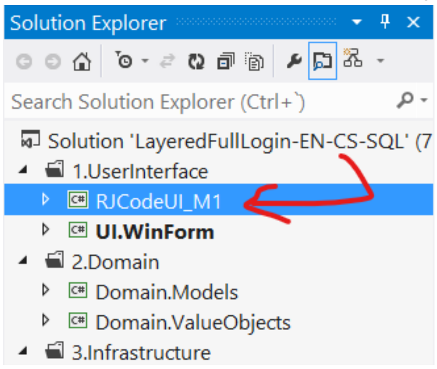


2. **We locate** the **RJCodeUI_M1.csproj** file in the downloaded package -> We **select** the RJCodeUI_M1.csproj file -> Finally we click on **Open**.



Add Existing Project
« English › RJ Modern UI-CSWF-EN-M1 › RJCodeUI_M1
Search RJCodeUI_M1
Organize
New folder
Microsoft Visual S
Projects
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
Local Disk (D:)
Name
Date modified
Type
Size
bin 2/16/2021 11:44 AM File folder
Files 2/16/2021 11:44 AM File folder
Models 2/16/2021 11:44 AM File folder
obj 2/16/2021 11:44 AM File folder
Properties 2/16/2021 11:44 AM File folder
Resources 2/16/2021 11:44 AM File folder
RJControls 2/16/2021 11:44 AM File folder
RJForms 2/16/2021 11:44 AM File folder
Settings 2/16/2021 11:44 AM File folder
TestAndDemo 2/16/2021 11:44 AM File folder
Utils 2/16/2021 11:44 AM File folder
RJCodeUI_M1.csproj 1/17/2021 9:38 AM Visual C# Project f... 17 KB
File name: RJCodeUI_M1.csproj
All Project Files (*.csproj;*.fspro
Open
Cancel
Solution Explorer
Search Solution Explorer (Ctrl+`)
Solution 'LayeredFullLogin-EN-CS-SQL' (6
1.UserInterface
UI.WinForm
2.Domain
Domain.Models
Domain.ValueObjects
3.Infrastructure
Infra.Common
Infra.DataAccess
Infra.EmailServices
1
2
3

3. **You can now use and work on the project** RJ Code Moden UI-M1 (**Don't forget to set the project as the startup project).**



RECOMMENDATION:

It is not possible to update the appearance of your existing forms and controls *with the RJCodeUI_M1 project. Therefore,* ***I recommend redoing the entire User Interface*** *(Presentation) layer of your project in the* ***RJCodeUI_M1 project****. As explained later.*

3. How to use the custom forms?

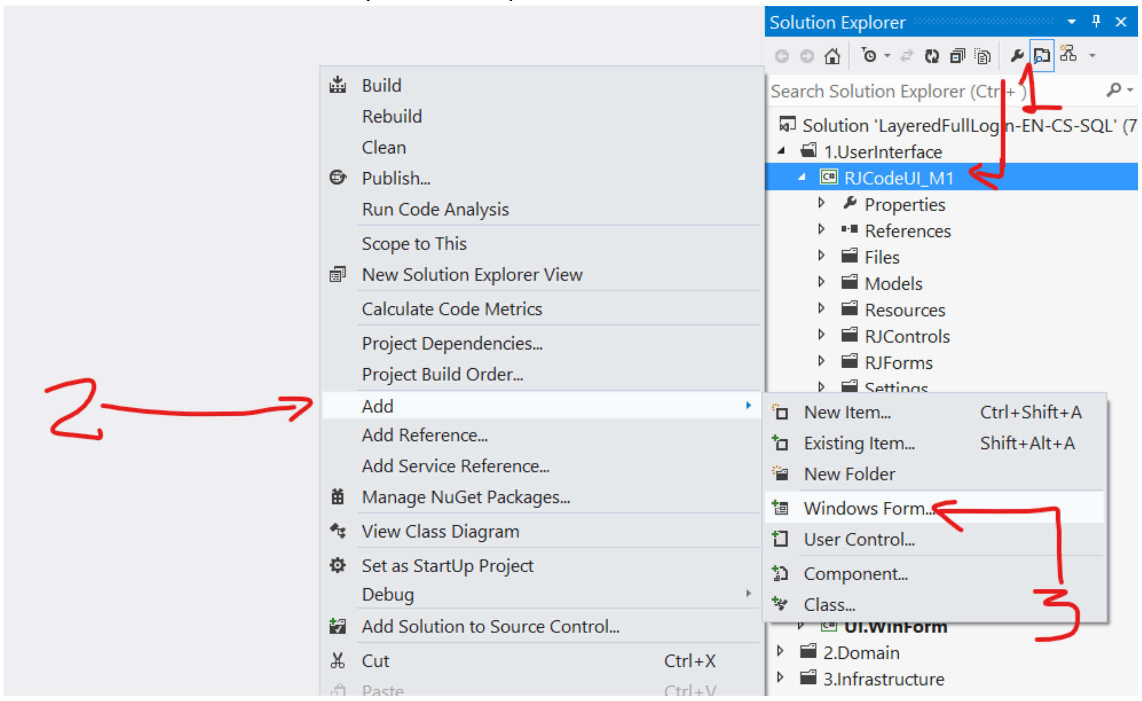
As stated above, **you need to completely redo the User Interface layer** of your existing project **in the RJCodeUI_M1 project** that you added. **I do not recommend editing the project's test and demo forms** (This includes LoginForm, MainForm and all forms in the TestAndDemo folder), as these help you understand and interact with the source code. Therefore, **I recommend creating new forms.**

To use custom forms base (RJBaseForm, RJMainForm, and RJChildForm), just inherit one of them, as shown below.

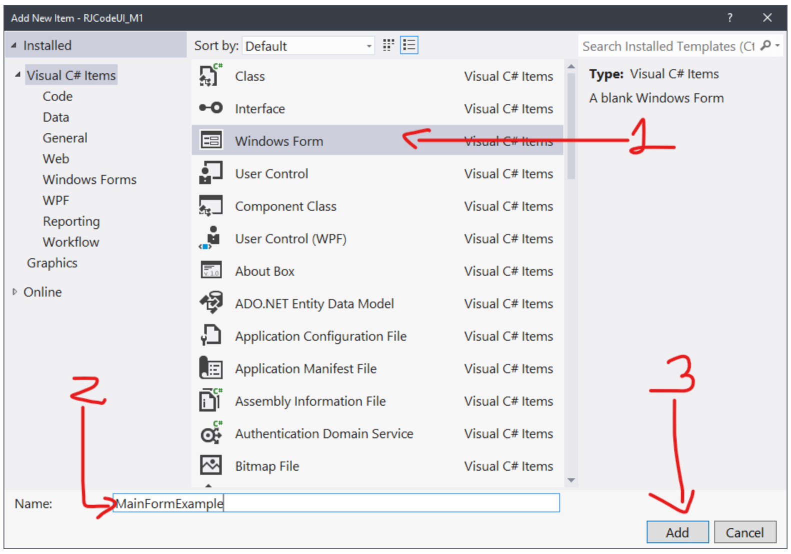
3.1. Create main form

To create a main form **inherit the RJMainForm class**, following the steps below:

1. **Add a new Windows Form** as you normally would.



