

Learning to Program in
>> Visual Basic



PG ONLINE

S Langfield

Learning to Program in
**Visual
Basic .NET** >>

S Langfield

Published by
PG Online Limited
The Old Coach House
35 Main Road
Tolpuddle
Dorset
DT2 7EW
United Kingdom
sales@pgonline.co.uk
www.pgonline.co.uk



PG ONLINE

Graphics: PG Online Ltd

Design and artwork: PG Online Ltd

First edition 2019

A catalogue entry for this book is available from the British Library

ISBN: 978-1-910523-18-6

Copyright © S Langfield, PM Heathcote, 2017

All rights reserved

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without the prior written permission of the copyright owner.

Printed on FSC certified paper

Printed by Bell and Bain Ltd, Glasgow, UK.



Preface

Programming is fun! Trial and error is to be encouraged and you should type all of the examples and try all the exercises to get used to entering and debugging programs and to see how the programs run.

This book is intended for individuals and students who may have done some programming in other languages, but are not familiar with Visual Basic .NET. It is intended that users of the book should work through the book sequentially, starting at Chapter 1. However, it will be a useful reference book for students on a programming course or anyone working on a programming project.

It teaches basic syntax and programming techniques and introduces a number of useful features such as:

- **Developing graphical user interfaces** (GUIs) with the visual designer in Visual Studio.
- **SQLite**, which enables the creation and processing of a database from within a Visual Basic .NET program. This provides an alternative to writing to a text file when data needs to be stored and retrieved.
- **The Visual Studio debugger**, which can be used to help find elusive logic errors.

Questions and exercises are included throughout every chapter. Over 120 VB programs for all the examples and exercises given in the book may be downloaded from **www.pgonline.co.uk**. We strongly advise you to write your own code and check your solutions against the sample programs provided.

This book is a companion volume to the book Learning to Program in Python by P.M. Heathcote, and uses the same format. Questions and exercises from that text have been used throughout this book, with answers and programs rewritten in Visual Basic.

Enjoy – the sky's the limit!

Downloading Visual Basic .NET

VB.NET is a high-level programming language, implemented on the .NET Framework. Microsoft® launched VB.NET in 2002 as the successor to its original Visual Basic language. Microsoft's integrated development environment (IDE) for developing programs in VB.NET is called Visual Studio. Visual Studio Express and Visual Studio Community are freeware. VB.NET is available to be downloaded free from <https://visualstudio.microsoft.com/vs/express/>. The programs have been written and tested in Visual Studio Express 2019. Many schools and individuals may prefer to use alternative development environments and the book is equally applicable to these.

Contents

Chapter 1 – Input, output and assignment	1
Programming in VB .NET	1
Programming in VB .NET console mode	2
Adding comments	4
Programming conventions	5
Data types	5
Rounding a result	6
Variables	7
Augmented assignment operators	9
The Write and WriteLine statements	9
The ReadLine statement	11
Chapter 2 – Strings and numbers	14
String functions	14
Syntax errors	16
Numbers	16
Inputting numbers	17
Converting between strings and numbers	18
Functions and methods	16
Constants	19
Chapter 3 – Selection	20
Programming constructs	20
Boolean conditions	21
The Elself clause	21
Case statements	22
Nested selection statements	23
Complex Boolean expressions	23
Generating a random number	24

Chapter 4 – Iteration	26
The For loop	26
The While loop	28
The Do ... Loop statement	30
String processing	31
Substrings	32
Interrupting execution	32
Chapter 5 – Arrays and tuples	34
Arrays	34
Operations on arrays	35
Array processing	36
Two-dimensional arrays	37
Tuples	39
Arrays of tuples	40
Structures	40
Arrays of structures	41
Chapter 6 – Validating user input	43
Validating user input	43
The ASCII code	44
Functions Asc() and Chr()	46
Regular expressions	47
Chapter 7 – Searching and sorting	51
Dictionary data structure	51
Adding to a list	54
Sorting a list	54
Storing a list in a dictionary	55
Chapter 8 – Procedures and functions	57
Types of subroutine	57
Built-in subroutines	58
Writing your own functions	59
Writing your own procedures	59
Using parameters and return values	60
Local and global variables	63

