

The background of the slide is a light gray gradient. It is decorated with numerous realistic water droplets of various sizes. Some droplets are large and prominent, while others are small and scattered. They are primarily located in the top-left, bottom-right, and bottom-center areas, with a few smaller ones near the top center.

PROGRAMMING IN VISUAL BASIC

CMPT 110

CHAPTER 2 – VISUAL BASIC CONTROLS, AND EVENTS

SCHNEIDER, 10TH ED.

2.1 AN INTRODUCTION TO VISUAL BASIC

2.2 VISUAL BASIC CONTROLS

2.3 VISUAL BASIC EVENTS

OBJECTIVES FROM UNIT 2 STUDY GUIDE

- TO UNDERSTAND THE CONCEPT OF A "**CONTROL OBJECT**" AS IT PERTAINS TO WINDOWS AND PROGRAMMING IN VISUAL BASIC.
- TO LEARN **HOW TO LAUNCH THE VISUAL STUDIO** INTEGRATED DEVELOPMENT ENVIRONMENT AND CREATE A NEW PROJECT.
- TO FAMILIARIZE YOURSELF WITH THE **LABEL, TEXTBOX, BUTTON, AND FORM CONTROL OBJECTS** AND THE **PROPERTIES** ASSOCIATED WITH THEM.
- TO UNDERSTAND THE IMPORTANCE OF PREPARING A "**REQUIREMENTS DEFINITION**" FOR GATHERING INFORMATION THAT WILL BE NEEDED IN A PROJECT AND HOW TO CREATE ONE.
- TO BE ABLE TO DESIGN SIMPLE WINDOWS INTERFACES USING THE BASIC CONTROL OBJECTS.

PREAMBLE

EVENT-DRIVEN AND OOP

VISUAL BASIC HAS BEEN DESCRIBED AS
AN "EVENT-DRIVEN" AND AN "OOP LANGUAGE".

WHAT DO THESE TERMS MEAN?

EVENT-DRIVEN AND OOP

AN **"EVENT-DRIVEN" PROGRAMMING LANGUAGE** IS ONE WHERE PROGRAMS ARE WRITTEN TO RESPOND TO ACTIONS TAKEN IN A WINDOWS ENVIRONMENT. THESE ACTIONS, CALLED "EVENTS," INCLUDE ALL THE TYPICAL THINGS THAT A USER CAN DO IN A WINDOW WITH A KEYBOARD AND MOUSE, INCLUDING:

- CLICK ON A MOUSE BUTTON.
- CLICK ON A VIRTUAL BUTTON IN A WINDOW.
- SCROLL HORIZONTALLY OR VERTICALLY THE TEXT IN A WINDOW.
- ENTER DATA THROUGH AN INPUT FIELD.
- DISPLAY TEXT OR IMAGES IN A WINDOW.
- CLICK ON RADIO-BUTTONS OR CHECK BOXES.

EVENT-DRIVEN AND OOP

- AN **"OBJECT-ORIENTED" PROGRAMMING LANGUAGE** [COVERED IN THE LAST PPT] IS ONE WHERE PROGRAMS ARE WRITTEN TO CREATE AND MANIPULATE VIRTUAL REPRESENTATIONS OF REAL-WORLD COLLECTIONS OF OBJECTS. THE OBJECTS THAT WE MAY WISH TO REPRESENT (OR "MODEL") MAY BE AS COMPLEX AS A PERSON OR AS SIMPLE AS A CHECK-BOX IN A WINDOW.
- HOW THESE OBJECTS ARE TO BE REPRESENTED IN A COMPUTER PROGRAM MUST USUALLY BE DEFINED BY A PROGRAMMER SUCH AS YOU. YOU MUST DECIDE ON WHAT CHARACTERISTICS OR PROPERTIES ARE NECESSARY TO DESCRIBE SUFFICIENTLY ANY MEMBER BELONGING TO A PARTICULAR COLLECTION OF OBJECTS SO THAT IT CAN BE MODELED SATISFACTORILY IN A COMPUTER PROGRAM.
- IN "PROGRAMMER-SPEAK," THE MEMBERS ARE CALLED "INSTANCES" AND THE COLLECTION IS CALLED A "CLASS," AND SO YOU WILL FIND PROGRAMMERS TALKING ABOUT "INSTANCES OF A CLASS" RATHER THAN MEMBERS OF A COLLECTION.

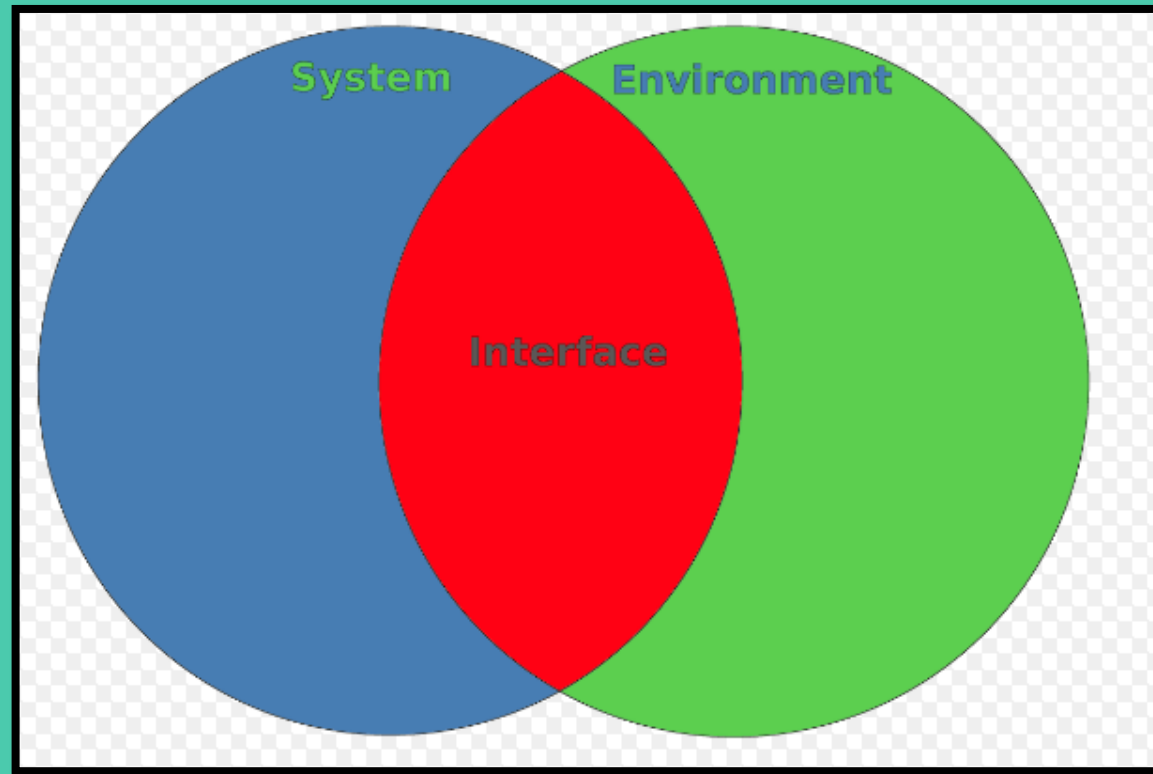
EVENT-DRIVEN AND OOP

UNDERSTANDING THE CONCEPTS OF "EVENT" AND "OBJECT" ARE CENTRAL TO WRITING PROGRAMS IN VISUAL BASIC, SO YOU WILL FIND MUCH OF STUDY GUIDE AND THE TEXTBOOK DEVOTED TO HOW TO WRITE PROGRAMS THAT RESPOND TO EVENTS AND HOW TO MANIPULATE OBJECTS.

WHEN PROVIDING VISUAL BASIC PROGRAMS AND SOURCE CODE AS EXAMPLES IN POWERPOINTS, THE FOLLOWING CONVENTIONS WILL BE ADOPTED:


- THIS POWERPOINT WILL USE VISUAL BASIC CLASS NAMES, SUCH AS "BUTTON" AND "TEXTBOX" WHEN REFERRING TO OBJECTS SUCH AS "BUTTON" AND "TEXT BOX."
- TWO IMPORTANT OBJECT CLASSES IN VISUAL BASIC ARE "**TEXTBOXES**" AND "**LISTBOXES**." THE TEXTBOOK SUGGESTS USING A LISTBOX FOR DISPLAYING OUTPUT. FOR NOW THIS STUDY GUIDE WILL USE THE TEXTBOX FOR THIS PURPOSE. WE WILL INTRODUCE THE LISTBOX LATER. IF YOU ARE UNFAMILIAR WITH WINDOWS, YOU SHOULD REVIEW THE RELEVANT SECTIONS IN APPENDIX B OF THE TEXTBOOK. NOTEPAD MAY ALSO BE USEFUL TO YOU IN CREATING TEST FILES FOR LATER ASSIGNMENTS IN THE COURSE.
- **IN THIS POWERPOINT, WORDS AND PHRASES IN ITALICS ARE TO BE REPLACED BY VISUAL BASIC IDENTIFIERS AND VALUES TO CREATE INSTANCES OF THE DESCRIPTION PROVIDED IN ITALICS.**

INTERFACE DESIGN



INTERFACE DESIGN

in·ter·face

/ˈin(t)ərˌfās/ 

noun

1. a point where two systems, subjects, organizations, etc., meet and interact.
"the interface between accountancy and the law"
2. **COMPUTING**
a device or program enabling a user to communicate with a computer.

verb

1. interact with (another system, person, organization, etc.).
"his goal is to get people interfacing with each other"
2. **COMPUTING**
connect with (another computer or piece of equipment) by an interface.

