Chapter 2 – 1st Application

2.1 Becoming Familiar with the Integrated Development Environment

Visual Basic .NET provides a complete environment to assist developers in creating their applications.

This environment may vary slightly from computer to computer.

The majority of the key items are essentially the same.

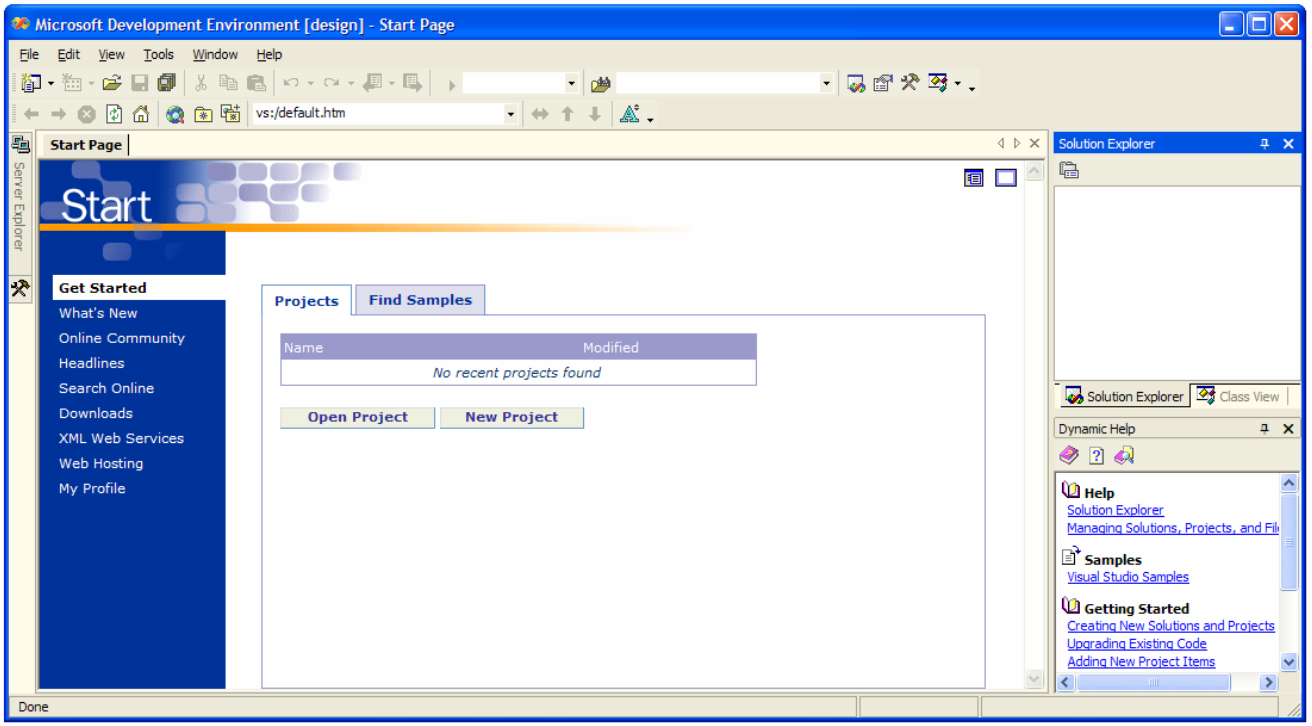
The first step in creating an application with Visual Basic .NET is to familiarize yourself with the main components of the new integrated development environment.

Chapter 2 – 1st Application

Get Started page

Step 1. When Visual Basic .NET is started, the **"Get Started" window** for Visual Studio .NET, similar to a home page of a Web site on the Internet, appears. This page gives you access to the following:

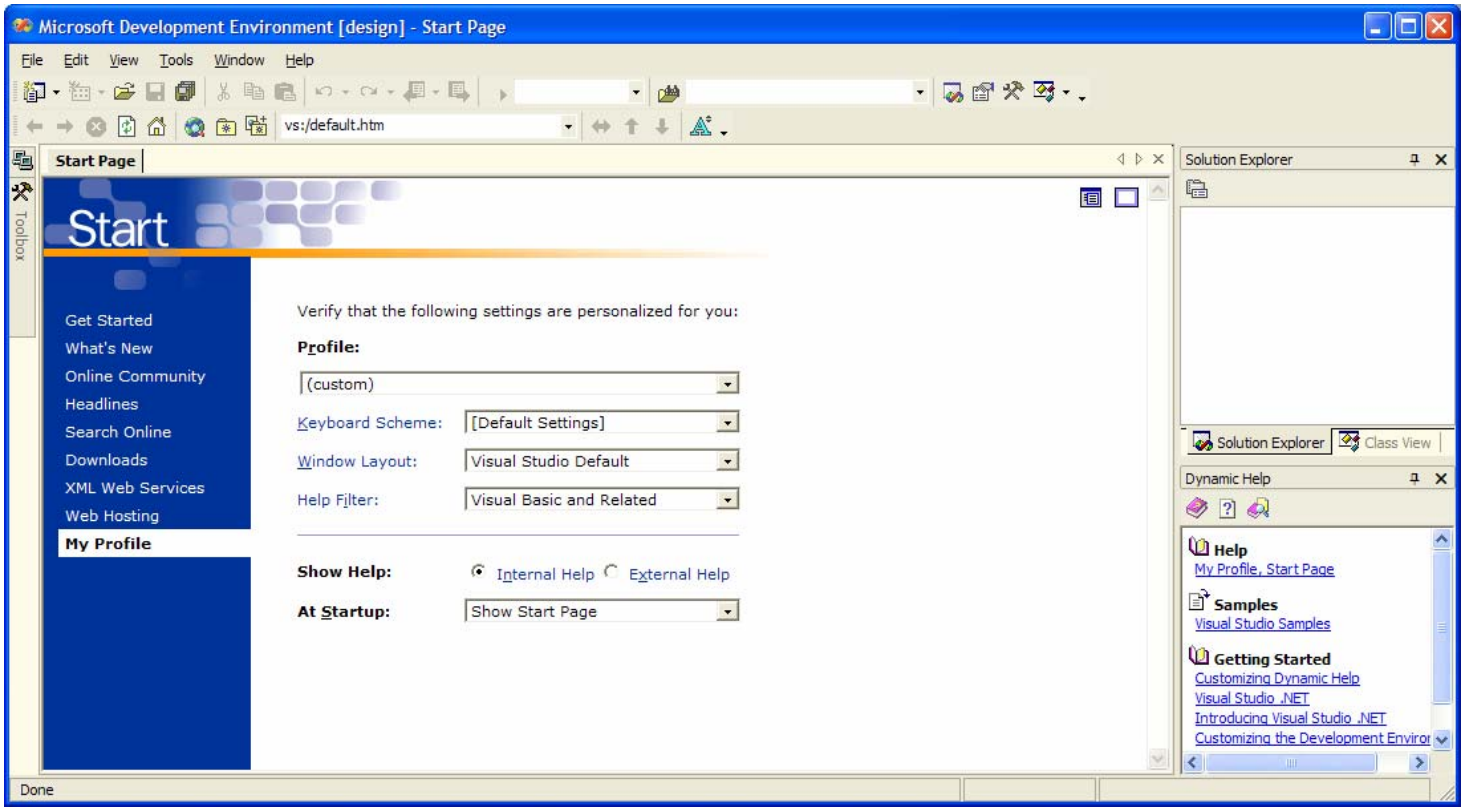
- Recently used projects
- The ability to open projects not listed in the recent projects
- An option to create new projects
- Links to other resources



Chapter 2 – 1st Application

My Profile page

Step 2. The very first time you run Visual Basic .NET, you will be presented with the **My Profile screen** which will allow you to personalize the development environment settings. Once the features are set, the screen will no longer appear. Select My Profile from the VS Start Page to revisit it.



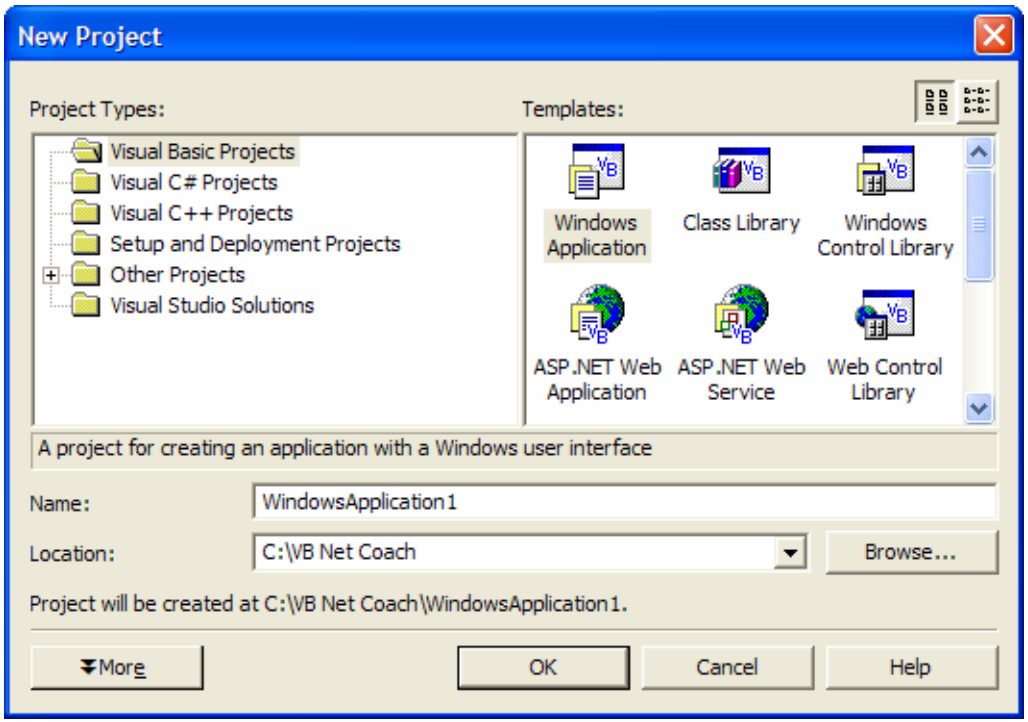
Chapter 2 – 1st Application

New Project window

Step 3. If you select New Project from the VS Start Page, the **New Project window** will appear.

For now, stick to a Windows application. Make sure Visual Basic Projects is selected from the Project Types and Windows Application is selected from the Templates.

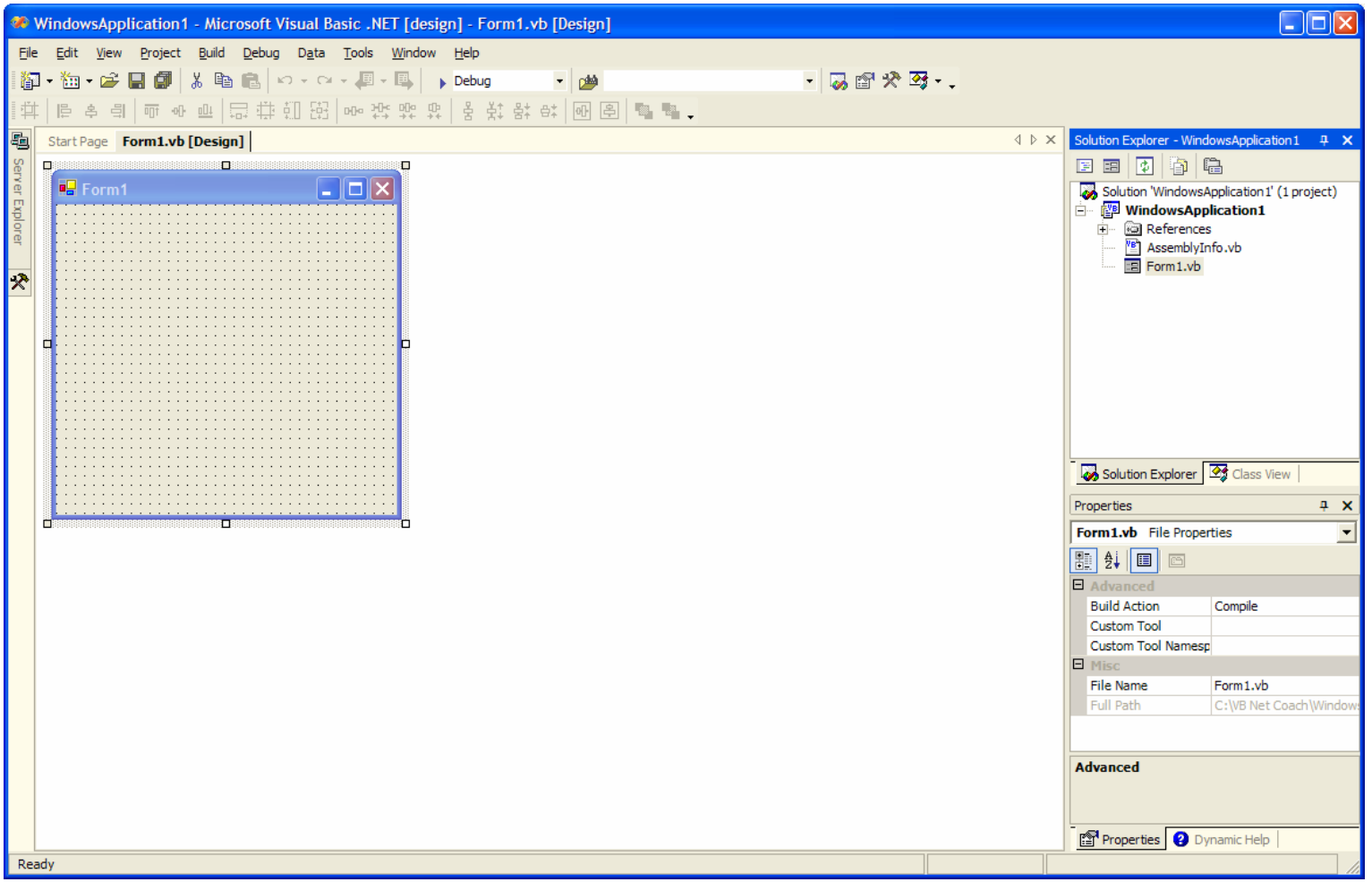
Specify a name and place to store your new project.