Chapter 9 – Reading and writing files	67
Storing data	67
Records and fields	68
Opening, reading and closing a text file	68
Writing to a file	72
Formatting output	75
File processing	77
Chapter 10 – Databases and SQL	80
Flat file databases	80
Records, fields and primary keys	81
Querying a database	82
Adding a record to a database table	82
Updating a record	83
Deleting records from a table	83
Chapter 11 – Using SQLite	85
Using SQL commands in a VB.NET program	85
Creating a database	85
Importing data from a text file	89
Creating a new database and loading it with data	90
Querying the database	92
Adding records entered by the user	94
Error handling	95
Deleting a record	96
Updating the database	97
Chapter 12 – Introduction to the graphical user interface	100
Creating a Windows Forms App	100
Controls	102
The “Hello World” program	102
Responding to user input	104
Setting Form properties	105

Chapter 13 – Developing a Windows application	107
Sample Application 1	107
Designing the data input window	107
Building the form	108
Sample Application 2	111
Sample Application 3	114
Chapter 14 – Program design	117
Planning a program	117
The sample task	118
Chapter 15 – Testing and debugging	120
Drawing up a test plan	120
Using the debugging feature	123
Index	127

Chapter 1

Input, output and assignment

Objectives

- Write a simple console application
- Use string, numeric and Boolean data types and operators
- Learn the rules and guidelines for declaring and naming variables
- Use input and output statements

Programming in VB .NET

Visual Basic (VB) is a popular programming language. A VB program is simply a series of instructions written according to the rules or **syntax** of the language, usually designed to perform some task or come up with a solution to a problem. You write the instructions, and then the computer translates these instructions into binary machine code which the computer can execute. It will do this using a translator program, called a **compiler**.

VB.NET is the latest version of the Visual Basic programming language. It comes with an **integrated development environment** (IDE) which enables you to enter your program, save it, edit it, translate it to machine code and run it once it is free of syntax errors. If you have written a statement incorrectly, it will be reported by the IDE or the compiler as a syntax error, and you can correct it and try again.

You can write simple programs as a console application or you can write Windows® based applications using Windows forms (see Chapters 12 and 13).

Augmented assignment operators

These operators provide a shortcut to writing assignment statements.

Operator	Example	Equivalent to
<code>+=</code>	<code>score += 1</code>	<code>score = score + 1</code>
<code>-=</code>	<code>score -= losses</code>	<code>score = score - losses</code>
<code>*=</code>	<code>score *= 2</code>	<code>score = score * 2</code>
<code>/=</code>	<code>score /= total</code>	<code>score = score / total</code>
<code>\=</code>	<code>score \= 7</code>	<code>score = score \ 7</code>

Table 1.4: Augmented assignment operators

Q4

Write statements using augmented assignment operators to do the following:

- Add 1 to `counter`.
- Double a variable called `housePrice`.
- Subtract a variable called `penalty` from a variable called `hits`.
- Divide `totalCostOfMeal` by 3.

The Write and WriteLine statements

You have already seen the `Console.Write` and `Console.WriteLine` statements used to display text on the screen.

Using an ampersand (&) as a concatenation operator

You can use `&` to separate strings or variables:

```
Dim length As Integer = 15
Console.WriteLine("The length is " & length & " metres")
```

produces the output:

```
The length is 15 metres
```

Using `&` has the advantage that you do not have to worry about the variable type. You can mix integers, real numbers and strings in the `Write` statement.

Using + as a concatenation operator

Instead of using `&`, you can use a `+` operator to concatenate strings in the output. However, you cannot join a string to an integer or real number, so you first have to convert any numbers to strings.

Q1

What will happen if you write

```
Dim name = {"Mark", "Juan", "Ali", "Cathy",
            "Sylvia", "Noah"}
For index = 0 To 6
    Console.WriteLine(name(index))
Next
```

Operations on arrays

Some array methods are shown in the table below. Assume `a = {45, 13, 19, 13, 8}`.