THE USER INTERFACE (UI)

User interface

From Wikipedia, the free encyclopedia

For the boundary between computer systems, see [Interface \(computing\)](#). For other uses, see [Interface](#).

The **user interface** (UI), in the [industrial design](#) field of [human–computer interaction](#), is the space where interactions between humans and machines occur. The goal of this interaction is to allow effective operation and control of the machine from the human end, whilst the machine simultaneously feeds back information that aids the operators' [decision-making](#) process. Examples of this broad concept of user interfaces include the interactive aspects of [computer operating systems](#), [hand tools](#), [heavy machinery](#) operator controls, and [process controls](#). The design considerations applicable when creating user interfaces are related to or involve such disciplines as [ergonomics](#) and [psychology](#).



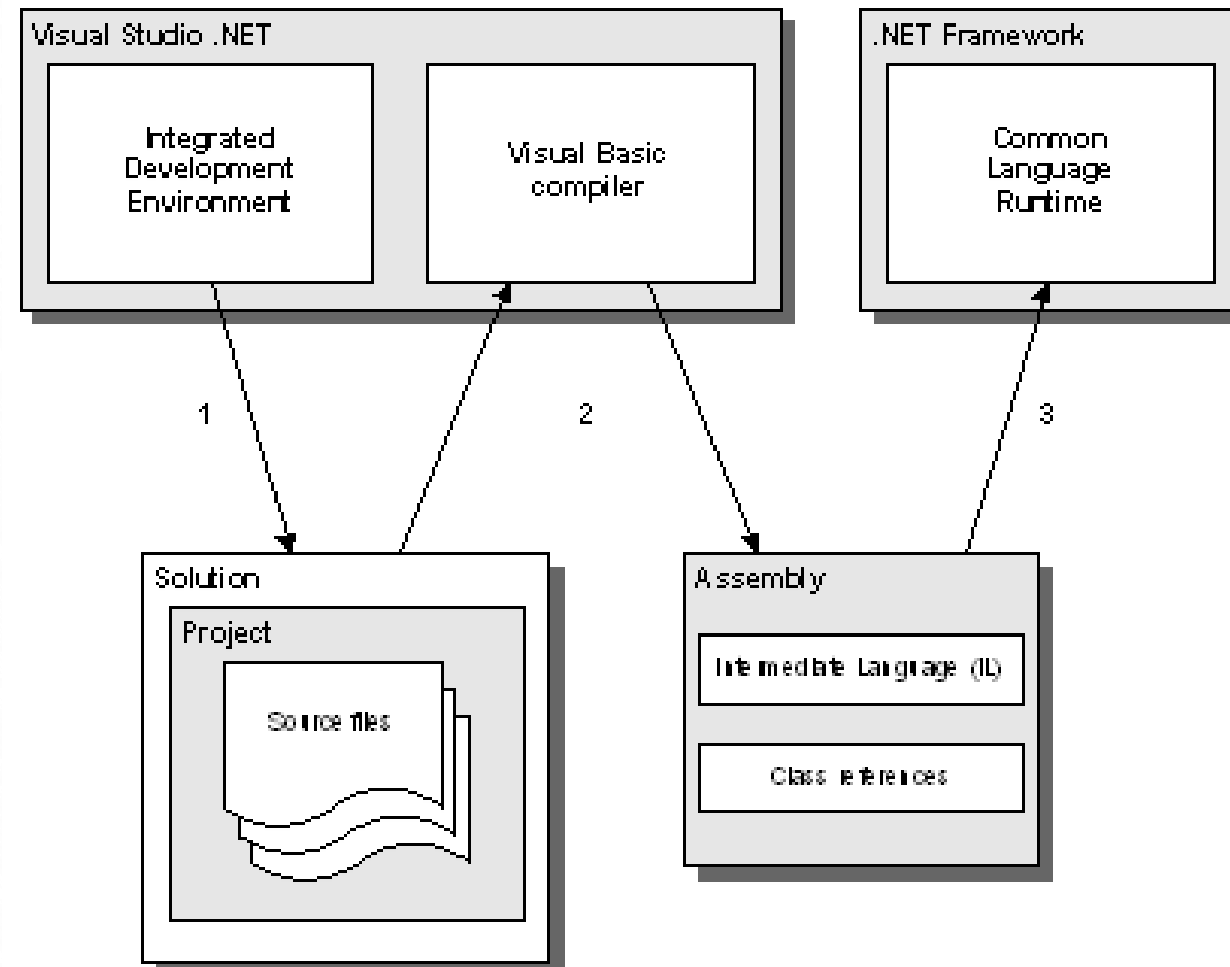
Example of a tangible user interface.

VISUAL STUDIO – THE IDE

INTEGRATED DEVELOPMENT ENVIRONMENT: ALLOWS THE AUTOMATION OF MANY OF THE COMMON PROGRAMMING TASKS IN ONE ENVIRONMENT

- WRITING THE CODE
- CHECKING FOR SYNTAX (LANGUAGE) ERRORS
- COMPILING AND INTERPRETING (TRANSFERRING TO COMPUTER LANGUAGE)
- DEBUGGING (FIXING RUN-TIME OR LOGIC ERRORS)
- RUNNING THE APPLICATION

VB COMPILING AND EXECUTION



PROJECT AND SOLUTION CONCEPTS

- **USER CREATES A NEW PROJECT IN VISUAL STUDIO**
 - A SOLUTION AND A FOLDER ARE CREATED AT THE SAME TIME WITH THE SAME NAME AS THE PROJECT
 - THE PROJECT BELONGS TO THE SOLUTION
 - MULTIPLE PROJECTS CAN BE INCLUDED IN A SOLUTION
- **SOLUTION**
 - CONTAINS SEVERAL FOLDERS THAT DEFINE AN APPLICATION'S STRUCTURE
 - SOLUTION FILES HAVE A FILE SUFFIX OF .SLN
- **PROJECT: CONTAINS FILES FOR A PART OF THE SOLUTION**
 - PROJECT FILE IS USED TO CREATE AN EXECUTABLE APPLICATION
 - A PROJECT FILE HAS A SUFFIX OF .VBPROJ
 - EVERY PROJECT HAS A TYPE (CONSOLE, WINDOWS, ETC.)
 - EVERY PROJECT HAS AN ENTRY POINT: A SUB PROCEDURE NAMED MAIN OR A FORM

END PREAMBLE



THE BASICS OF VISUAL BASIC



VISUAL BASIC

- PROGRAMMING FOR THE WINDOWS USER INTERFACE (OR ANY GUI) IS EXTREMELY COMPLICATED.
- VISUAL BASIC PROVIDES A CONVENIENT METHOD FOR BUILDING A GUI.
- VISUAL BASIC CAN INTERFACE WITH CODE WRITTEN IN C, FOR EFFICIENCY (FRONT-END).
- THE INSTRUCTIONS EXECUTED IN THE A VB PROGRAM ARE CONTROLLED BY EVENTS.



WHAT VISUAL BASIC ISN'T

- VISUAL BASIC IS *NOT* A POWERFUL PROGRAMMING LANGUAGE THAT ENABLES YOU TO DO ANYTHING YOU WANT.
- VISUAL BASIC IS *NOT* ELEGANT OR FAST.
- VISUAL BASIC IS *NOT* A REPLACEMENT FOR A LANGUAGE SUCH AS C.
- VISUAL BASIC IS *NOT* LIKE MOST COMMON COMPUTER LANGUAGES.