Chapter 2 – 1st Application

Integrated Development Environment

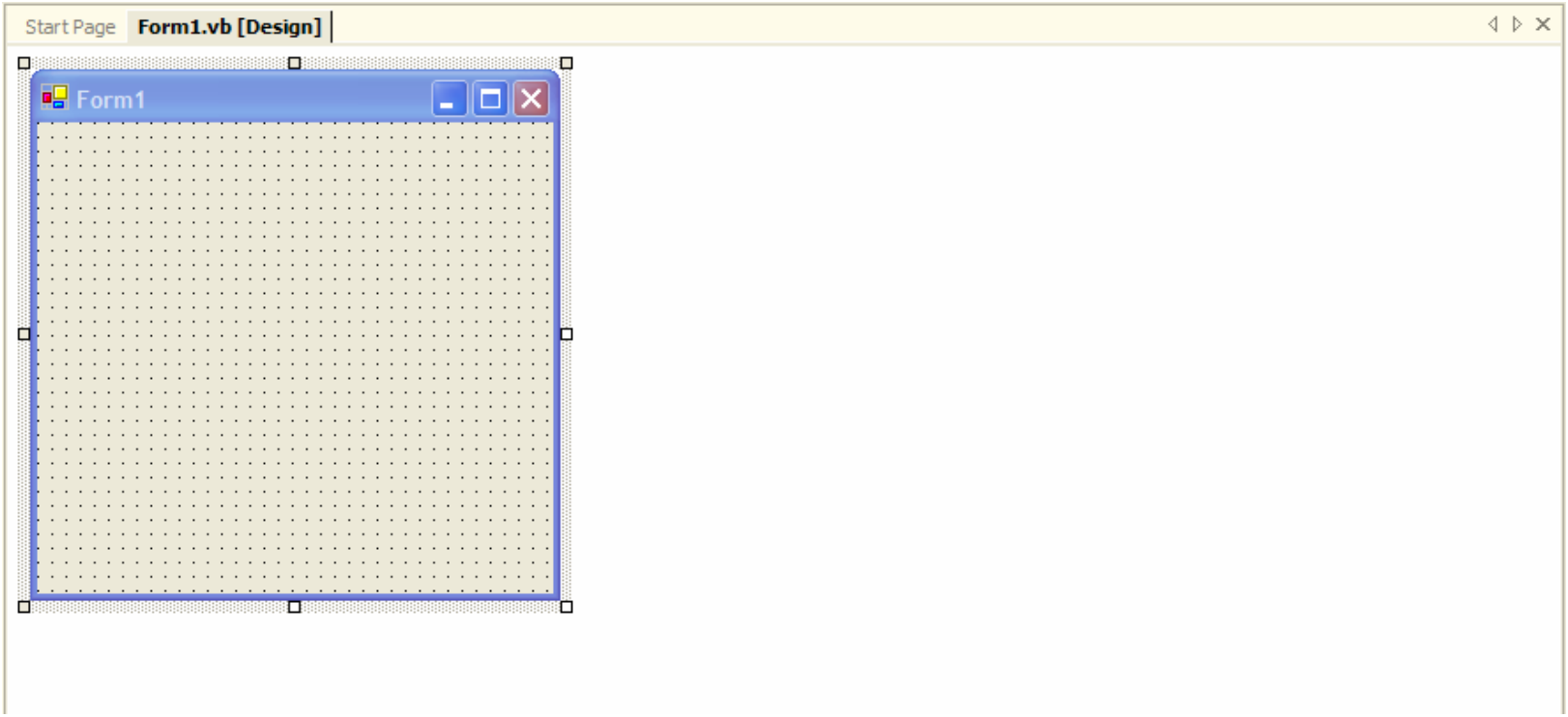
Once the application has been created, you will see the **IDE**.



Chapter 2 – 1st Application

Form Window

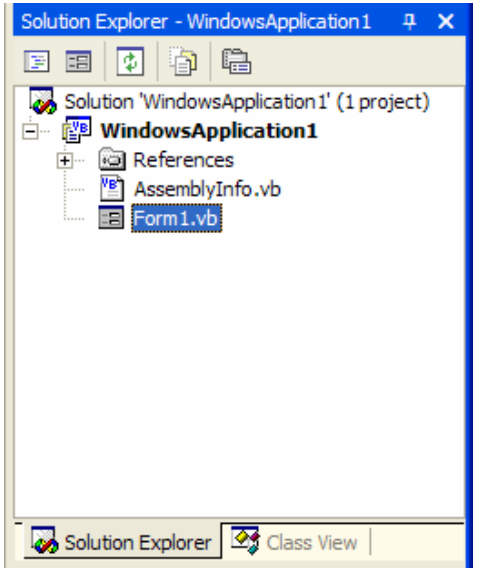
The **Form window** is like a painter's canvas. It is where you will lay out the design of a form for your application and the interface for placing the code associated with the components.



Chapter 2 – 1st Application

Solution Explorer

The **solution explorer window** wraps all of the components of an application into one interface. Here you can access all of the forms, projects, and other modules that are combined to build your application.

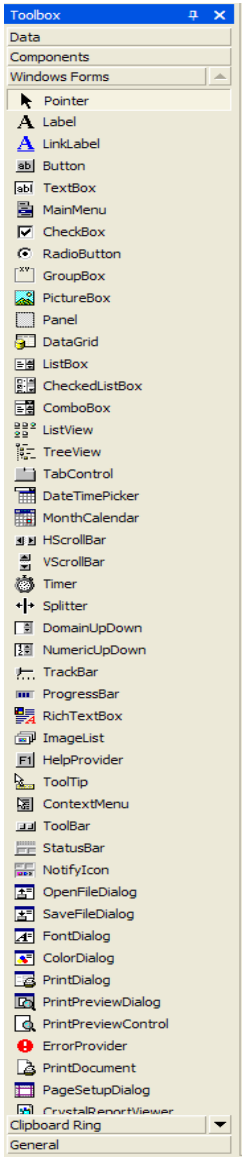


Standard Toolbar

Visual Basic has a main or **Standard toolbar** as well as additional toolbars to give the developer easy access to commonly used operations.

Chapter 2 – 1st Application

Toolbox Window



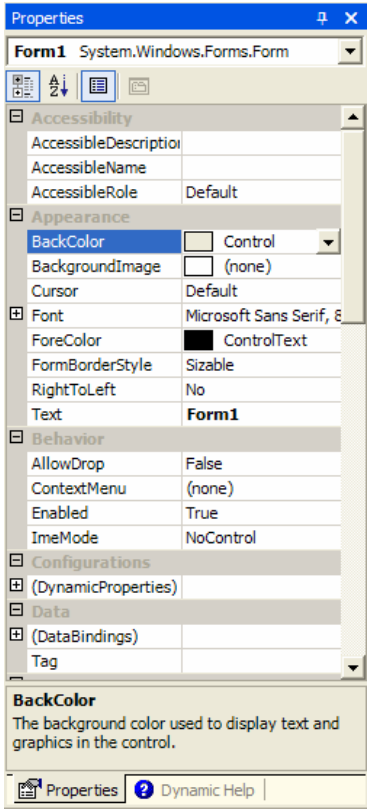
Toolbox window contains objects called controls that assist in creating applications. The controls are broken into sections.

Chapter 2 – 1st Application

Properties Window

Components of Visual Basic have many properties associated with them. A **property** is a way of customizing the appearance and behavior of a control in Visual Basic .NET.

Amongst other properties, the **Properties window** allows the developer to control the color, font, and size of Visual Basic .NET constructs. The current property selected is highlighted, and additional information explaining the purpose of that property is displayed at the bottom of the window. Properties are sometimes grouped together.



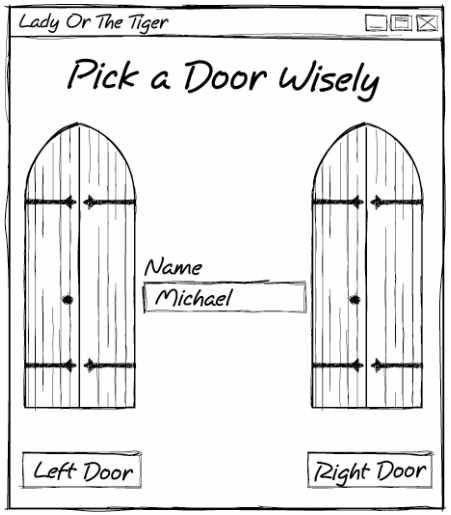
Menu Bar

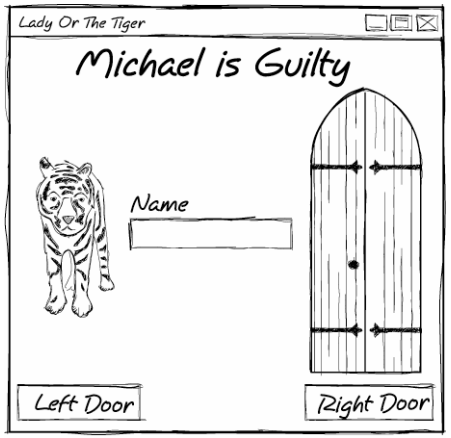
The Visual Basic .NET **Menu bar** is located just below the Title bar. It contains shortcuts to commonly used commands.

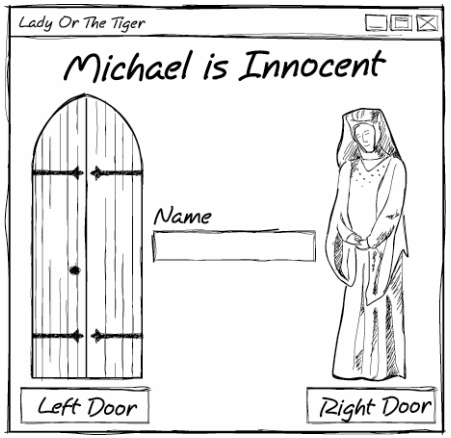
Chapter 2 – 1st Application

2.2 Creating Your First Application

For your first application, you are going to create a Form that acts out the classic tale of "The Lady or the Tiger" by Frank R. Stockton. You will develop this application in a stepwise fashion.



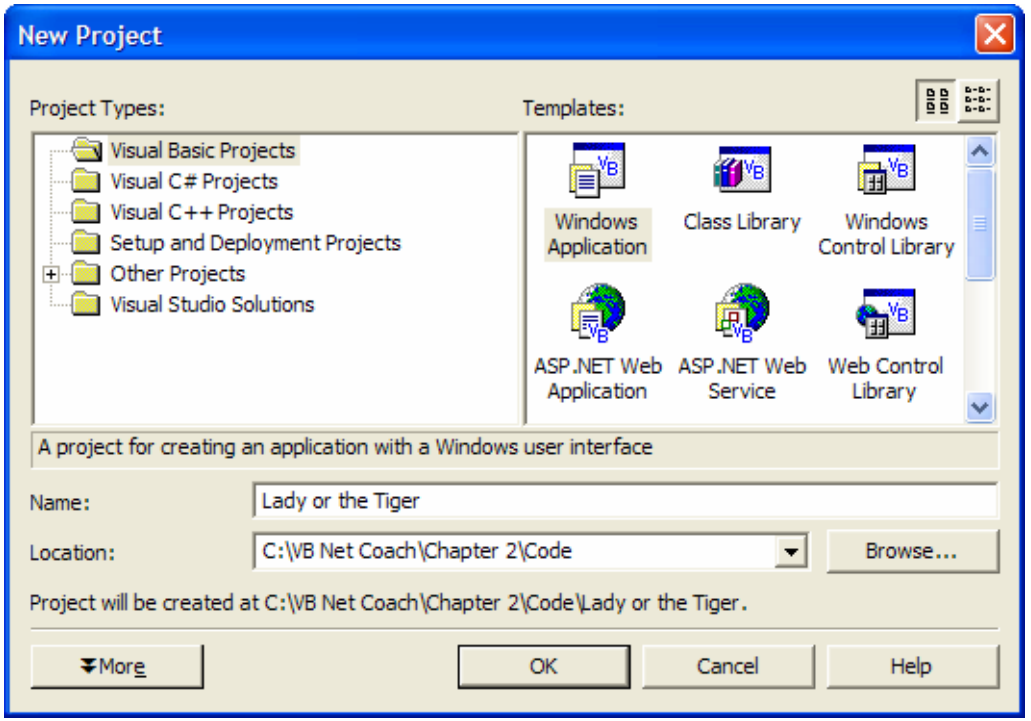




Chapter 2 – 1st Application

Setting the Project Name and Form Name

Step 1: From the Start window, click on New Project. The New Project window will appear.