Array operation	Description	Example	array contents	Return value
Count()	Counts the elements in the array	<code>a.Count()</code>	{45, 13, 19, 13, 8}	5
Length()	Return the number of elements	<code>a.Length()</code>	{45, 13, 19, 13, 8}	5
Contains(item)	Returns True if item exists in array, False otherwise	<code>a.Contains(13)</code>	{45, 13, 19, 13, 8}	True
Max()	Returns the largest value	<code>a.Max()</code>	{45, 13, 19, 13, 8}	45
Min()	Returns the smallest value	<code>a.Min()</code>	{45, 13, 19, 13, 8}	8
Sum()	Returns the sum of all elements of a numeric array	<code>a.Sum()</code>	{45, 13, 19, 13, 8}	98
Average()	Returns the average of all elements of a numeric array	<code>a.Average()</code>	{45, 13, 19, 13, 8}	19.6

Table 5.1 Array operations

5

Example 2

Determine whether the number 100 is in the array

`numbers = {56, 78, 100, 45, 88, 71}`, and if so, print its index.

```
'Program name: Ch 5 Example 2 Array of numbers
Dim numbers = {56, 78, 100, 45, 88, 71}
Dim index = Array.IndexOf(numbers, 100)

If index = -1 Then
    Console.WriteLine("100 is not in the array")
Else
    Console.WriteLine("100 is at index number: " & index)
End If
```

An array is a static data structure, that means it is fixed in size. The elements of an array must all be of the same data type.

produces the output:

```
Key: Wesley      Value: 5
Key: Jo          Value: 9
Key: Betty       Value: 6
Key: Robina      Value: 5
```

To look up the mark obtained by a particular student (the key), write the name of the dictionary followed by the key in brackets. You cannot index a dictionary in the same way as an array, using an index number – an item can only be accessed through its key.

The statement

```
Console.WriteLine("Betty: " & studentMarks("Betty"))
```

produces the output:

```
Betty: 6
```

Q1 Write a statement to print the mark obtained by Robina.

The table below shows some of the most useful built-in dictionary methods.

Method	Description
ContainsKey(key)	Finds if a key is present in the dictionary
Add(key, value)	Adds a key to the dictionary
Keys	Returns all the keys in the dictionary
Remove(key)	Removes an item from the dictionary

Table 7.1: Useful dictionary methods

Example 1a: Looking up a value

If you try to print the mark for a student whose name is not in the dictionary, it will return an error. To avoid this, you should test whether the key is in the dictionary before trying to access it.

```
Dim studentMarks As New Dictionary(Of String, Integer) From
    {"Wesley", 5}, {"Jo", 9}, {"Betty", 6}, {"Robina", 5}
Dim name As String
Console.Write("Enter a student name to look up: ")
name = Console.ReadLine()
If studentMarks.ContainsKey(name) Then
    Console.WriteLine("Mark: " & studentMarks(name))
Else
    Console.WriteLine("Name not found")
End If
```

Example 6

Write a program that uses StreamWriter to create a new file called **temperatures.txt**, or append records to the file if one already exists.

```
Imports System.IO
' Program name: Ch 9 Example 6 writing to a file using
StreamWriter
Module Module1
    Sub Main()
        Dim tempsFile As New StreamWriter("temperatures.txt",
            True)
        Dim city, localTime As String
        Dim temperature As Integer
        Console.WriteLine("Writes data to temperatures.txt")
        Console.WriteLine("If file does not exist, it will
            be created")
        Console.Write("Enter city name, xxx to end: ")
        city = Console.ReadLine()
        Do While Not city = "xxx"
            Console.Write("Enter temperature: ")
            temperature = Console.ReadLine()
            Console.Write("Enter local time: ")
            localTime = Console.ReadLine()
            tempsFile.WriteLine(city & "," & temperature &
                "," & localTime)
            Console.Write("Enter city name: ")
            city = Console.ReadLine()
        Loop
        tempsFile.Close()
        Console.Write("Press Enter to exit ")
        Console.ReadLine()
    End Sub
End Module
```

Q1 Write a program to read and print all the records in temperatures.txt.

