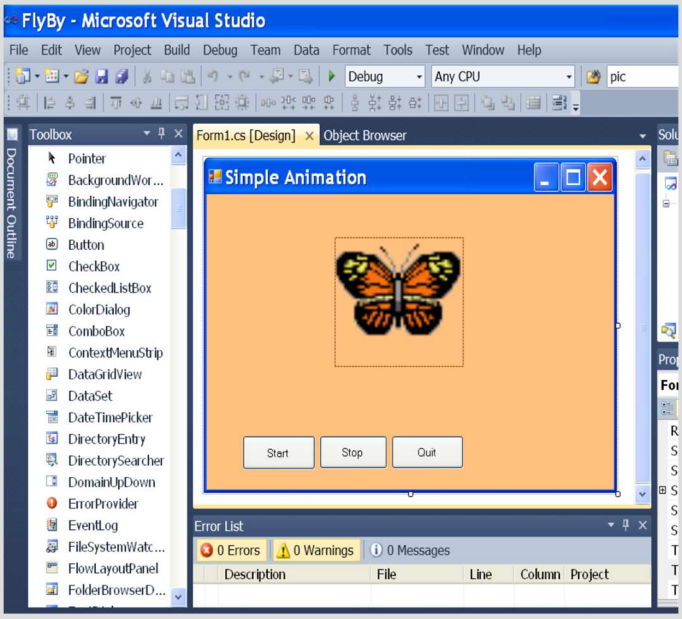
Windows Programming

bucks new university

.NET Windows Programming Using C#



Application Programming (CO453)

Part B Weeks 5-8

C# Windows Programming
Unit 4

Multi-Form Projects

Introduction

- Our Windows projects so far have used just one form to deliver most of the action.
- When we added forms to a project, this was mostly for cosmetic reasons.
  - for example we used a Splash Screen to introduce our Calculator project.
- In this section we shall try to make better use of forms and this will require us to be able to pass information from one form to another.
- We shall create a simple project to allow us to calculate payments in a restaurant

4.1 Starting the Tipster Project

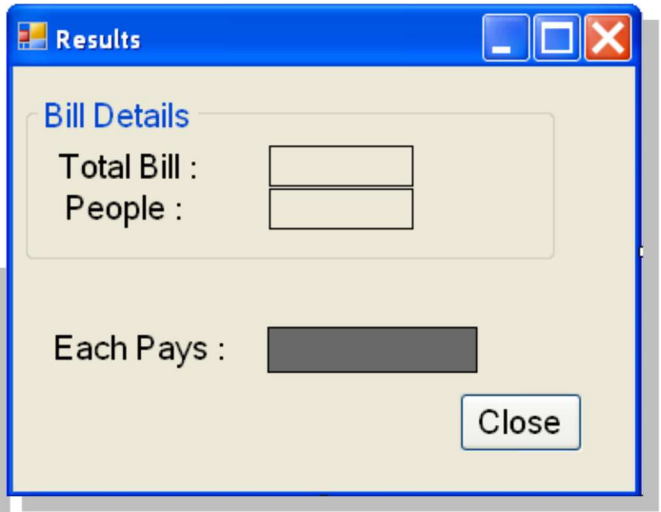
- Start a new Windows project called **Tipster**
  This will first allow you to enter a bill total and work out the cost per person.
- Design the main data entry form so that it looks something like the one shown here, with a textbox, listbox, picturebox, 2 buttons and some labels.
- The listbox should allow you to select a number from 1 to 10

Task 4.1: Tipster 1

- Create the Data Entry Form for the Tipster project as shown above

4.2 Adding a Form to Display Results

- Add a new Results form to your Project
- Design this form with labels, a button and a GroupBox so that it looks something like the one shown here.
- At the moment there will be no numbers displayed



Task 4.2: Tipster 2

- Create the Results Form for the Tipster project as shown here
- Get this to show up when the **Calculate** is clicked and close when the **Close** button is clicked
- For help see the next page note