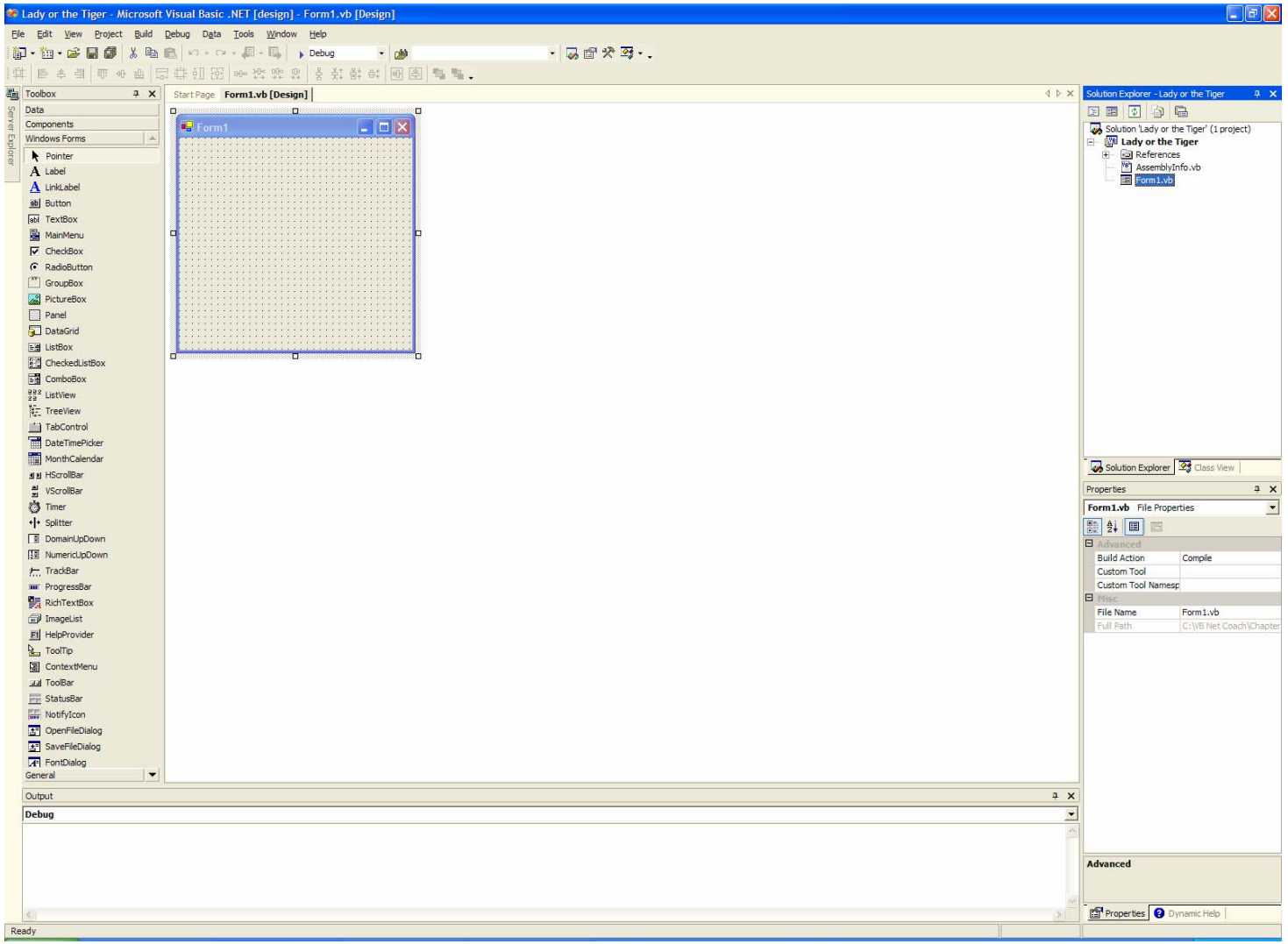


Step 2: Specify the name of the application as "`Lady or the Tiger`".

Step 3: Specify the location as "`C:\VB Net Coach\Chapter 2\Code\`".

Chapter 2 – 1st Application

Step 4: Click on the OK button and the development environment will appear.



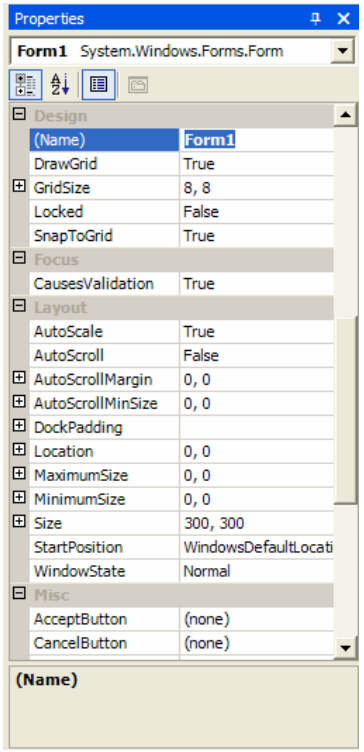
Chapter 2 – 1st Application

Setting the Form Name

Step 5: You should rename the default form of the application `Form1` to something more meaningful. Visual Basic .NET allows you to set a value to display as the name for the form, to set the name you refer to the form from within your program, and to set the actual file name.

When naming Forms in Visual Basic, your name should follow Microsoft's naming convention. The prefix identifies what type of object you are naming. In this case, use the prefix `frm` to indicate that your name is associated with a Form object.

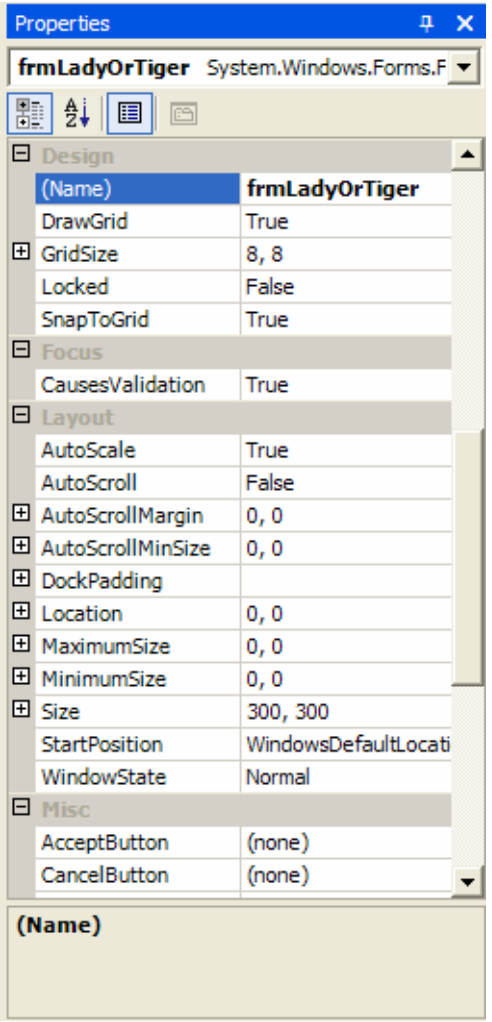
Double-click on the window containing the name `Form1` to highlight it.



Chapter 2 – 1st Application

Setting the Form Name

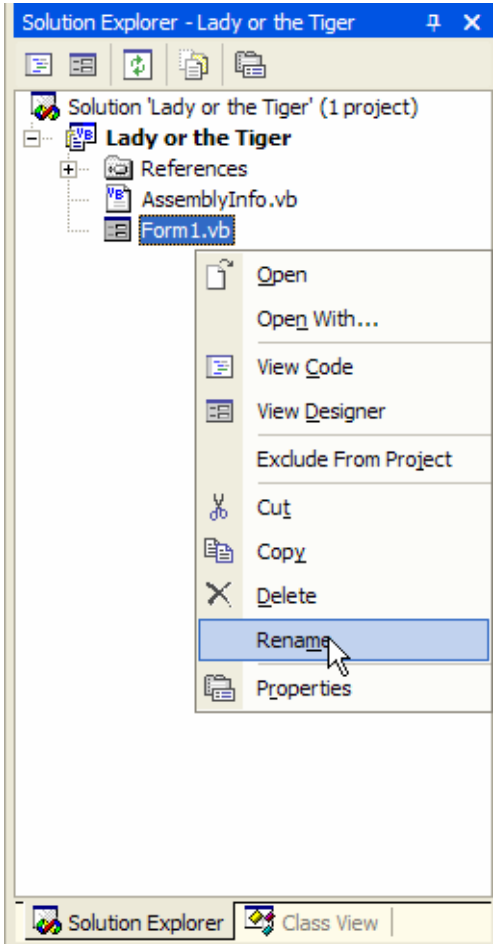
Step 6: Type the name for the form, `frmLadyOrTiger` in the Properties window.



Chapter 2 – 1st Application

Setting the Form Name

Step 7: You can rename the file by right-clicking on the file name in the Solution Explorer. A pop-up window will appear that will allow you to select Rename.

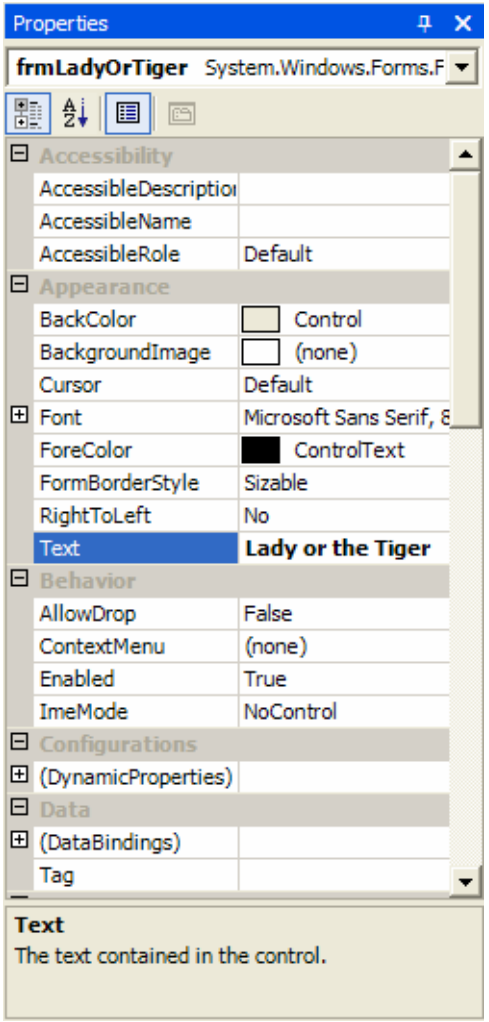


Step 8: Enter the file name for the form, `LadyOrTiger.vb`

Chapter 2 – 1st Application

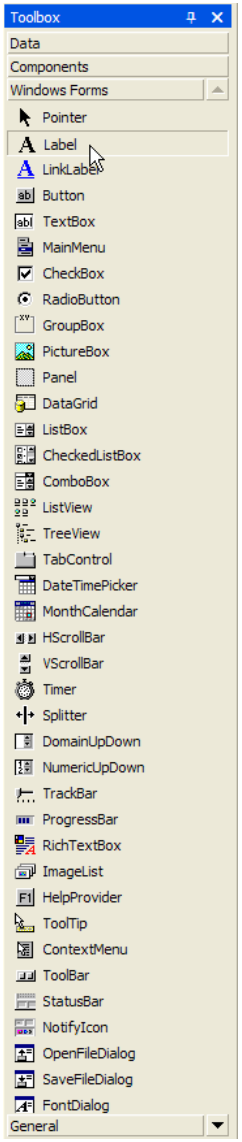
Setting the Form Text

Step 9: To set the display name for the form set the Text property to **Lady Or The Tiger**.