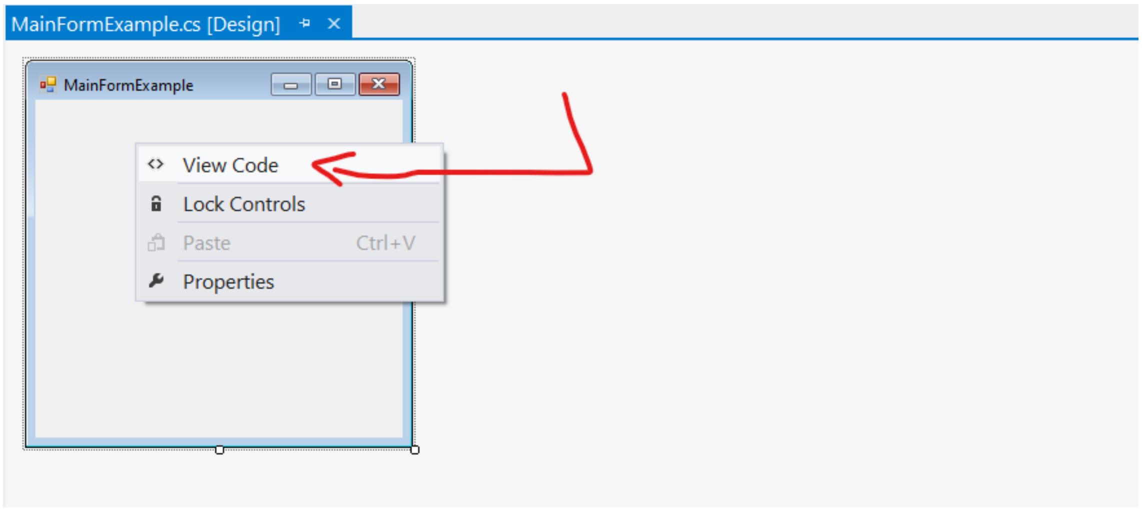
2. **Put a name** for the form (In my case Example Main Form) and click on **Add**.


Add New Item - RJCodeUI_M1
Installed
Visual C# Items
Code
Data
General
Web
Windows Forms
WPF
Reporting
Workflow
Graphics
Online
Sort by: Default
Search Installed Templates (Ct
Class Visual C# Items
Interface Visual C# Items
Windows Form Visual C# Items
User Control Visual C# Items
Component Class Visual C# Items
User Control (WPF) Visual C# Items
About Box Visual C# Items
ADO.NET Entity Data Model Visual C# Items
Application Configuration File Visual C# Items
Application Manifest File Visual C# Items
Assembly Information File Visual C# Items
Authentication Domain Service Visual C# Items
Bitmap File Visual C# Items
Add
Cancel
1
2
3

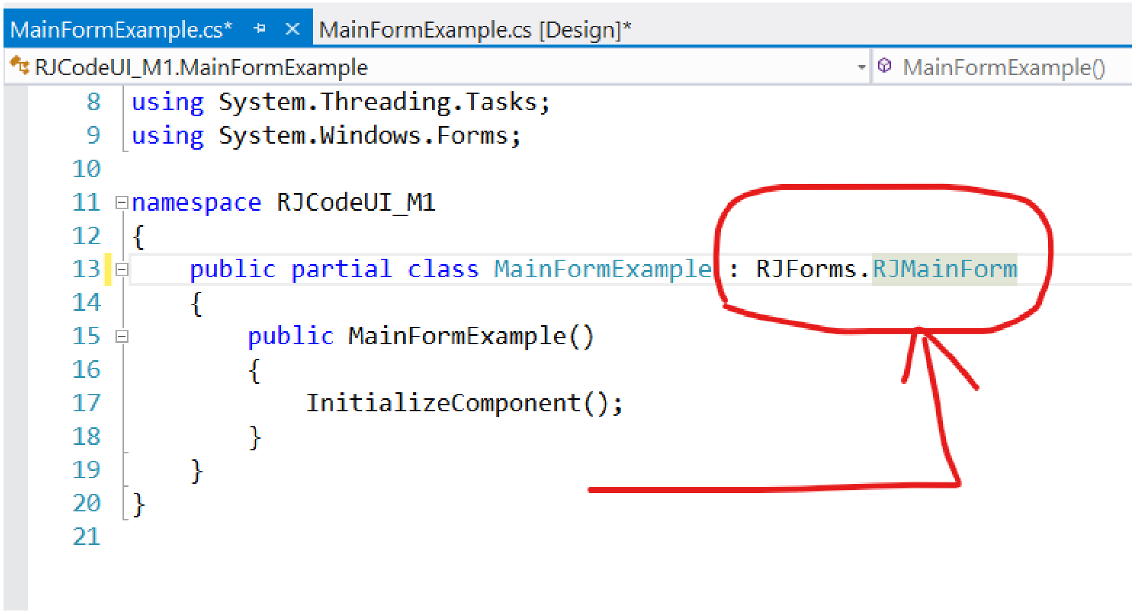
Type: Visual C# Items
A blank Windows Form

Name: MainFormExample

3. **Go to the form code** added (Right click-> View code).



4. Once in the code, **inherit the RJMainForm class**.

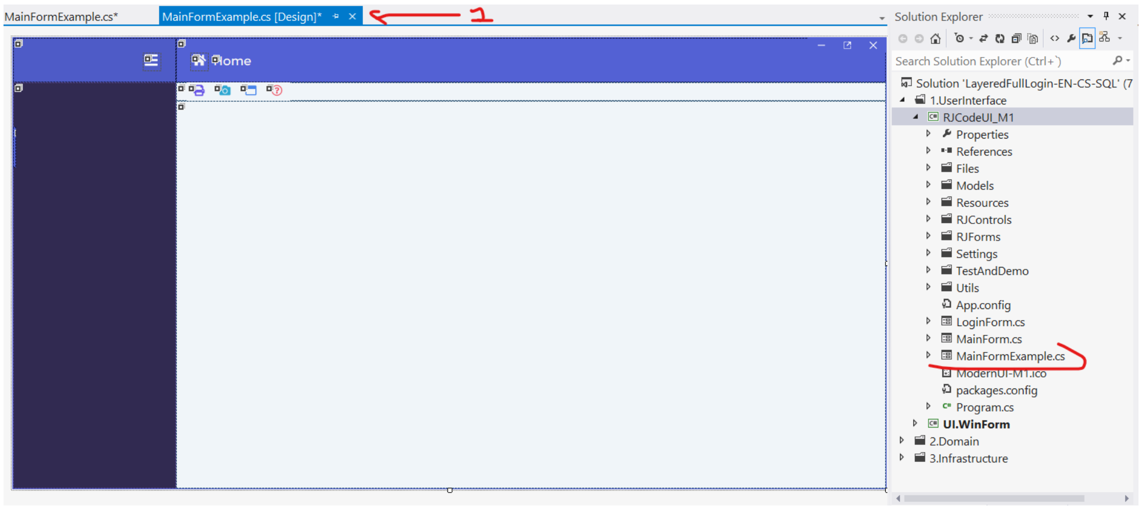


```
using System.Threading.Tasks;
using System.Windows.Forms;

namespace RJCodeUI_M1
{
    public partial class MainFormExample : RJForms.RJMainForm
    {
        public MainFormExample()
        {
            InitializeComponent();
        }
    }
}