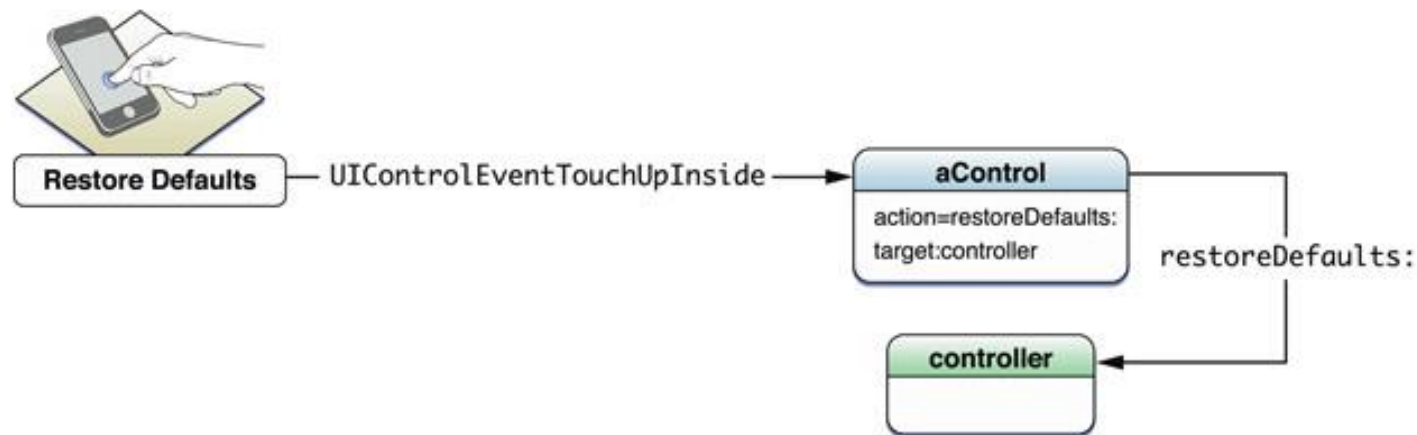
WHEN YOU PROGRAM IN VB

- YOU DRAW PICTURES OF YOUR USER INTERFACE.
- YOU DRAW BUTTONS, TEXT BOXES, AND OTHER USER-INTERFACE ITEMS.
- YOU ADD LITTLE SNIPPETS OF CODE TO HANDLE THE USER INTERACTION.
- YOU ADD INITIALIZATION CODE, USUALLY AS THE LAST STEP.
- IF YOU LIKE, YOU CAN CODE MORE COMPLEX FUNCTIONS. (BUT MANY DO NOT.)

CONTROL OBJECTS

CONTROL OBJECTS

- A CONTROL IS A TYPE OF VIEW IN A USER INTERFACE THAT SENDS A MESSAGE TO ANOTHER OBJECT WHEN A USER MANIPULATES IT IN A CERTAIN WAY, SUCH AS TAPPING A BUTTON OR DRAGGING A SLIDER. A CONTROL IS THE AGENT IN THE *TARGET-ACTION MODEL*.
- A STORES THE INFORMATION NECESSARY FOR SENDING THE MESSAGE: A REFERENCE TO THE OBJECT TO RECEIVE THE MESSAGE (THE TARGET) AND A SELECTOR THAT IDENTIFIES THE METHOD TO INVOKE ON THE TARGET (THE ACTION).
- WHEN A USER MANIPULATES THE CONTROL IN A SPECIFIC WAY, IT SENDS A MESSAGE TO THE APPLICATION OBJECT, WHICH THEN FORWARDS THE ACTION MESSAGE TO THE TARGET.



CONTROL OBJECTS

TO KEEP THINGS SIMPLE, THE EXAMPLES IN THIS PPT AS WELL AS YOUR PROGRAMS WILL INITIALLY EMPLOY ONLY FIVE CLASSES OF OBJECTS:

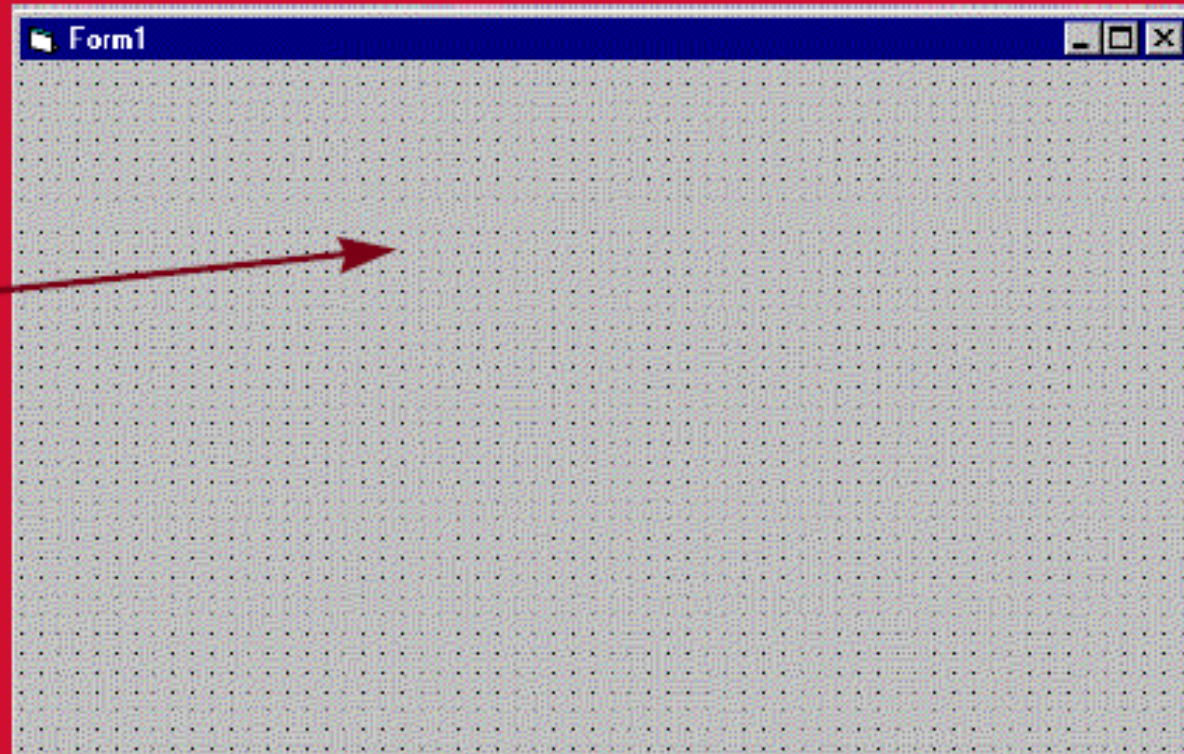
- **LABEL OBJECTS** PROVIDE YOU WITH A WAY TO PLACE TITLES IN YOUR WINDOW. THE ACTUAL TITLE DISPLAYED IS A PROPERTY OF THE LABEL OBJECT AND IS PROVIDED AS PART OF THE PROGRAMMING.
- **TEXTBOX OBJECTS** ALLOW YOU OR ANOTHER USER TO ENTER DATA INTO A PROGRAM OR TO DISPLAY OUTPUT SO THAT THE USER CAN OBSERVE THE RESULT.
- **BUTTON OBJECTS** PROVIDE A VIRTUAL BUTTON FOR YOU OR ANOTHER USER TO CLICK ON IN ORDER TO INITIATE SOME ACTION. THE EFFECT OF THE ACTION IS USUALLY DESCRIBED BY A CAPTION ON THE BUTTON. THE TEXT OF THIS CAPTION IS A PROPERTY OF THE BUTTON OBJECT AND IS PROVIDED BY THE PROGRAMMER IN THE DEVELOPMENT OF THE APPLICATION.
- **LISTBOX OBJECTS** ALLOW YOU TO DISPLAY A LIST OF VALUES AS OUTPUT.
- **FORM OBJECTS** ARE USED TO REPRESENT THE ENTIRE INPUT/OUTPUT INTERFACE WINDOW. MOST APPLICATIONS REQUIRE ONE OR MORE FORM OBJECTS, DEPENDING ON WHETHER A SINGLE WINDOW IS NEEDED TO COLLECT ALL THE INPUT AND DISPLAY ALL THE OUTPUT. A COMBINATION OF OBJECTS SUCH AS LABELS, TEXTBOXES, AND BUTTONS IS PLACED ON EACH FORM OBJECT.