Chapter 2 – 1st Application

Placing a Label Control on the Form



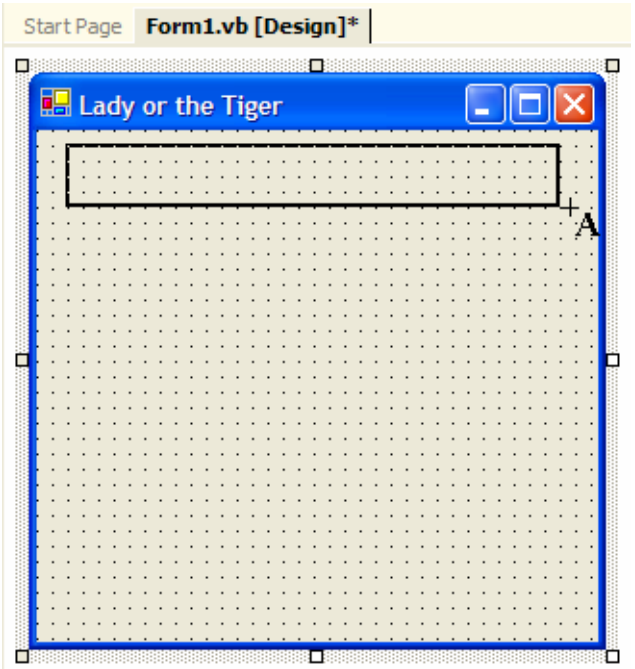
A label control allows you to add static text to a form that will typically not change during the running of your application.

Step 1: Select the label control by clicking on the **icon** for the Label control .

Chapter 2 – 1st Application

Placing the Label Control

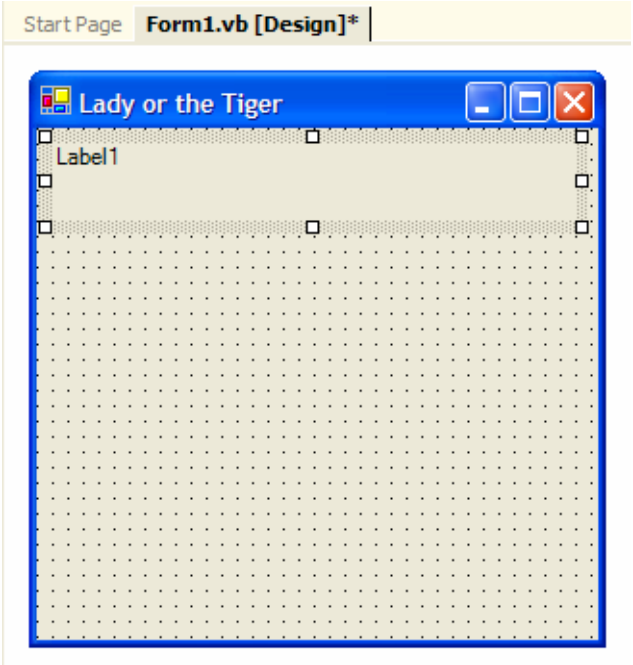
Step 2: Place the label in the desired position on the form.



Chapter 2 – 1st Application

Placing the Label Control

Step 3: Release the mouse button, thereby completely specifying the location, width, and height of the text box.



Chapter 2 – 1st Application

Placing the Label Control

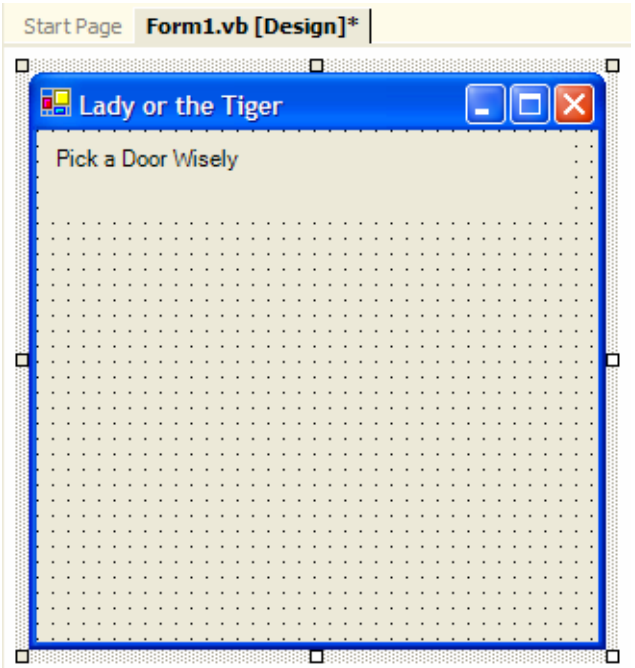
Step 4: Make sure that your `Label` control is roughly the same size as shown. If it is not, there are two ways that you can modify your control.

You can adjust it visually.

You can adjust it by modifying properties of the Label control. The `Width and` Height properties can be set to the exact values. Set yours to 248, 40, respectively.

Step 5: Set the Name property of the Label to `lblTitle`.

Step 6: Click the `Text` property and type `Pick a Door Wisely.`

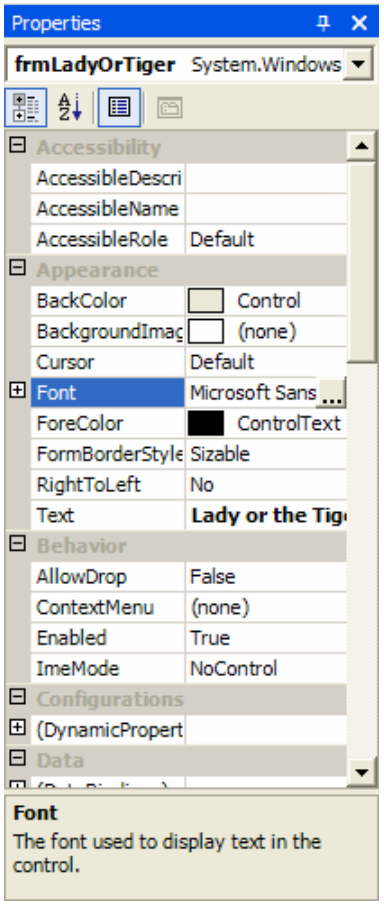


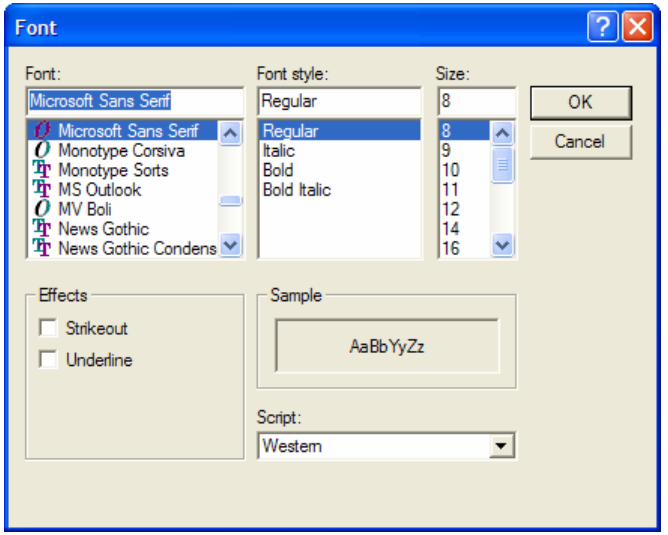
Chapter 2 – 1st Application

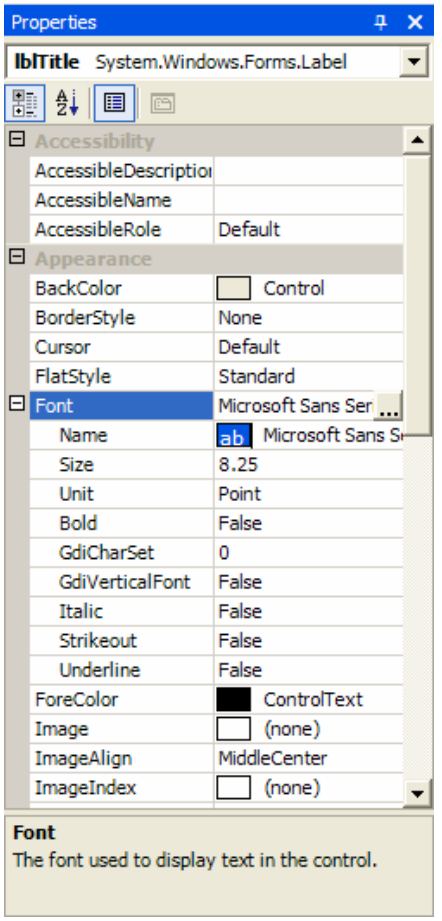
Changing the Label Size and Style

Step 7: We can set a font's type face, size, and style.

Set the `Font.Style` to `Bold` and the `Size` to `18`.





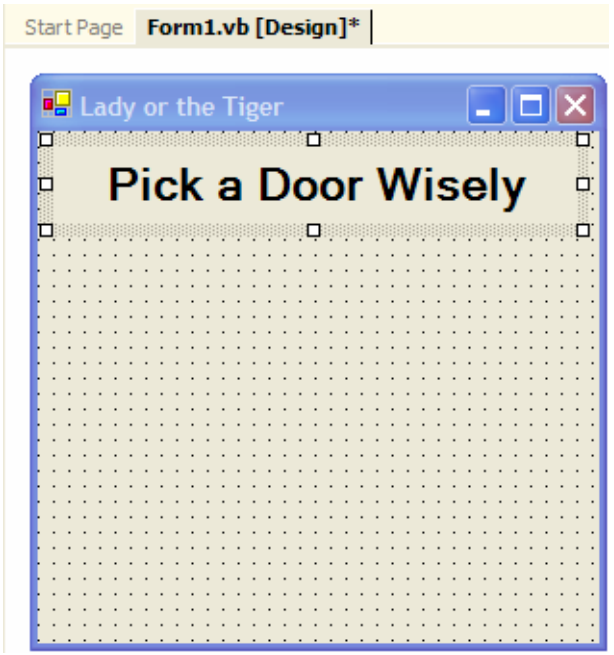


Chapter 2 – 1st Application

Aligning the Label Text

Step 8: The final step is to align the text so that it is centered in the form.

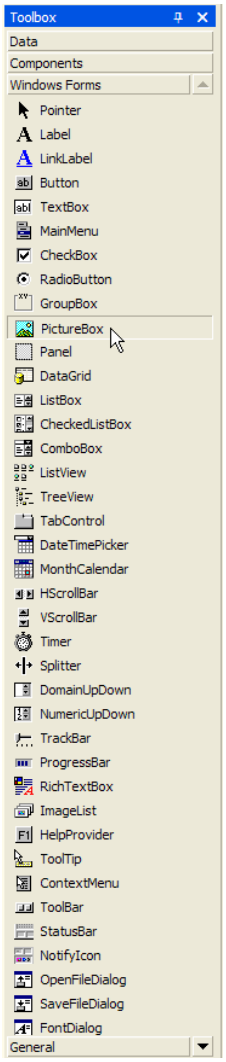
This can be done visually or by clicking on the `TextAlign` property and selecting `MiddleCenter` from the pull-down menu.



Chapter 2 – 1st Application

2.3 Picture Box Control

Visual Basic allows you to easily add pictures to your form. In your case, you wish to add a picture of a door twice. To display a picture, you will use the **picture box control**.



Select PictureBox Control

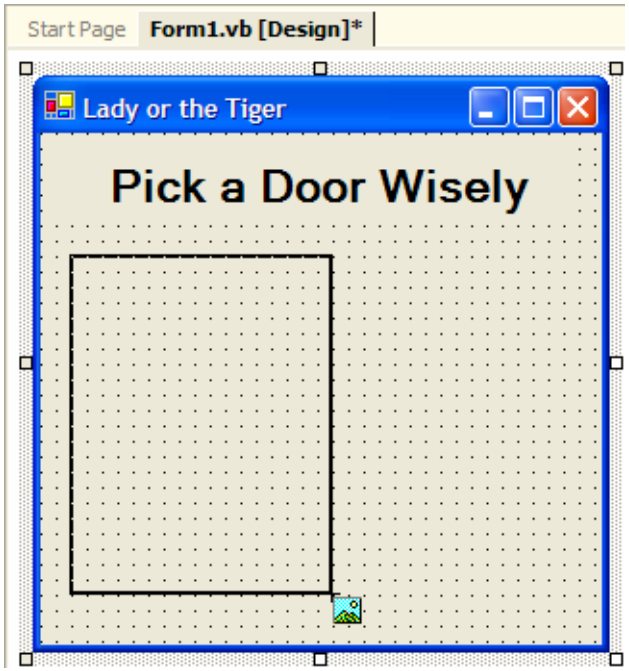
Step 1: Start by clicking on the PictureBox control in the control toolbox.

Chapter 2 – 1st Application

Placing the PictureBox Control

Step 2: Click just below and to the left of the text in the Label control you previously placed.

Hold the mouse button down and release it with the mouse pointer near the bottom of the form and aligned in between the o's of the word "Door" in the label control.



