1a

Introduction to Visual C#

OBJECTIVES

In this lecture you will learn:

- The basics of the Visual Studio Integrated Development Environment (IDE) that assists you in writing, running and debugging your C# programs.
- Visual Studio's help features.
- Key commands contained in the IDE's menus and toolbars.
- The purpose of the various kinds of windows in the Visual Studio 2017 IDE.

OBJECTIVES

- What visual programming is and how it simplifies and speeds program development.
- To create, compile and execute a simple C# program that displays text and an image using the Visual Studio IDE and the technique of visual programming.

Outline

2.1 Introduction

- **Visual Studio** is Microsoft's Integrated Development Environment (IDE) for various .NET programming languages.
- Visual Studio allows you to drag and drop predefined components into place- a technique called **visual programming**.

2.2 Overview of the Visual Studio IDE

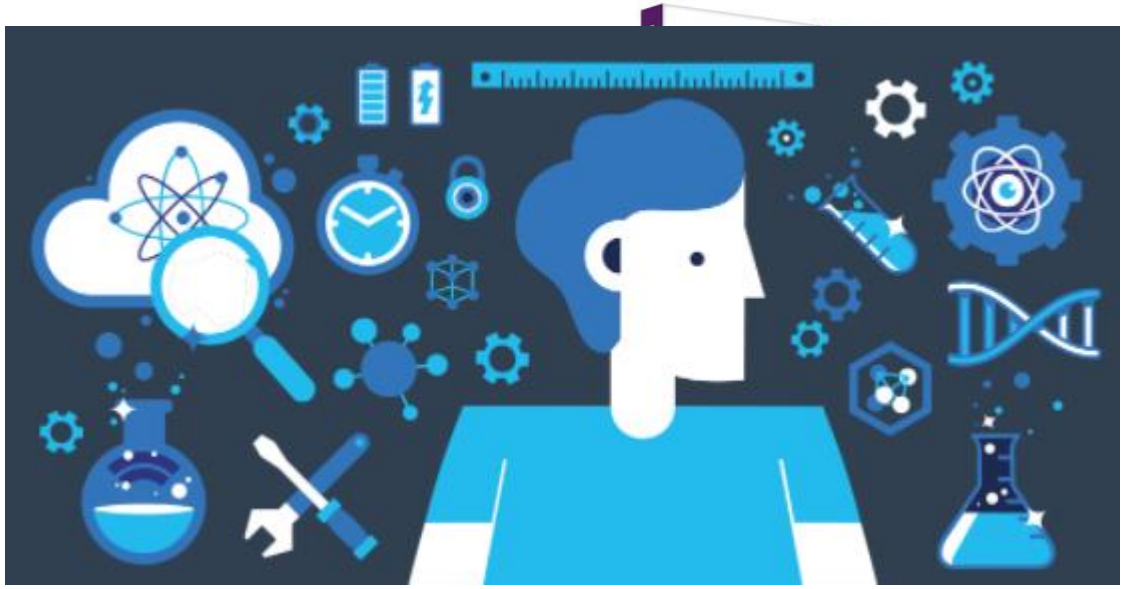
- The following examples use *Microsoft Visual C# 2017 Express Edition* .
- Full versions of Visual Studio 2017 include support for other languages.
- An overview of **all the VS 2013 editions** is available on https://www.visualstudio.com/downloads/
- For the installation of Visual Studio 2017 study

  Installing Visual Studio 2017

2.2 Overview of the Visual Studio 2017 IDE

There are four editions of **VS 2017** for developers

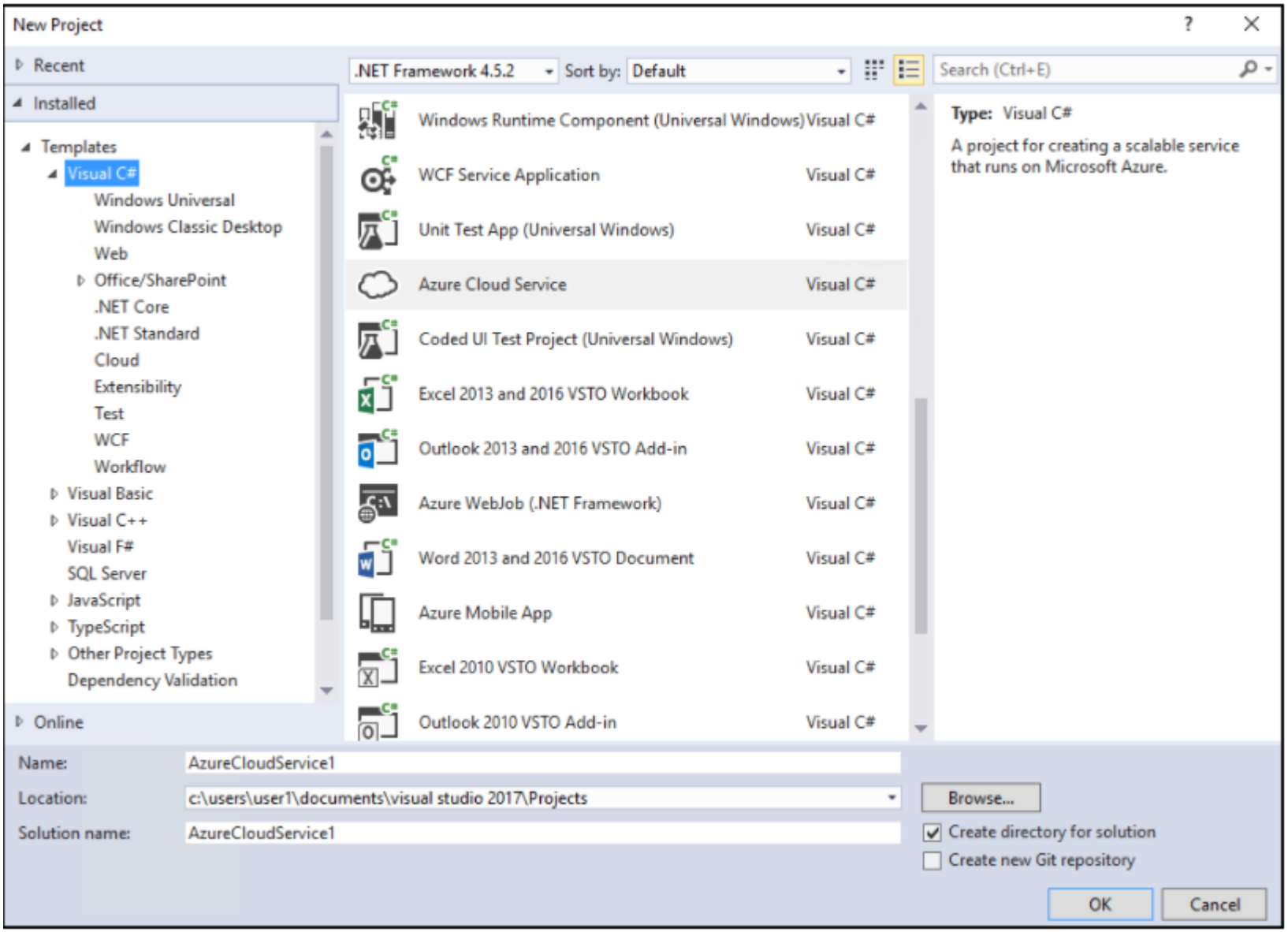
- **Community**
- **Professional**
- **Enterprise**
- **Code**



2.2 Overview of the Visual Studio 2017 IDE

Supported Features	Visual Studio Community	Visual Studio Professional	Visual Studio Enterprise
⊕ Supported Usage Scenarios	●●●○	●●●●	●●●●
Development Platform Support [2]	●●●●	●●●●	●●●●
⊕ Integrated Development Environment	●●●○	●●●○	●●●●
⊕ Advanced Debugging and Diagnostics	●●○○	●●○○	●●●●
⊕ Testing Tools	●○○○	●○○○	●●●●
⊕ Cross-platform Development	●●○○	●●○○	●●●●
⊕ Collaboration Tools and Features	●●●●	●●●●	●●●●

2.2 Visual Studio services



Windows apps

Develop apps and games to reach every device running Windows.

Windows apps

- Load a sample
- Do a tutorial
- Browse the docs

Universal apps

- Load a sample
- Do a tutorial
- Browse the docs
- Watch a video

Mobile Apps

Create native or hybrid apps targeting Android, iOS, and Windows.

Xamarin

- Load a sample
- Do a tutorial
- Browse the docs
- Watch a video

Apache Cordova

- Load a sample
- Do a tutorial
- Browse the docs
- Watch a video

Games

Design, code, and debug games with cutting-edge graphics and scripting tools.

DirectX

- Load a sample
- Do a tutorial
- Browse the docs
- Watch a video

Unity

- Load a sample
- Do a tutorial
- Browse the docs
- Watch a video

Azure apps

Build, manage, and deploy cloud scale apps to Azure with ease.

- Load a sample
- Do a tutorial
- Browse the docs
- Watch a video

Web apps

Develop modern web apps with flexibility and powerful open tools.

- Do a tutorial
- Browse the docs
- Watch a video

Office

Utilize powerful tools for all types of Office development.

- Load a sample
- Do a tutorial
- Browse the docs
- Watch a video

Extensions

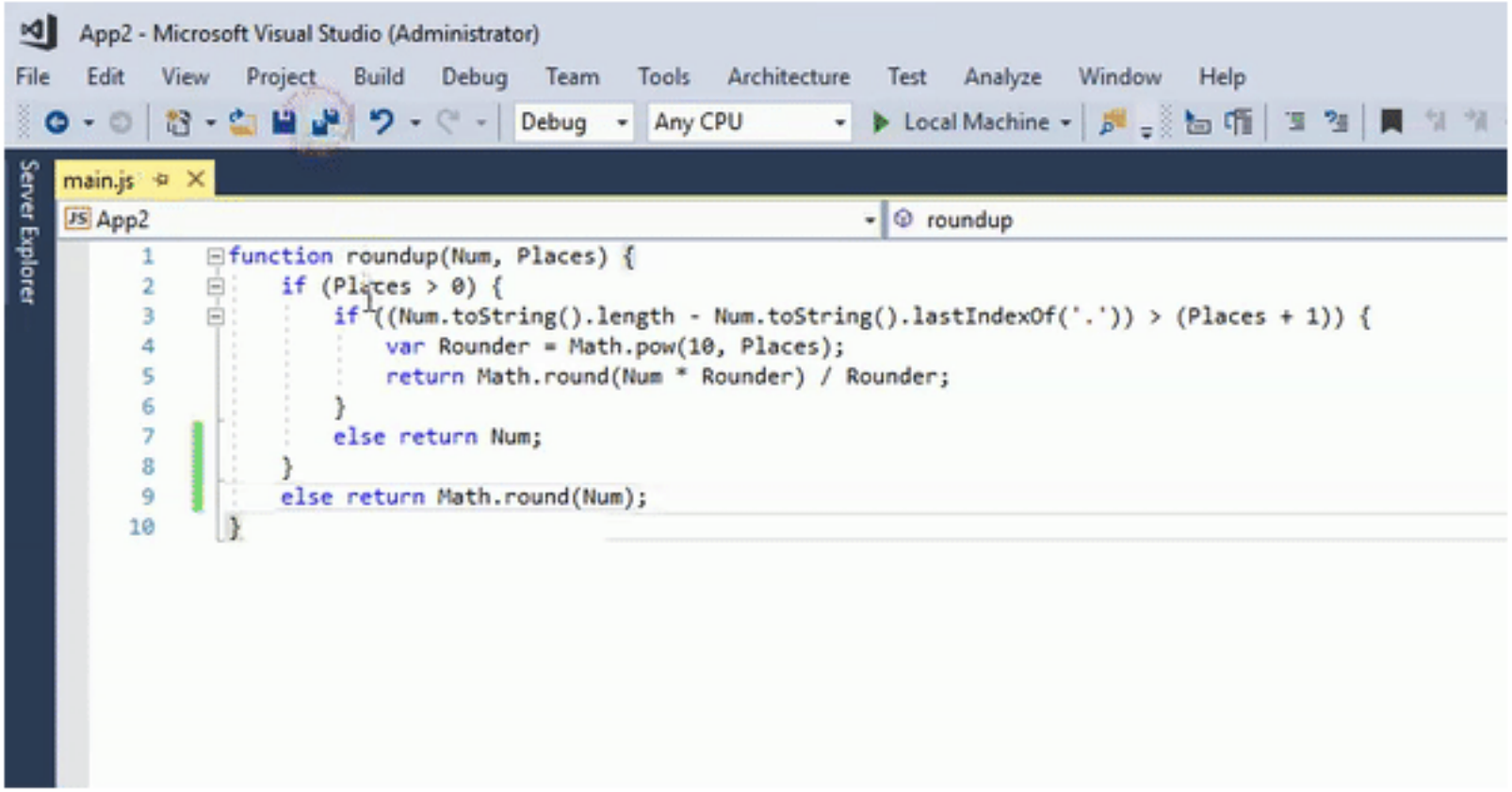
Write your own extensions for Visual Studio.

- Sample code
- Documentation

2.2 Visual Studio services

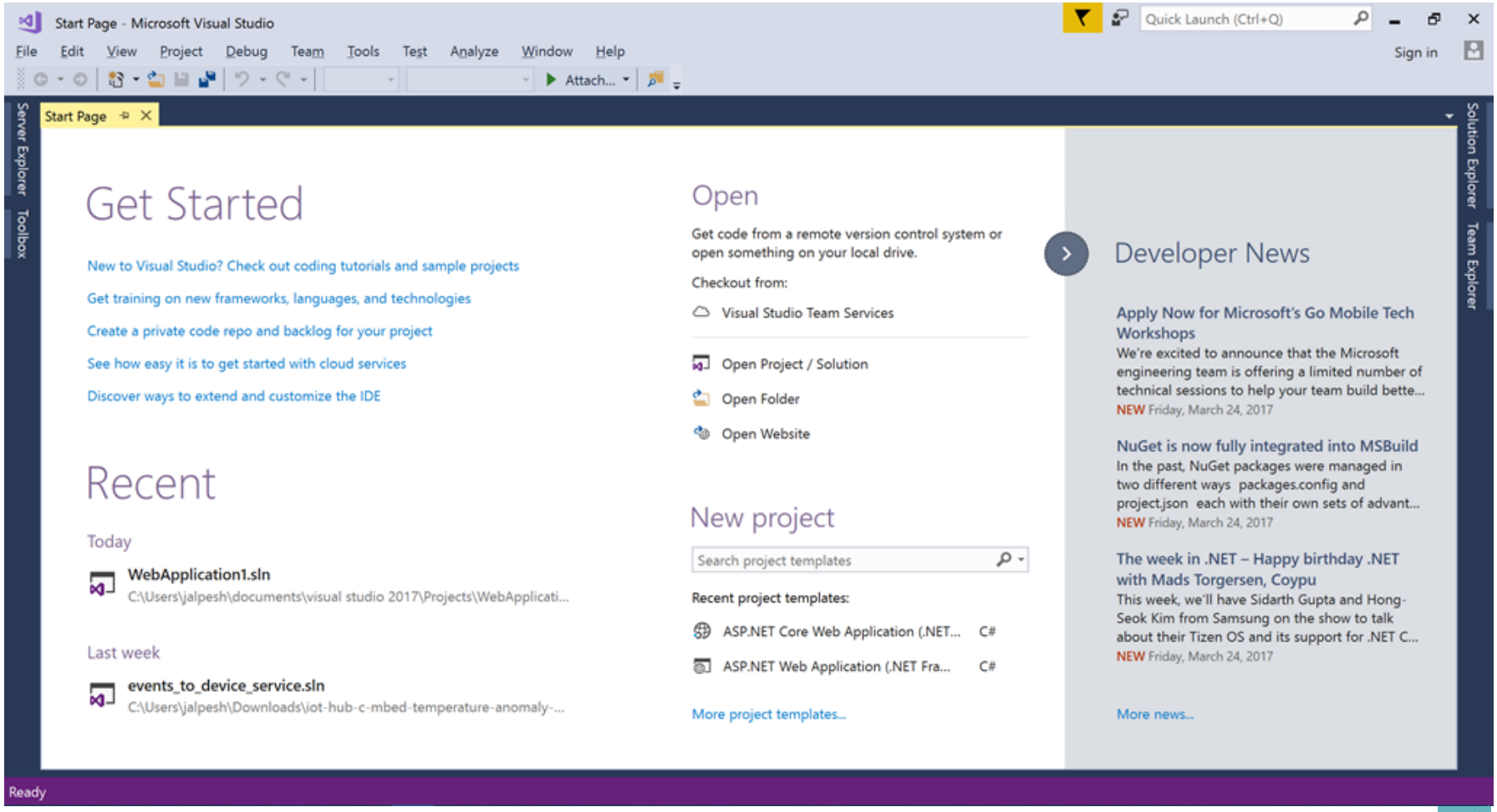
Write code in a world class editing environment.

To learn more, see Writing Code in the Code and Text Editor.