```

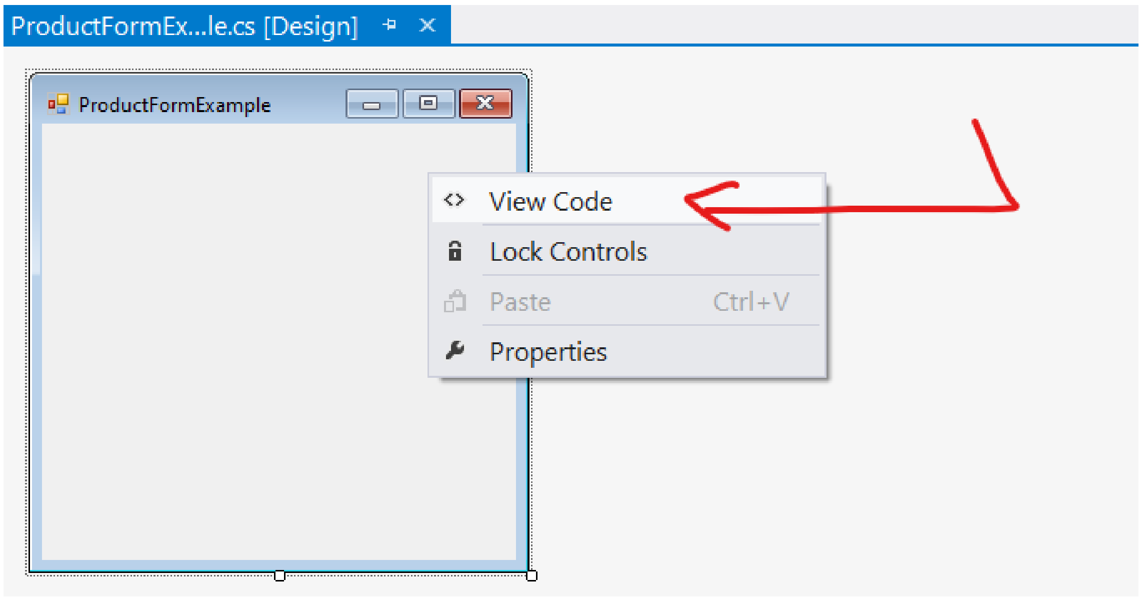
5. Lastly, **go to the form designer and you will have the applied appearance** of the main form as shown below.



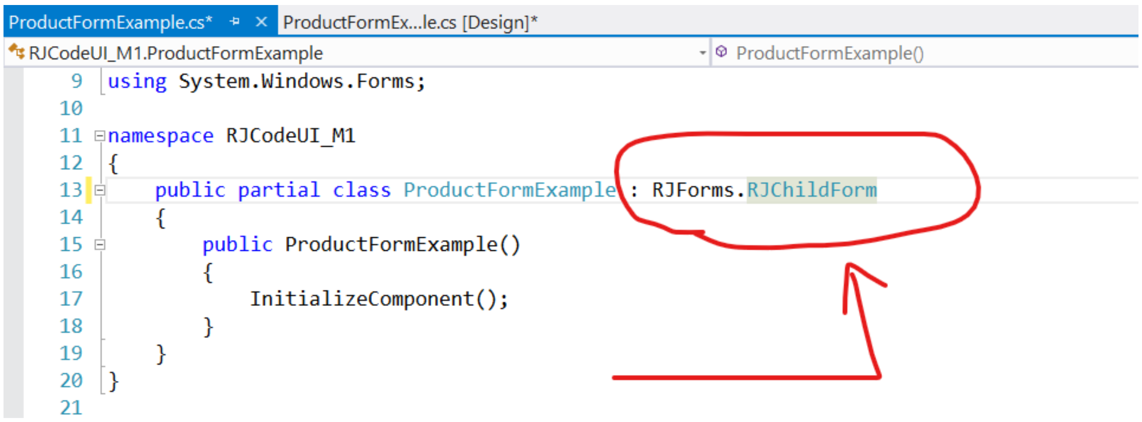
3.2. Create child forms

To create a child form **inherit the RJChildForm class**, following the steps below:

1. **Add a new Windows Form** and go to the form code as before (Step 1-3).



2. Once in code, **inherit the RJChildForm class**.

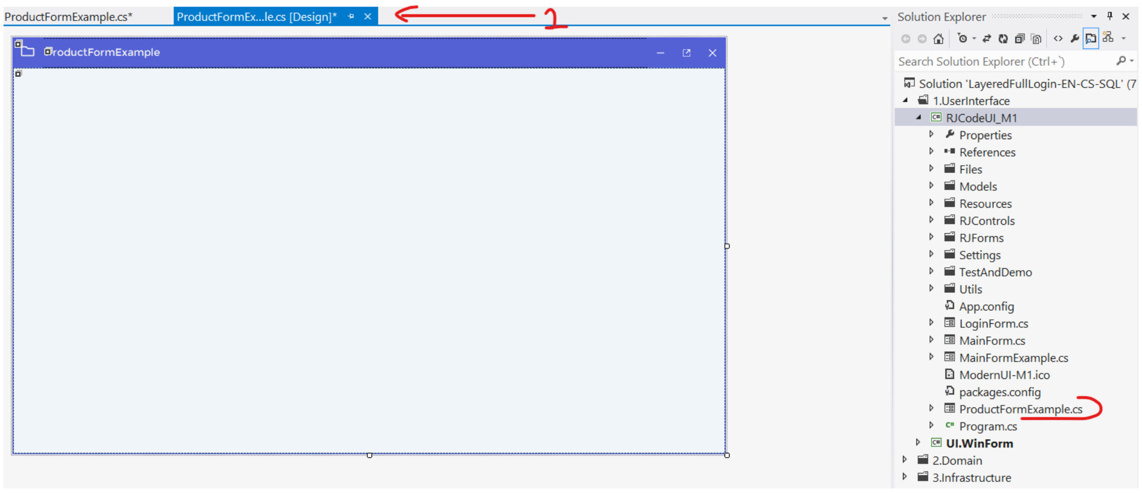

ProductFormExample.cs*
ProductFormEx...le.cs [Design]*
RJCodeUI_M1.ProductFormExample
ProductFormExample()

```csharp
using System.Windows.Forms;

namespace RJCodeUI_M1
{
    public partial class ProductFormExample : RJForms.RJChildForm
    {
        public ProductFormExample()
        {
            InitializeComponent();
        }
    }
}

```

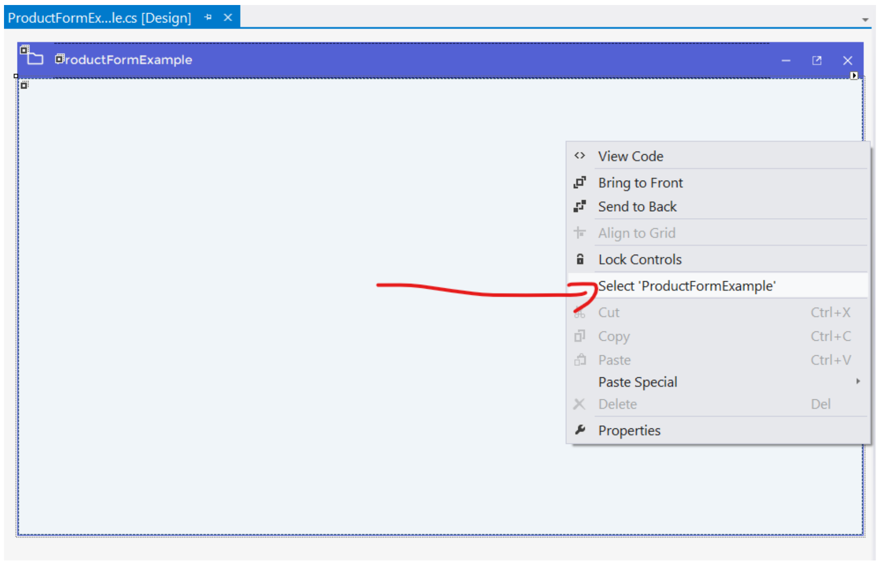
3. Lastly, **go to the form designer and you will have the appearance applied.**