Chapter 2 – 1st Application

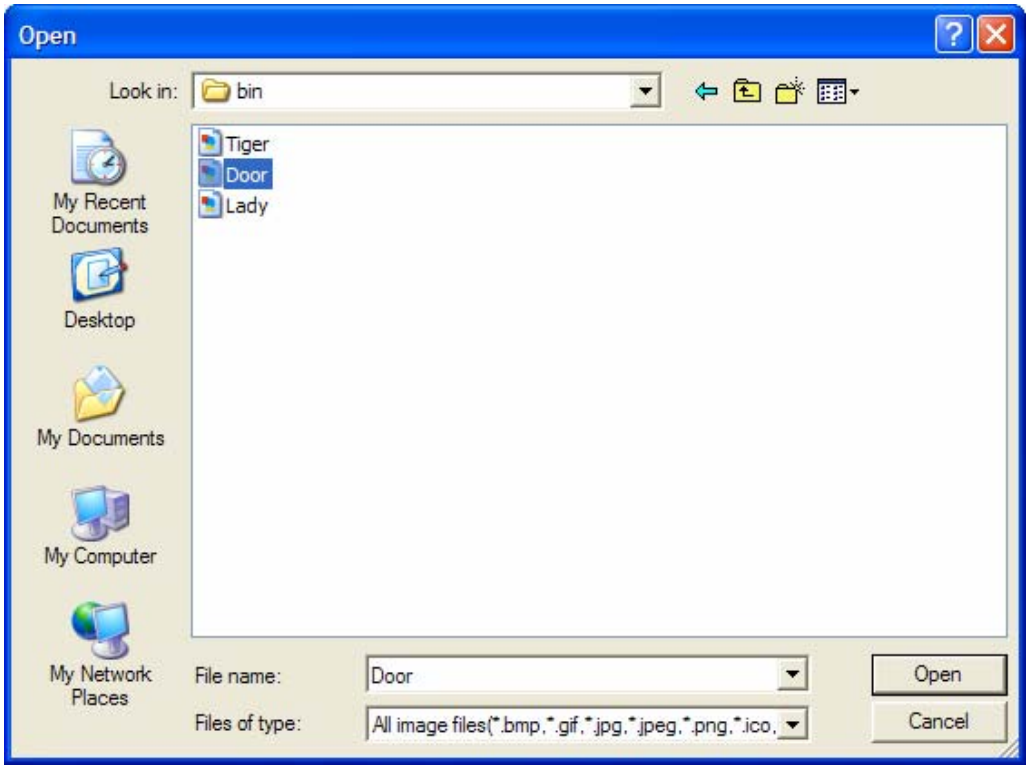
Setting the Name of the Control

Step 3: Set the `Name` property of the `PictureBox` to `picLeftDoor`.

Setting the Picture to Display in the Control

Step 4: Click on the `Image` property

Click on the ellipses (…) and a dialog box will appear to select the graphic file to display within the `PictureBox` control.



Chapter 2 – 1st Application

Resizing the Picture Box Control

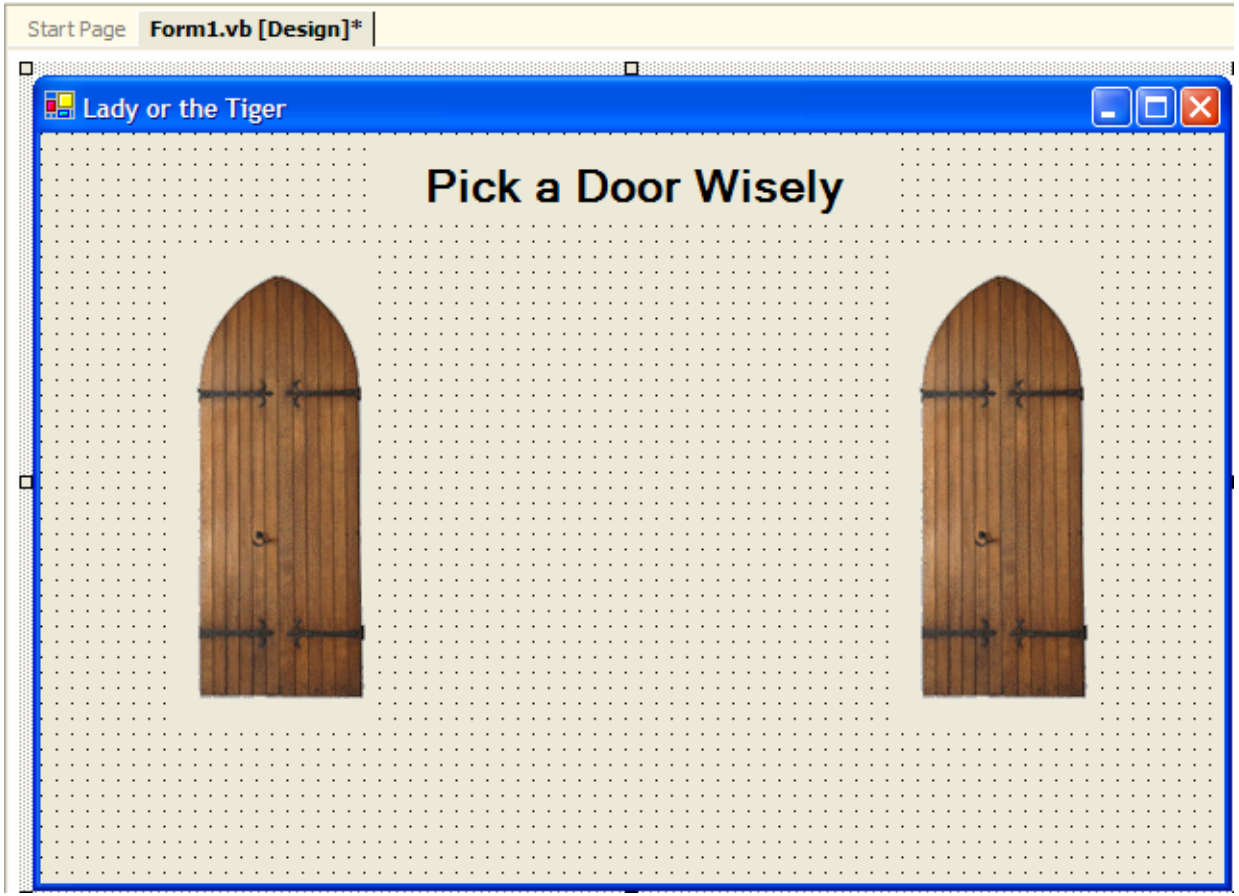
Step 5: Since the form is not large enough, increase its size by setting the `Width` and `Height` to `624, 424`, respectively. Click anywhere on the form a control is not already placed to select the form and display its properties.

Increase the size of the PictureBox by setting the `Height` and `Width` properties to 100, 248, respectively.

Chapter 2 – 1st Application

Add Other Door

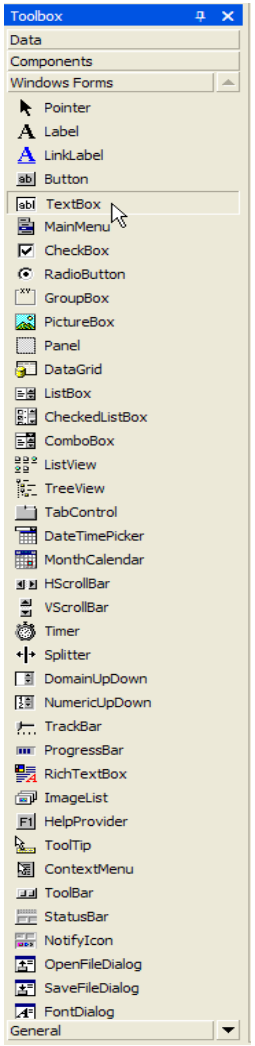
Step 6: The last step is to repeat the process and create another Picture Box, `picRightDoor`, with the same picture.



Chapter 2 – 1st Application

2.4 Text Box Control

By using a **text box control** you can place an area on the form where users of the application may enter any text he or she wishes.



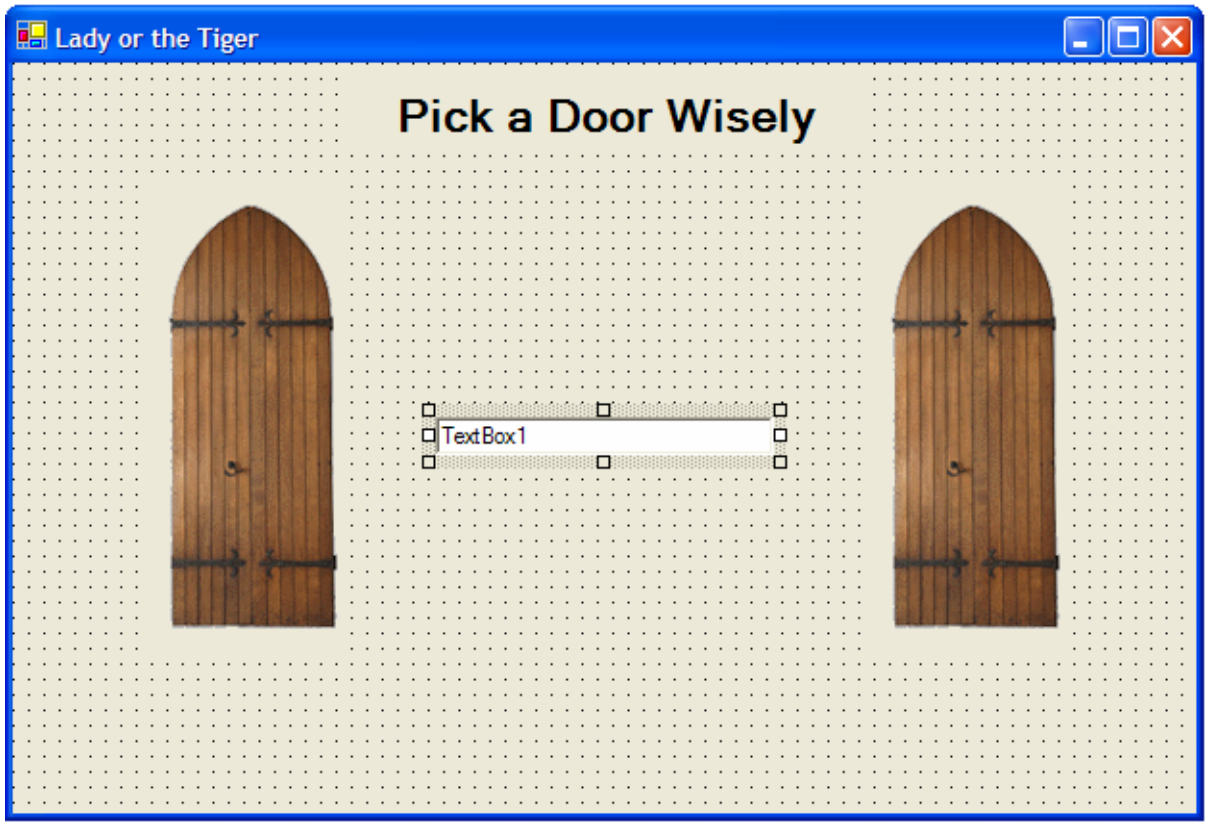
Select the Text Box From the Control Toolbox

Step 1: Select the `TextBox` control from the control toolbox.

Chapter 2 – 1st Application

Add Text Box to Form

Step 2: Place a text box on the form in the same manner as the other controls.



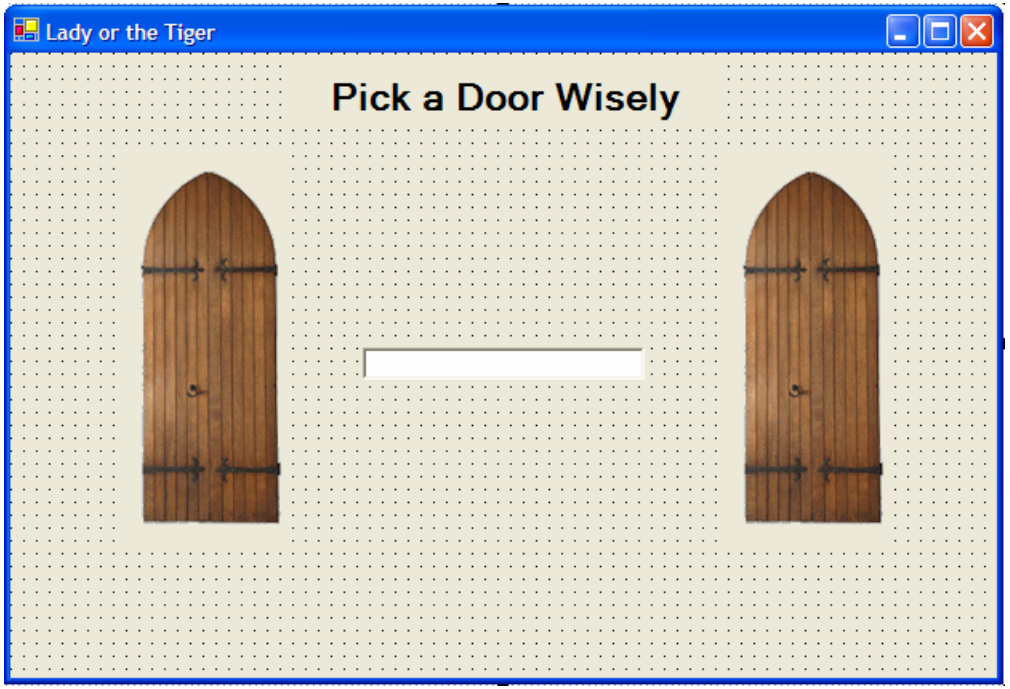
Chapter 2 – 1st Application

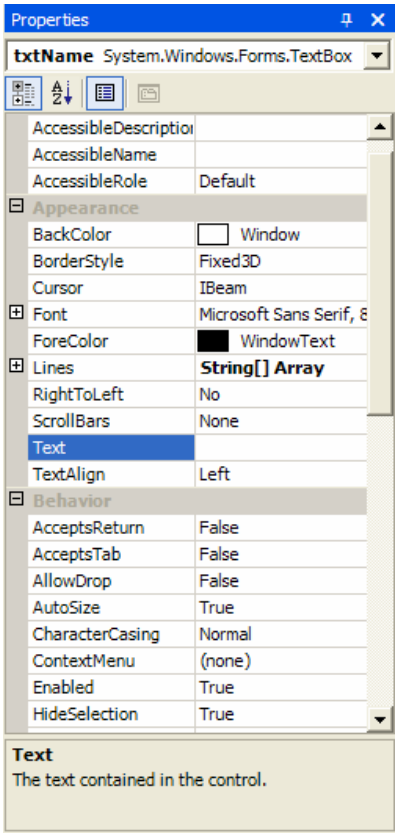
Setting the Name of the Control

Step 3: Set the `Name` property of the text box to `txtName` .