CONTROL OBJECTS

- THESE ARE THE COMPONENTS OF THE FORM THAT YOU (THE PROGRAMMER) DEEM NECESSARY TO CONVEY WHAT TO DO TO THE USER AND PROVIDE A MEANS FOR THE USER TO SUPPLY ANY REQUIRED INPUT AND OBSERVE ANY RESULTING OUTPUT.
- BECAUSE INSTANCES OF ALL OF THESE TYPES OF OBJECTS EXCEPT FORMS PROVIDE THE USER WITH THE ABILITY TO AFFECT THE BEHAVIOR OF A PROGRAM, VISUAL BASIC PROGRAMMERS REFER TO THESE OBJECTS AS "CONTROL OBJECTS" OR SIMPLY "CONTROLS." THE "TOOLBOX" WINDOW OF THE IDE PROVIDES THESE AND MANY MORE CONTROL OBJECTS FOR YOU TO USE AS A PROGRAMMER IN IMPLEMENTING YOUR VISUAL BASIC PROGRAMS. TO DISPLAY THE TOOLBOX, CHECK THE TAB LABELLED "TOOLBOX" ON THE UPPER LEFT SIDE OF THE WINDOW.
- TO BE SURE YOU CAN GET STARTED AND TO FAMILIARIZE YOURSELF WITH THESE CONTROLS, YOU SHOULD PERFORM THE WALK-THROUGHS PROVIDED IN SECTION 2.2 OF THE 10TH EDITION OF THE TEXTBOOK (VB CONTROLS – **EMAILED TO YOU IN PDF FORM**). BE SURE YOU ACTUALLY DO THEM RATHER THAN JUST READING THAT SECTION OF THE TEXT. IN GENERAL, YOU SHOULD MAKE IT A PRACTICE TO ENTER AND RUN MOST, IF NOT ALL, PROGRAMMING EXAMPLES THAT THIS STUDY GUIDE SUGGESTS YOU EXAMINE. YOU CANNOT LEARN HOW TO PROGRAM WITHOUT ACTUALLY "DOING IT." LIKE ANY NATURAL LANGUAGE WHERE YOU MUST SPEAK AND WRITE IN THE LANGUAGE IN ORDER TO LEARN IT, SO IT IS THE CASE THAT YOU CANNOT LEARN A PROGRAMMING LANGUAGE WITHOUT ACTUALLY WRITING PROGRAMS AND RUNNING THEM. ALSO, JUST WRITING THE ASSIGNMENT PROGRAMS IS CERTAINLY NOT ENOUGH PRACTICE!

THE VB INTERFACE

Draw Your
Program
Here!



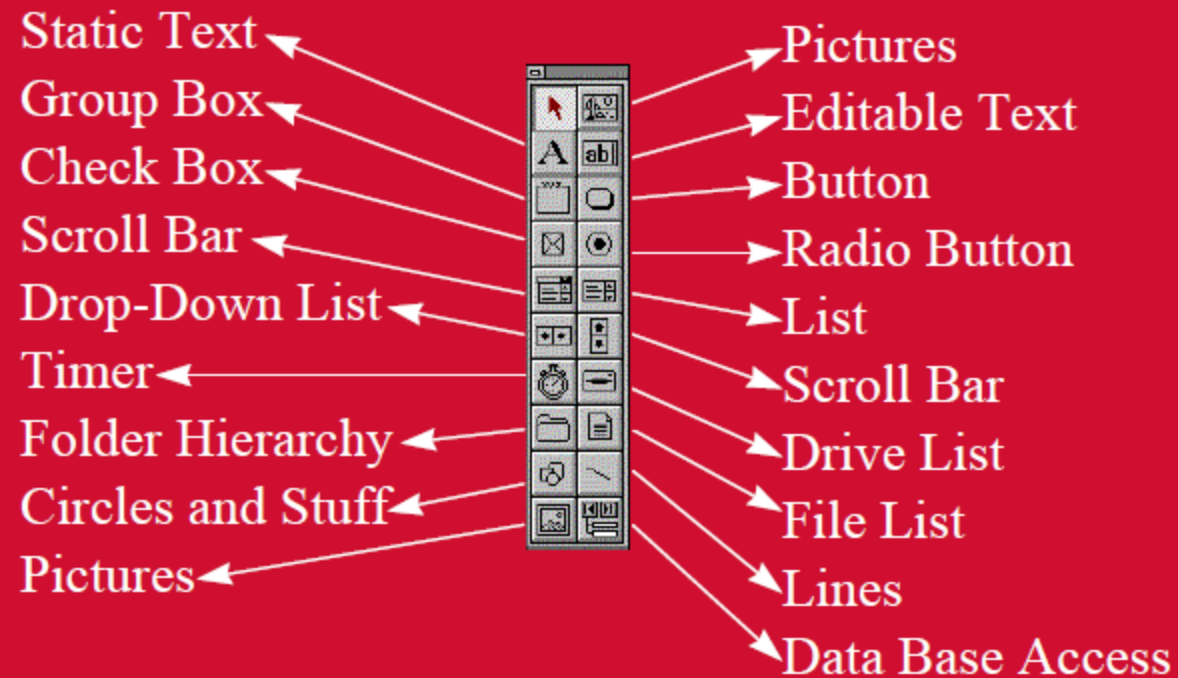
DRAWING THE PROGRAM

Select A Control From Here
(Click on the appropriate button)

Then Draw the control on the form



TYPES OF CONTROLS



And the List Goes On and On ...

HOW YOU DEVELOP A VISUAL BASIC PROGRAM

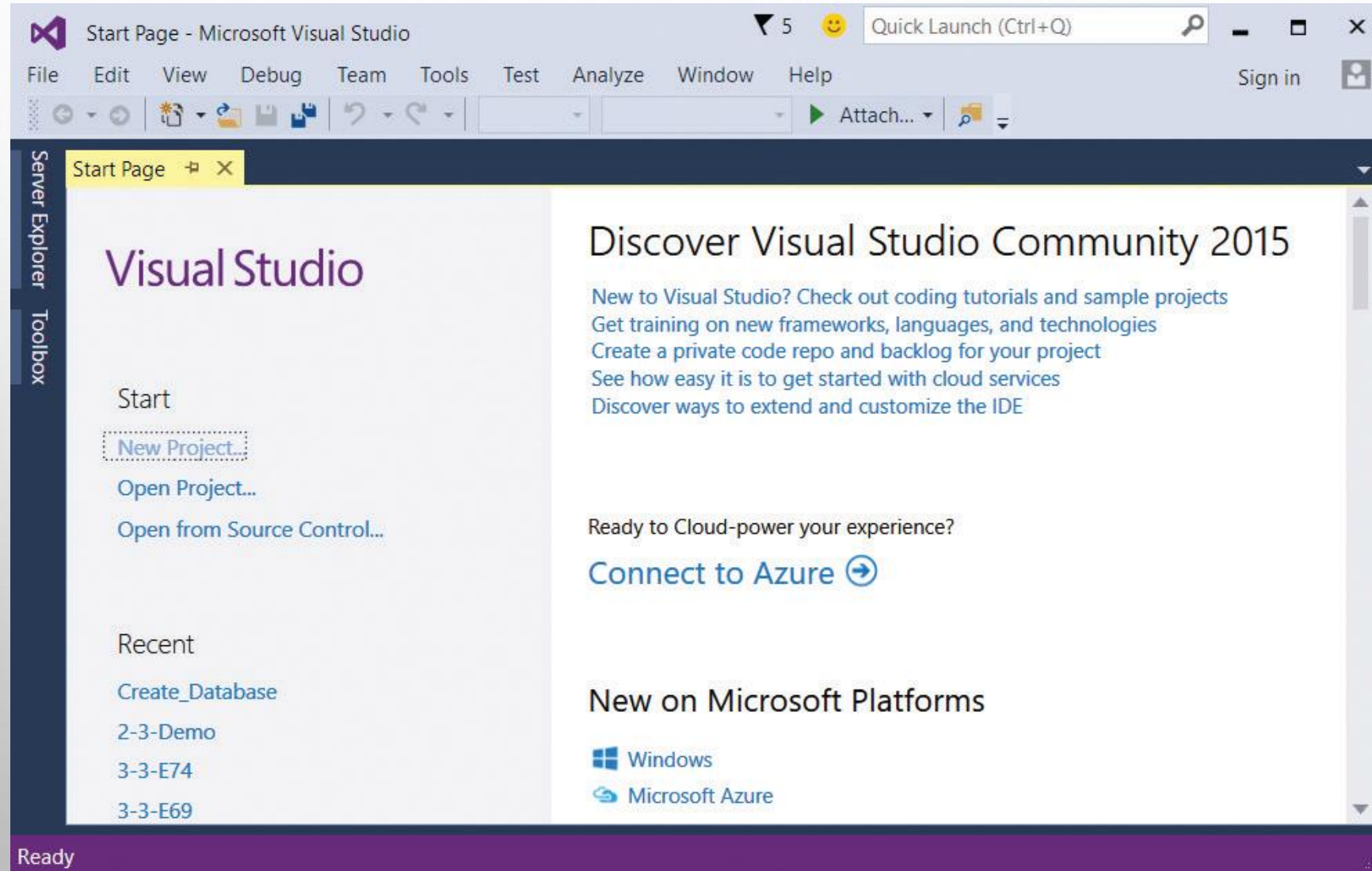
HOW YOU DEVELOP A VISUAL BASIC PROGRAM

- DESIGN THE INTERFACE FOR THE USER.
- DETERMINE WHICH EVENTS THE CONTROLS ON THE WINDOW SHOULD RECOGNIZE.
- WRITE THE EVENT PROCEDURES FOR THOSE EVENTS.