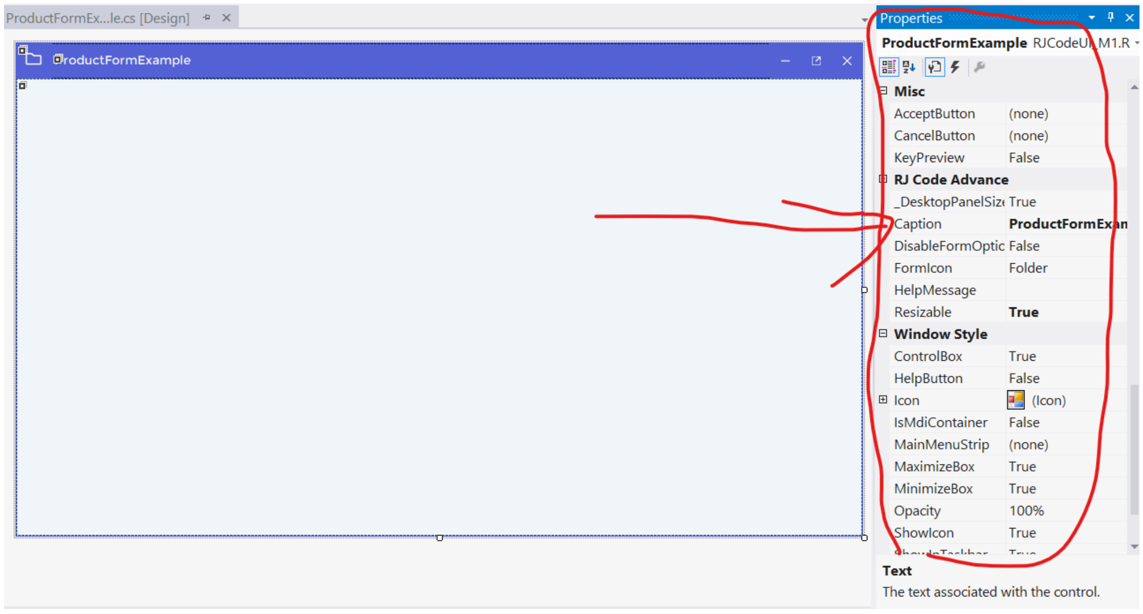
3.3. How to change the properties?

Due to the fact that the base forms RJMainForm and RJChildForm already have controls added for appearance, **it is not possible to directly select the form and change the properties**. To be able to do it you have to do it in the following way:

1. **Right click on the form** or the designer space-> **Click on Select 'Form name'**.



2. You can now change the properties of the form from the toolbox.



ProductFormExample RJCodeUI_M1.R

Misc	
AcceptButton	(none)
CancelButton	(none)
KeyPreview	False
RJ Code Advance	
_DesktopPanelSiz	True
Caption	**ProductFormExan**
DisableFormOptic	False
FormIcon	Folder
HelpMessage	
Resizable	**True**
Window Style	
ControlBox	True
HelpButton	False
Icon	(Icon)
IsMdiContainer	False
MainMenuStrip	(none)
MaximizeBox	True
MinimizeBox	True
Opacity	100%
ShowIcon	True

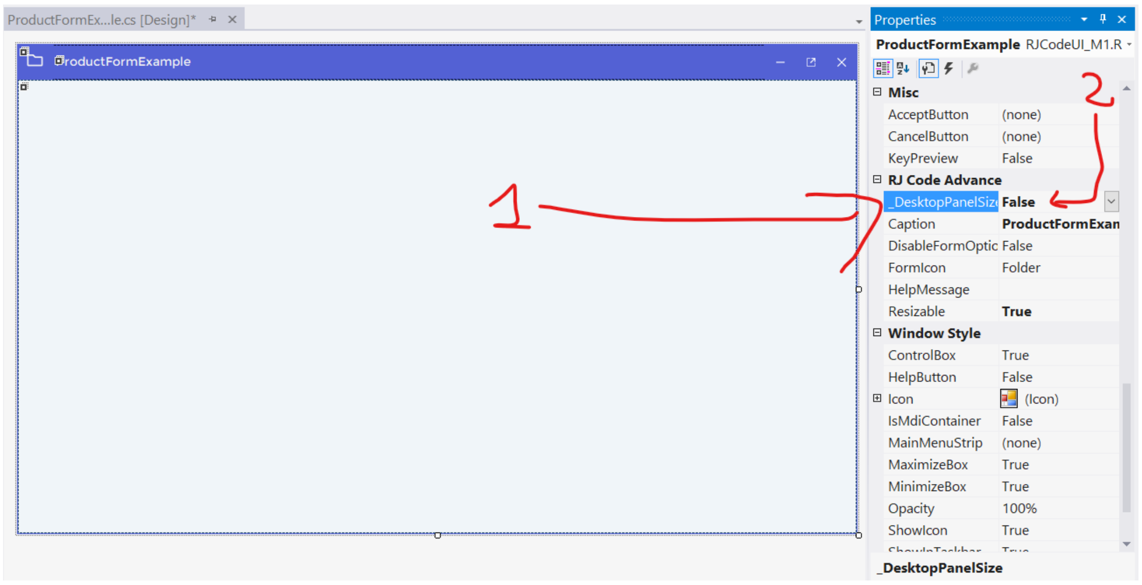
Text
The text associated with the control.

3.4. How to change the width of the child form?

By Default, the size of the child form is the same as the size of the desktop panel of the main form and it **is not possible to change the width, it only allows you to change the height**, because this makes it easier for you to design the form in a more elegant and exact way for the desktop of the main form.

However, in many cases we do not need to open the child form on the desktop of the main form and we want a different size.

1. To change the width of the form, go to the **form properties and set the _DesktopPanelSize property to FALSE**.

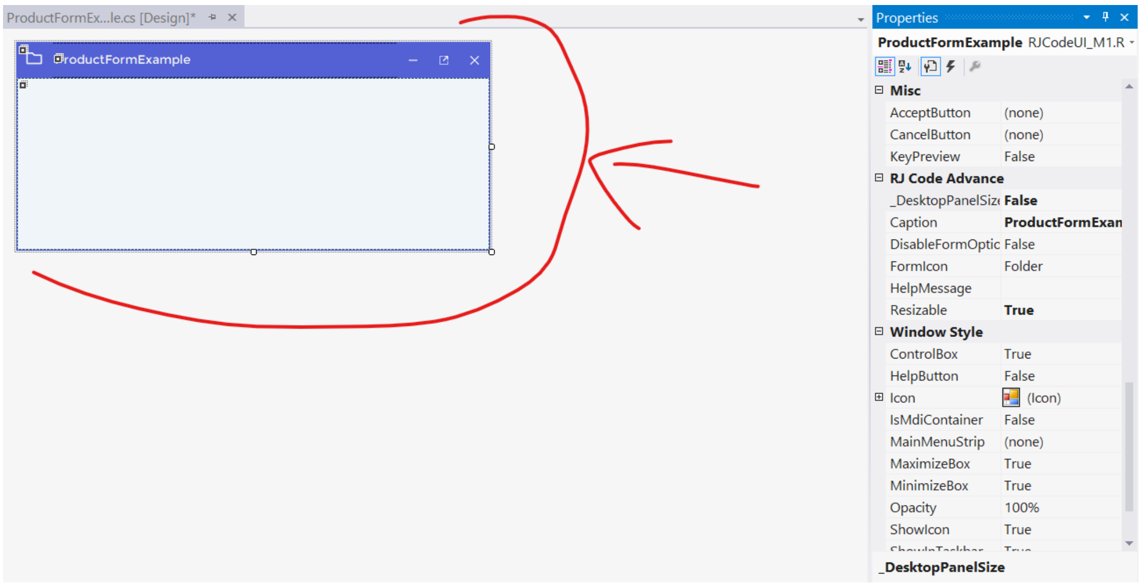


ProductFormEx...le.cs [Design]*
ProductFormExample
1
2
Properties
ProductFormExample RJCodeUI_M1.R