Clearing the Default Text

Step 4: Click on the `Text` property and erase `"TextBox1"`.

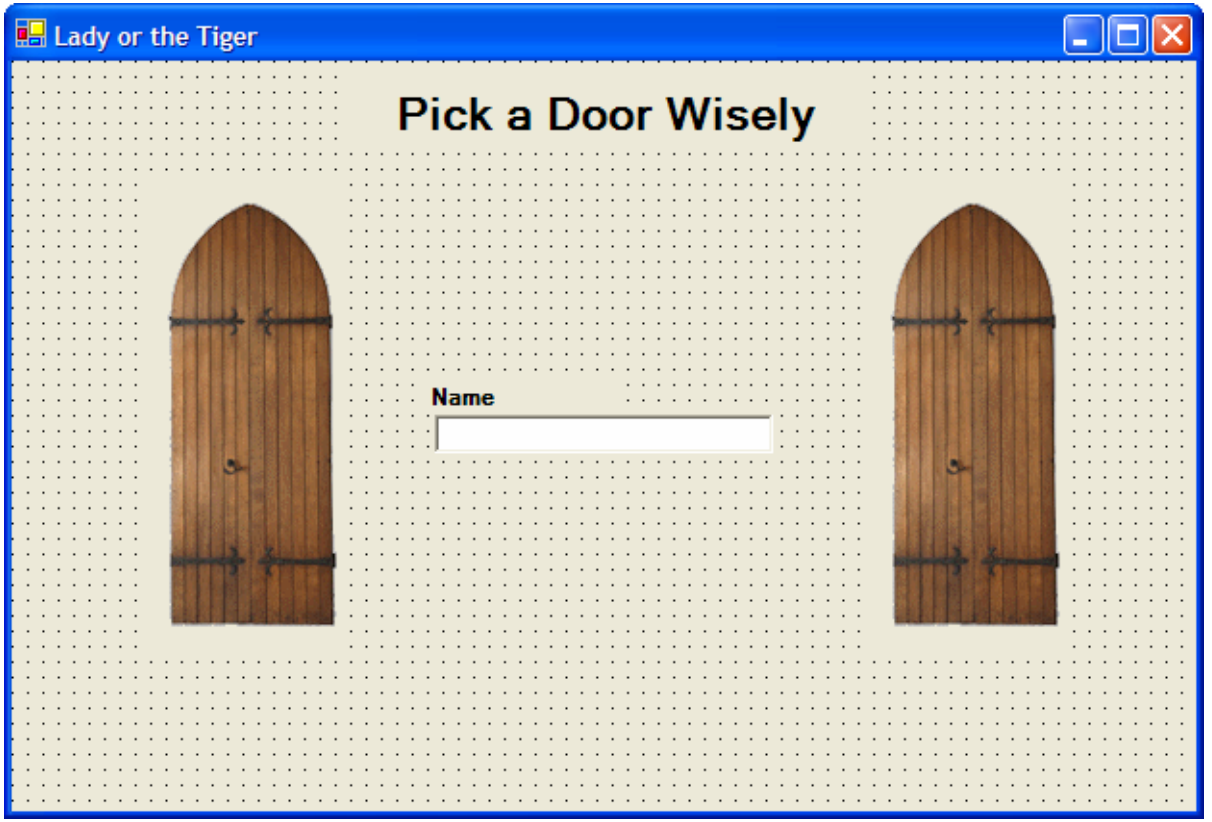




Chapter 2 – 1st Application

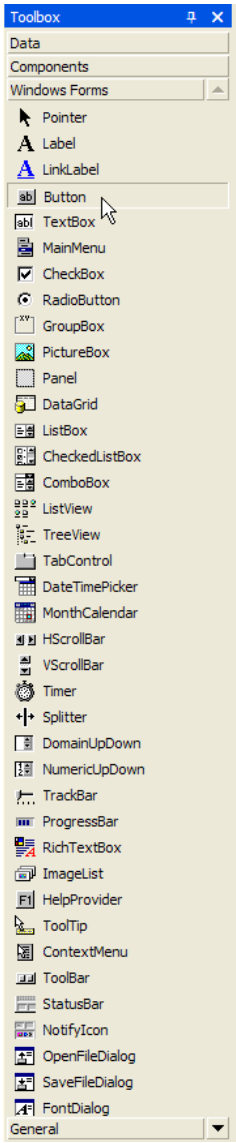
Adding a Label above the Text Box

Step 5: Add a Label control above the text box to indicate the text box is for the peasant's name. Set the `Name` of the label to `lblName`, the `Font Bold` to `True` and the `Text` to `Name`.



Chapter 2 – 1st Application

2.5 Button Control



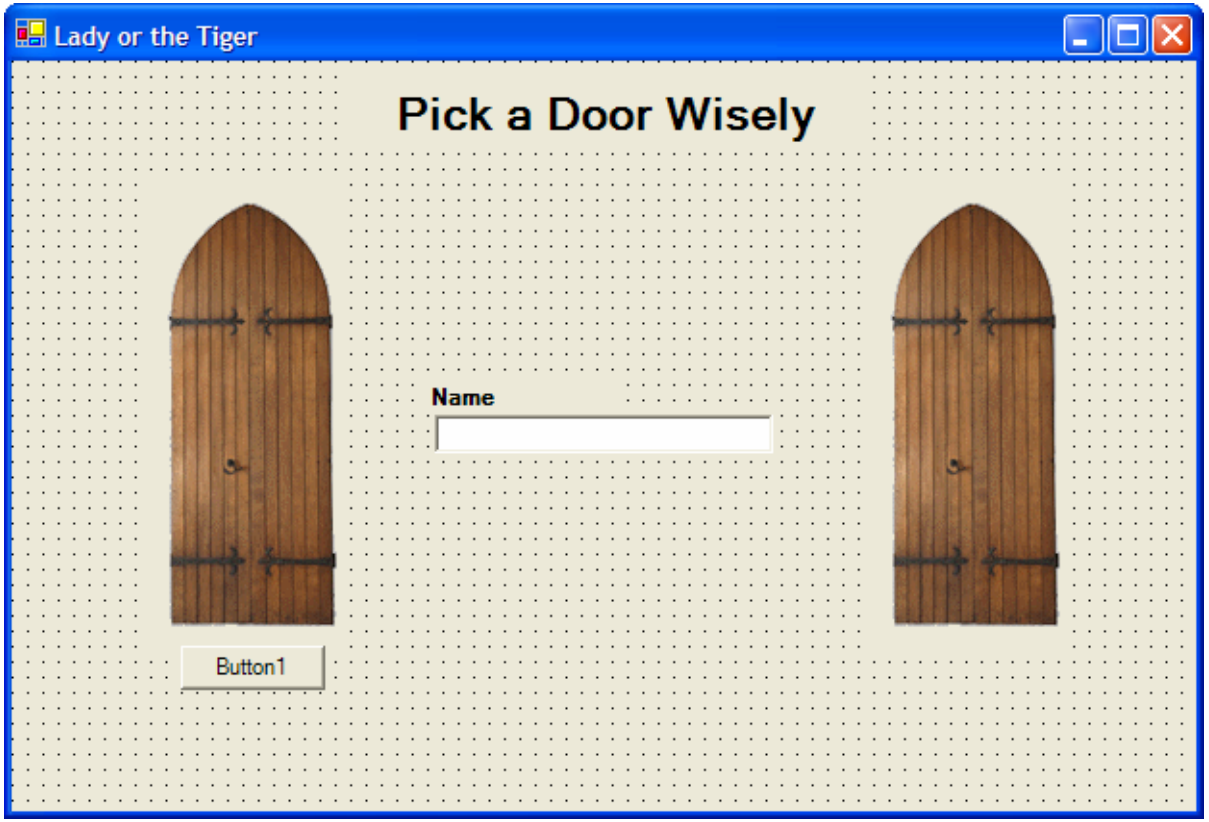
Select the Button Control From the Toolbox

Step 1: Select the Button control from the control toolbox.

Chapter 2 – 1st Application

Add Command Button to Form

Step 2: Place a Button on the form in the same manner as the other controls.



Chapter 2 – 1st Application

Setting the Name of the Control

Step 3: Set the `Name` property to `btnLeftDoor` .

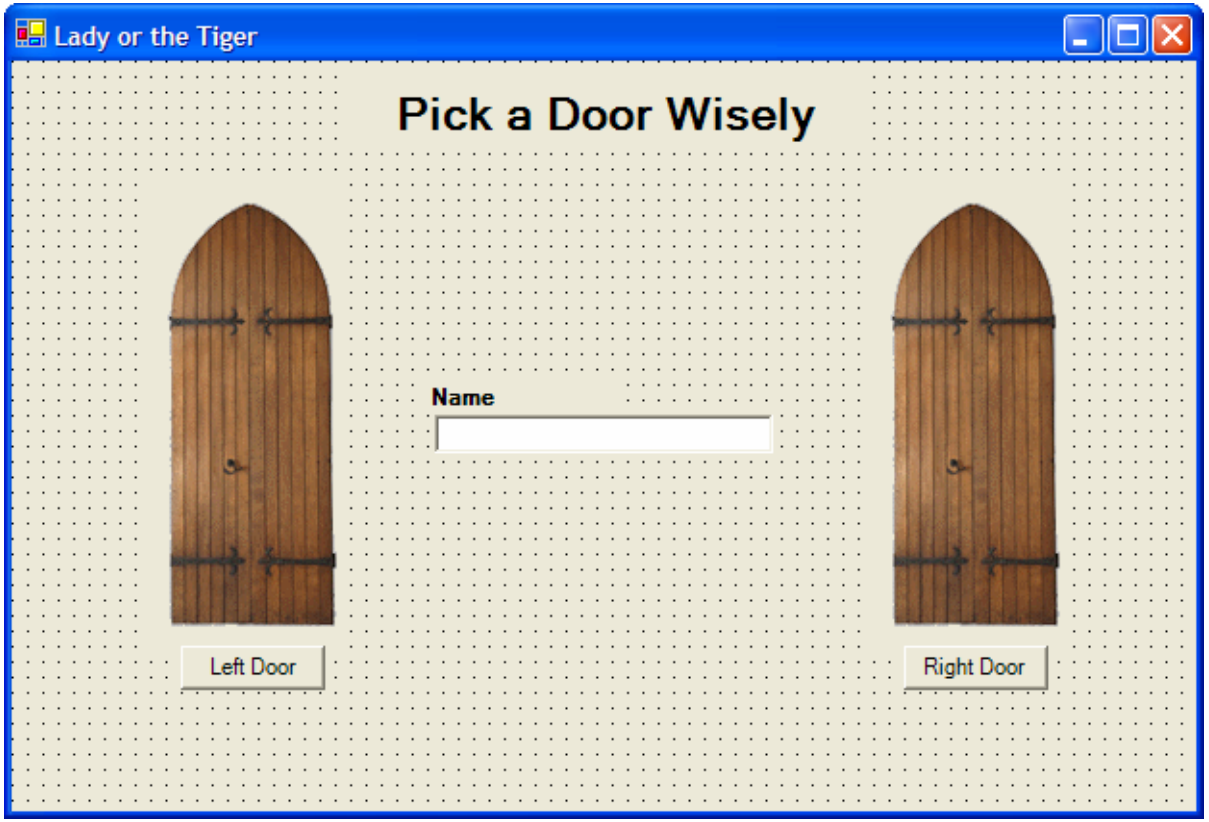
Clearing the Default Text

Step 4: Set the `Text` property to `"Left Door"`.

Chapter 2 – 1st Application

Add the Other Button

Step 5: The last step is to repeat the process and create another Button, `btnRightDoor`.