2.2 Overview of the Visual Studio IDE

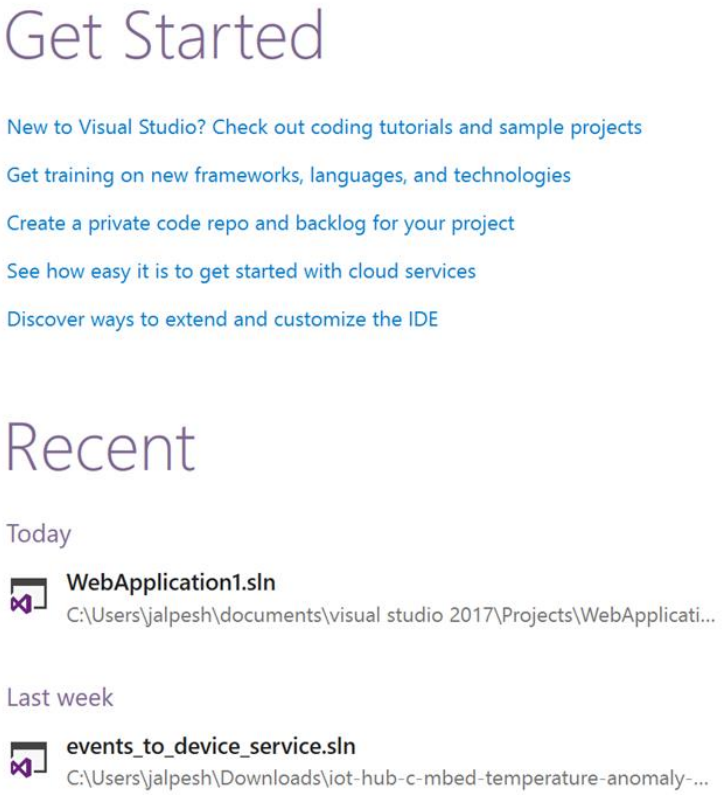
- Select **Start > All Programs > Microsoft Visual C# 2017 Community Edition** to display the **Start Page**



2.2 Overview of the Visual Studio IDE

Get Started section

Contains how you can get started with Visual Studio. It contains a variety of links including how to get started with Visual studio to how to extend visual studio.

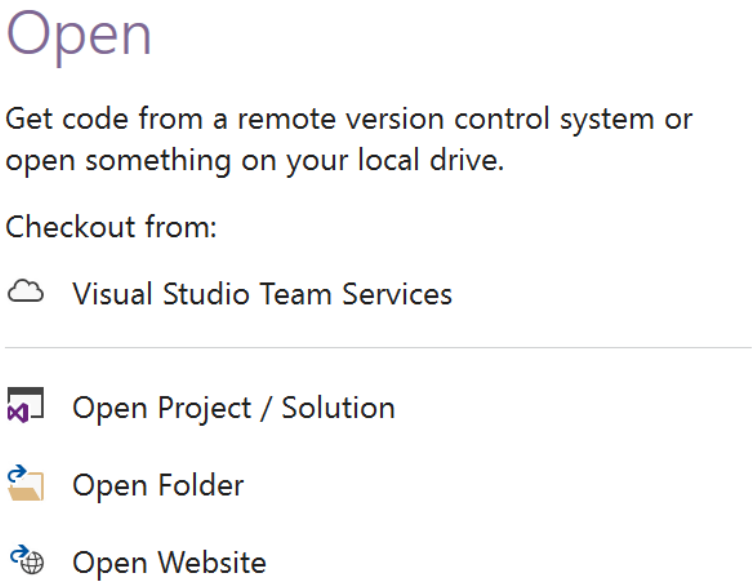


2.2 Overview of the Visual Studio IDE

Open and New Project Section:

The **open section** contains four things.

- A link to **connect Visual Studio team services** from where you can directly connect to a team services project and open it from there.
- **Open Project/Solution**: - Open project or solution works in same as the earlier version of Visual Studio. It open project or solution available on your computer.
- **Open Folder**: - It will open a folder available on your computer and then display all the code files available in that particular folder.
- **Open Web Site**: This works same as earlier of Visual Studio. It opens an ASP.NET Web Sites available on your local computer.

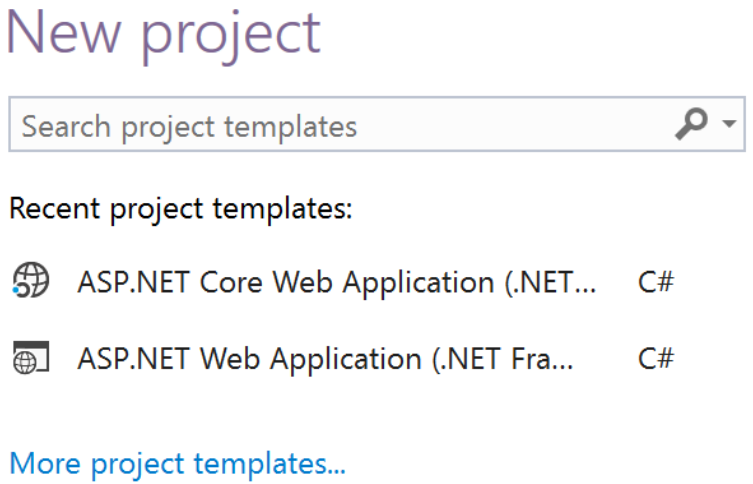


2.2 Overview of the Visual Studio IDE

Open and New Project Section:

The New Project section is brand new in Visual Studio 2017. By default, it shows the recent project's templates you recently used and also there is search box to search the templates available in Visual studio 2017.

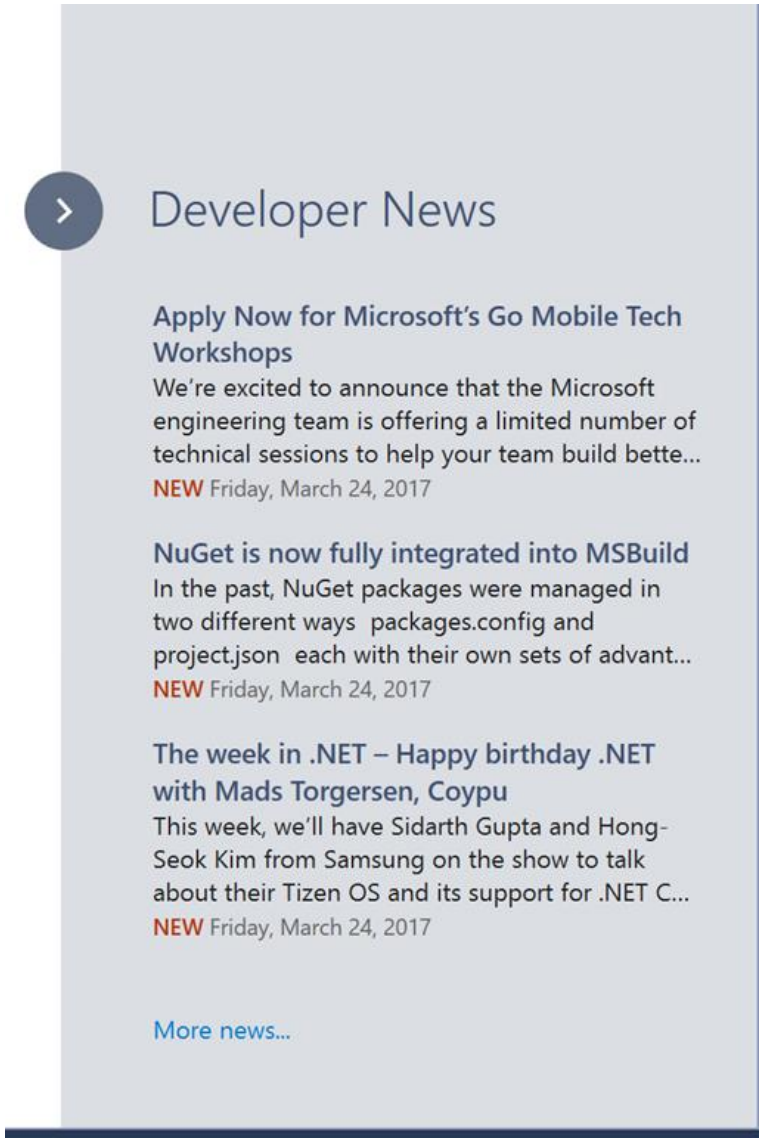
Even you search templates via putting some text in the search box.



2.2 Overview of the Visual Studio IDE

Developer News Section:

This developer new section contains new feeds from various sites.



2.3 Create a program in Visual Studio

- A **project** is a group of related files, such as the code files and any images that make up a program.
- **Solutions** contain one ore more projects.

To create a new project

1. Open Visual Studio.
2. On the menu, choose **File**, **New**, **Project**. (Use the default project values.)

As an alternative, you can **create a new project by using the Start Page**.

2.3 Create a program in Visual Studio

The **New Project dialog** displays.

- **Templates** are project types users can create in Visual C#.

1. The **New Project** dialog box shows several project templates. Choose the **Windows Universal** category under **Visual C#**
2. Choose the **Blank App (Universal Windows)** template, and then choose the **OK** button.

This creates a **new blank Universal Windows app** project **using Visual C# and XAML** as the programming languages. Wait for a bit while Visual Studio sets up the project for you. If you are prompted for any information, just accept the default values for now.

2.3 Create a program in Visual Studio

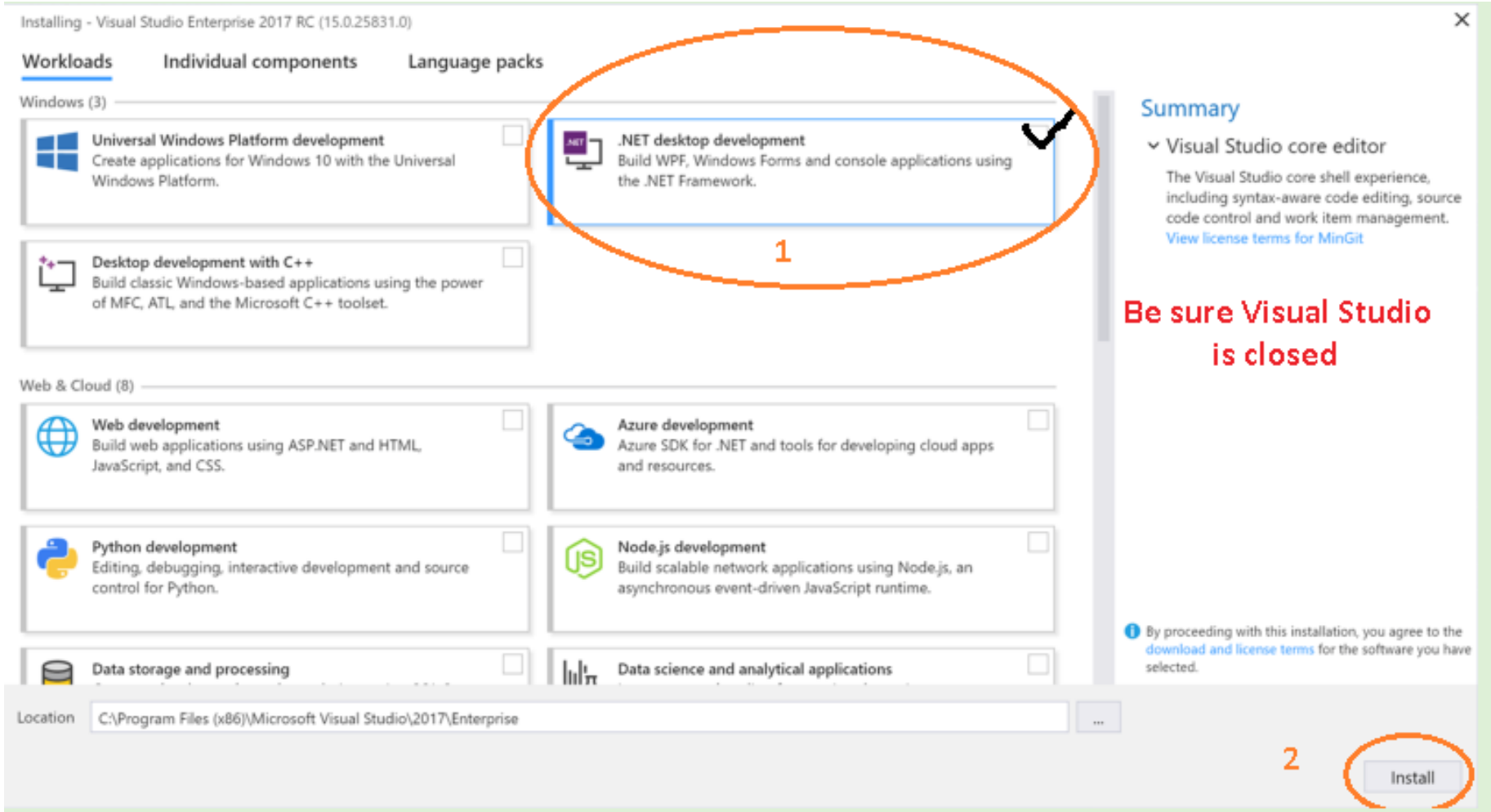
Note:

A **Windows Forms application** executes within a Windows operating system and has a **graphical user interface (GUI)**. In Visual Studio 2017 it is optionally supported.

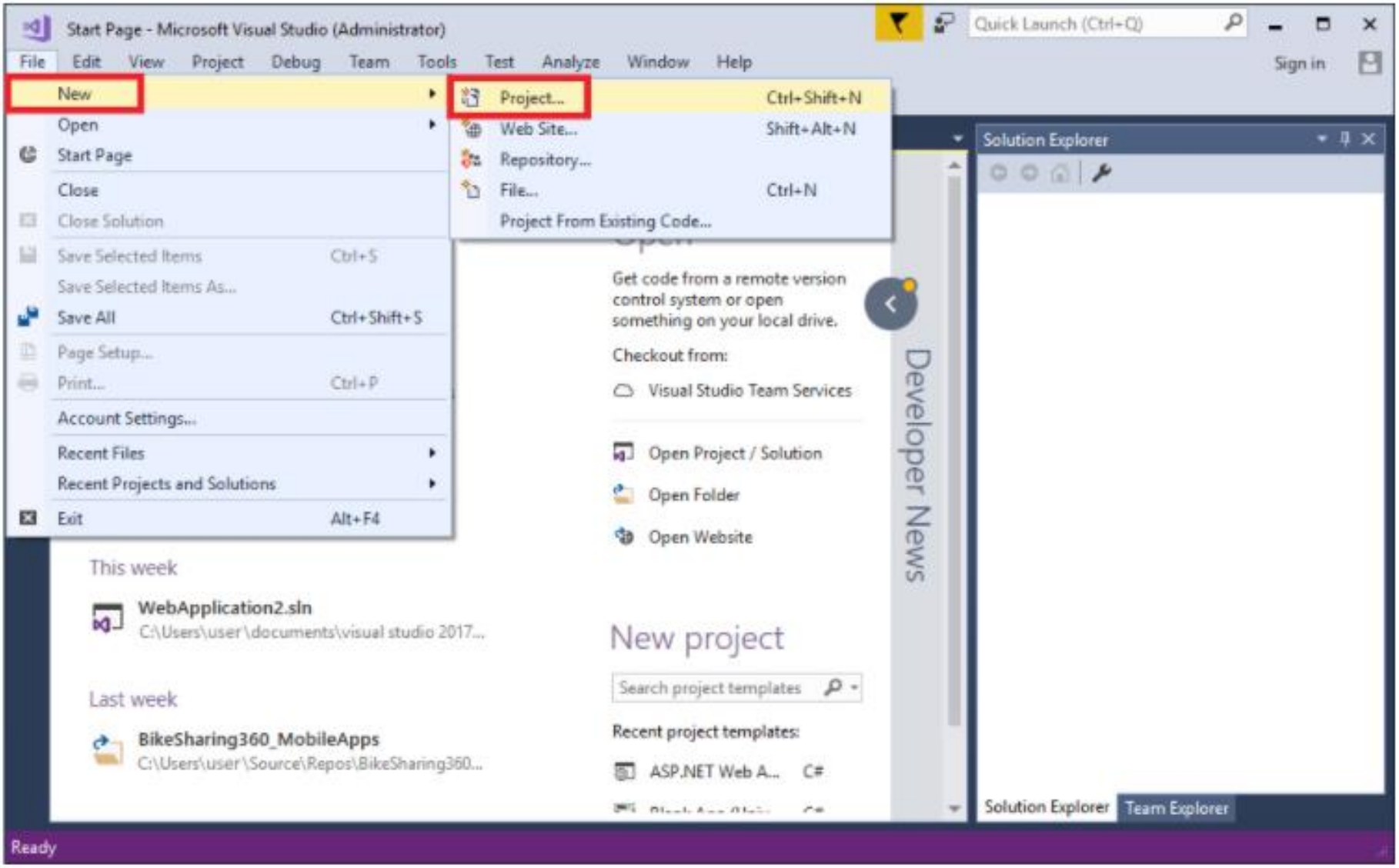
To add support for this template

1. Go to the Visual Studio Installer when you click to File->New->Project
2. **Open** Visual Studio Installer on the left Bottom
3. Then once the installer is opened close Visual Studio
4. Then on the installer screen go to the .NET Desktop Development and the click Install or Modify Button.(be sure Visual Studio is closed before install it).