Misc	
AcceptButton	(none)
CancelButton	(none)
KeyPreview	False
RJ Code Advance	
_DesktopPanelSiz	**False**
Caption	**ProductFormExan**
DisableFormOptic	False
FormIcon	Folder
HelpMessage	
Resizable	**True**
Window Style	
ControlBox	True
HelpButton	False
Icon	(Icon)
IsMdiContainer	False
MainMenuStrip	(none)
MaximizeBox	True
MinimizeBox	True
Opacity	100%
ShowIcon	True

_DesktopPanelSize

2. Now **you can change the width of the form** at will (If you only want to change the height of the form, it is not necessary (nor do I recommend it) to perform these steps).



ProductFormEx...le.cs [Design]*
ProductFormExample
Properties
ProductFormExample RJCodeUI_M1.R

Misc	
AcceptButton	(none)
CancelButton	(none)
KeyPreview	False
RJ Code Advance	
_DesktopPanelSiz	**False**
Caption	**ProductFormExan**
DisableFormOptic	False
FormIcon	Folder
HelpMessage	
Resizable	**True**
Window Style	
ControlBox	True
HelpButton	False
Icon	(Icon)
IsMdiContainer	False
MainMenuStrip	(none)
MaximizeBox	True
MinimizeBox	True
Opacity	100%
ShowIcon	True

_DesktopPanelSize

4. How to use custom controls?

To use the project's custom controls, simply open the Visual Studio toolbox and start dragging the controls you need onto the form as you normally would (In case the controls do not appear, compile the project), below are some examples .

4.1. Example 1- Design the main form

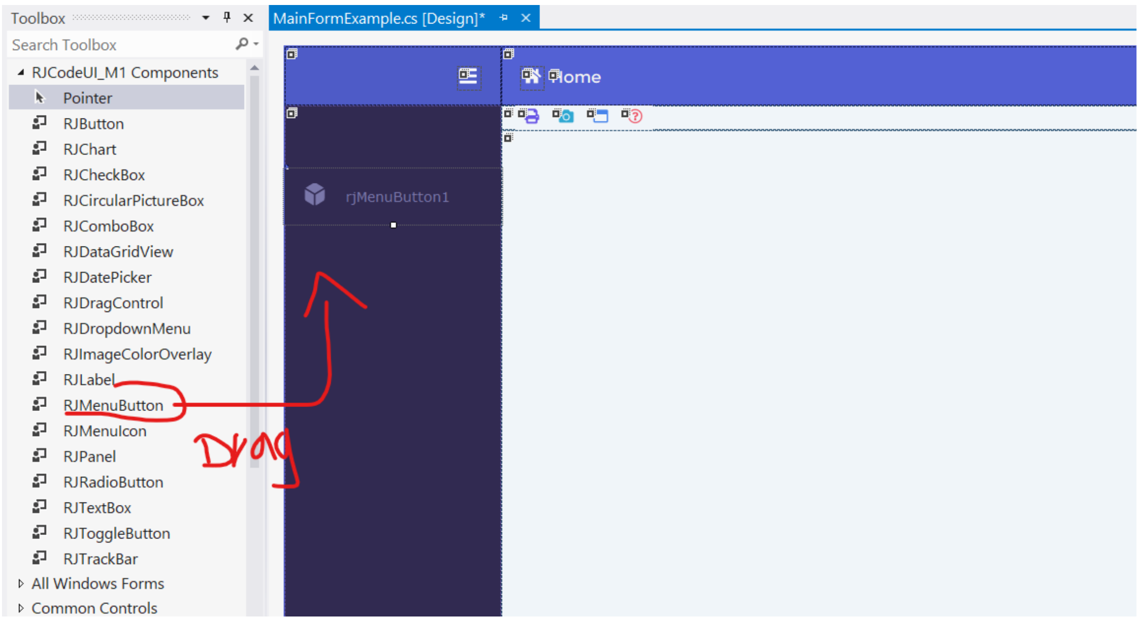
It is worth mentioning that in the project there are specific controls that you can add to the main form, these are:

- ✓ **RJMenuButton** (It is optimized only to be used in the side menu of the main form).
- ✓ **RJMenuIcon** (By default it is optimized to be used in the title bar of the main form, you can change that by setting the BackIcon property to true).
- ✓ **RJDropdownMenu** (Drop-down menu that you can associate with the above controls: RJMenuButton and RJMenuIcon, no need to code to display).
- ✓ **RJLabel**(You can set the Style property to BarCaption or BarText and use it in the title bar of the main form.)

You can add other controls, for example RJCircularPictureBox to display the photo of the logged in user in the application.

Add a menu button (RJMenuButton)

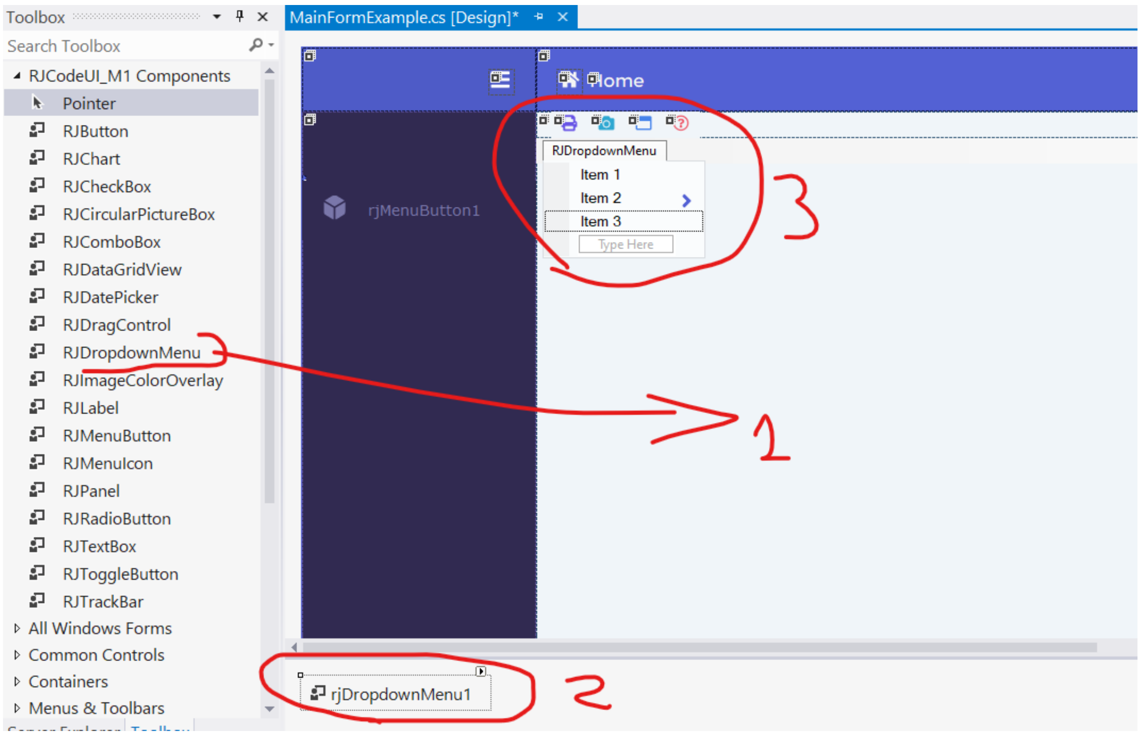
Drag control RJMenuButton from the toolbox to the **main form side menu panel** as shown in the image.