How to Show a new Form (also see Unit 1)

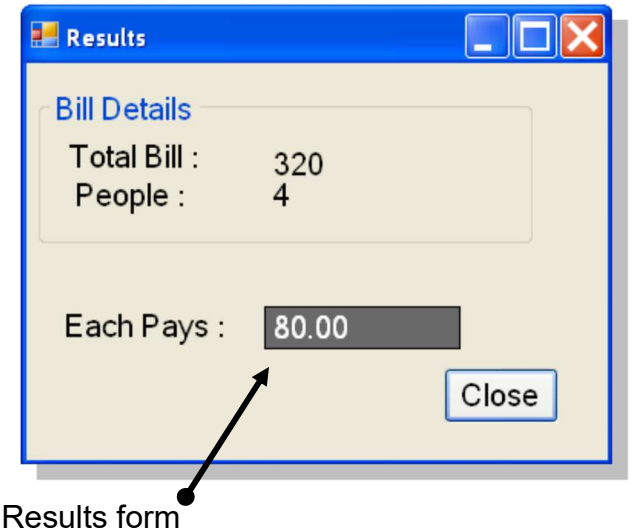
- First you need to create a new form Object using the form's Class name e.g. if the form is named frmResults then this could be done like this:
  frmResults newForm = new frmResults();
- Then you can show your new form object like this:
  newForm.Show();
- This code is added to a button's Click method so it runs when the button is clicked
- Note: The current form can be closed by using **this.Close();**

4.3 Moving Data between Forms

Now we want to be able to move data from our main form to the display form .. see the example here.

Important Note

- By default when you add objects to a form they are set to **Private**
- This means that they are not available to other forms.
- To make them available, we need to set their **Modifiers** property to **Public**



- Select one of the data labels on the Results form
- In the **Properties** window, find the **Modifiers** property and change this to **Public**
- Repeat this for the two other data labels.
- Now program the **Calculate** button to also copy the Bill amount from the textbox to the label on the new form. For example:
  newForm.lblTotal.Text = txtBill.Text;
- Add more code to also copy the people number from the listbox to the new form.
- Finally add code to display the amount each person is to pay .. this is done by dividing the total bill by the number of people. Note that you will have to convert the Text to a number (double) before calculating. Here is some information from chapter 2 .. use this to help you add the necessary code :

```
double  n1, n2, answer;                    // define 3 number variables
n1 = Convert.ToDouble(txtFirstNum.Text);   // convert the first number
n2 = Convert.ToDouble(txtSecNum.Text);     // convert second number
answer = n1/n2;                            // divide them
lblResult.Text = answer.ToString();        // put string into result label
```

Task 4.3: Tipster 3

- Code the **Calculate** button so all Results are displayed correctly on the form
- Program the **Quit** and **Close** buttons too

4.4 Using try/catch .. to trap errors

It is possible that someone might type something daft like 'smsm' by mistake for the bill amount. We can trap errors or exceptions using the very useful try .. catch .. instruction.

- The **try .. catch ..** instruction has many different versions but we shall just consider the simplest one here.
- The basic structure is this:

```
try
{
        // put instructions here that you want to check
}
catch
{
        // display an error message here if an exception occurs
}
```

- Here is an example:

```
try
{
        double n1, n2, answer;
        n1 = Convert.ToDouble(txtBill.Text);
           // the rest of the calculations also go here
}
catch
{
        MessageBox.Show("Error in Data Entry");
}
```

- The **catch** block is only done if there is an error in the **try** block
- Here is the possible result:

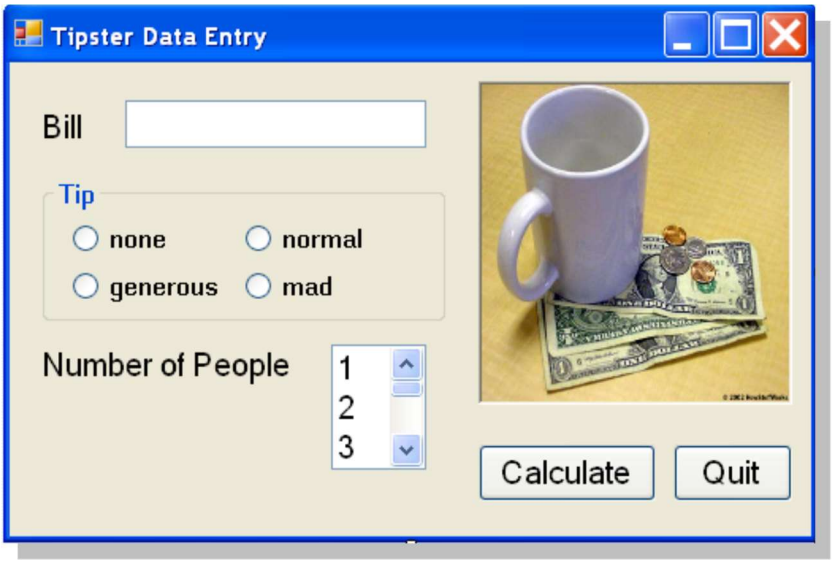
Task 4.4: Using try/catch

- Program the **Calculate** button using **try** and **catch** to trap errors like this

4.5 Tipping

Now add a GroupBox and 4 radiobuttons as shown to allow the user to select a tip to be added to the Bill.

- None is just that .. 0 (stingy or what?)
- Normal is 10% of the bill
- Generous is 15% of the bill
- Mad is 20% of the bill



Task 4.5: Tipping

- Complete the form to allow tipping as shown above
- Add code so that the appropriate tip is added to the bill (you may want to look at chapter 5 for help)
- Add another **Bill + Tip** label to the Results form and ensure that this new total is now used to calculate the amount per person

Task 4.6: Test Plan

- Design a Test Plan for your Tipster project and use it to test your final product

Unit 4: Independent Study

Bucks New University is hosting a conference called "Education in the 21st Century". Teachers from three local schools have been invited to attend.

- Attendees can have lunch and/or dinner
- Each school is charged £50 per person to attend the conference and £10 for lunch and £15 for dinner
- You have been asked to design a prototype program to collect data from each teacher as they arrive at reception and produce bills depending on choices made.

Program Specification

There are 3 forms in the project. Your task is to design the main data entry form, the bill form and the totals form. Then link these together.

- All forms are well designed with pictures, good layout and choice of fonts and colours

4.7: The Data Entry Form

Design a form with the following functionality:

1. The attendee should be able to put their name into a textbox
2. They should be able to choose their school from a dropdown list
3. The various dining options should be available via radiobuttons or checkboxes
4. A **Quit** button on the form quits the program
5. A **See Bill** button allows the user to see their Bill (see 6.8)
6. A **Totals** button allows university staff to see the totals (see 6.9)

4.8: The Bill Form

This form should have a suitable BackgroundImage and the following functionality:

1. A button returns back to the main Data Entry form .. this can be done by simply closing the form
2. The name of the teacher and school name from the main form are displayed in a heading
3. The choices are also displayed together with a Total Bill.
4. An **Accept Bill** button asks the teacher if they are happy with their choices and if so, various totals and counts are updated (see 6.9)

4.9: The Totals Form

This form should have the following functionality:

1. There is a Back button that closes the form
2. The total number of teachers attending the conference so far is displayed
3. The individual school totals are displayed (see hints)
4. The total bills for each school are displayed

Some Hints and Reminders

1. Most of the programming will need to be done on the main **DataEntry** form.
2. You will need to define a number of variables to hold the various **totals** and bills, etc. You should do this near the beginning of the form code, after the form header.

e.g.

```
namespace Project
{
    public partial class frmDataEntry : Form
    {
            int totalDelegates = 0;
```

3. By default when you add objects like textboxes etc. to a form they are set to **Private**
4. This means that their contents are not available to other forms.
5. To make them available, you must to set the **Modifiers** property to **Public**