Chapter 2 – 1st Application

2.6 Basic Event Handling

You now will add functionality so that clicking on your button will cause the form to display either a young maiden or a tiger.

If the young maiden is displayed, the label should change to the person's name and the words " is `Innocent`". If the tiger appears, the label should change to the person's name and " is `Guilty`".

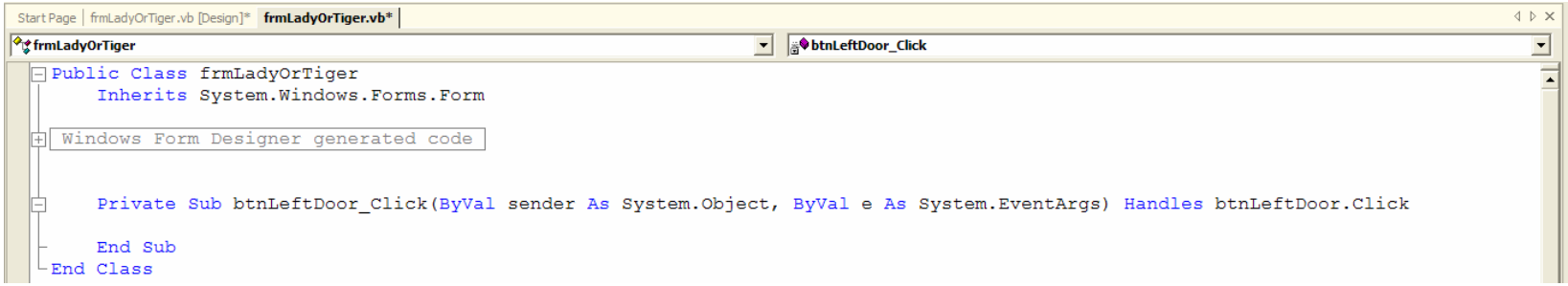
You will also notice whatever text is in the text box is erased.

This involves your first real coding. In this case, your coding will be triggered by an event.

Chapter 2 – 1st Application

Display the Code for the Button

Step 1: Double-click on the `btnLeftDoor` button to display the code.



The first few lines define the object you are working in. In this example you are creating a form called `frmLadyOrTiger`.

The first two words, `Public Class`, indicate that you are creating a template for an object that can be accessed by any object in the project.

The code for the event starts with the words `Private Sub`

`btnLeftDoor_Click()` indicates that this code is attached to the `btnLeftDoor` button and will be exected when a `Click` event occurs.

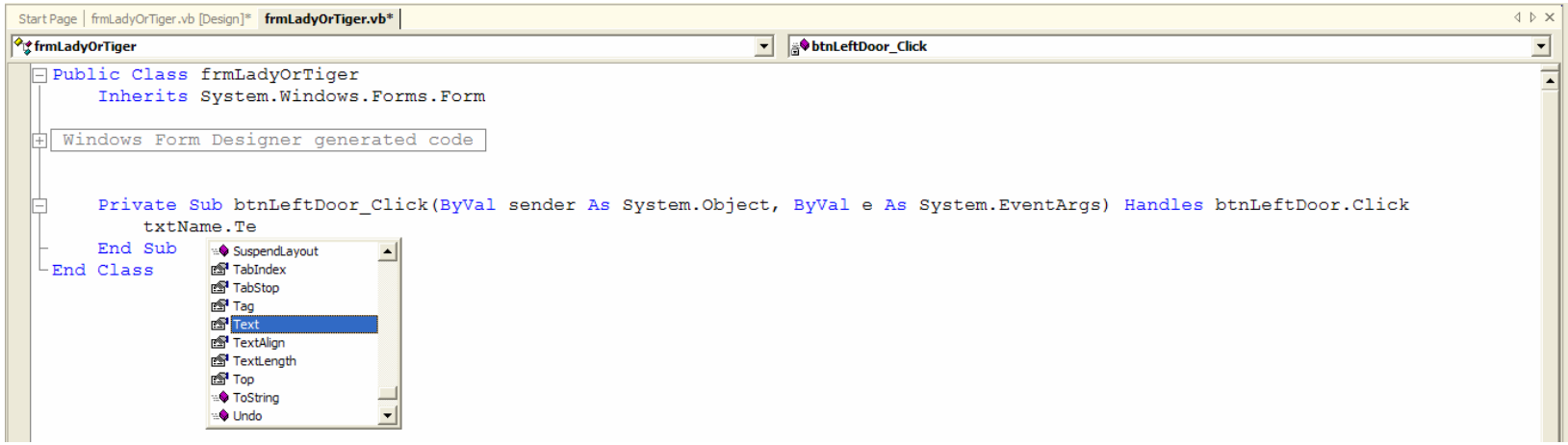
`End Sub` indicates the ending of the event.

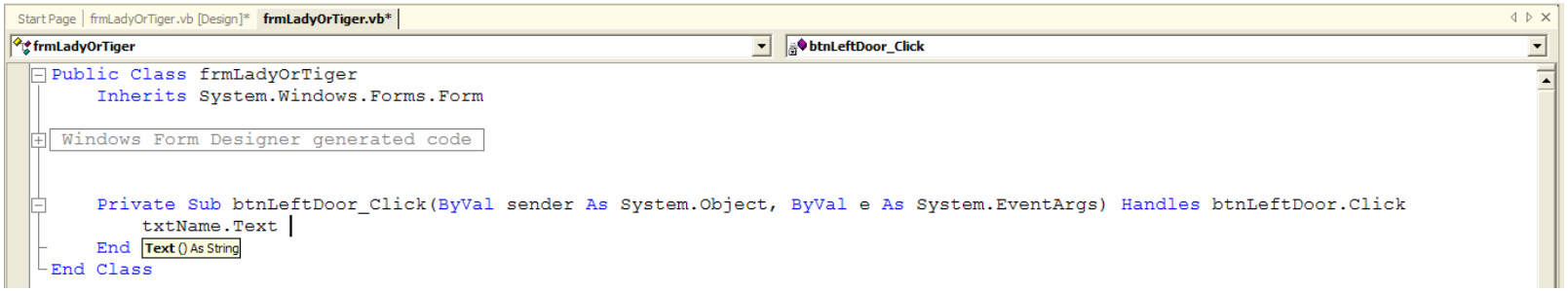
Chapter 2 – 1st Application

Clearing the Text Box

Step 2: Remove any text that is placed in the `txtName` text box by setting the `Text` property.

You can use the pull-down menu to select the property you wish to set or type it in.





Chapter 2 – 1st Application

Step 1 Continued: To cause a text box's `Text` property to change, type an equal sign followed by the new value you want to display in quotation marks

```
Start Page | frmLadyOrTiger.vb [Design]* | frmLadyOrTiger.vb*
frmLadyOrTiger                          btnLeftDoor_Click

Public Class frmLadyOrTiger
    Inherits System.Windows.Forms.Form

 Windows Form Designer generated code


    Private Sub btnLeftDoor_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnLeftDoor.Click
        txtName.Text() = ""
    End Sub
End Class
```

Now you can run your application by clicking on the Start button in the Standard toolbar. Type a name into the text box like “Michael” and click on the button `btnLeftDoor`. Watch how the text in the text box is removed.

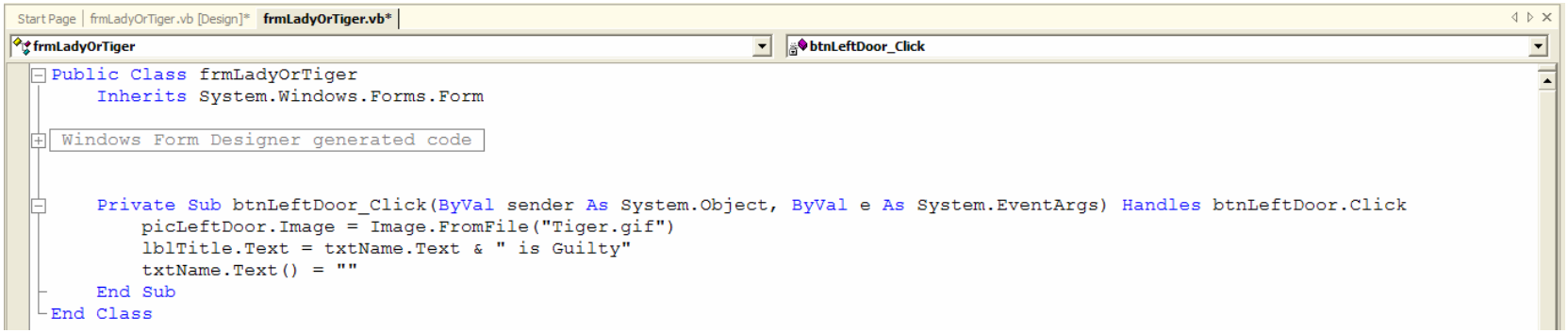
Chapter 2 – 1st Application

Changing the Picture and Label

Step 2 : Set the `Image` property of `picLeftDoor` to the new graphic.

Type `Image.FromFile("Picture name and path goes here")` to the right of the equal sign. In your case the picture name and path are `"Tiger.gif"`.

Change the `Label` control `lblTitle` to "`Michael is Guilty:`



In order to combine the name from `txtName` text box and the text "`is Guilty`", you must use a concatenation operation. The `&` in between `txtName.Text` and "`is Guilty`" will do that.

Chapter 2 – 1st Application

Now you can run your application. Click on the `btnLeftDoor` button to see the final result.

Chapter 2 – 1st Application

Code the Right Doors Button

Step 3 : Set the code for the `btnRightDoor` command button in a similar fashion.

```
Start Page | frmLadyOrTiger.vb [Design] | frmLadyOrTiger.vb
frmLadyOrTiger                          btnRightDoor_Click

Public Class frmLadyOrTiger
    Inherits System.Windows.Forms.Form

 Windows Form Designer generated code


    Private Sub btnLeftDoor_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnLeftDoor.Click
        picLeftDoor.Image = Image.FromFile("Tiger.gif")
        lblTitle.Text = txtName.Text & " is Guilty"
        txtName.Text() = ""
    End Sub

    Private Sub btnRightDoor_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnRightDoor.Click
        picRightDoor.Image = Image.FromFile("Lady.gif")
        lblTitle.Text = txtName.Text & " is Innocent"
        txtName.Text() = ""
    End Sub
End Class
```

Here is the final result of clicking the `btnRightDoor` button.

Chapter 2 – 1st Application

Saving the Project

You can save your project by either clicking on the Save icon in the Standard toolbar, or Selecting File and Save from the Menu bar.

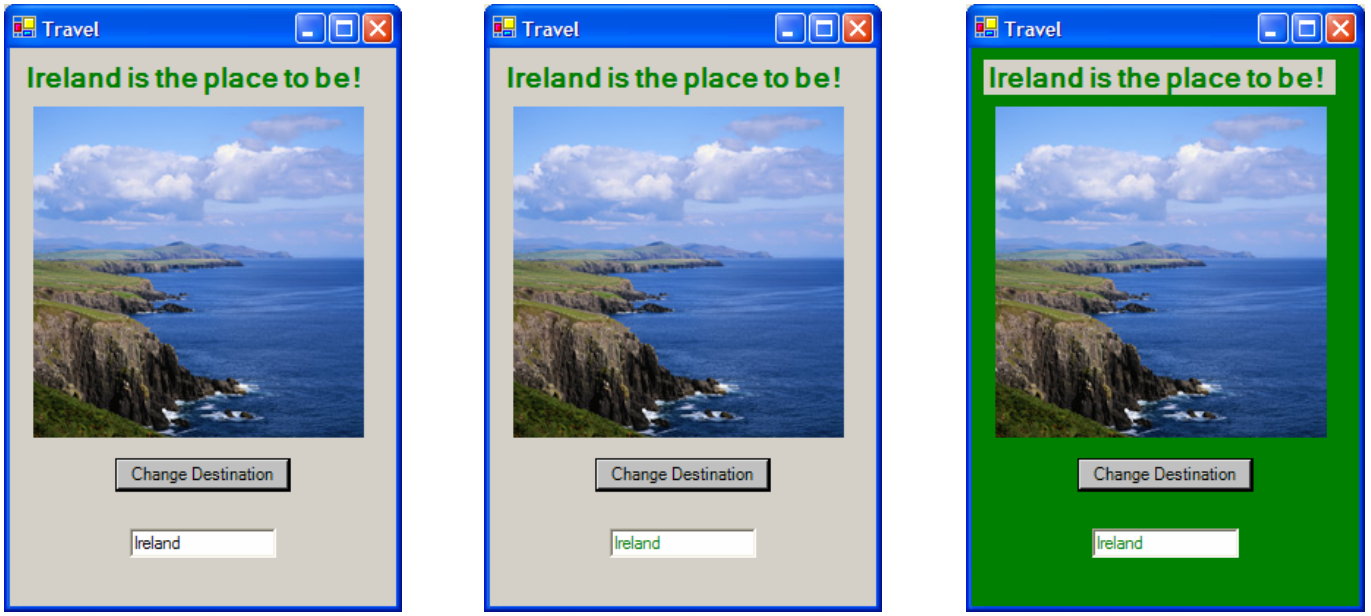
Chapter 2 – 1st Application

2.7 Use of Color

Visual Basic .NET gives you the ability to set the color of almost every control. Your use of color should be conservative, consistent, and logical.