Example 7

Read the data in the file **temperatures.txt**, convert all the Centigrade temperatures to Fahrenheit and print out both the Centigrade and Fahrenheit temperatures.

```
Imports System.IO
' Program name: Ch 9 Example 7 process temperatures file
Module Module1
    Sub Main()
        Dim tempsFile = 1
        Dim city, localTime As String
        Dim temperatureC As Integer
        Dim temperatureF As Double
```

Controls

Controls are GUI objects, such as buttons and text entry fields, that are used to interface with the program. They can also be used to display information to the user in the form of a label or a graphic. When adding a control to the form designer, it is a good idea to give it a sensible name, so that your program code will be easier to maintain. The convention is to use a prefix with each control identifier that represents the control's type. For example, the identifier for a confirm button might be `btnConfirm`.

Here is a list of prefixes for some common components along with an example:

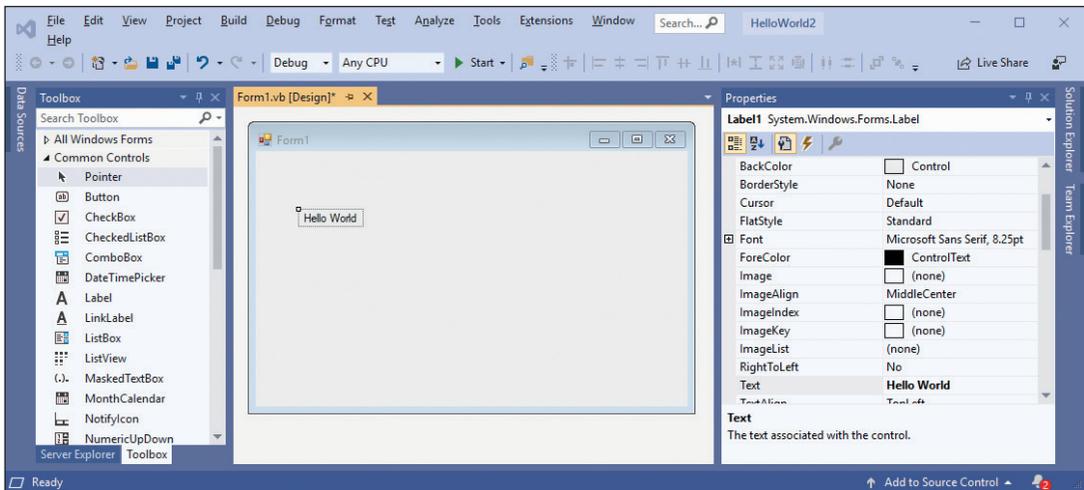
Components	Prefix	Example
Button	btn	btnConfirm
Checkbox	chk	chkSunday
Form	frm	frmConfirmation
Label	lbl	lblInstructions
Textbox	txt	txtFirstName
ComboBox	cbo	cboTasks
GroupBox	grp	grpDeliveryMethod

The "Hello World" program

Example 1

Select the **Label** component from the **toolbox** and drag it onto the form.

In the **Properties** window change the **Text** property to "Hello World".



Sample application 3

This application allows a user (for example, a teacher) to create a multiple-choice test consisting of several questions which could be saved in a text file or database. The input window will look like this:

Create a new **Windows Forms App** project and add the following components to the form. The values for key **properties** are shown below.

Component	Property	Value
Form1	Text	Question entry
	BackColor	192, 255, 255
GroupBox1	Text	""
	BackColor	224, 224, 224
Label1	Text	Name of test
	Font	Arial, 12pt, Bold
TextBox1	Name	txtTestName
GroupBox2	Text	""
	BackColor	224, 224, 224
Label2	Text	Question number
Label3	Text	Question
Label 4	Text	Possible answers
Label5	Text	Correct answer:
TextBox2	Name	txtQuestionNumber

Index

A

ampersand 9
 append mode 72
 array 34
 2D 37
 array processing 36
 Asc() 46, 58
 ASCII code 44
 assignment operator 7
 assignment statement 8
 attribute 81
 augmented assignment
 operators 9

B

Boolean 6, 21
 expressions 23
 condition 28
 breakpoint 123
 built-in functions 58
 button 103

C

call statement 59
 camel case 8
 case statements 22
 Chr() 46
 Clear button 110
 close() method 88
 command object 87
 comments 4
 commit() method 88
 compiler 1
 composite formatted strings 75
 concatenate 5, 9
 connection object 87
 console mode 2
 constants 19
 control 102
 conventions 5
 cursor object 87