2.3 Create a program in Visual Studio



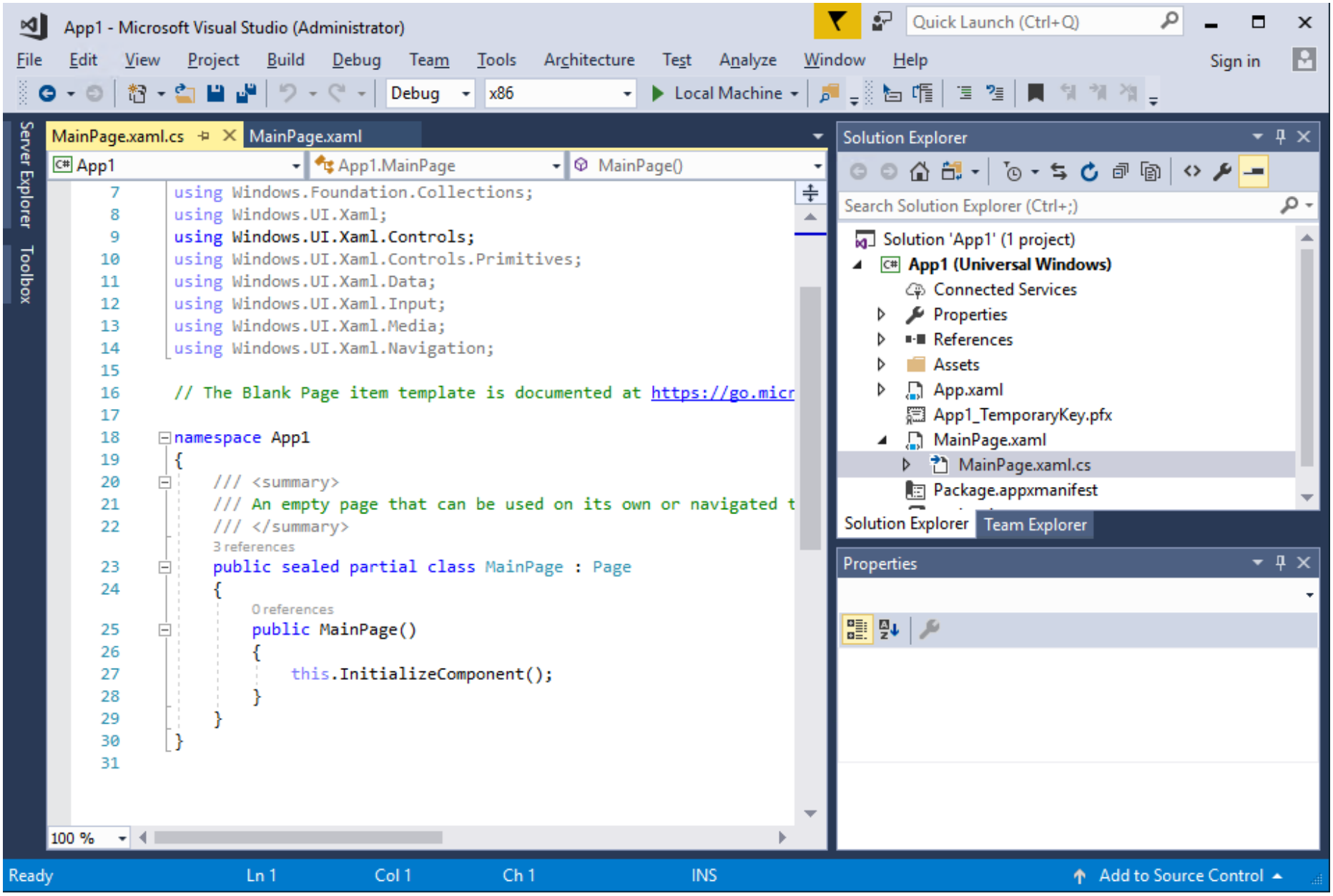
2.3 Create a program in Visual Studio



2.3 Create a program in Visual Studio

Shortly, you should see something like the following screenshot. Your **project files are listed on the right side** in a window called **Solution Explorer**.

2.3 Create a program in Visual Studio



2.3 Create a program in Visual Studio

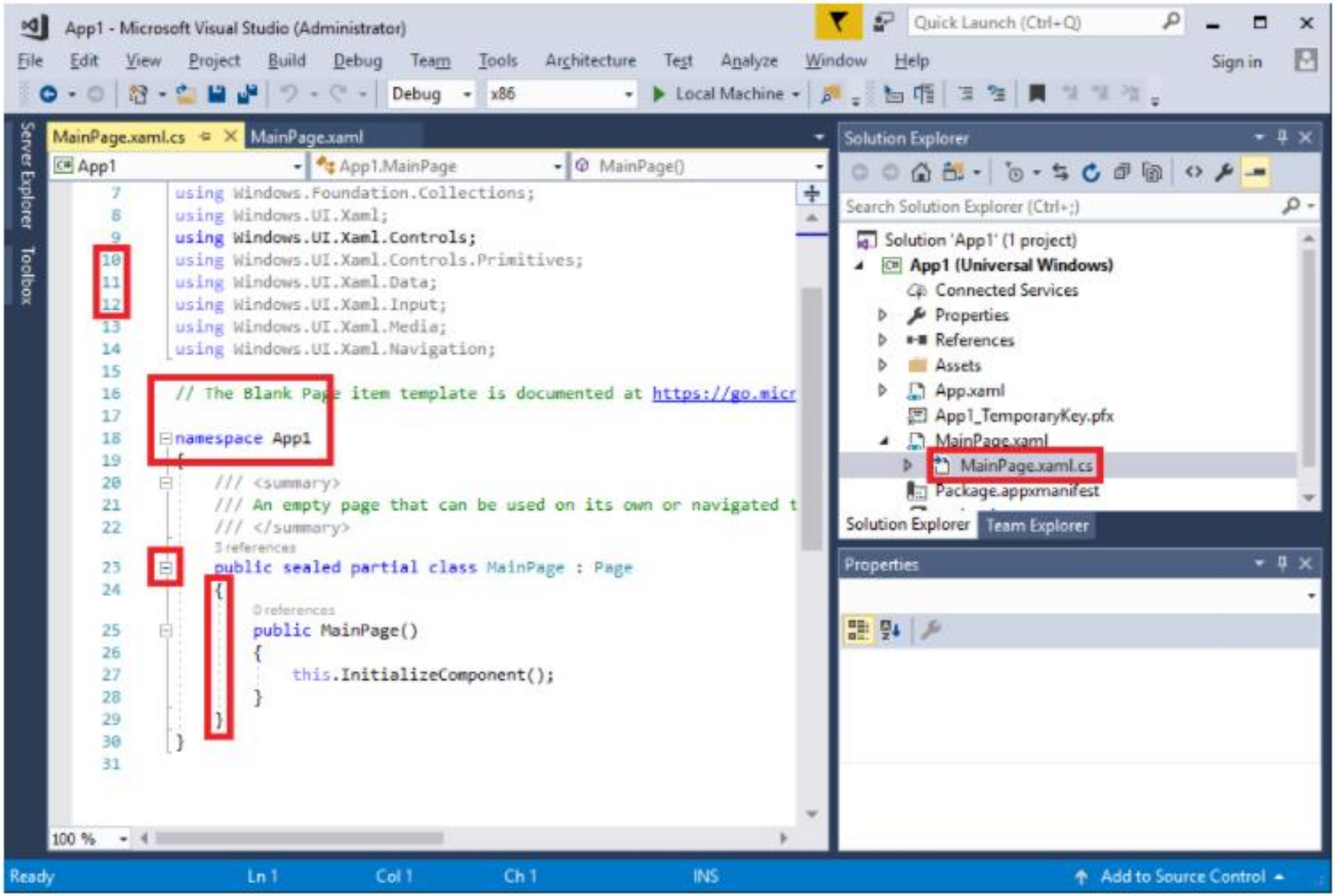
In **Solution Explorer**, choose the **little black triangle** next to the **MainPage.xaml** file to expand it, and you should see a **MainPage.xaml.cs** file underneath. Choose this file (which **contains C# code**) to open it.

The C# code in **MainPage.xaml.cs** appears **in the code editor on the left side** of the screen.

Notice that the **code syntax is automatically colorized** to indicate different types of code, such as statements or comments. In addition, **small, vertical dashed lines in the code indicate which braces match one another**, and line numbers help you locate code later.

You can choose the small, boxed minus signs to **collapse or expand code**. This code outlining feature lets you hide code you don't need, helping to minimize onscreen clutter.

2.3 Create a program in Visual Studio



2.3 Create a program in Visual Studio

Add a button to the **XAML** form **to give users a way to interact** with your app.

To do this, open the **MainPage.xaml** file.

This shows **a split view**: a **designer above**, for visually placing controls, and a **code view below**, which shows the XAML code behind the designer.

When you run the program later, what you see in the designer becomes a window that users will see, a "form," and **the underlying XAML determines what appears on the form**.

2.3 Create a program in Visual Studio

Add a button to the **XAML** form **to give users a way to interact** with your app.

To do this, open the **MainPage.xaml** file.

This shows **a split view**: a **designer above**, for visually placing controls, and a **code view below**, which shows the XAML code behind the designer.

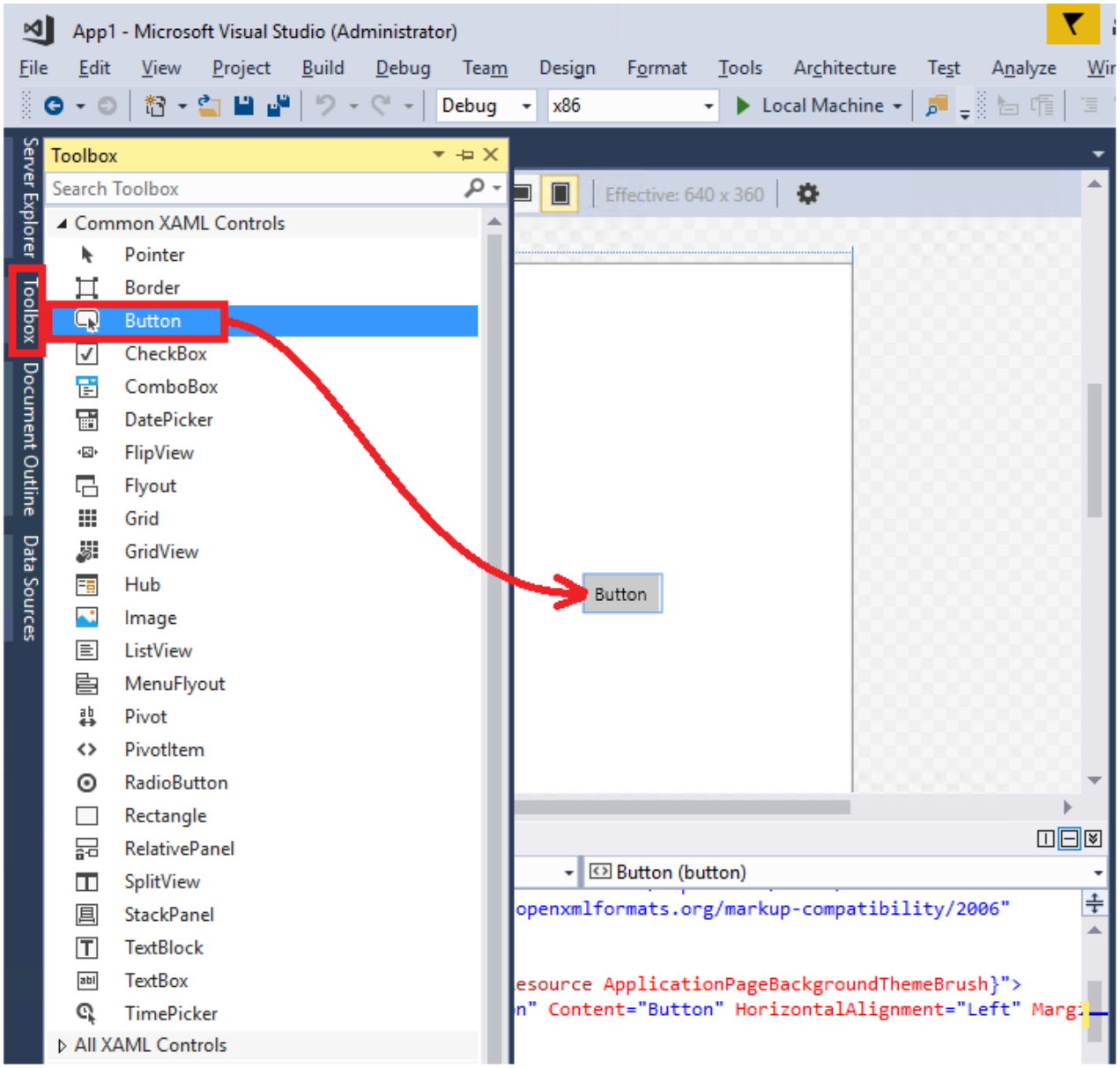
When you run the program later, what you see in the designer becomes a window that users will see, a "form," and **the underlying XAML determines what appears on the form**.

On the left side of the screen, **choose** the **Toolbox** tab to **open the Toolbox**.

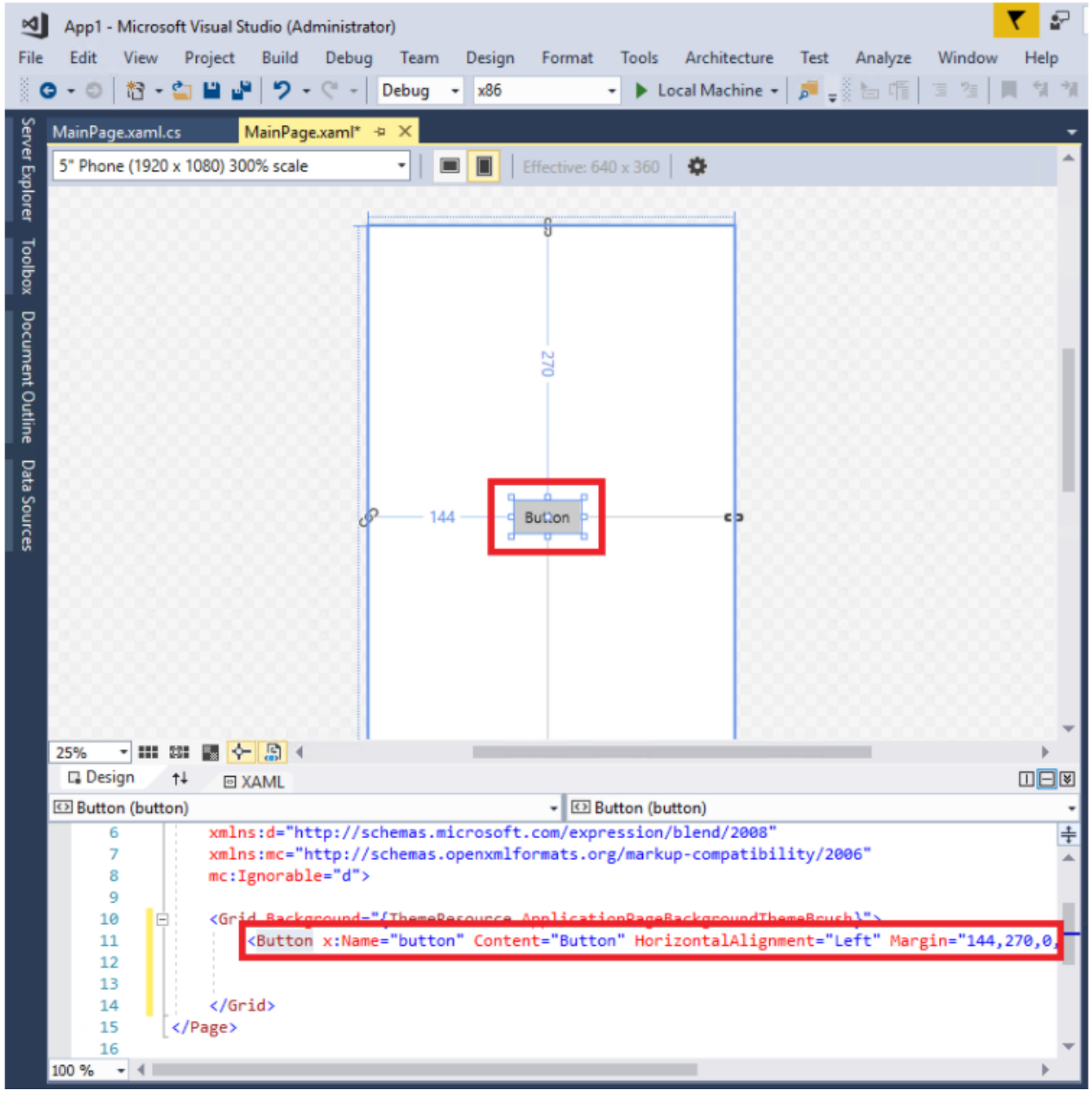
The Toolbox contains a number of **visual controls** that you can add to forms. For now, **we'll just add a button** control.

Expand the **Common XAML Controls** section and **then drag the Button control out** to about the middle of the form.