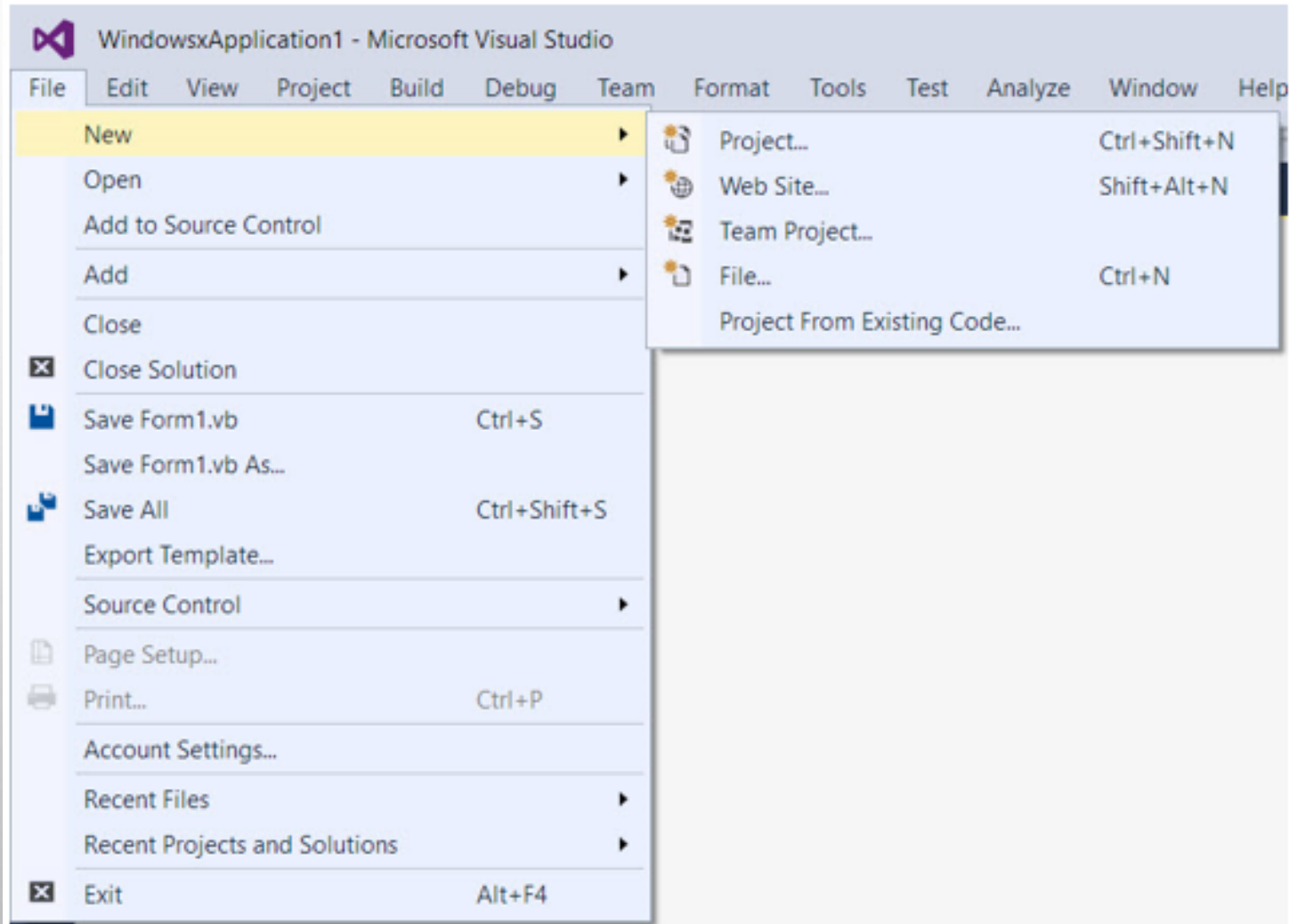
2.2 VISUAL BASIC CONTROLS

- STARTING A NEW VISUAL BASIC PROGRAM
- TEXT BOX CONTROL
- BUTTON CONTROL
- LABEL CONTROL
- LIST BOX CONTROL
- NAME PROPERTY / FONTS / AUTO HIDE
- POSITIONING AND ALIGNING CONTROLS
- MULTIPLE CONTROLS
- SETTING TAB ORDER