For now the control behaves like a normal menu button, you can turn it into a dropdown menu button, by adding the RJDropdownMenu control and associating it with the menu button, as demonstrated below:

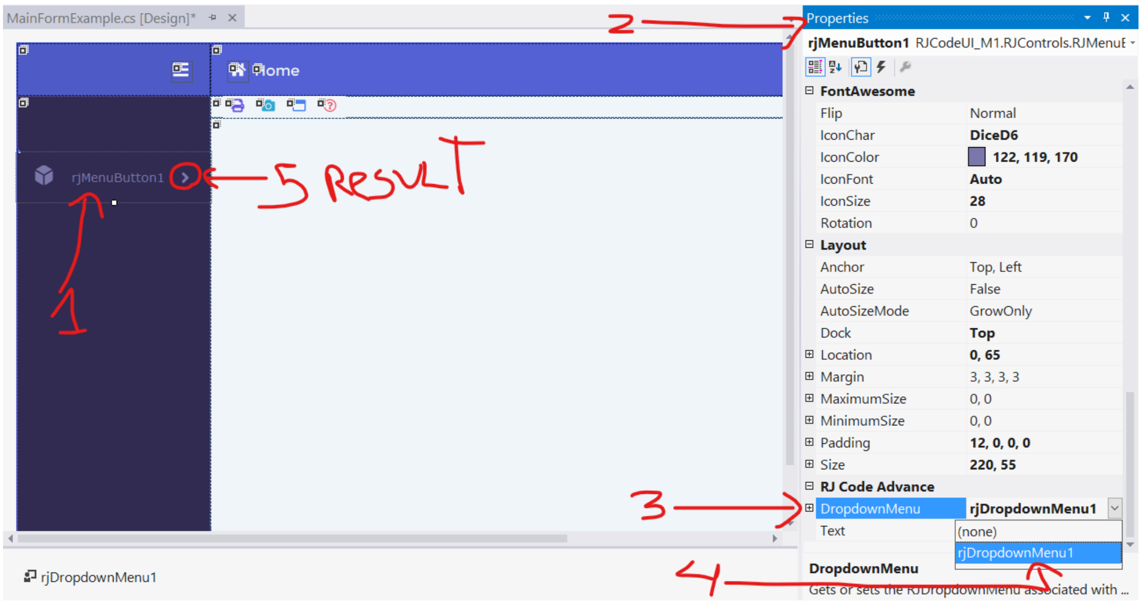
Add a dropdown menu (RJDropdownMenu)

1) **Drag control RJDropdownMenu** from the toolbox to the form and add the elements and sub-elements you want.



*It is important to remember that the **RJDropdownMenu control inherits from the ContextMenuStrip control**, this control is a component. Components **generally do not have a visual representation and are not children of the form**. Therefore, adding a component to the form **places them at the bottom of the form designer workspace**.*

2) **Associate control RJDropdownMenu** with the **RJMenuButton** or RJMenuIcon control, to do this, do the following: **Select the menu button**-> Go to the **properties**-> Locate the **DropdownMenu property ->** Finally **select the drop-down menu that you want to associate.**

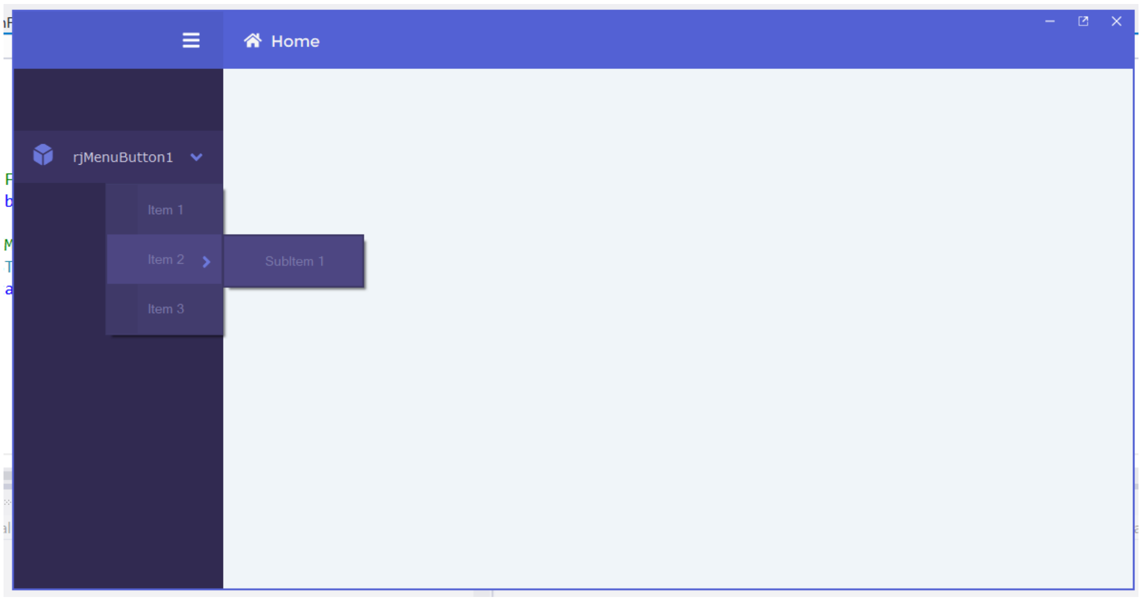


rjMenuButton1 RJCodeUI_M1.RJControls.RJMenuB

FontAwesome	
Flip	Normal
IconChar	**DiceD6**
IconColor	■ **122, 119, 170**
IconFont	**Auto**
IconSize	**28**
Rotation	0
Layout	
Anchor	Top, Left
AutoSize	False
AutoSizeMode	GrowOnly
Dock	**Top**
Location	**0, 65**
Margin	3, 3, 3, 3
MaximumSize	0, 0
MinimumSize	0, 0
Padding	**12, 0, 0, 0**
Size	**220, 55**
RJ Code Advance	
DropdownMenu	**rjDropdownMenu1**
Text	(none) / rjDropdownMenu1