2.3 Create a program in Visual Studio

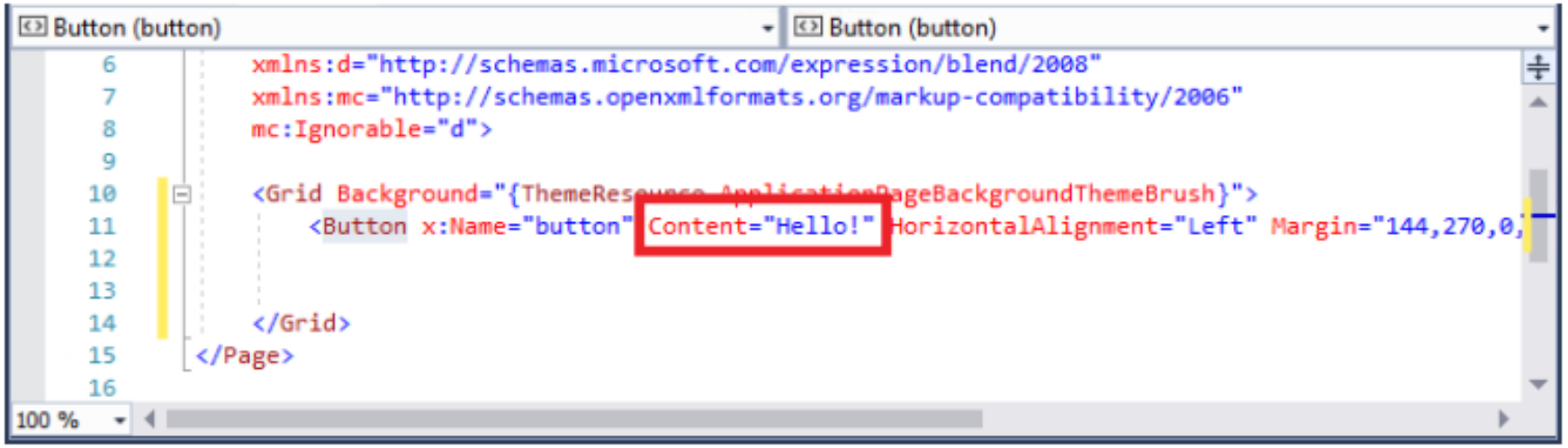


2.3 Create a program in Visual Studio



2.3 Create a program in Visual Studio

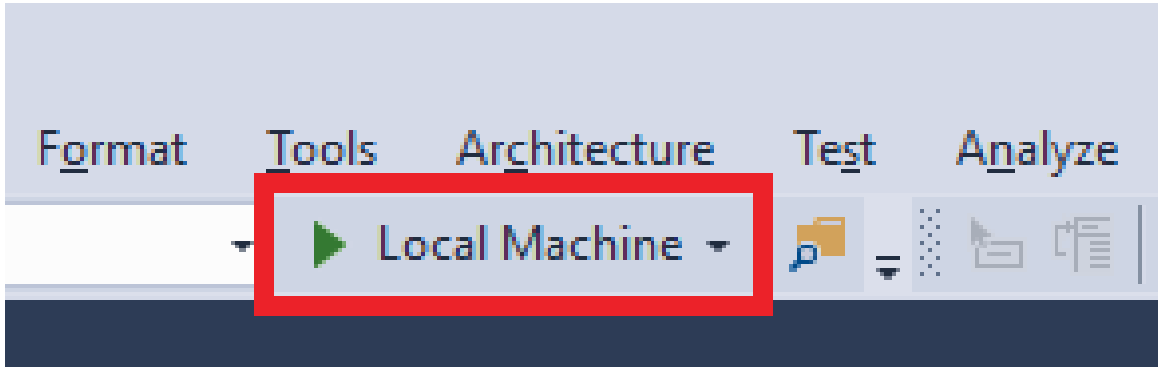
- The button is on the designer, and **its underlying code (highlighted) is automatically added** to the designer's XAML code.
- Let's change some of the **XAML** code. **Rename** the text in the button code from Button to **Hello!**.



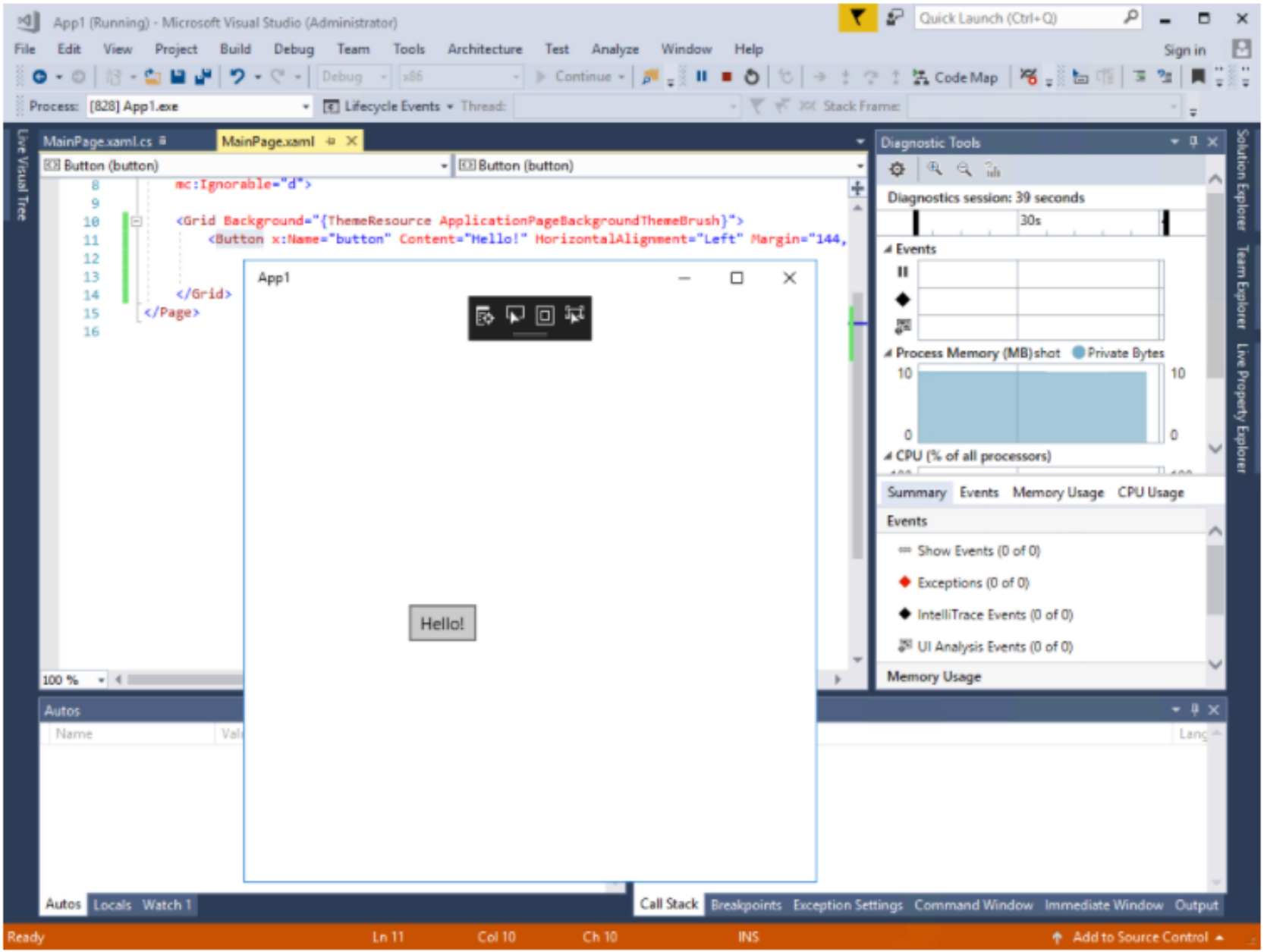
2.3 Create a program in Visual Studio

Now, **start** the app.

- You can do this by choosing the **Start (Start button)** button on the toolbar, or by choosing the **F5** key, or on the menu, choosing **Debug, Start Debugging**.
- The app begins its build process and status messages appear in the Output window. Soon, you should see the form appear with your button in it. You now have a running app!



2.3 Create a program in Visual Studio



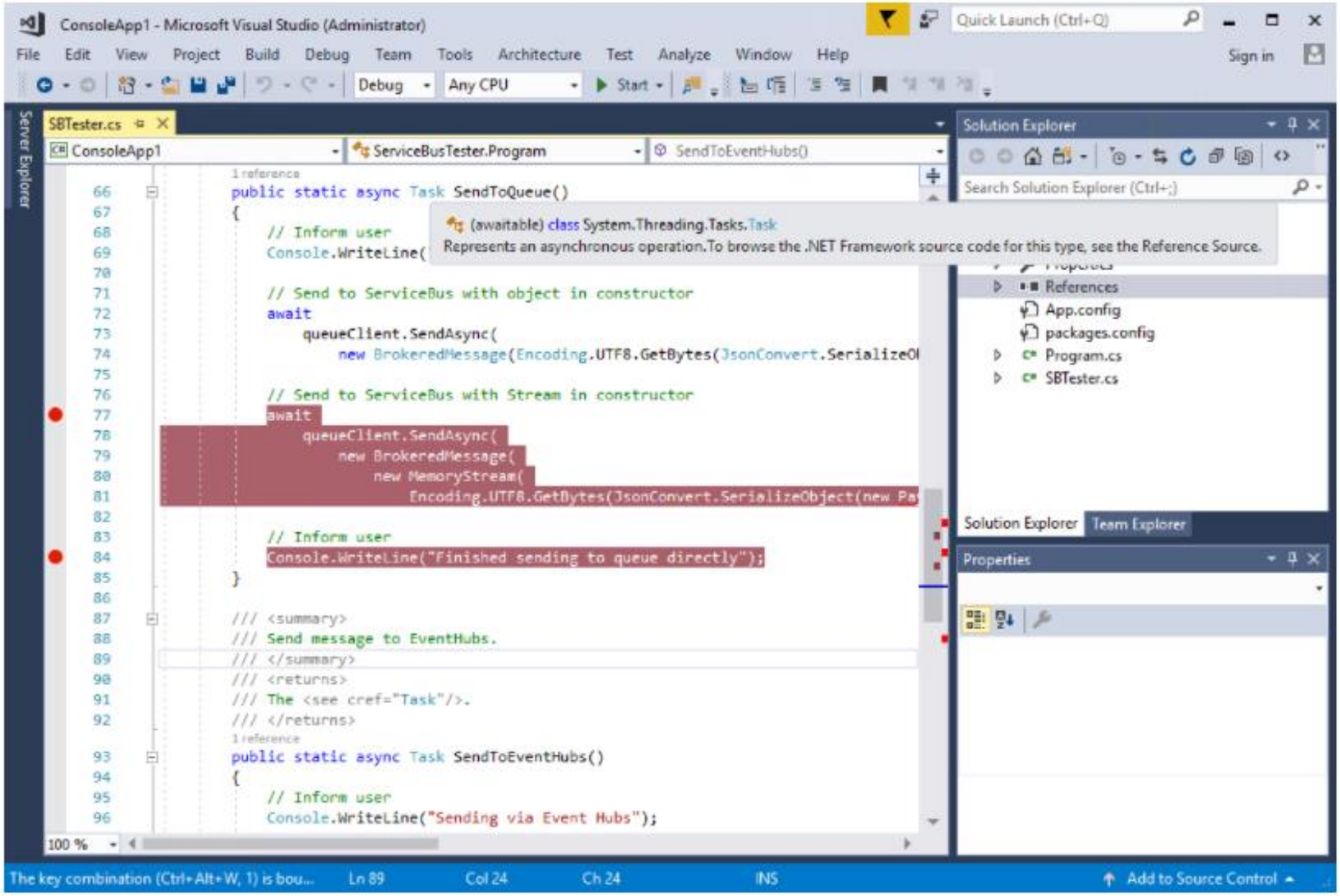
2.4 Debug, test, and improve your code

When you write code, you need to run it and **test it for bugs** and **performance**.

Visual Studio's cutting edge debugging system enables you to **debug code running in your local project**, on a remote device, or on an emulator such as the ones for Android or Windows Phone devices.

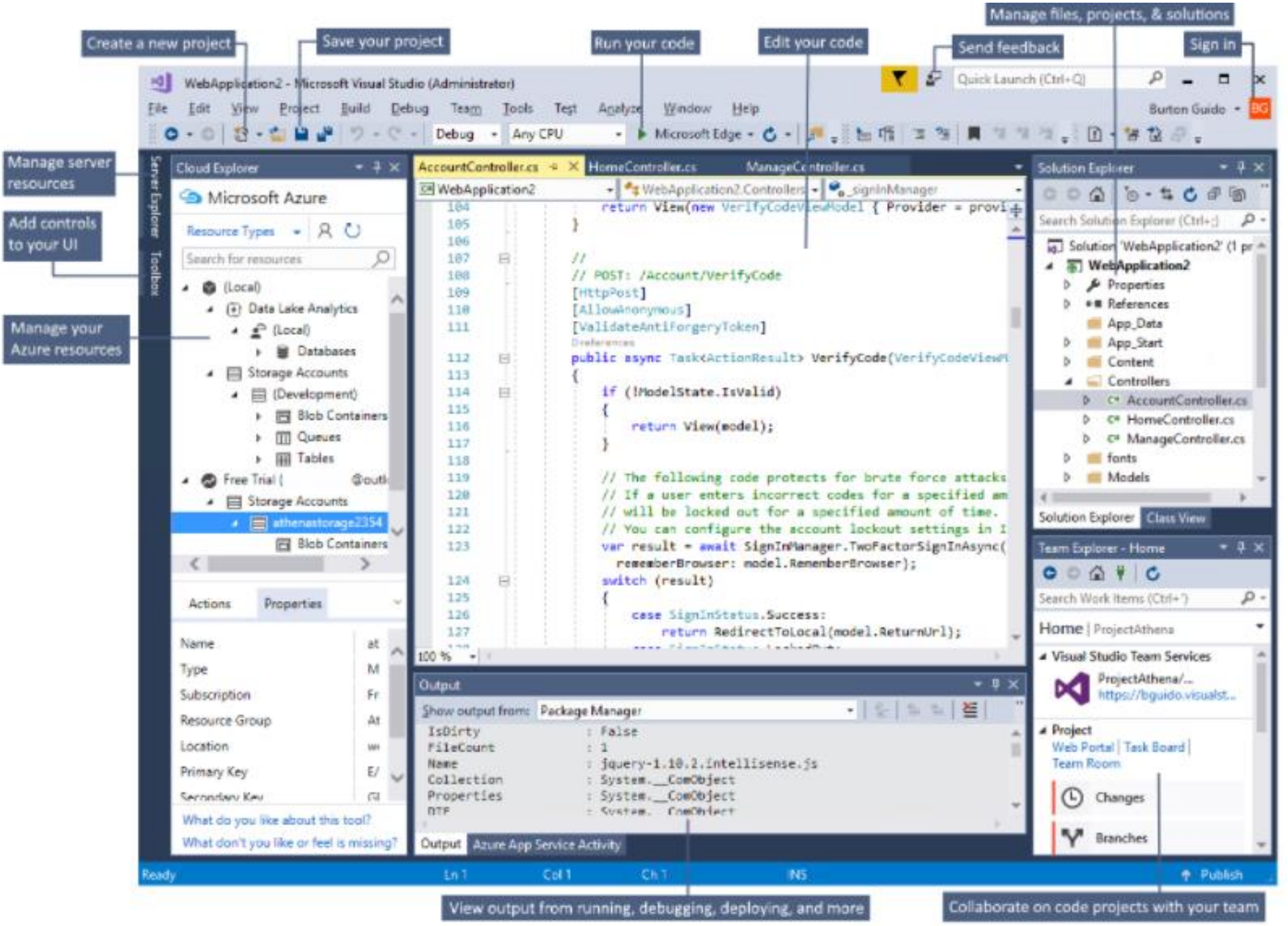
You can **step through code** one statement at a time and inspect variables as you go, you can step through multi-threaded applications, and you can set breakpoints that are only hit when a specified condition is true. You can **monitor the values of variables as the code runs**, and more. All of this can be managed in the code editor itself, so that you don't have to leave your code.

2.4 Debug, test, and improve your code



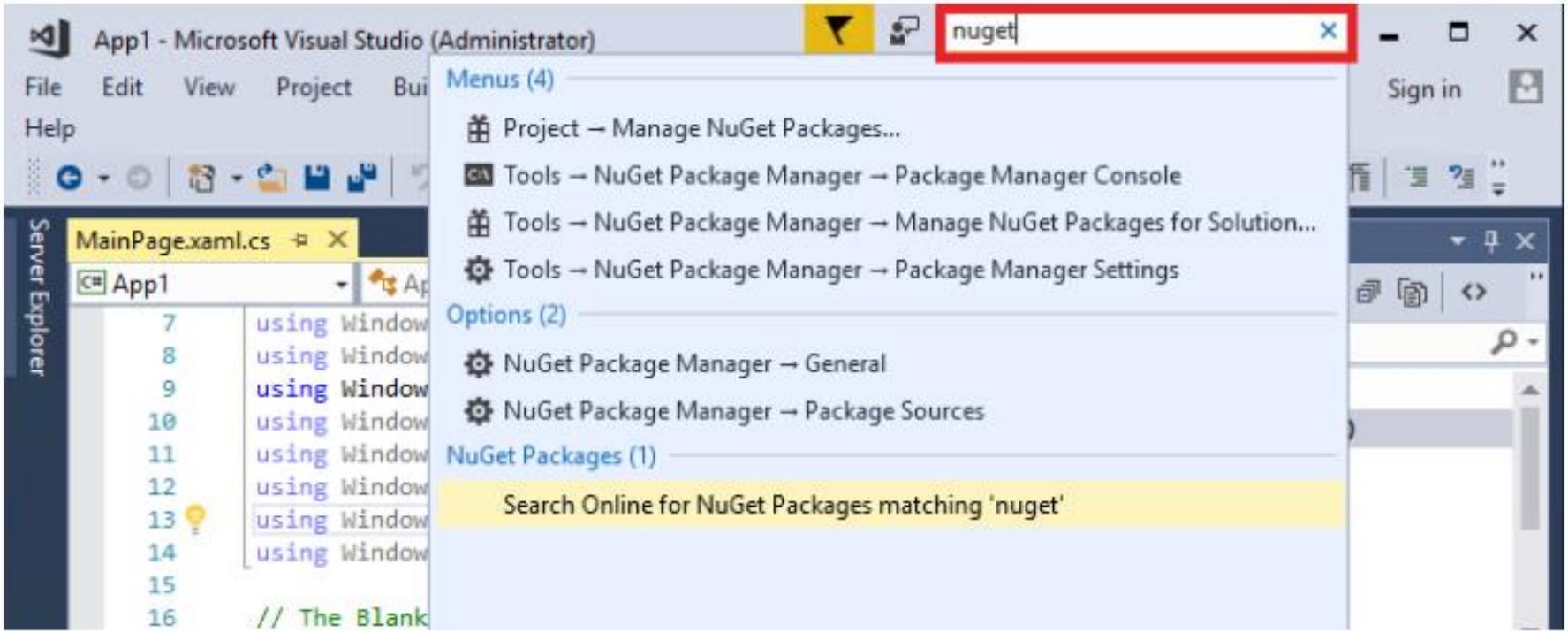
2.4 Quick tour of the IDE

- Solution Explorer lets you view, navigate, and manage your code files.
- The Editor window shows your code and enables you to edit source code and designer data.
- The Output window shows output messages from compiling, running, debugging, and more.
- Team Explorer lets you track work items and share code with others using version control technologies such as Git and Team Foundation Version Control (TFVC).
- Cloud Explorer lets you view and manage your Azure resources, such as virtual machines, tables, SQL databases, and more.