D

database
 attribute 81
 connection object 87
 creating 85
 error handling 95
 field 81
 flat file 80
 primary key 81
 query 92
 record 81
 updating 97
 database table
 add new record 82
 data structure
 static 35
 data types 5
 Boolean 6
 collection 51
 floating-point 5
 numeric 5
 string 5
 debugging 120, 123
 decimal 5
 Designer view 104
 dictionary 51
 key 51
 value 51
 Do ... Loop 30
 dot notation 40
 double 5
 Do Until 30
 Do While 30

E

Elseif 21
 encapsulation 64
 event subroutines 110
 ExecuteNonQuery() method 88

F

fields 40, 68
 file processing 77
 flat file 80
 For Each loop 31
 For loop 26
 nested 27
 format check 43, 48
 format items 75
 format modifiers 76
 format operators 75
 formatting output 75
 functions 18, 57
 string 14

G

global variable 63
 group boxes 108
 GUI application 107

I

identifier 81
 index 31
 indexing arrays 34
 Indexing strings 31
 initialise 36
 input statement 33
 integer 5
 integer division 5
 integrated development
 environment 1
 interrupting execution 32
 InvalidCastException 10
 iteration 20

K

key-value pair 53



L

Len() function 43
 length check 43
 length of a string 15
 LIMIT parameter 93
 list 54
 see also array
 lists 54
 local variable 63
 logical operators 7
 logic error 123
 loop
 endless 32
 for 26
 while 28

M

message box 112
 methods 18
 Mod 5
 mode
 append 72
 write 72
 modulus 5
 multi-line statement 11
 mutable 39, 51

N

nested
 For loops 27
 selection statements 23
 nested selection 23
 numbers 16

O

object 18
 operator 7
 operators
 augmented 9
 Boolean 21

P

parameter 58, 60
 primary key 81
 printing 10
 procedure 57
 Properties window 101
 pseudocode 118

Q

query 80

R

random number 24
 reading files 67
 ReadLine 11, 58
 records 68, 81
 deleting 96
 regular expressions 47
 relational operator 7
 Return statement 60
 return values 60
 Reverse() method 54
 Rnd() function 24
 rounding 6

S

selection 20
 sequence 20
 single 5
 sorting a list 54
 Sort() method 54
 SQL 80
 command object 87
 SQLite 81, 85
 StreamReader 70
 StreamWriter 73
 string 5, 14
 concatenate 5
 index 31
 length 15
 processing 31
 Submit button 110
 subroutine 57
 substrings 32
 syntax 1
 syntax errors 16

T

terminate 32, 122
 testing 120
 test plan 121
 text editor 67
 text file
 closing 68
 reading 68
 trace 123
 Try...Catch 96
 tuple 39
 two-dimensional arrays 37
 type check 44

V

validation 43
 length check 43
 regular expressions 47
 type check 44
 variable
 Boolean 21
 convert 18
 declaring 7
 global 63
 initialise 36
 local 63
 naming guidelines 8
 vbCrLf 10
 vbTab 10

W

While loop 28
 widget 106
 Windows form
 closing 113
 Windows Forms app 100
 write statement 9
 mixing variables 12
 WriteLine 9, 58
 writing files 67

X

XML documentation block 60

Learning to Program in Visual Basic >>



This book is intended for individuals and students learning to program. You may already have done some programming in other languages, but not be familiar with Visual Basic. Novice programmers should work through the book sequentially, starting at Chapter 1. It will also be a useful reference book for students on a programming course or anyone working on a programming project.

It teaches basic syntax and programming techniques, and introduces databases, SQL and SQLite, and the development of a graphical user interface for a Windows application.

Questions and exercises are included in every chapter. Answers to these as well as over 120 VB programs for all the examples and exercises given in the book may be downloaded from www.pgonline.co.uk. These programs enable users of the book to try out the in-text examples and check possible solutions to the exercises.

Cover picture:

'Fosse No1'

Oil on linen,

30x30cm © Barbara Burns 2015

www.slaneart.com



PG ONLINE