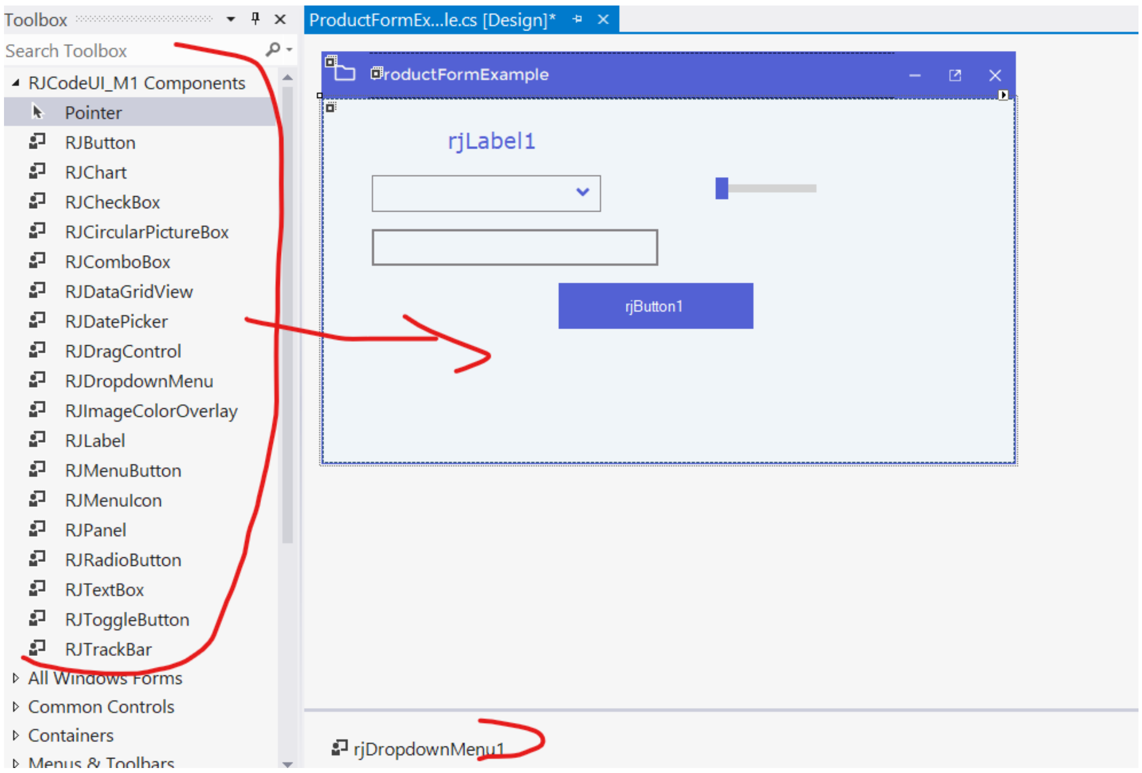
DropdownMenu
Gets or sets the RJDropdownMenu associated with ...

3) Result.

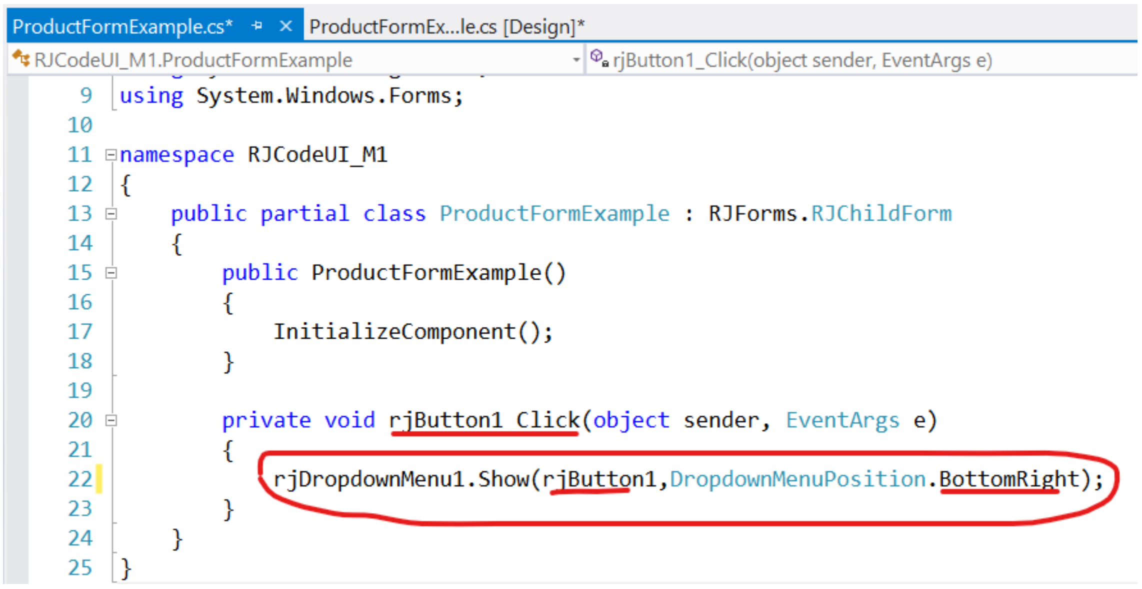


4.2. Example 2- Design a child form

As before, just **drag the controls you need onto the form and set the properties.**



You may be wondering, **how to use the drop down menu in child forms**. Well, I remind you again that the **RJDropdownMenu control inherits from the ContextMenuStrip** control, therefore **use the Show () method** of the control **in any event of any control**, as you normally would with the ContextMenuStrip control. For example in the Click event of the button rjBotton1.



ProductFormExample.cs* ProductFormEx...le.cs [Design]*

RJCodeUI_M1.ProductFormExample rjButton1_Click(object sender, EventArgs e)

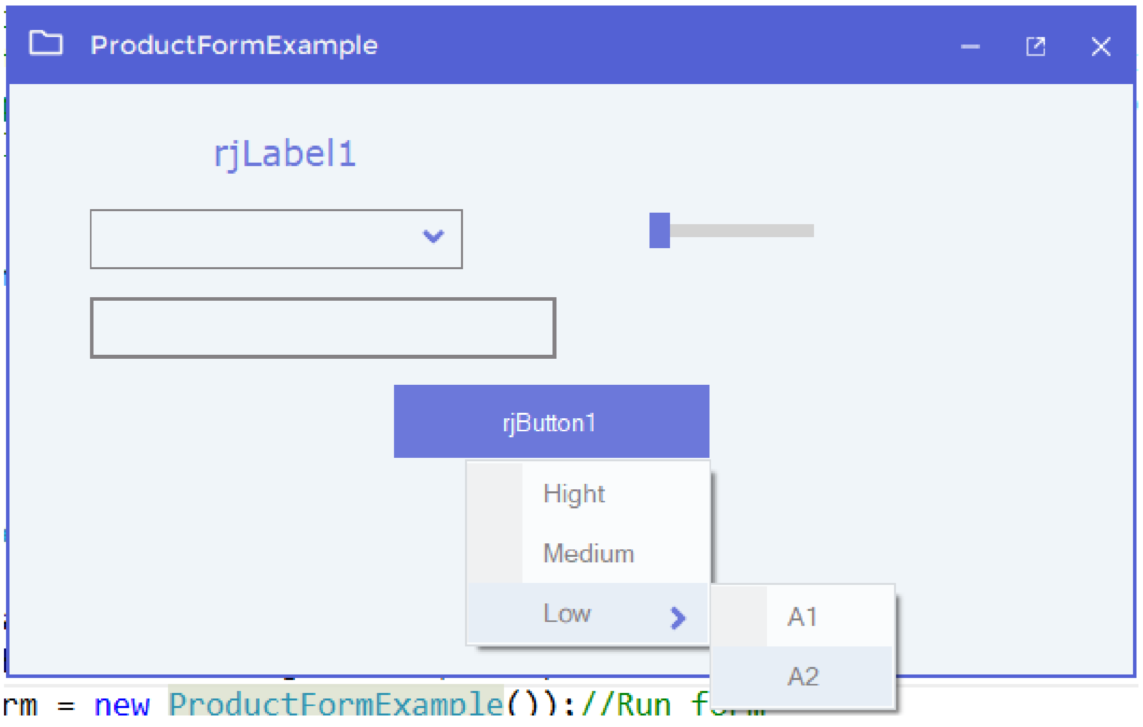
```
using System.Windows.Forms;

namespace RJCodeUI_M1
{
    public partial class ProductFormExample : RJForms.RJChildForm
    {
        public ProductFormExample()
        {
            InitializeComponent();
        }

        private void rjButton1_Click(object sender, EventArgs e)
        {
            rjDropdownMenu1.Show(rjButton1,DropdownMenuPosition.BottomRight);
        }
    }
}
```

This is the **demonstration of the new Show () method** that makes it **easy to quickly configure the position of the drop-down menu**. However, you can still use the other original Show () methods of the ContextMenuStrip control.