Create a new project
Save your project
Run your code
Edit your code
Send feedback
Manage files, projects, & solutions
Sign in
Manage server resources
Add controls to your UI
Manage your Azure resources
WebApplication2 - Microsoft Visual Studio (Administrator)
Cloud Explorer
Microsoft Azure
AccountController.cs
HomeController.cs
ManageController.cs
Solution Explorer
Team Explorer - Home
Output
Ready
View output from running, debugging, deploying, and more
Collaborate on code projects with your team

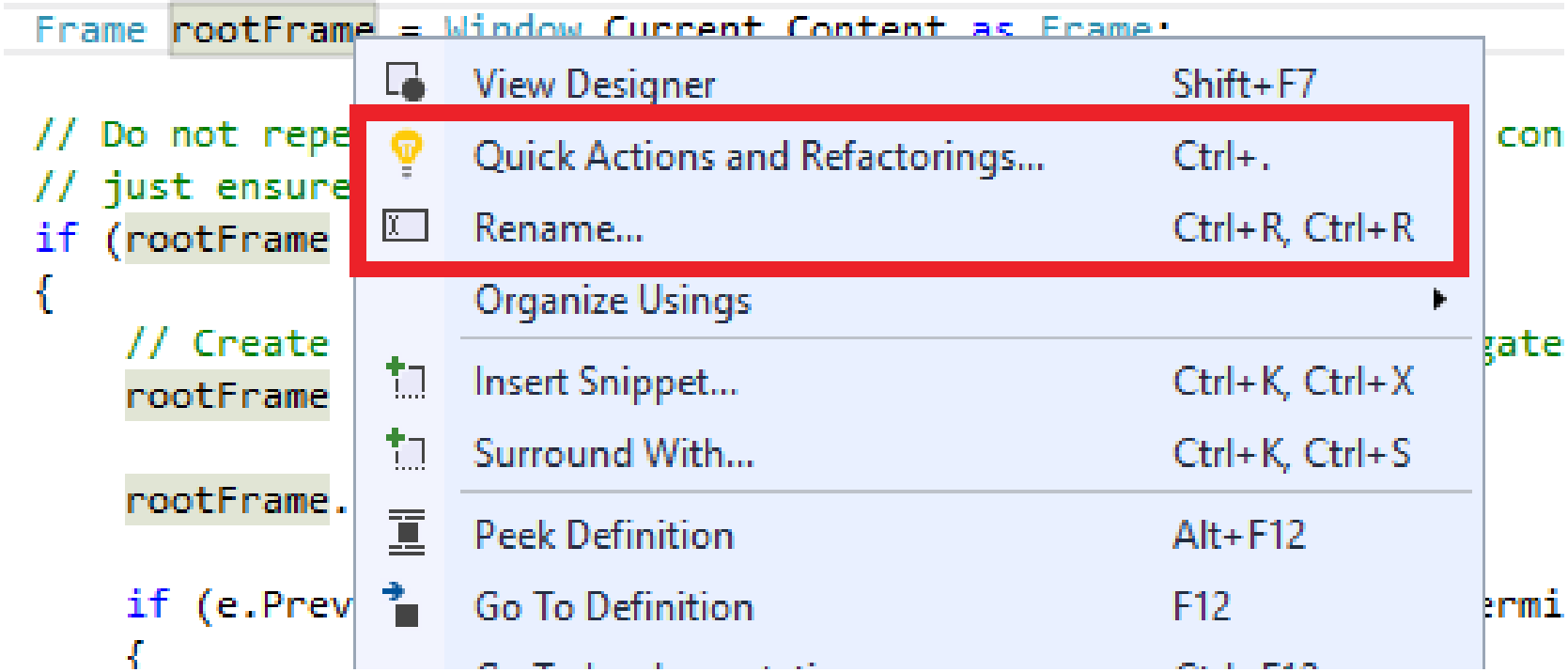
2.5 Common productivity features in Visual Studio

The **Quick Launch** search box is a great way to rapidly find what you need in Visual Studio. Just start entering in the name of whatever you are looking for, and Visual Studio lists results that take you exactly where you want to go. Quick Launch also shows links that start the Visual Studio Installer for any workload or individual component.



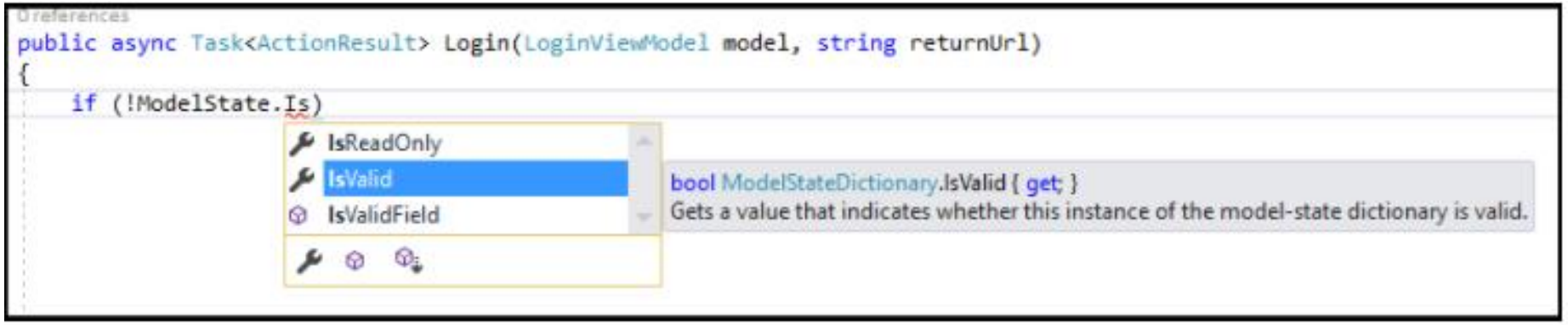
2.5 Common productivity features in Visual Studio

Refactoring includes operations such as intelligent renaming of variables, moving selected lines of code into a separate function, moving code to other locations, reordering function parameters, and more.



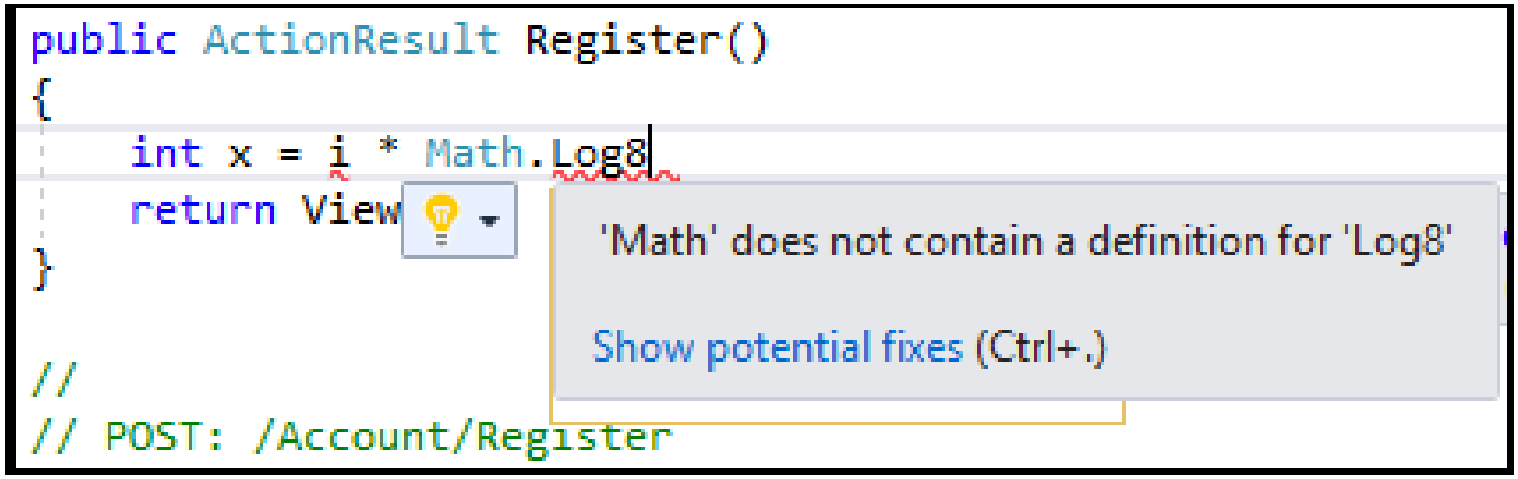
2.5 Common productivity features in Visual Studio

IntelliSense is an umbrella term for a set of popular features that display type information about your code directly in the editor and, in some cases, write small bits of code for you. It's like having basic documentation inline in the editor, which saves you from having to look up type information in a separate help window. IntelliSense features vary by language.



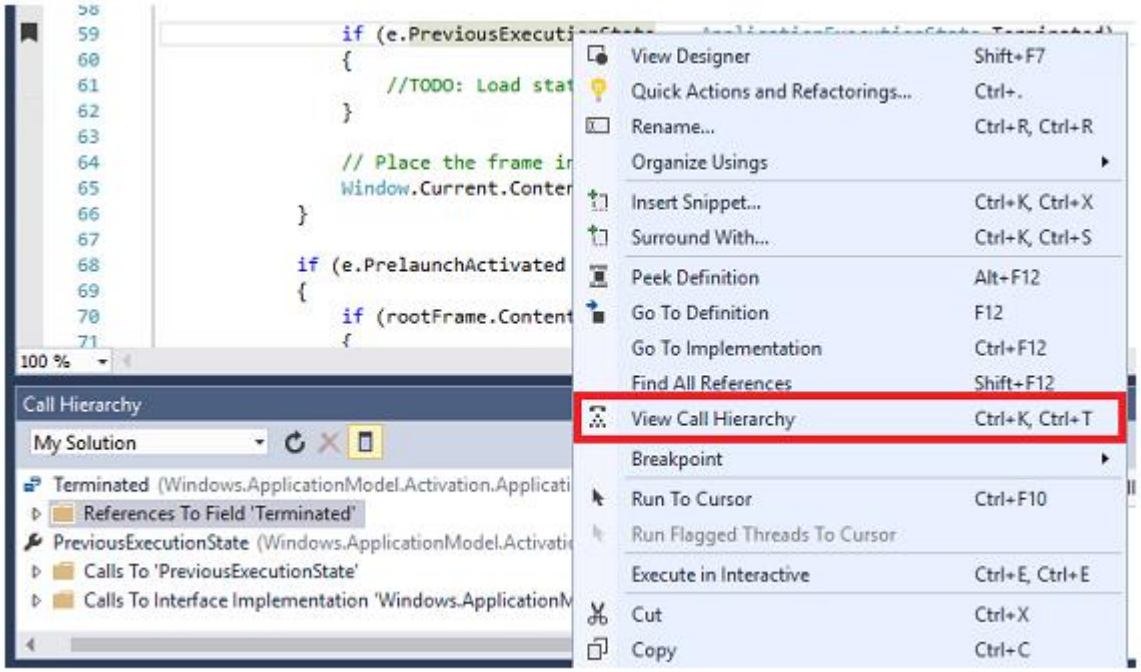
2.5 Common productivity features in Visual Studio

Squiggles are wavy red underlines that **alert you to errors** or **potential problems in your code in real time** as you type. This enables you to fix them immediately without waiting for the error to be discovered during compilation or run time. If you hover over the squiggle, you see additional information about the error. A light bulb may also appear in the left margin with suggestions for how to fix the error.



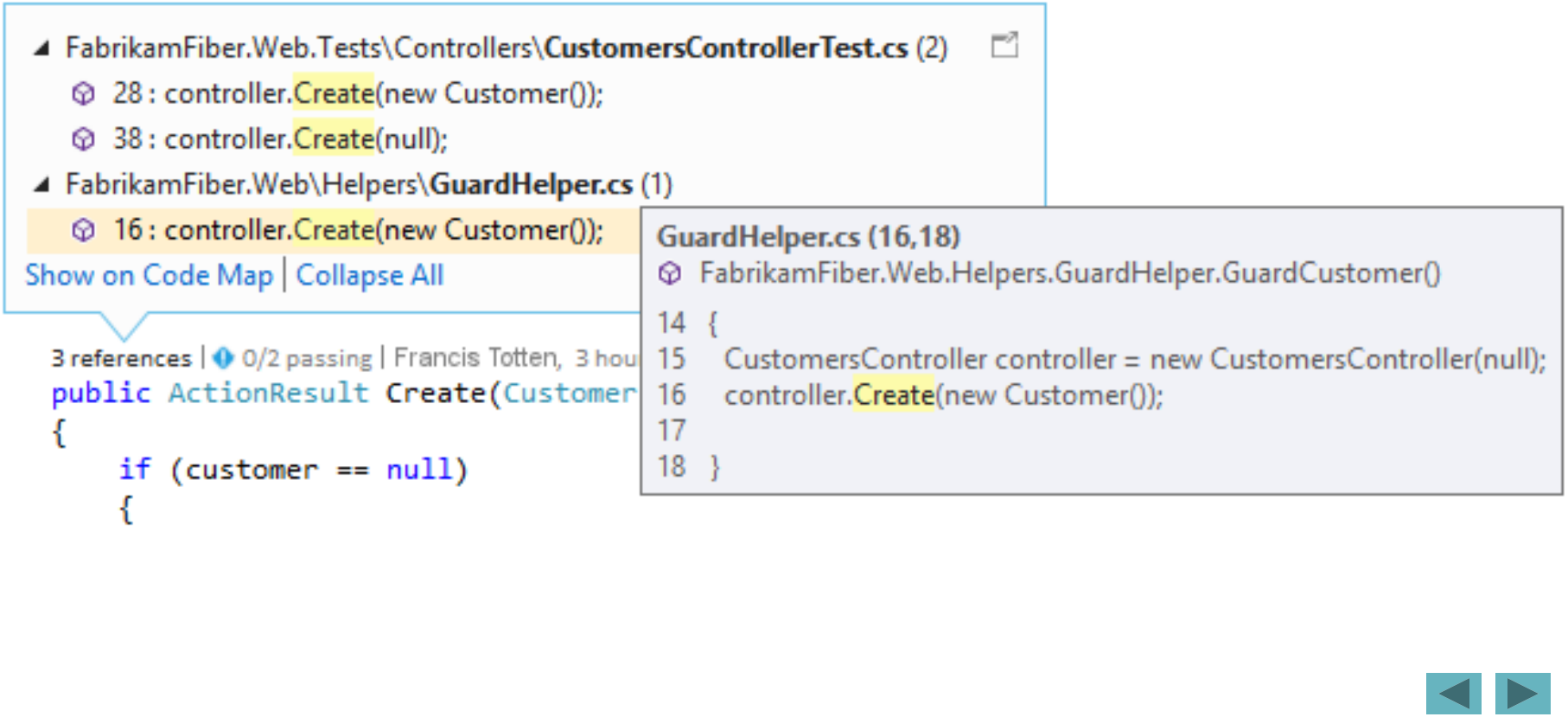
2.5 Common productivity features in Visual Studio

The **Call Hierarchy** window can be opened on the text editor context menu to show the methods that call, and are called by, the method under the caret (insertion point).



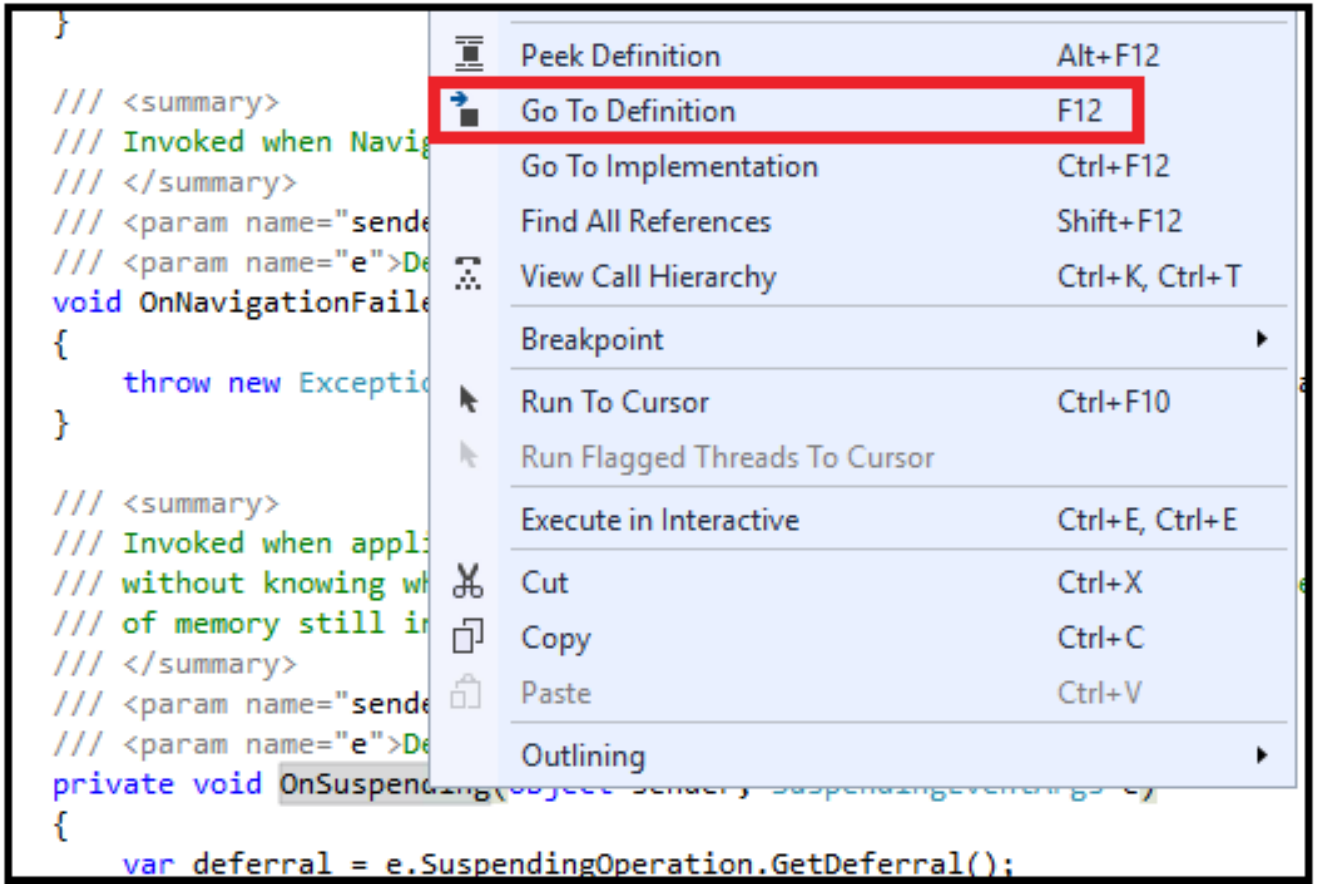
2.5 Common productivity features in Visual Studio

CodeLens enables you to **find references and changes to your code**, **linked bugs**, **work items**, **code reviews**, and **unit tests**, all without leaving the editor.