Imagine if you were developing a travel application.

Which do you like best?



Which application looks the best is a matter of preference. Personally, I like the second one the best.

Chapter 2 – 1st Application

Changing Color

The two properties that were used in this example are `ForeColor` and `BackColor`. Each can be set interactively or programmatically. To set the color interactively, perform the following:

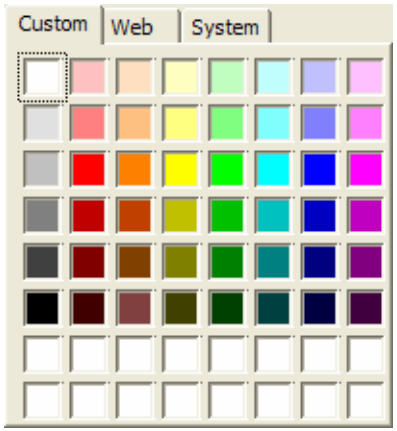
Step 1: Click on the control that you wish to change the color of.

Step 2: Click on the property you wish to change. (`ForeColor` or `BackColor`).

Step 3: Click on the drop-down arrow to get the `Pallet window` to appear.

Step 4: Click on the Custom tab of the pop-up window.

Step 5: Click on the color you wish to select:



Chapter 2 – 1st Application

Programatically Changing Color

We can also set colors programmatically by using any of the predefined Visual Basic .NET colors. If we wanted to set the vacation application to display Egypt instead of Ireland, we could use the following code:

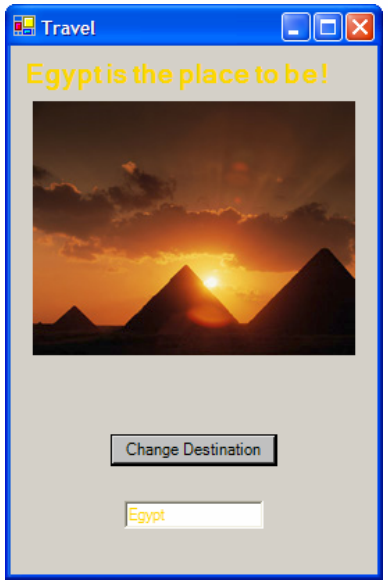
Start Page | frmTravel.vb [Design] | **frmTravel.vb**

frmTravel | (Declarations)

```
Public Class frmTravel
    Inherits System.Windows.Forms.Form

    Windows Form Designer generated code

    Private Sub btnChangeDestination_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnChangeDestination.Clic
        lblTitle.Text = "Egypt is the place to be!"
        picPhoto.Image = Image.FromFile("Egypt.jpg")
        txtName.Text = "Egypt"
        txtName.ForeColor = Color.Gold
        lblTitle.ForeColor = Color.Gold
    End Sub
End Class
```



Chapter 2 – 1st Application

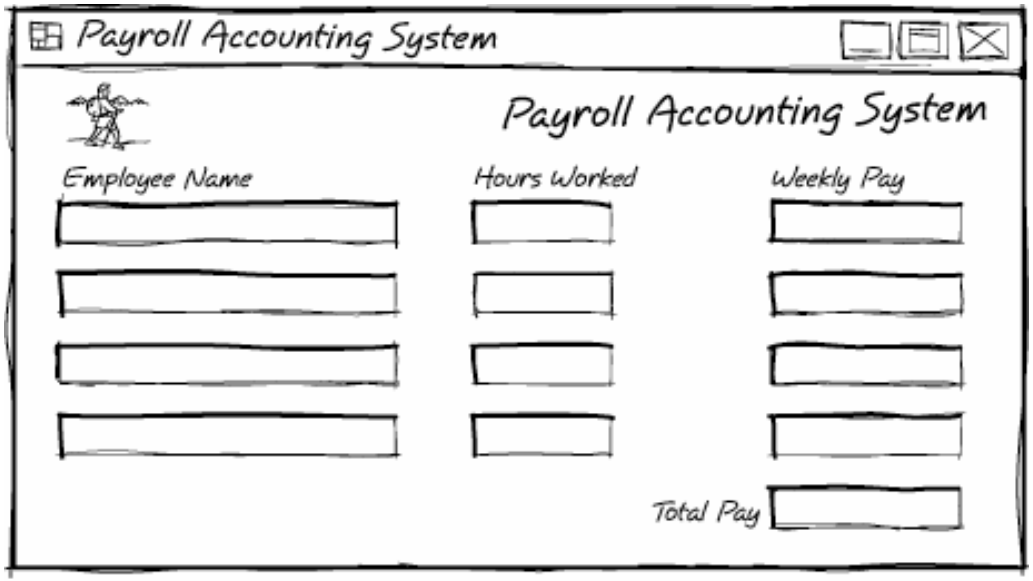
2.8 Case Study

Problem Description

A company that sells products on the Internet, Walking Promotions, wants to develop an application to track its financial data.

Create an application that allows the entry of the names of the employees, how many hours they worked that week, and a place to display their payment for the week as well as a total cost of payroll.

Beautify the form by adding the company's logo to the form as well as a title.



Chapter 2 – 1st Application

2.8 Case Study

Problem Discussion

While you do not know enough to create an application that will actually process the payroll, you can at least set up the user interface of the application.

It will require using text boxes, picture boxes, and label controls.

To make programming easier in the future, it is a good idea to name all of the controls something a little more specific than the default values.

This will make them more discernible later.

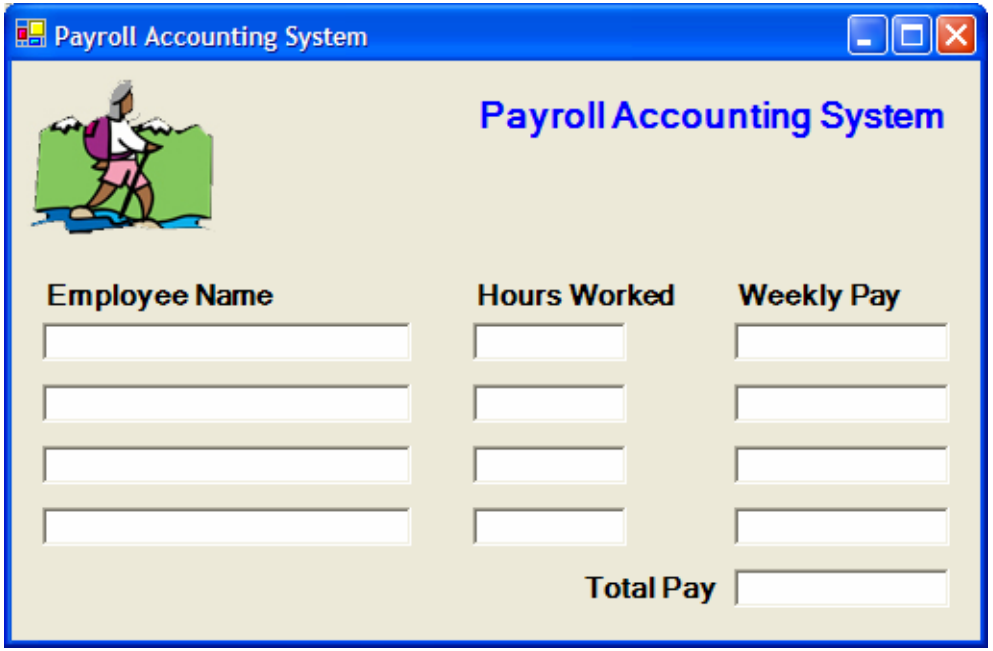
Chapter 2 – 1st Application

Problem Solution

The project requires a single form and a graphic file containing the logo.

The graphic file will be called `WalkingPromotionsLogo.jpg`.

While you have many options in how you lay out your solution, you want to come up with a simple, intuitive solution. Your completed application should look as follows:



Chapter 2 – 1st Application

Adding the Logo

To add the Walking Promotion's logo to a blank form, follow the steps that you used when adding a `PictureBox` control in Section 2.3.

Step 1: Select the `PictureBox` control from the control toolbar.

Step 2: Place the mouse pointer over the area of the form you wish to place the upper-left corner of the logo.

Step 3: Hold the mouse button down and drag the pointer to the lower-right corner where the logo will be placed.

Step 4: Release the mouse button and the `PictureBox` control will be placed on the form.

Step 5: Click on the `Image` property in the `Properties window` and click on the `WalkingPromotionsLogo.jpg` to select the appropriate graphic.

Step 6: Click on the `Name` property and change the `Name` property to `picLogo`.

Chapter 2 – 1st Application

Adding the Labels

You need to add a total of five labels to the form. Add the first four now and the last one after you add the text box controls so that they line up appropriately. To add the first control, perform the following steps:

Step 1: Select the `Label` control from the `toolbox`.

Step 2: Place the mouse over the area of the form you wish to place the upper-left corner of the `Label` control.

Step 3: Hold the mouse button down and drag the pointer to the lower-right corner where the `Label` control will be placed.

Step 4: Release the mouse button and the `Label` control will be placed on the form.

Step 5: Click on the `Text` property in the `Properties window` and type "`Payroll Account System`".

Step 6: Click on the `Font` property in the `Properties window` and set the `Size` to `14` and the `Bold` property to `True`.

Step 7: Click on the `Name` property and change the `Name` property to `lblTitle`.

Chapter 2 – 1st Application

Adding the Labels Continued

The following are the properties for the other three label controls you need to add to the form:

Name: lblEmployeeName	Name: lblHoursWorked	Name: lblWeeklyPay
Text: Employee Name	Text: Hours Worked	Text: Weekly Pay
Font: Size=11, Bold=True	Font: Size=11, Bold=True	Font: Size=11, Bold=True
Size: Width=128, Height=23	Size: Width=112, Height=23	Size: Width=104, Height=23
Location: X=16, Y=112	Location: X=240, Y=112	Location: X=376, Y=112

Chapter 2 – 1st Application

Adding the Text Boxes

You need to add a total of 13 text box controls to the form. Remember to line them up when you are finished. To add the first text box perform the following steps:

Step 1: Select the `TextBox` control from the control toolbar.

Step 2: Place the mouse over the area of the form you wish to place the upper-left corner of the `TextBox` control.

Step 3: Hold the mouse button down and drag the pointer to the lower-right corner of where the `TextBox` control will be placed.

Step 4: Release the mouse button and the `TextBox` control will be placed on the form.

Step 5: Click on the `Name` property in the `Properties window` and type `txtEmployee1`.

Step 6: Click on the `Text` property in the `Properties window` and clear the default text so that nothing is displayed in the text box when you run the application.

Chapter 2 – 1st Application

Adding the Text Boxes Continued

The following are the properties for the remaining text box controls you need to add to the form:

Name: txtEmployee2	Name: txtEmployee3	Name: txtEmployee4
Text:	Text:	Text:
Size: Width=192, Height=20	Size: Width=192, Height=20	Size: Width=192, Height=20
Location: X=16, Y=168	Location: X=16, Y=200	Location: X=16, Y=232
Name: txtHours1	Name: txtHours2	Name: txtHours3
Text:	Text:	Text:
Size: Width=80, Height=20	Size: Width=80, Height=20	Size: Width=80, Height=20
Location: X=240, Y=136	Location: X=240, Y=168	Location: X=240, Y=200
Name: txtHours4	Name: txtWeeklyPay1	Name: txtWeeklyPay2
Text:	Text:	Text:
Size: Width=80, Height=20	Size: Width=112, Height=20	Size: Width=112, Height=20
Location: X=240, Y=232	Location: X=376, Y=136	Location: X=376, Y=168
Name: txtWeeklyPay3	Name: txtWeeklyPay4	Name: txtTotalPay
Text:	Text:	Text:
Size: Width=112, Height=20	Size: Width=112, Height=20	Size: Width=112, Height=20
Location: X=376, Y=200	Location: X=376, Y=232	Location: X=376, Y=264

Chapter 2 – 1st Application

Adding the Final Label

The following are the properties for last label control you need to add to the form:

Name: lblTotalPay
Text: Total Pay
Font: Size=11, Bold=True
Size: Width=72, Height=23
Location: X=296, Y=264

Chapter 2 – 1st Application

Coach's Corner

Tab Index

When you create applications, you often add controls in the order you think about them, not necessarily in the order the user wishes to enter data.

When you place controls on a form, they receive a default `TabIndex` that follows the order they are added to the form.

A well-designed program should start with the focus in the upper-left corner. When the `<TAB>` key is pressed, you should move to the next logical control.

You can control the tab index of the form by setting the `TabIndex` property of each control to the logical sequence starting with 0.

Tab Stop

Often you do not wish a control to be in the tab order at all.

You should set the `TabStop` property of the control to `False`.

A `False` setting will remove the control from the tab loop.

A `True` setting will place it in the tab loop. PictureBox and Label controls usually do not have a tab order.