Some Extra Useful C# .NET Stuff

Some Display Instructions

```
_____.BackColor = Color.Blue;          // set a background colour
_____.ForeColor = Color.Yellow;        // set a foreground colour
_____.Text = "Hello " + name ;         // set the Text property
```

Delay for a time

```
System.Threading.Thread.Sleep (1000);
// gives a delay of 1 second (1000 milliseconds)
```

The Math.Pow() function

Use the Math.Pow() function to return the power of a number: e.g.

```
cube = Math.Pow(number, 3);
```

The Math.Abs() function

This function returns the value of a number, ignoring any + or - sign e.g.

```
value = Math.Abs(number);
```

Converting Text into upper case

```
lblName.Text = lblName.Text.ToUpper() ;
// converts a label's Text into upper case (e.g. yes becomes YES)
```

Random Number Generation

- First we must create a new object from the Random class .. e.g.

```
Random  r = new  Random();          // creates a new object called r
```

- Then you can use **r.Next()** to pick the next number e.g.

```
n = r.Next(6) + 1;          // puts either 1,2,3,4,5 or 6 into n
```

Output of Decimal Places

If you want to display a double type of variable (num) to 2 decimal places:in a label:

```
lblAnswer.Text = "The answer is " + num.ToString("0.00") ;
```

Number Conversion

e.g. Convert text from a textbox to the right type using Convert e.g.

```
num = Convert.ToDouble(txtAmount.Text);    or
num = Convert.ToInt32(txtAmount.Text);
```

Closing Down

```
Application.Exit();
```

Graphics Summary (after first getting a Graphics object g)

Clearing the Screen

```
g.Clear (BackColor);          // clear using the current backcolor
```

Rectangles and Squares

```
g.DrawRectangle (myPen, x, y, w, h);      or   g.FillRectangle(myBrush, x, y, w, h);
```

Ellipses and Circles

```
g.DrawEllipse (myPen, x, y, w, h);        or   g.FillEllipse(myBrush, x, y, w, h);
```

Polygons

```
g.DrawPolygon (myPen, shape);             or   g.FillRectangle(myBrush, shape);
```

Lines

```
g.DrawLine (myPen, x, y, w, h);
```

Text

```
g.DrawString (s, myFont, myBrush, x, y);
```

Pens and Brushes (for drawing and filling graphics)

```
Pen  myPen = new Pen(Color.Green, 5);          // create a new green pen, size 5
Brush  myBrush = new SolidBrush(Color.Red);    // create a new red brush
```

RGB Colours (up to 256 values each for Red Green and Blue combinations)

```
this.BackColor  = Color.FromArgb(100, 50, 10);      // set a RGB background for this form
```

Appendix B:
Assessment of CO453 Application Programming

1. The module is assessed by coursework that consists of a series of directed study exercises and programming projects that must be recorded in a logbook.

2. The logbook must be an e-book and should contain your designs, algorithms, test plans, source code and results of your work.

3. The directed study includes **independent study tasks** and **programming projects**.

4. The **classwork** component of the directed study is assessed each week in your practical sessions. You **MUST** be observed doing the classwork in the computer laboratories during these timetabled sessions. You must record your classwork in a logbook and this will be presented for inspection at designated times. Your attendance and achievement will be recorded weekly.

5. The **independent** component of the directed study is your own unaided work. This work must be recorded in your logbook. The **independent** directed study is assessed when logbooks are submitted

6. The **programming projects** are your own unaided work and must be recorded in your logbook. They are assessed at various times during the timetabled practical session.

7. The assessed directed study is contained in several directed study packs.
   The weighting of the assessment to the final grade will be in the order of the following:

100% C# .NET Windows, Independent Study and C# .NET Windows Project –

15%; C# .NET Windows, Independent Study 60%; and C# .NET Windows Project 15%; contribution.)