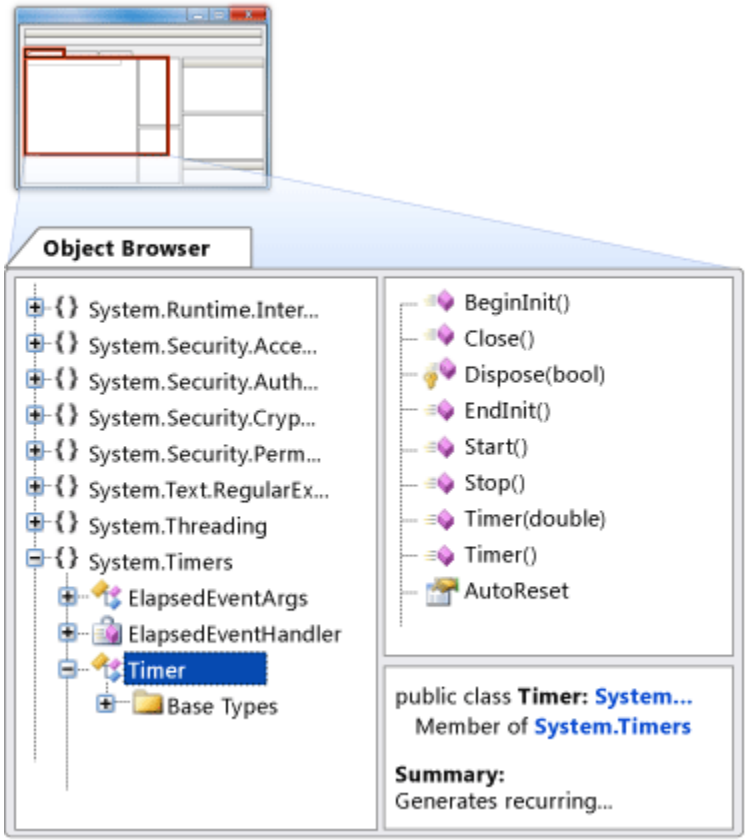
2.5 Common productivity features in Visual Studio

The **Go To Definition** context menu **option takes you directly to the place where the function or object is defined**. Other navigation commands are also available by right-clicking in the editor.



2.5 Common productivity features in Visual Studio

The **Object Browser**, enables you to **inspect .NET or Windows Runtime assemblies** on your system to **see what types they contain** and **what members** (properties, methods, events, etc.) **those types contain**.



2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image

- Visual C# has preexisting controls used to build and customize programs (Fig. 2.26).
- A `Label` contains descriptive text.
- A `PictureBox` displays an image, such as the Deitel bug mascot.

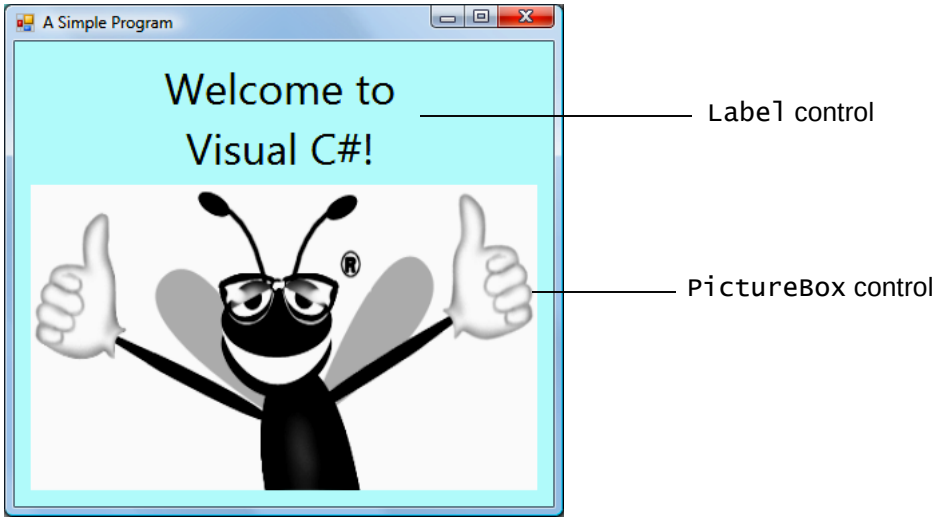


Fig. 2.26 | Simple program executing.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Select **File > New Project...** and create a new **Windows Forms Application** (Fig. 2.27).
- Name the project **ASimpleProgram** and click **OK**.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

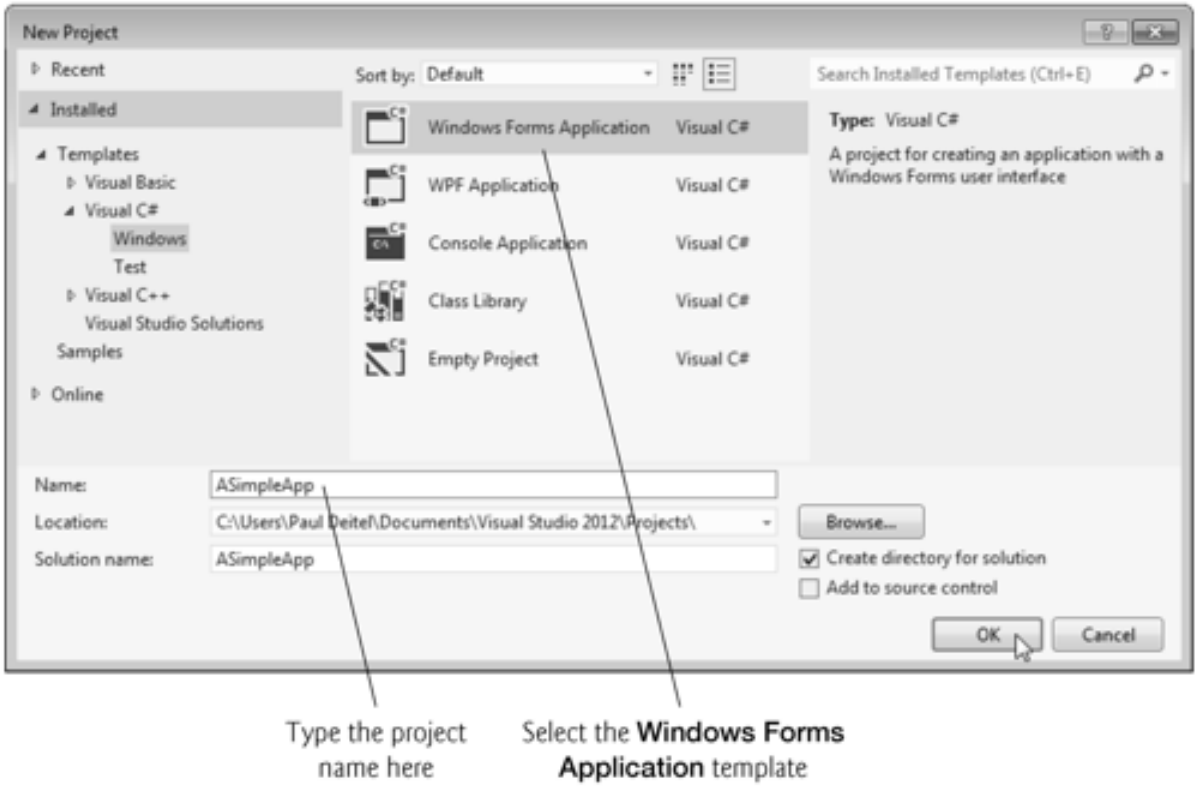


Fig. 2.26 | **New Project** dialog.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Select **File > Save All** to display the **Save Project dialog** (Fig. 2.28).

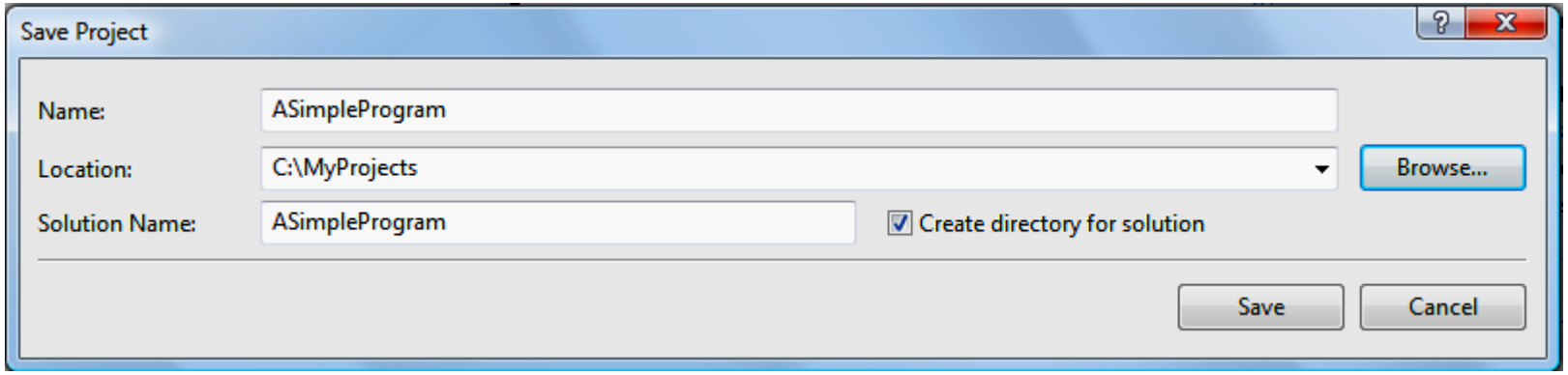


Fig. 2.28 | Save Project dialog.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Click the **Browse...** button, which opens the **Project Location dialog** (Fig. 2.29).
- Navigate through the directories and select one in which to place the project.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

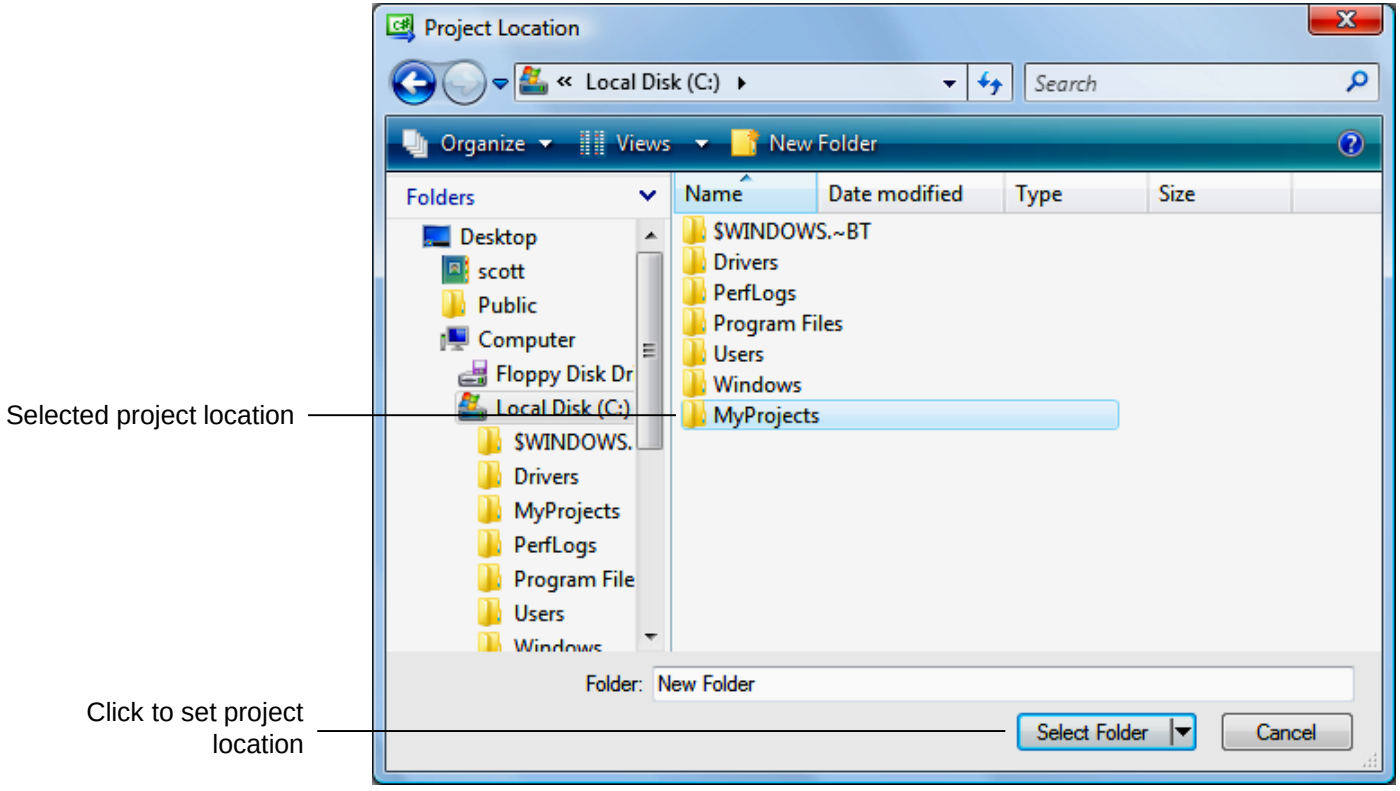


Fig. 2.29 | Setting the project location in the **Project Location** dialog.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Click anywhere in the `Form` to display the `Form`'s properties in the **Properties** window.
- Click in the textbox to the right of the **`Text` property** box and type "`A Simple App`" (Fig. 2.30).

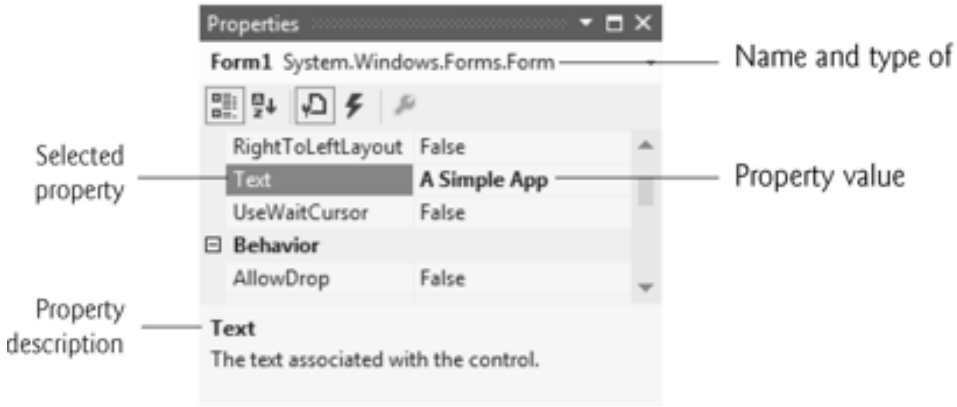


Fig. 2.30 | Setting the `Form`'s `Text` property in the **Properties** window.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Press *Enter*—the `Form`'s title bar is updated immediately (Fig. 2.31).
- Resize the `Form` by clicking and dragging one of the **enabled sizing handles** (Fig. 2.31).

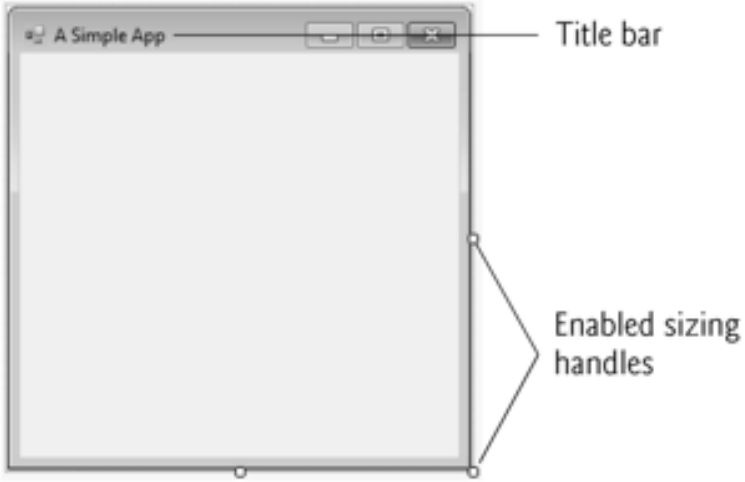


Fig. 2.31 | `Form` with enabled sizing handles.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Select the bottom-right sizing handle and drag it down and to the right to make the `Form` larger
- You can also resize a `Form` by setting its `Size` property.