Result:

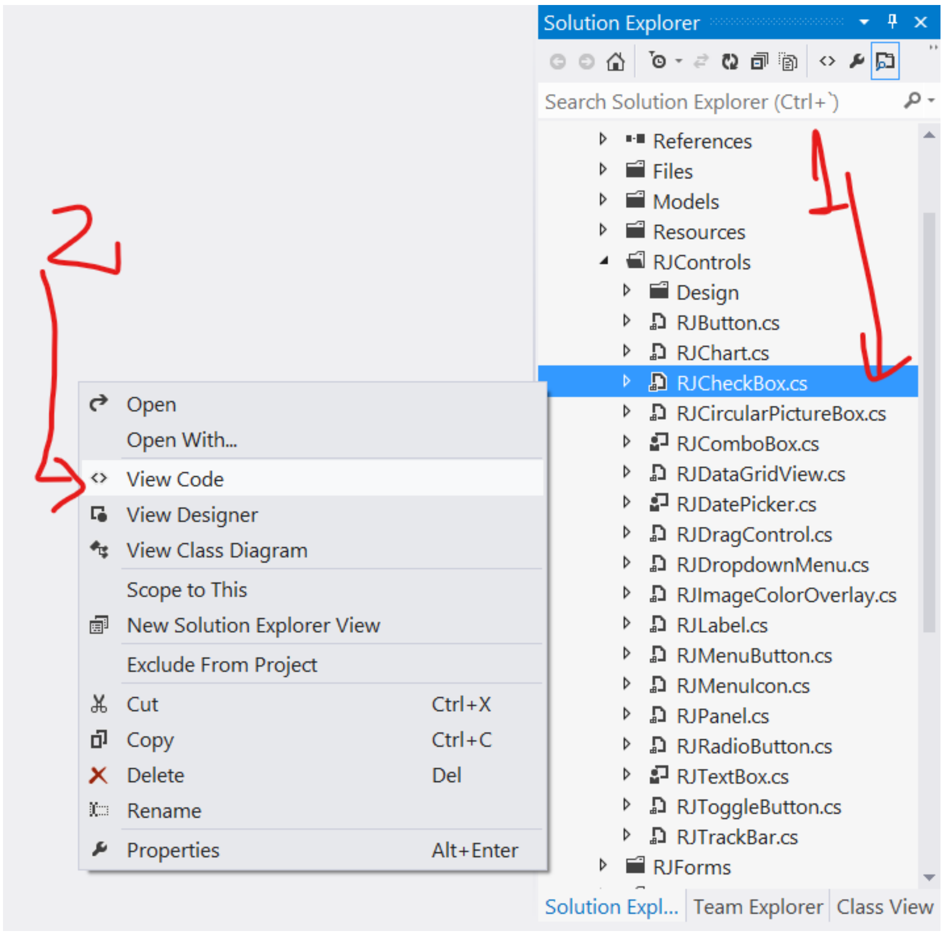


```
rm = new ProductFormExample());//Run fo
```

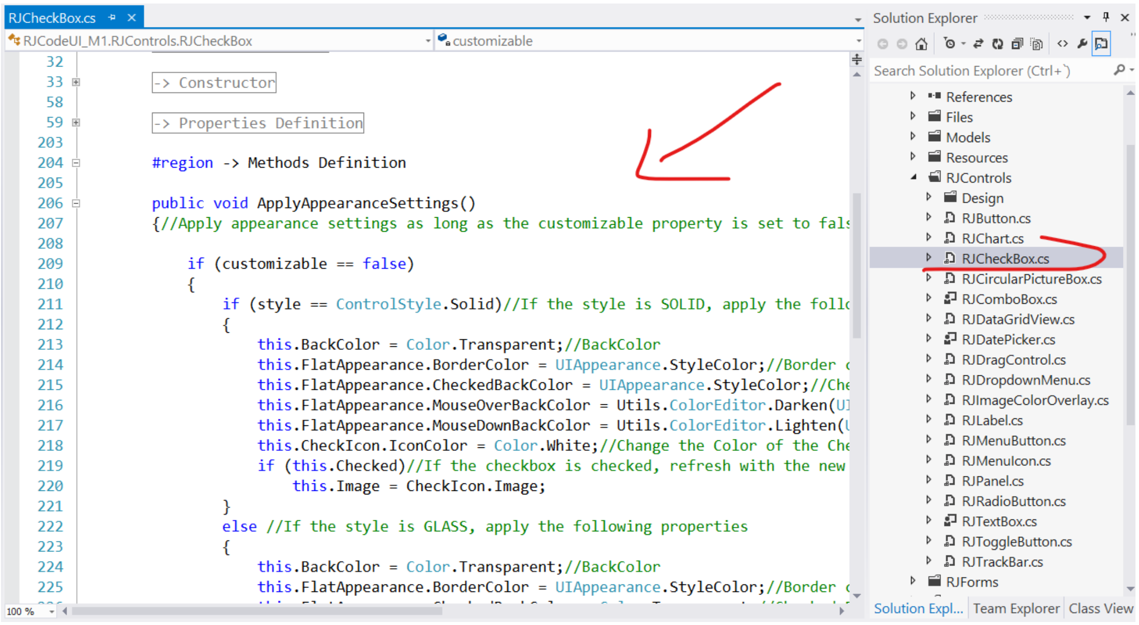
5. How to modify the project components?

You can modify or add properties and methods of any control or custom form, but simply locate (With the help of the class diagram and demo) the class-> field, property or method that you want to modify.

1) To make changes, **open the control or form code**: Right click on the class-> Select View Code.



2) Modify the source code.



```
        -> Constructor

        -> Properties Definition

        #region -> Methods Definition

        public void ApplyAppearanceSettings()
        {//Apply appearance settings as long as the customizable property is set to fals

            if (customizable == false)
            {
                if (style == ControlStyle.Solid)//If the style is SOLID, apply the follo
                {
                    this.BackColor = Color.Transparent;//BackColor
                    this.FlatAppearance.BorderColor = UIAppearance.StyleColor;//Border c
                    this.FlatAppearance.CheckedBackColor = UIAppearance.StyleColor;//Che
                    this.FlatAppearance.MouseOverBackColor = Utils.ColorEditor.Darken(UI
                    this.FlatAppearance.MouseDownBackColor = Utils.ColorEditor.Lighten(U
                    this.CheckIcon.IconColor = Color.White;//Change the Color of the Che
                    if (this.Checked)//If the checkbox is checked, refresh with the new
                        this.Image = CheckIcon.Image;
                }
                else //If the style is GLASS, apply the following properties
                {
                    this.BackColor = Color.Transparent;//BackColor
                    this.FlatAppearance.BorderColor = UIAppearance.StyleColor;//Border c
```

Every time you make changes to controls or forms, you must Compile / Build **the project** to apply the changes at design time.

Well that's it, I hope it has helped you with any of your doubts.