Log Books

For the module you should submit (unless otherwise instructed):

- Clearly numbered Task headings
- Simple description of task(s)
- Sample Screen Shots
- Fully commented Source code of relevant sections using highlighter pen to show added code where necessary
- Comments on problems encountered etc.

Grade related criteria for Programming - CO453

A	Where the student has demonstrated clear evidence of an excellent understanding of the theories and principles together with a high degree of analytical accuracy, good design skills, implementing fully tested solutions that show reliability, maintainability, readability and minimal complexity and correct form of presentation skills. *To acquire the knowledge and skills to demonstrate the above the student will normally be expected to attend the lecture and practical sessions and attempt at least 85% of directed study for each week.*
B	Where the student has demonstrated clear evidence of a good understanding of the theories and principles together with a good analytical ability, good design skills, implementing solutions that show reliability, maintainability, readability and minimal complexity and correct form of presentation skills. *To acquire the knowledge and skills to demonstrate the above the student will normally be expected to attend the lecture and practical sessions and attempt at least 75% the directed study for each week.*
C	Where the student has demonstrated a reasonable understanding of the theories and principles together with a reasonable analytical ability, design skills, implementing solutions that appreciate the need for reliability, maintainability, readability and minimal complexity and reasonable presentation skills. *To acquire the knowledge and skills to demonstrate the above the student will normally be expected to attend the lecture and practical sessions and attempt at least 66% of the directed study for each week.*
D	Where the student has demonstrated an understanding of the theories and principles of analysis, design, implementation and presentation skills. *To acquire the knowledge and skills to demonstrate the above the student will normally be expected to attend the lecture and practical sessions and attempt at least 50% of the directed study for each week.*
E	Where the student has made a genuine attempt to acquire the knowledge and skills but requires further application and study to demonstrate an understanding of the theories and principles of analysis, design, implementation and presentation skills. *In order to demonstrate a genuine attempt the student will normally be expected to attend the lecture and practical sessions and attempt at least 40% of the directed study.*
F	Where the student has clearly not acquired sufficient knowledge and skills and not attempted or coped with the directed study with any degree of competence regarding theories, principles, analysis, design, and implementation and presentation skills or where the student has NOT attended for assessment or where the student has copied work from an alternative source.

Module Name and code	Application Programming CO453
Staff: Richard Jones, Richard Mather, Carlo Lusuardi & Nick Day	

Learning Outcomes:

- Analyse a simple requirement in a structured manner
- Design, document, implement and test reliable, maintainable programs as solutions to simple problems
- Use structured techniques of design and implementation and good documentation practice.
- Use software development tools.

Teach WK	Uni WK	LECTURE/TUTORIAL	PRACTICAL
1	18	**C# (Console) 4 Methods and Parameters**	C# Directed Study Unit 4
2	19	**C# (Console) 5 Arrays**	C# Directed Study Unit 5
3	20	**C# (Console) The Project**	C# Directed Study Project Unit
4	21	**Workshop**	
5	22	**Windows C#1 Introduction & Splash Screen**	C# Directed Study Unit 1
6	23	**Windows C#2 SPS Game**	C# Directed Study Unit 2
7	24	**Windows C#3 Other .NET Controls**	C# Directed Study Unit 3
8	25	**Windows C#4 Multiform projects**	C# Directed Study Unit 4
	26-28	**Spring Recess – (Easter)**	
10	29	**Workshop**	
11	30	**Windows C#5 Animation**	C# Directed Study Unit 5
12	31	**Windows C#6 Graphics**	C# Directed Study Unit 6
13	32	**Windows C#7 Web Programming**	C# Directed Study Unit 7
14	33	**Windows C#8 The .Net Project**	C# Directed Study Unit 8
15	34	**Workshop**	

Course Texts:

Comprehensive Course Notes are provided

Bradley & Millspaugh, ***Programming in C#***, 2010, pub: McGraw Hill

Deitel & Deitel, ***Visual C# 2010 How to Program***, 2011, pub: Pearson