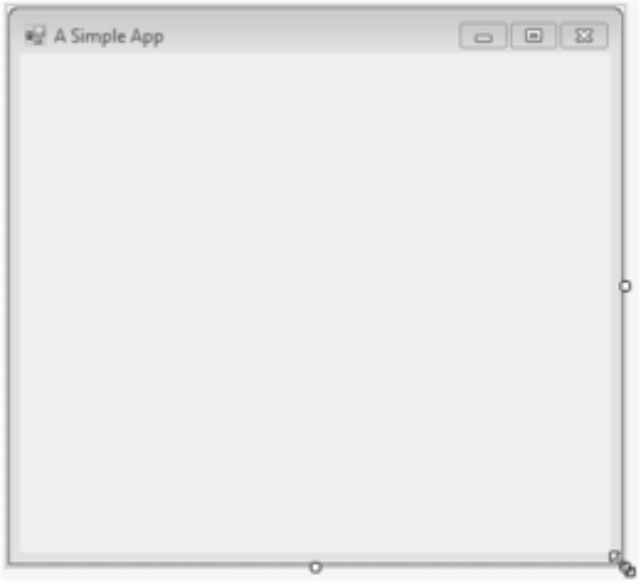


Fig. 2.32 | Resized `Form`.

2.6 Using Visual Programming to Create a Simple Program (Cont.)

- Clicking `BackColor` in the **Properties** window causes a down-arrow button to appear (Fig. 2.33).
- When clicked, the arrow displays tabs for **Custom**, **Web** and **System** colors.
- Click the **Custom tab** to display the **palette** and select light blue.

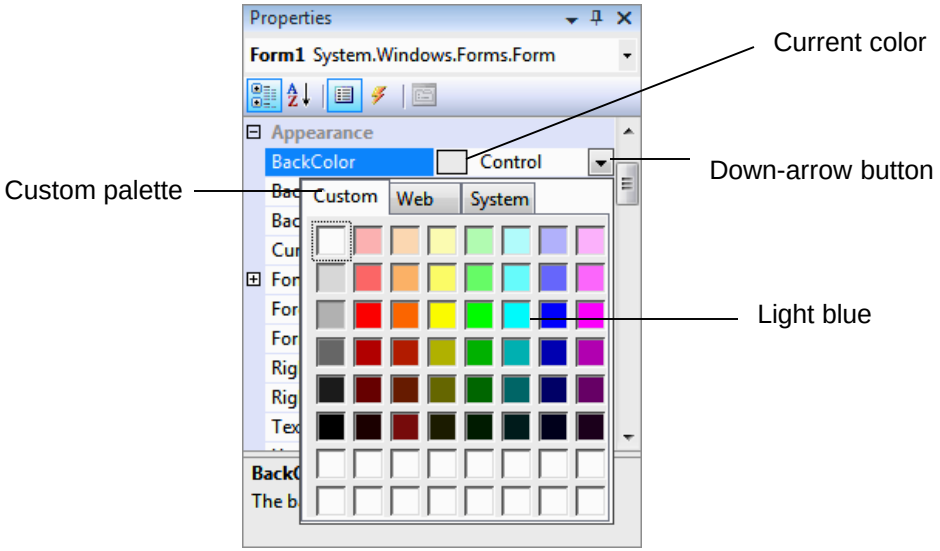


Fig. 2.33 | Changing the `Form`'s `BackColor` property.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Once you select the color, the `Form`'s background changes to light blue (Fig. 2.34).

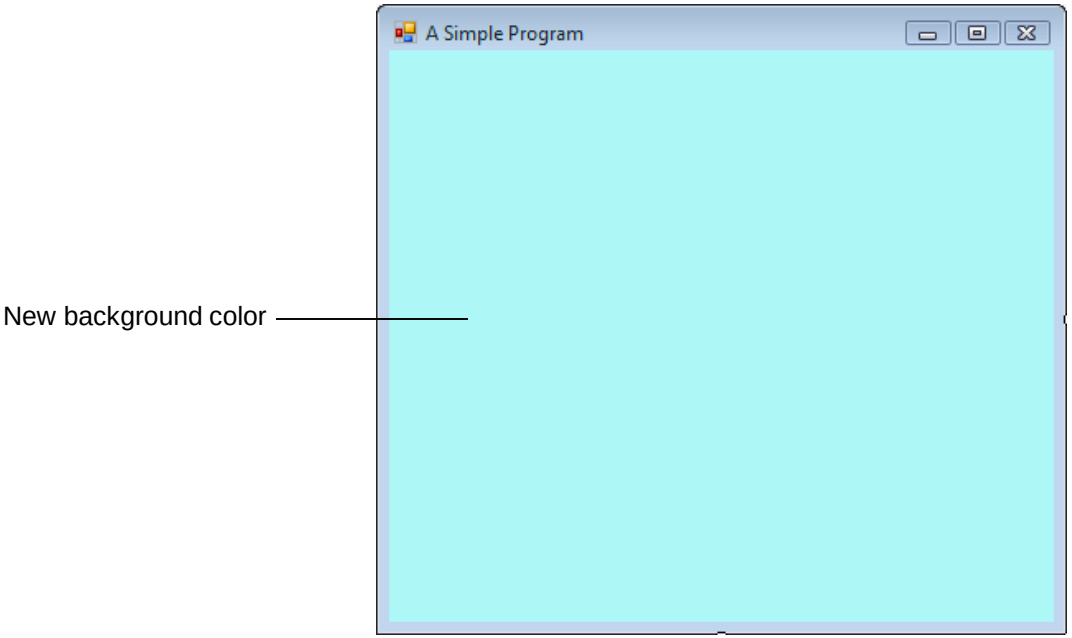


Fig. 2.34 | `Form` with new `BackColor` property applied.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Double click the `Label` control in the **Toolbox** to add a `Label` (Fig. 2.35).
- You also can "drag" controls from the **Toolbox** to the `Form`.

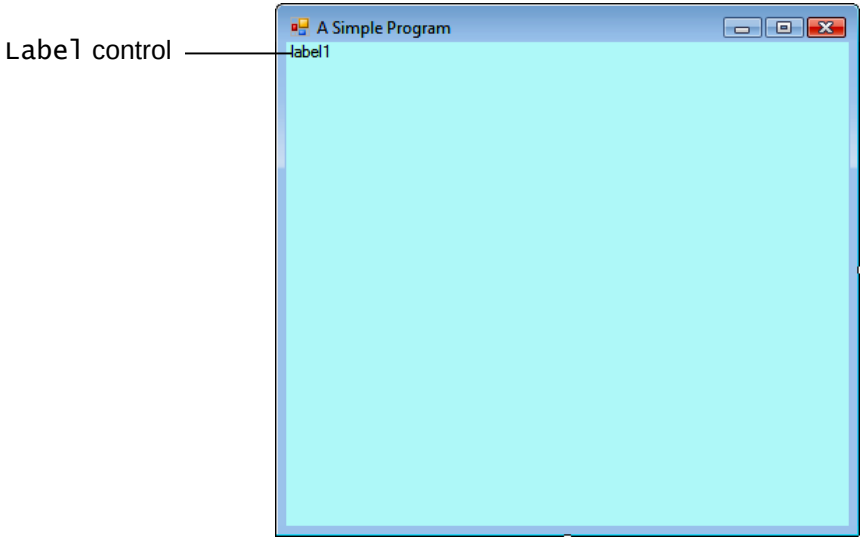


Fig. 2.35 | Adding a `Label` to the `Form`.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Select the `Label` to make its properties appear in the **Properties** window (Fig. 2.36).
- Set the `Label`'s `Text` property to **Welcome to Visual C#!**.
- The **`AutoSize` property** is set to True, which allows the `Label` to resize to fit its text.
- Set the `AutoSize` property to `False`.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

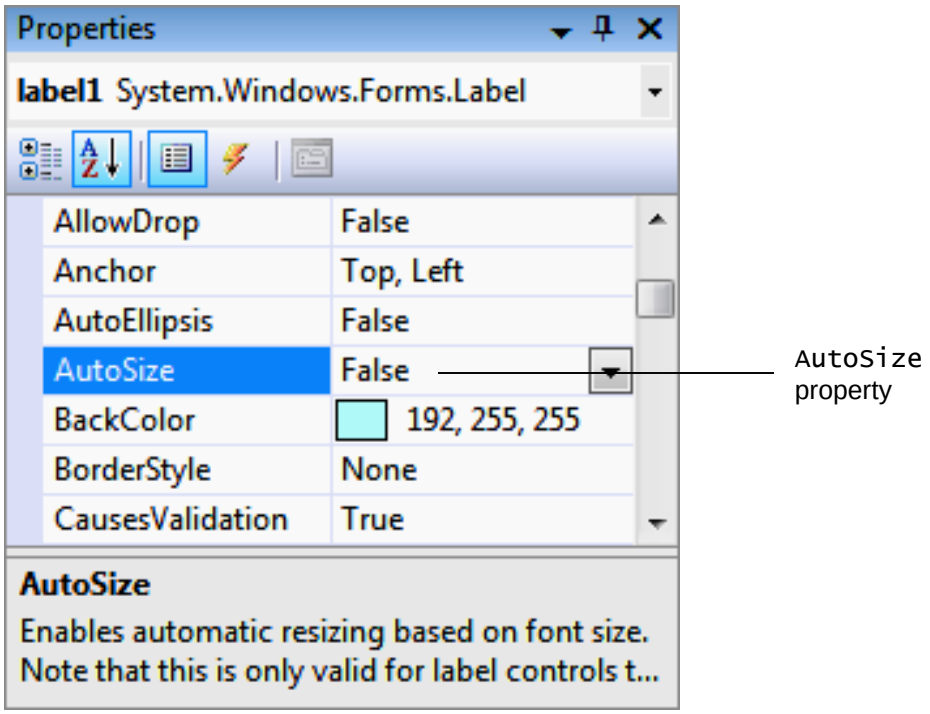


Fig. 2.36 | Changing the `Label`'s `AutoSize` property to `False`.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Move the `Label` by dragging it or by using the left and right arrow keys (Fig. 2.37).
- When the `Label` is selected, you can also center the `Label` using the **Format** menu.

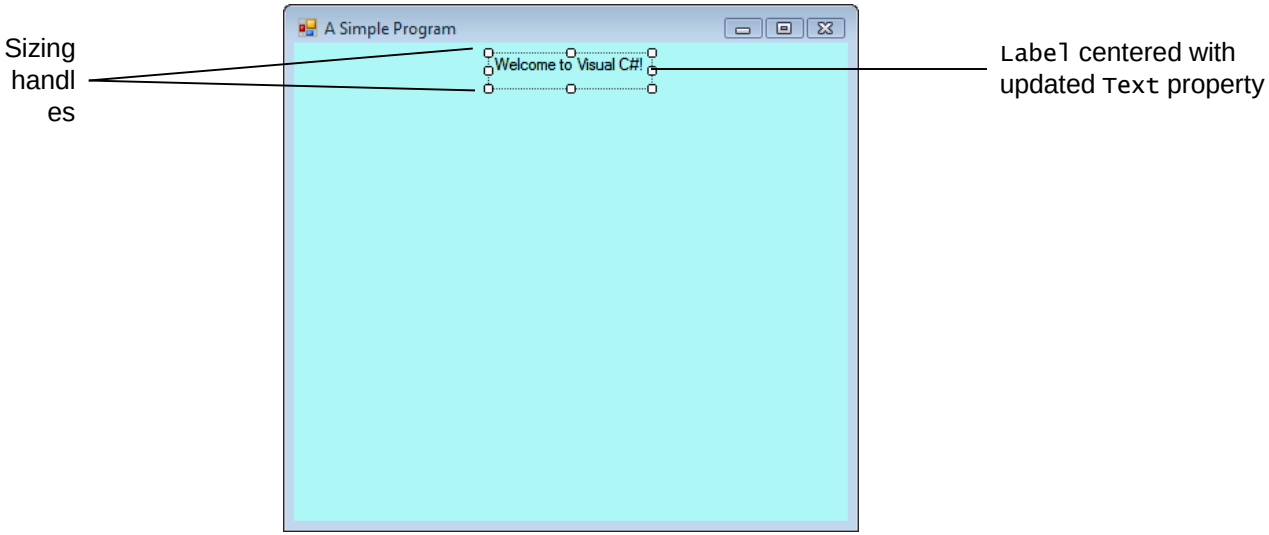


Fig. 2.37 | GUI after the `Form` and `Label` have been customized.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- To change the font of the `Label`'s text, select the **`Font` property** (Fig. 2.38).
- When the ellipsis button is clicked, a dialog appears that provides additional values.

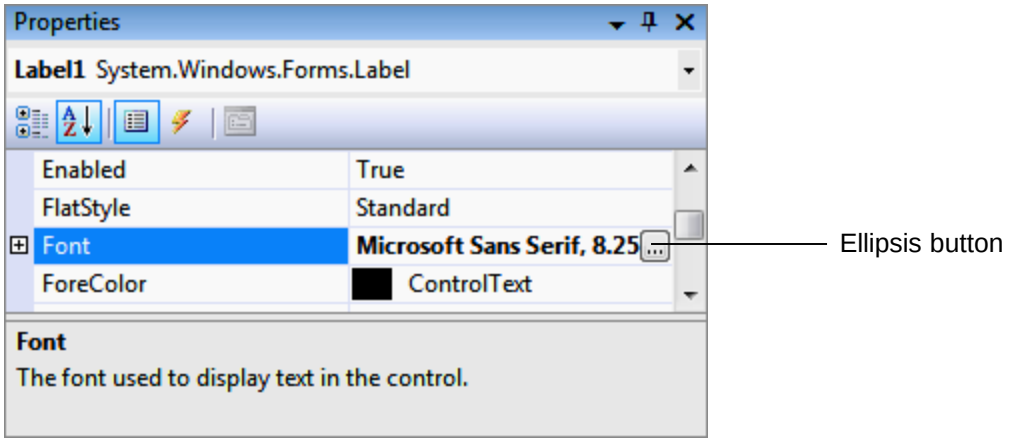


Fig. 2.38 | Properties window displaying the `Label`'s properties.

2.6 Using Visual Programming to Create a Simple Program (Cont.)

- The **Font dialog** (Fig. 2.39) allows you to select the font name, style and size.
- Under **Font**, select **Segoe UI**. Under **Size**, select **24** points and click **OK**.
- Resize the `Label` if it's not large enough to hold the text.

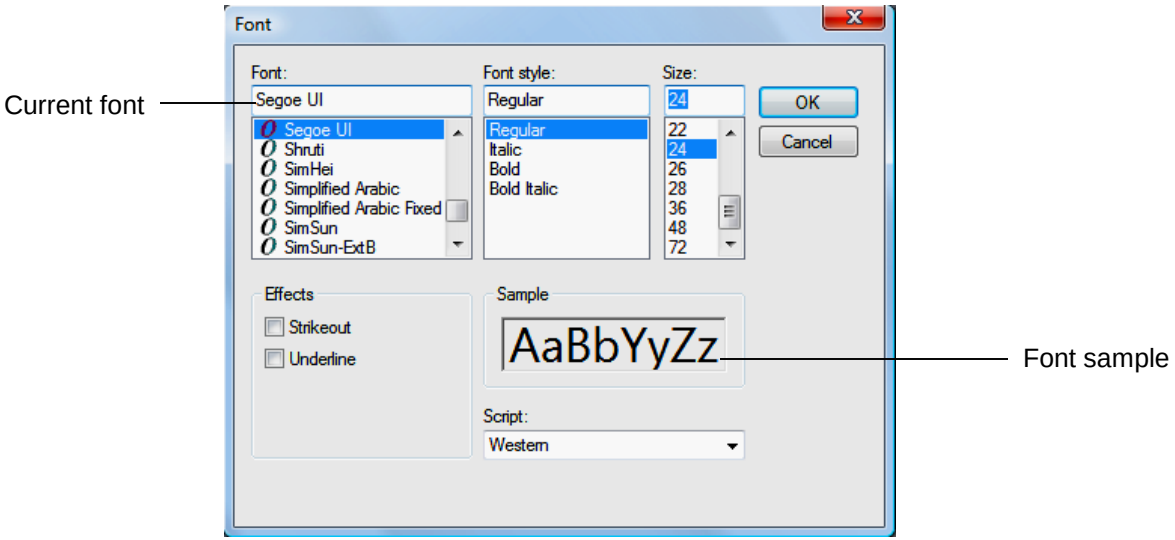


Fig. 2.39 | Font dialog for selecting fonts, styles and sizes.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Select the `Label`'s `TextAlign` property (Fig. 2.40).
- Set the `TextAlign` property to `MiddleCenter`.

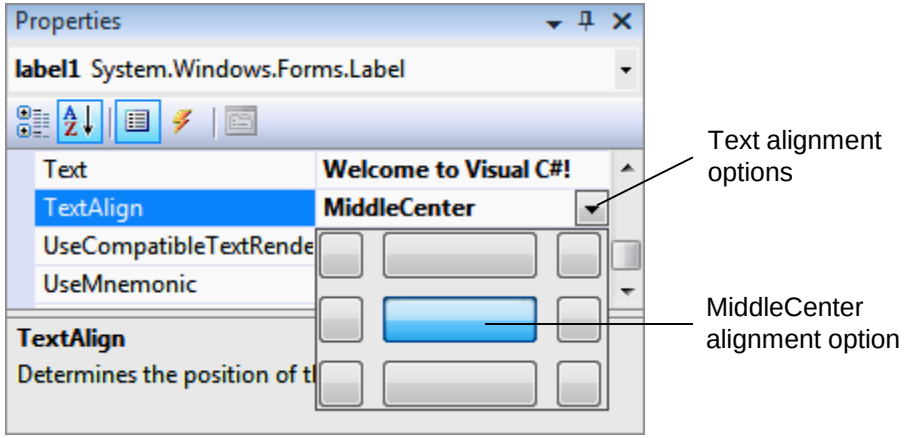


Fig. 2.40 | Centering the `Label`'s text.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Locate the `PictureBox` in the **Toolbox** and add it to the `Form` (Fig. 2.41).