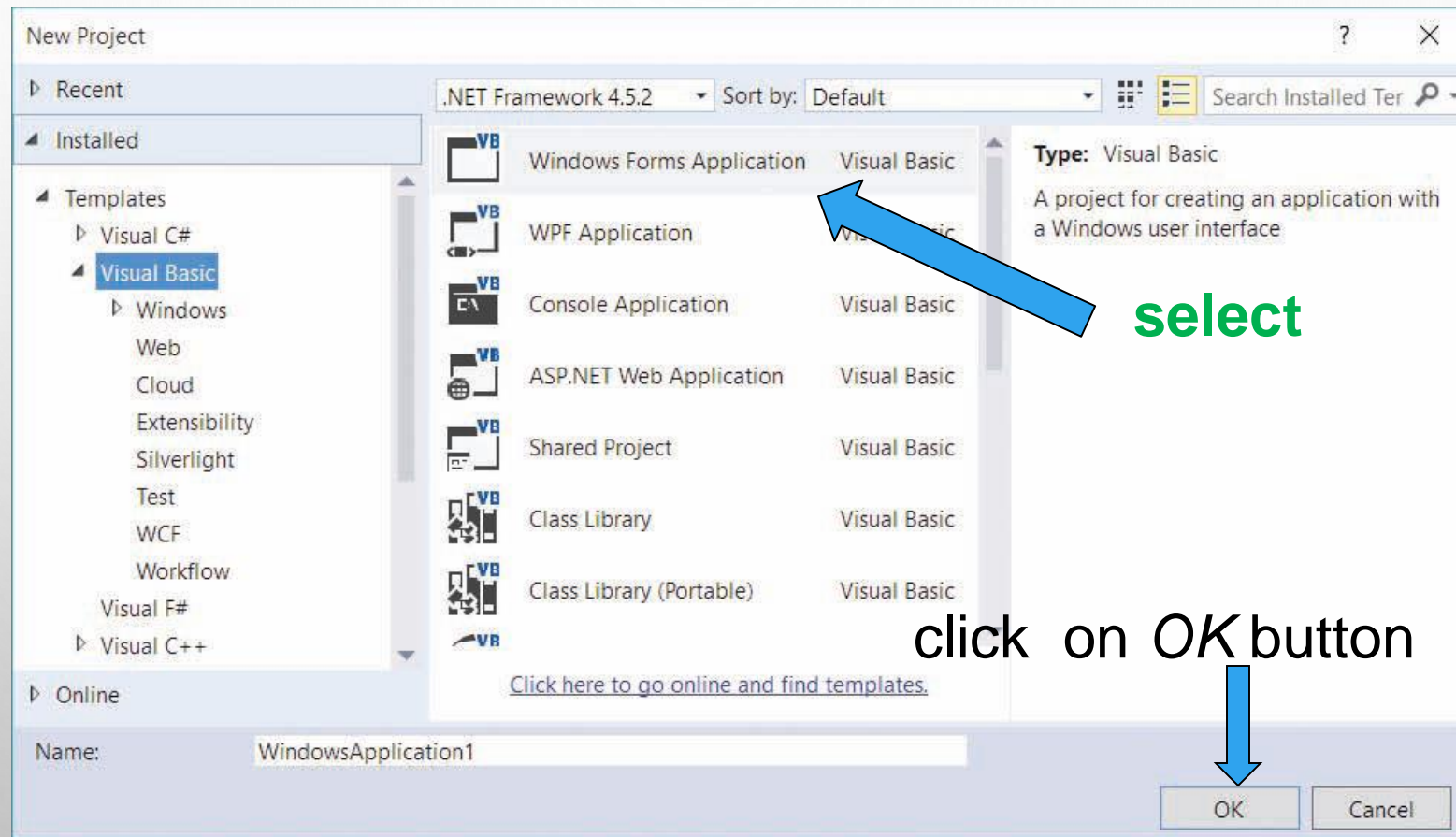
VISUAL STUDIO OPENING SCREEN



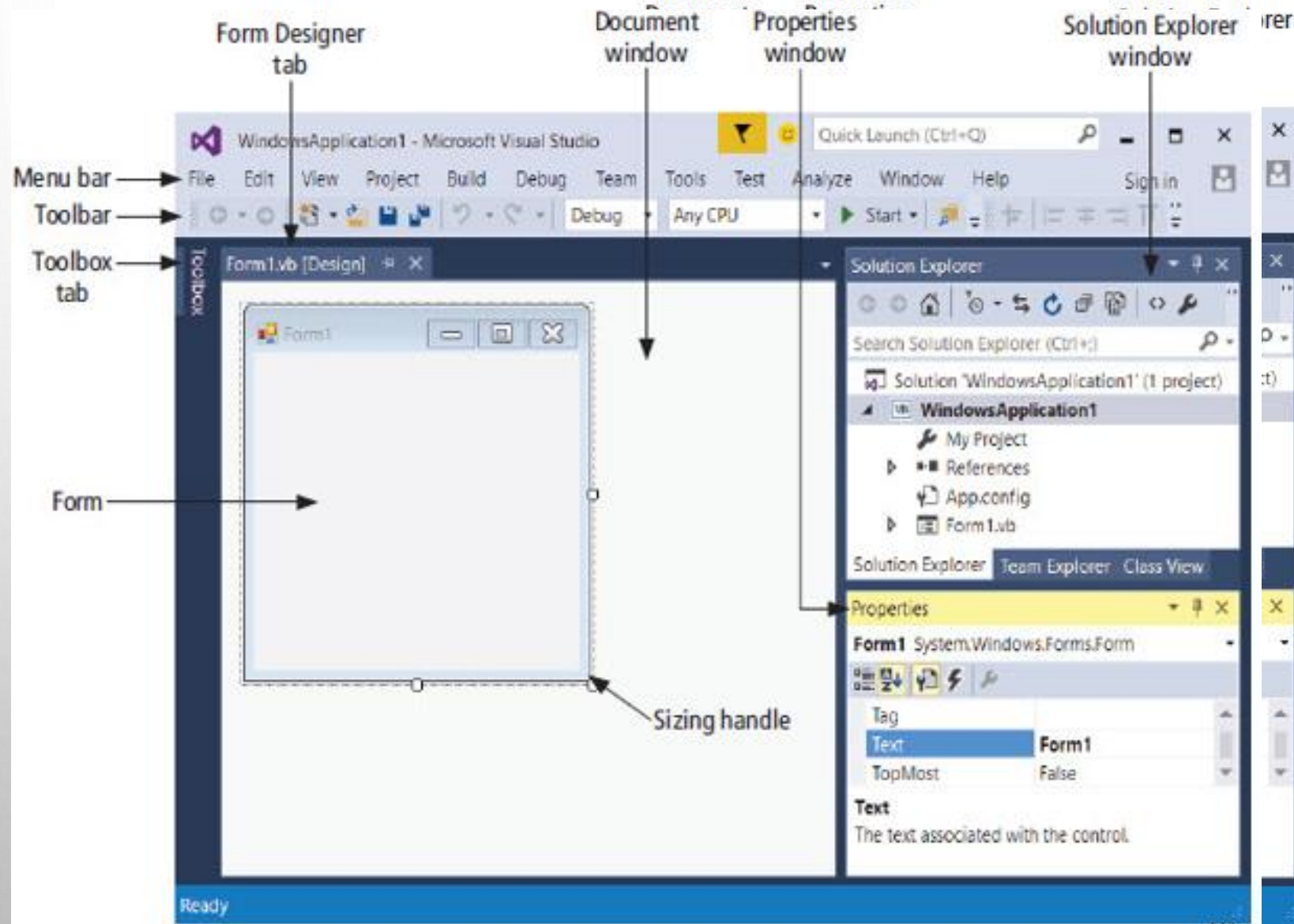
START A NEW PROJECT



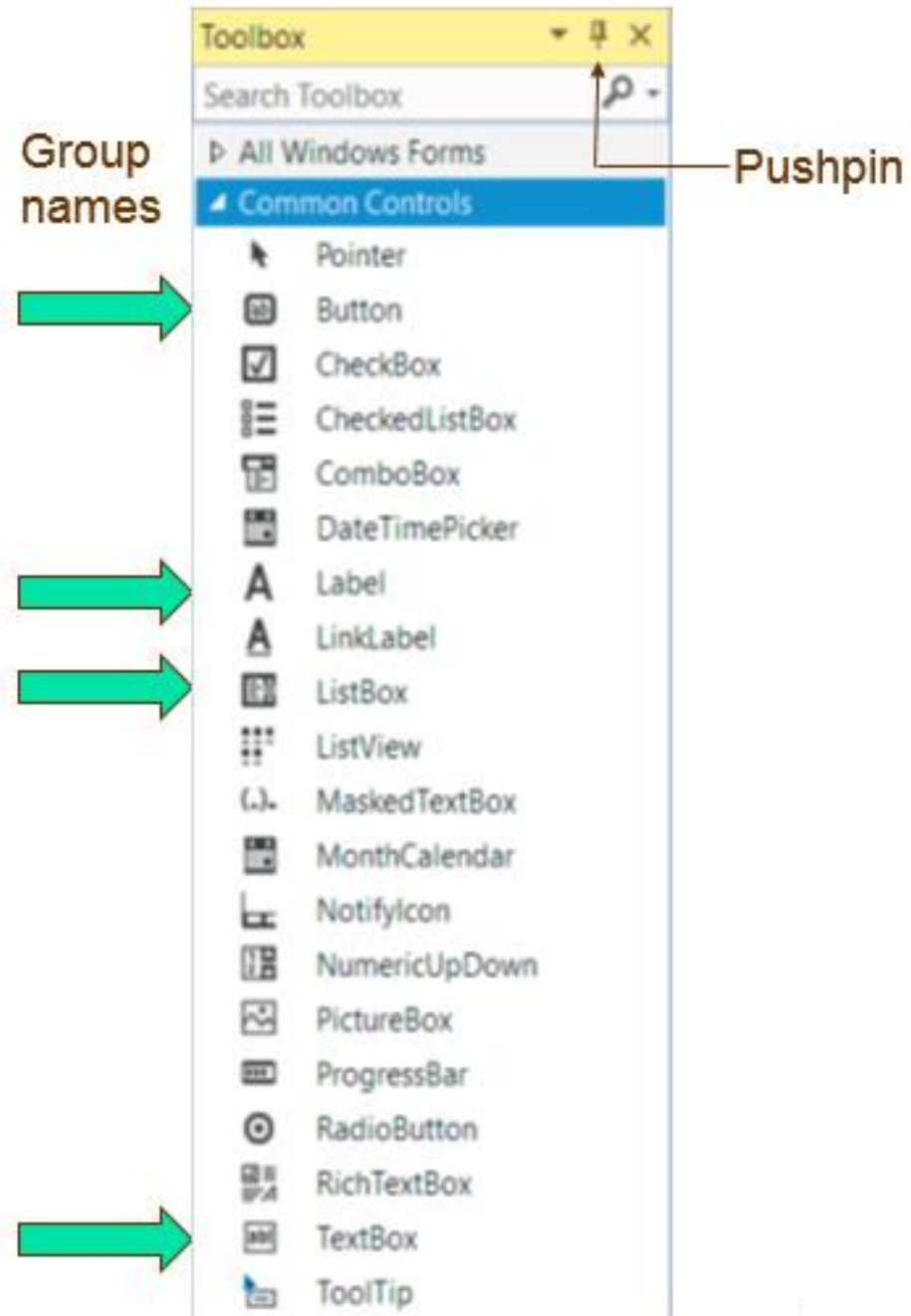
NEW PROJECT DIALOG BOX



INITIAL VISUAL BASIC SCREEN



TOOLBOX



4 WAYS TO PLACE A CONTROL FROM THE TOOLBOX ONTO THE FORM DESIGNER

- DOUBLE-CLICK
- DRAG AND DROP
- CLICK, POINT, AND CLICK
- CLICK, POINT, AND DRAG

INTERFACE DESIGN

- AN IMPORTANT TASK IN WRITING MOST PROGRAMS IS TO DEFINE A SUITABLE INPUT-OUTPUT INTERFACE. IT PROVIDES THE MEANS FOR USERS OF YOUR PROGRAM TO COMMUNICATE WITH IT IN A MEANINGFUL WAY. HENCE, WE OFTEN REFER TO THIS INTERFACE AS THE "USER INTERFACE" OF OUR PROGRAM. EVEN THOUGH YOU MAY NOT HAVE WRITTEN A PROGRAM PREVIOUSLY, IF YOU HAVE USED A COMPUTER YOU WILL BE FAMILIAR WITH DEALING WITH USER INTERFACES SINCE THAT IS THE PURPOSE OF A WINDOW. YOU CAN PROBABLY RECALL SOME EXAMPLES OF PROGRAMS WITH WINDOWS INTERFACES (ALSO CALLED "GRAPHICAL USER INTERFACES" OR "GUIs") THAT WERE EASY TO USE AND SOME THAT WERE DIFFICULT. FINDING THE RIGHT COMBINATION OF BOXES, LABELS, TEXT, PICTURES - ALL TYPICAL COMPONENTS OF A WINDOW - IS CENTRAL TO THE DESIGN OF A GOOD (THAT IS, "USER-FRIENDLY") USER INTERFACE.
- IN VISUAL BASIC AND MANY OTHER LANGUAGES, EACH OF THE COMPONENTS OF A GUI IS IMPLEMENTED AS AN INSTANCE OF SOME CLASS OF OBJECTS. THAT IS, EACH TYPE OF COMPONENT CAN BE DESCRIBED BY A NUMBER OF PROPERTIES. FOR EXAMPLE, A VIRTUAL BUTTON (THAT IS, SOMETHING IN A WINDOW YOU CAN CLICK ON TO MAKE SOMETHING HAPPEN) MIGHT HAVE PROPERTIES THAT RELATE TO ITS APPEARANCE, SUCH AS ITS SIZE, COLOR, AND THE TEXT OF THE LABEL APPEARING ON IT.

INTERFACE DESIGN

- IN VISUAL BASIC, THE TYPES OF COMPONENTS YOU MIGHT WISH TO INCLUDE IN A WINDOW ARE REPRESENTED AS CLASSES OF OBJECTS. THUS THERE ARE THE BUTTON CLASS FOR BUTTON OBJECTS AND THE TEXTBOX CLASS FOR TEXT FIELDS.
- TO AID PROGRAMMERS IN WRITING THEIR PROGRAMS, SUPPLIERS OF PROGRAMMING SOFTWARE OFTEN PROVIDE A WINDOW-BASED PROGRAM CALLED AN **INTEGRATED DEVELOPMENT ENVIRONMENT** OR IDE. THIS PROGRAM PROVIDES A NUMBER OF TOOLS THAT ALLOW PROGRAMMERS NOT ONLY TO WRITE AND EDIT PROGRAMS BUT ALSO TO CREATE THE USER INTERFACES FOR INTERACTING WITH THE PROGRAM. AS WELL, OTHER TOOLS ARE PROVIDED TO SAVE PROGRAMS AND USER INTERFACES AND OTHERWISE MANAGE THE DEVELOPMENT OF A TYPICAL PROGRAMMING PROJECT.

INTERFACE DESIGN

- FOR VB, MICROSOFT PROVIDES SUCH AN INTEGRATED ENVIRONMENT, CALLED "MICROSOFT VISUAL STUDIO," OR MORE SIMPLY "VS IDE." TO LAUNCH THE VS IDE FOR VISUAL BASIC:
 - CLICK ON START (LOWER LEFT-HAND CORNER).
 - CLICK ON "ALL PROGRAMS."
 - CLICK ON THE "MICROSOFT VISUAL STUDIO 2012" FOLDER.
 - CLICK ON "VISUAL STUDIO 2012" TO LAUNCH
- A WINDOW WILL APPEAR, CALLED THE **VISUAL STUDIO IDE START PAGE**. THIS IS THE STARTING POINT FOR DEVELOPING NEW PROGRAMS AS WELL AS EDITING EXISTING ONES. THIS WINDOW IS NOT EXCLUSIVELY FOR THE DEVELOPMENT OF VB PROGRAMS. INSTEAD, YOU CAN CHOOSE THE DESIRED PROGRAMMING LANGUAGE FROM A LIST OF CHOICES. FOR THIS COURSE, WHEN YOU START A NEW PROJECT, YOU SHOULD ALWAYS SELECT "NEW PROJECT," AND SINCE YOU WILL ALWAYS BE CONSTRUCTING PROGRAMS THAT ALLOW THE USER TO INTERACT WITH THE PROGRAM VIA A WINDOW, YOU SHOULD CHOOSE "WINDOWS FORMS APPLICATION" AS YOUR TEMPLATE.H.

A NOTE ON “REQUIREMENTS”

Software Requirements¹

Requirements are descriptions of the services that a software system must provide and the constraints under which it must operate

Requirements can range from high-level abstract statements of services or system constraints to detailed mathematical functional specifications

Requirements Engineering is the process of establishing the services that the customer requires from the system and the constraints under which it is to be developed and operated

Requirements may serve a dual function:

- As the basis of a bid for a contract
- As the basis for the contract itself

A NOTE ON “REQUIREMENTS”

Software Requirements¹

Requirements are descriptions of the services that a software system must provide and the constraints under which it must operate.

Requirements can range from high-level abstract statements of services or system constraints to detailed mathematical functional specifications.

Requirement engineering is the process of establishing the services that the customer requires from the system and the constraints under which it is to be developed and operated.

Requirements may serve a dual function:

- As the basis for the contract itself
- As the basis for the contract itself


See the “Creating Control Objects” section of the Study Guide (Important for HW2)



RUNNING CODE SNIPPETS WITH ONLINE COMPILERS



Compile and Execute VB.Net Online (Mono v4.6)

 Execute

 Embed

main.vb

Stdin

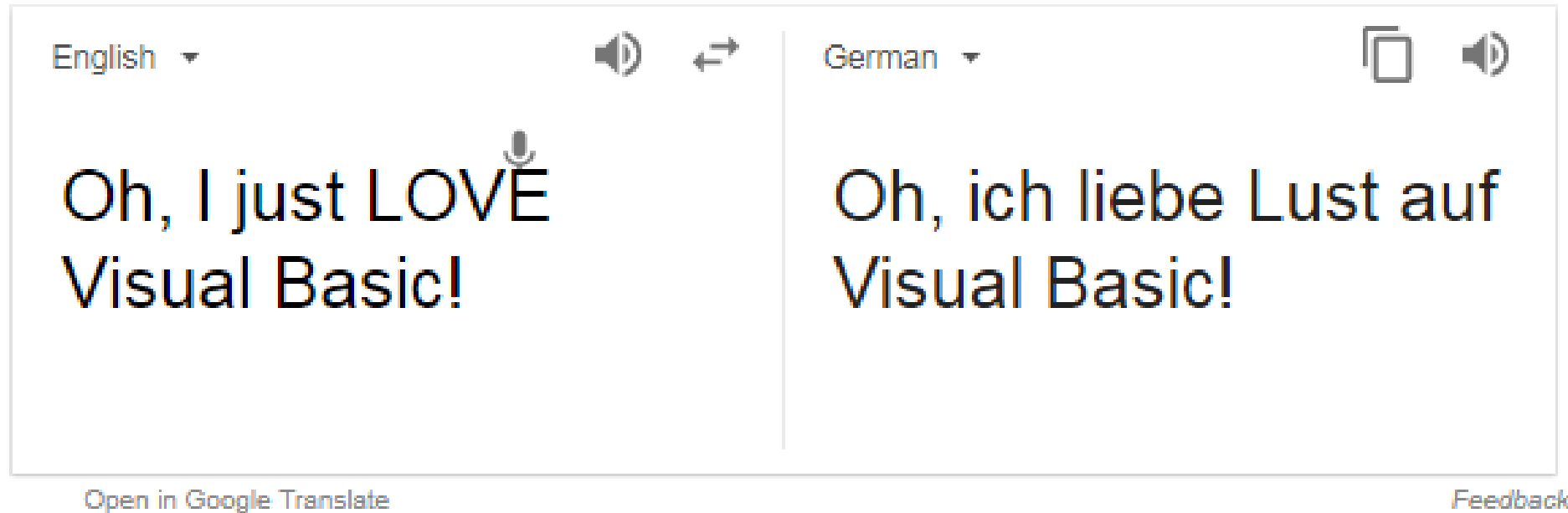
```
1 Module HelloWorld
2     ' every console app starts with Main
3     Sub Main( )
4         System.Console.WriteLine("Hello world!")
5     End Sub
6 End Module
```

 Result

```
$vbnc *.vb /out:main.exe
Visual Basic.Net Compiler version 0.0.0.5943 (Mono 4.6 - ta
Copyright (C) 2004-2010 Rolf Bjarne Kvinge. All rights rese

Assembly 'main, Version=0.0, Culture=neutral, PublicKeyToka
Compilation successful
Compilation took 00:00:00.5048620
$mono main.exe
Hello world!
```

ONLINE COMPILERS ARE ESSENTIALLY LANGUAGE TRANSLATORS



SYNTAX OF HELLO WORLD

Visual Basic Version of Hello, World

Visual Studio 2012 | [Other Versions](#) ▼

The following console program is the Visual Basic version of the traditional "Hello, World!" program, which displays the string "Hello, World!".

VB

```
' A "Hello, World!" program in Visual Basic.  
Module Hello  
    Sub Main()  
        MsgBox("Hello, World!") ' Display message on computer screen.  
    End Sub  
End Module
```

SYNTAX OF HELLO WORLD

Visual Basic Version of Hello, World

Visual Studio 2012 | [Other Versions](#) ▼

The following console program is the Visual Basic version of the

VB

```
' A "Hello, World!" program in Visual Basic.  
Module Hello  
    Sub Main()  
        MsgBox("Hello, World!") ' Display message  
    End Sub  
End Module
```

The important points of this program are the following:

- Comments
- The `Main` procedure
- Input and output
- Compilation and execution

SYNTAX OF HELLO WORLD

```
' A "Hello, World!" program in Visual Basic.  
Module Hello  
    Sub Main()  
        MsgBox("Hello, World!") ' Display message on computer screen.  
    End Sub  
End Module
```

Comments

The first line of the example contains a comment:

VB

```
' A "Hello, World!" program in Visual Basic.
```

The single quotation mark (') means that the rest of the line is a comment and will be ignored by the compiler. You can make an entire line a comment, or you can append a comment to the end of another statement, as follows:

VB

```
MsgBox("Hello, World!") ' Display message on computer screen.
```

SYNTAX OF HELLO WORLD

```
' A "Hello, World!" program in Visual Basic.  
Module Hello  
    Sub Main()  
        MsgBox("Hello, World!") ' Display message on computer screen.  
    End Sub  
End Module
```

The Main Procedure

Every Visual Basic application must contain a procedure called `Main`. This procedure serves as the starting point and overall control for your application. It is called when your module is loaded.