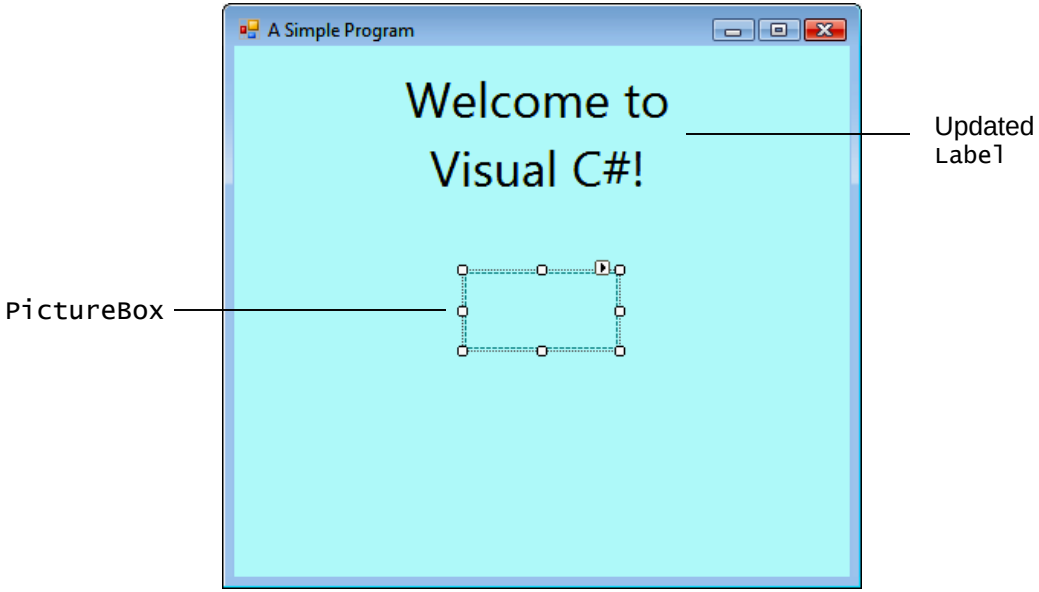


Fig. 2.41 | Inserting and aligning a `PictureBox`.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Click the `PictureBox` to display its properties in the **Properties** window (Fig. 2.42).
- The **`Image` property** displays a preview of the image, if one exists.

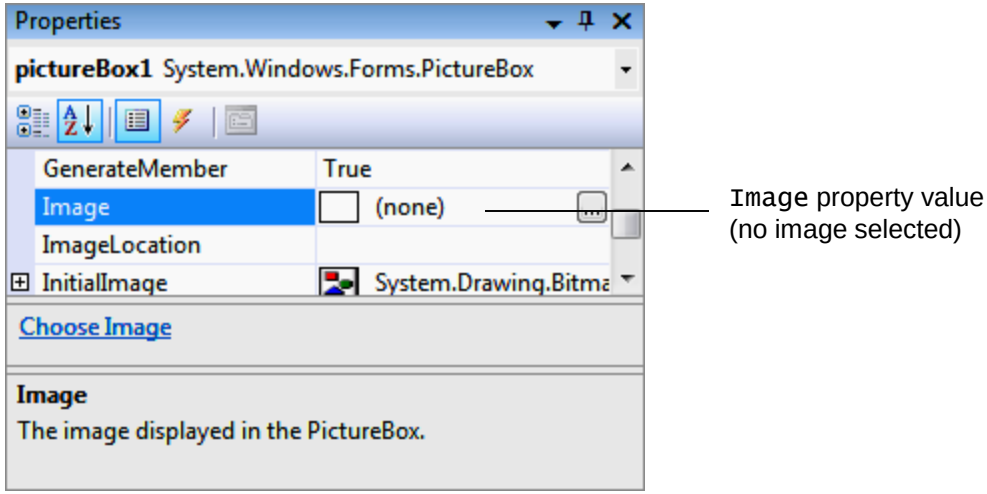


Fig. 2.42 | `Image` property of the `PictureBox`.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Click the ellipsis button to display the **Select Resource dialog** (Fig. 2.43).
- Click the **Import...** button to browse for the image to insert (`bug.png`)

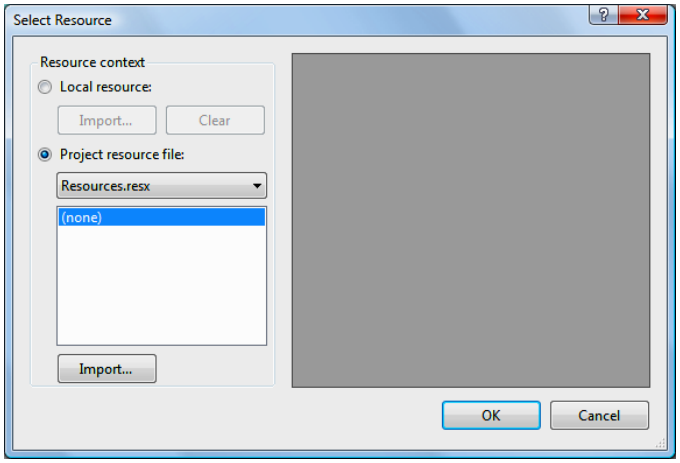


Fig. 2.43 | Select Resource dialog to select an image for the `PictureBox`.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- In the dialog that appears, locate the image file, select it and click **OK** (Fig. 2.44).
- Click **OK** to place the image in your program.

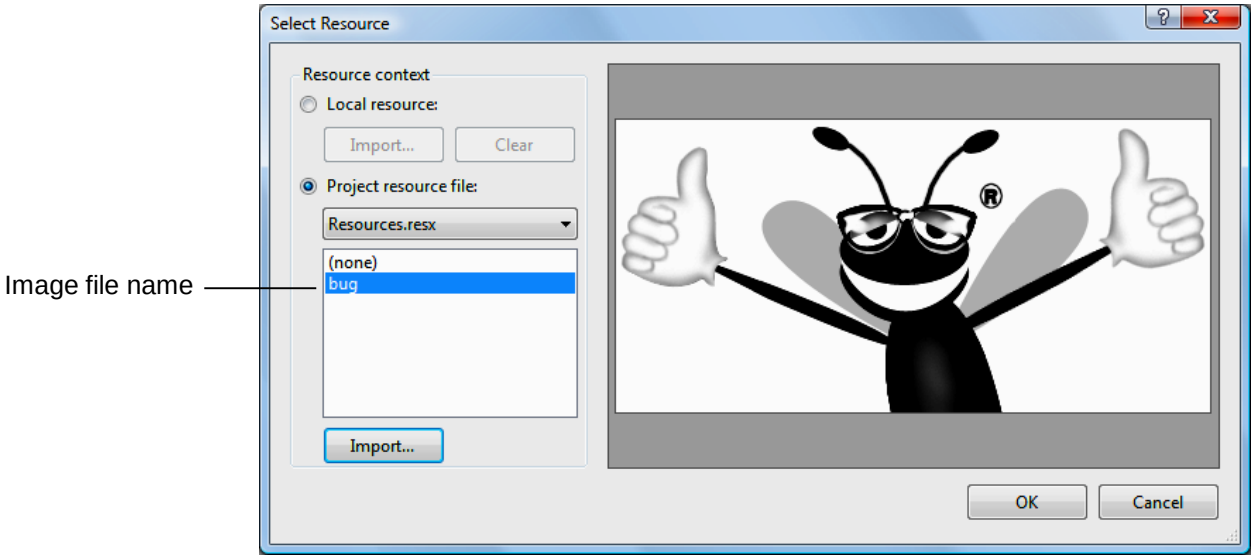


Fig. 2.44 | **Select Resource** dialog displaying a preview of selected image.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- To size the image to the `PictureBox`, change the **`SizeMode` property** to **`StretchImage`** (Fig. 2.45).

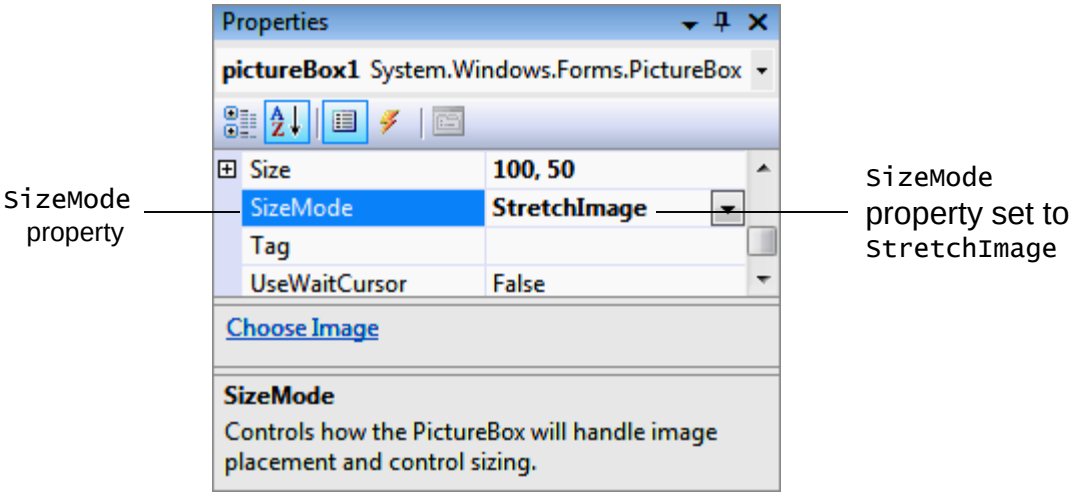


Fig. 2.45 | Scaling an image to the size of the `PictureBox`.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- Resize the `PictureBox`, making it larger (Fig. 2.46).
- Select **File > Save All** to save the entire solution.

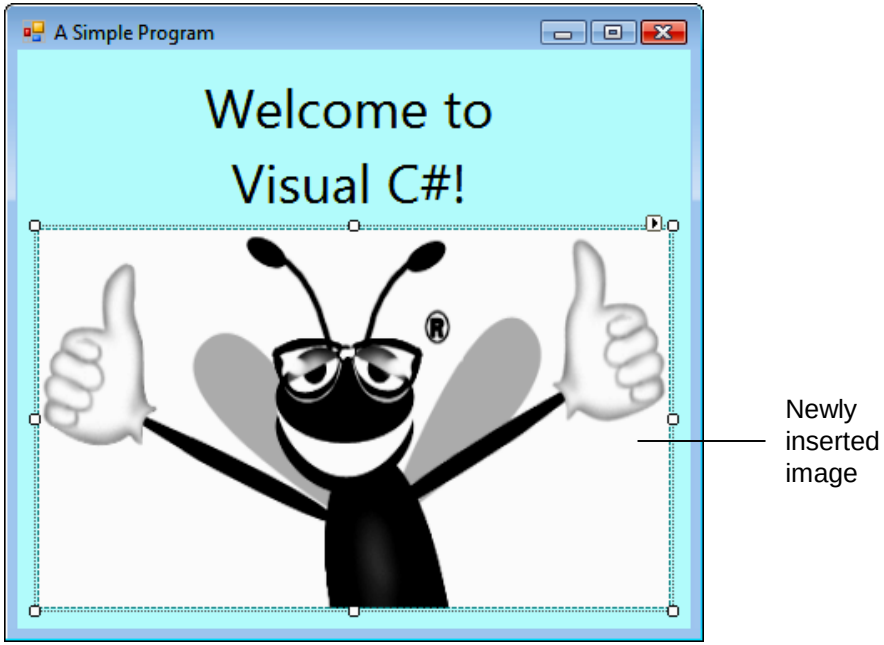


Fig. 2.46 | `PictureBox` displaying an image.

2.6 Using Visual Programming to Create a Simple Program that Displays Text and an Image (Cont.)

- In **run mode**, the program is executing, and some features are disabled (Fig. 2.47).

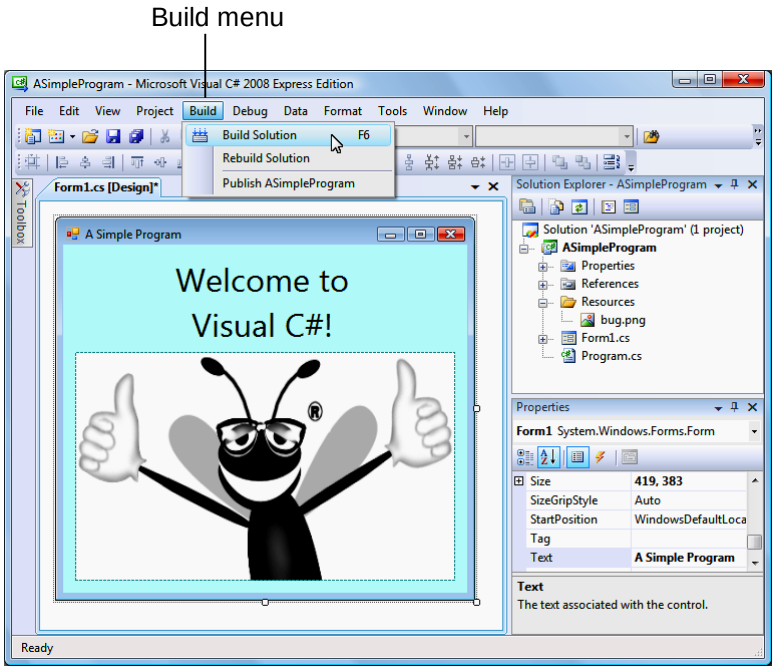


Fig. 2.47 | Building a solution.

2.6 Using Visual Programming to Create a Simple Program (Cont.)

- Select **Debug > Start Debugging** to execute the program (Fig. 2.48).

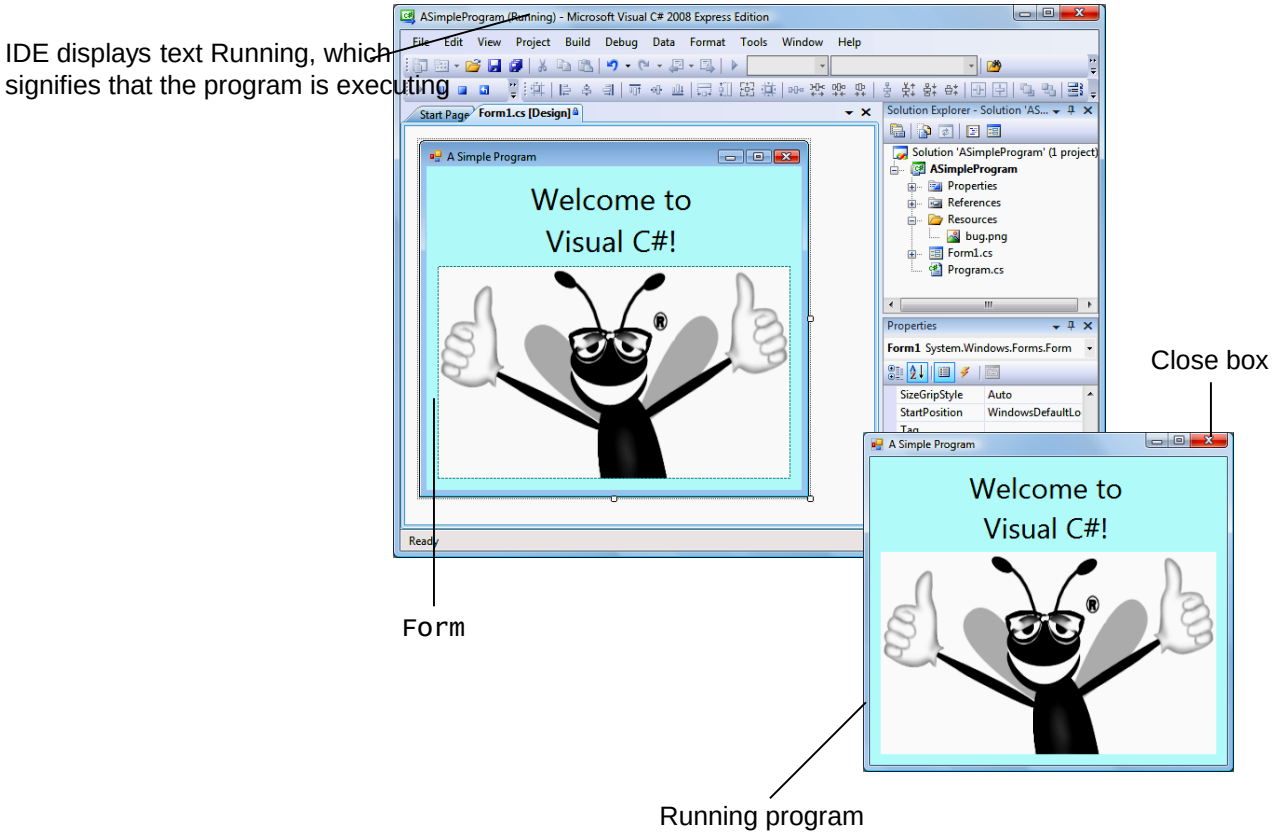


Fig. 2.48 | IDE in run mode, with the running program in the foreground.