There are four varieties of `Main`:

- `Sub Main()`
- `Sub Main(ByVal cmdArgs() As String)`
- `Function Main() As Integer`
- `Function Main(ByVal cmdArgs() As String) As Integer`

The most common variety of this procedure is `Sub Main()`. Unless you are creating a Windows Forms application, you must write the `Main` procedure for applications that run on their own. For more information, see [Main Procedure in Visual Basic](#).

SYNTAX OF HELLO WORLD

```
' A "Hello, World!" program in Visual Basic.  
Module Hello  
    Sub Main()  
        MsgBox("Hello, World!") ' Display message on computer screen.  
    End Sub  
End Module
```

Input and Output



This example uses the standard Visual Basic run-time library, which is available through the [Microsoft.VisualBasic](#) namespace. If you compile the program in the integrated development environment (IDE), you can use all the procedures and properties of [Microsoft.VisualBasic](#) without having to import it. If you compile from the command line, you must use the [Imports Statement \(.NET Namespace and Type\)](#) in your source code, or the [/imports \(Visual Basic\)](#) command-line compiler option, to make the [Microsoft.VisualBasic](#) members available to your program.

The `Main` procedure calls the `MsgBox` function to display a message box containing the string "Hello, World!":

VB

```
MsgBox("Hello, World!") ' Display message on computer screen.
```

SYNTAX OF HELLO WORLD

```
' A "Hello, World!" program in Visual Basic.  
Module Hello  
    Sub Main()  
        MsgBox("Hello, World!") ' Display message on computer screen.  
    End Sub  
End Module
```

Compilation and Execution



You can compile the "Hello, World!" program using either the Visual Studio integrated development environment (IDE) or the command line.

To compile and run the program from the command line

1. Create the source file using any text editor and save it with a file name such as `Hello.vb`.
2. To invoke the compiler, enter the following command:

```
vbc Hello.vb
```

If your source file does not include an **Imports** statement for the `Microsoft.VisualBasic` namespace, you can include the **/imports** command-line compiler option in the `vbc` command:

```
vbc Hello.vb /imports:Microsoft.VisualBasic
```

3. If your program does not contain any compilation errors, the compiler creates a `Hello.exe` file.
4. To run the program, enter the following command:

```
Hello
```

SYNTAX OF HELLO WORLD

```
' A "Hello, World!" program in Visual Basic.  
Module Hello  
    Sub Main()  
        MsgBox("Hello, World!") ' Display message on computer screen.  
    End Sub  
End Module
```

Compilation and Execution



You can optionally include the `/main` command-line compiler option in the `vbc` command to specify the namespace and module supplying `Main`.

To compile and run the program from the IDE

1. Create a Visual Basic console application project.
2. Copy the code into the project.
3. Choose the appropriate **Build** command from the **Build** menu, or press F5 to build and run (corresponding to **Start** in the **Debug** menu).

Filter

Main Procedure

References and the Imports Statement

Namespaces

Naming Conventions

Coding Conventions

Conditional Compilation

How to: Break and Combine Statements in Code

How to: Collapse and Hide Sections of Code

How to: Label Statements

Special Characters in Code

Comments in Code

Keywords as Element Names in Code

Me, My, MyBase, and My Class

Limitations

> Language Features

> COM Interop

Visual Basic Coding Conventions

07/20/2015 • 6 minutes to read • Contributors

Microsoft develops samples and documentation that follow the guidelines in this topic. If you follow the same coding conventions, you may gain the following benefits:

- Your code will have a consistent look, so that readers can better focus on content, not layout.
- Readers understand your code more quickly because they can make assumptions based on previous experience.
- You can copy, change, and maintain the code more easily.
- You help ensure that your code demonstrates "best practices" for Visual Basic.

Naming Conventions

- For information about naming guidelines, see [Naming Guidelines](#) topic.
- Do not use "My" or "my" as part of a variable name. This practice creates confusion with the `My` objects.
- You do not have to change the names of objects in auto-generated code to make them fit the guidelines.

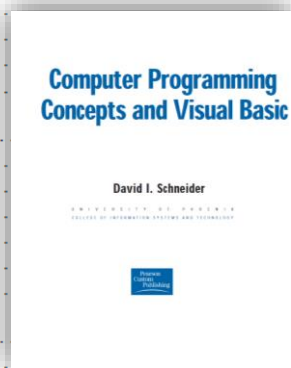
Layout Conventions

IMPORTANT NOTES

- STUDY CHAPTERS 2 AND 3 OF SCHNEIDER, 10TH ED. (EMAILED TO YOU ALREADY).
- KNOW THE STUDY GUIDE IN DETAIL.
- KNOW THE PPTS IN DETAIL.
- LET US LOOK AT THE DIFFERENCES BETWEEN THE SCHNEIDER EDITIONS...

CONTENTS

SECTION 1: PROBLEM SOLVING	1
1.1 PROGRAM DEVELOPMENT CYCLE	3
1.2 PROGRAMMING TOOLS	5
SECTION 2: FUNDAMENTALS OF PROGRAMMING IN VISUAL BASIC	15
2.1 VISUAL BASIC OBJECTS	17
2.2 VISUAL BASIC EVENTS	27
2.3 NUMBERS	34
2.4 STRINGS	42
2.5 INPUT AND OUTPUT	49
2.6 BUILT-IN FUNCTIONS	58
SUMMARY	67
PROGRAMMING PROJECTS	68
SECTION 3: GENERAL PROCEDURES	71
3.1 SUB PROCEDURES, PART I	73
3.2 SUB PROCEDURES, PART II	80
3.3 FUNCTION PROCEDURES	87
3.4 MODULAR DESIGN	93
SUMMARY	97
PROGRAMMING PROJECTS	97
SECTION 4: DECISIONS	101
4.1 RELATIONAL AND LOGICAL OPERATORS	103
4.2 IF BLOCKS	106
4.4 SELECT CASE BLOCKS	112
4.4 A CASE STUDY: WEEKLY PAYROLL	119
SUMMARY	126
PROGRAMMING PROJECTS	126
SECTION 5: REPETITION	129
5.1 DO LOOPS	131
5.2 PROCESSING LISTS OF DATA WITH DO LOOPS	135
5.3 FOR...NEXT LOOPS	141
5.4 A CASE STUDY: ANALYZE A LOAN	146
SUMMARY	153
PROGRAMMING PROJECTS	153
SECTION 6: ARRAYS	159
6.1 CREATING AND ACCESSING ARRAYS	161
6.2 USING ARRAYS	171
6.3 CONTROL ARRAYS	179
6.4 SORTING AND SEARCHING	184
6.5 TWO-DIMENSIONAL ARRAYS	196
6.6 A CASE STUDY: CALCULATING WITH A SPREADSHEET	200



Chapter 1 An Introduction to Computers and Problem Solving 1

- 1.1 An Introduction to Computing and Visual Basic 2
- 1.2 Program Development Cycle 5
- 1.3 Programming Tools 7

Chapter 2 Visual Basic, Controls, and Events 15

- 2.1 An Introduction to Visual Basic 2015 16
- 2.2 Visual Basic Controls 18
- 2.3 Visual Basic Events 37

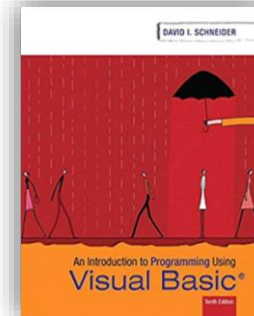
Summary 52

Chapter 3 Variables, Input, and Output 53

- 3.1 Numbers 54
- 3.2 Strings 72
- 3.3 Input and Output 93

Summary 109

Programming Projects 110



Chapter 4 Decisions 113

- 4.1 Relational and Logical Operators 114
- 4.2 If Blocks 122
- 4.3 Select Case Blocks 144
- 4.4 Input via User Selection 160

Summary 174

Programming Projects 175

Chapter 5 General Procedures 179

- 5.1 Function Procedures 180
- 5.2 Sub Procedures, Part I 197
- 5.3 Sub Procedures, Part II 213
- 5.4 Program Design 224
- 5.5 A Case Study: Weekly Payroll 228

Summary 236

Programming Projects 236

Chapter 6 Repetition 241

- 6.1 Do Loops 242
- 6.2 For ... Next Loops 257
- 6.3 List Boxes and Loops 273

Summary 284

Programming Projects 285

Chapter 7 Arrays 293

- 7.1 Creating and Using Arrays 294
- 7.2 Using LINQ with Arrays 321

PRESENTATION TERMINATED

