

Visual Basic 101 Course Handbook Supplement

By Richard Rost

Published By Amicron Computing

PO Box 1308, Amherst NY 14226 USA www.599cd.com

> First Printing **1/25/2006** Copyright 2006 by Amicron Computing All Rights Reserved



Welcome

Welcome to the 599CD Visual Basic 101 Handbook.

This handbook is designed to be a **supplement** to the full 599CD video course for **Visual Basic 101**. We recommend you use this handbook to follow along with the class videos. This handbook is not meant as a stand-alone study guide.

We do recommend that you watch the course videos one time through, paying attention to the lessons covered. Follow along with the course videos using this guide. Take notes on the pages where needed. Then, watch the videos a second time, practicing the examples yourself on your computer.

Table of Contents

Welcome	2
Table of Contents	2
Introduction	
Lesson 2. What is a program?	4
Lesson 3. Our First Program	7
Lesson 4. Properties	
Lesson 5. Text Boxes and Labels	
Lesson 6. Simple Calculator	45
Lesson 7. More Calculator	
Lesson 8. Compiling	71
Lesson 9. Review	



Introduction

Welcome to Visual Basic 101, brought to you by 599CD.com. I am your instructor, Richard Rost.



Objectives for today's class:

- Learn about Visual Basic
- Create a Program
- Learn Basic Commands like MsgBox
- Learn about Controls, such as buttons and text boxes
- Decision Making (if / then)
- Compiling your Program

Our goal for today is to get you up and writing programs in about an hour.

Pre-Requisites: Basic knowledge of Windows

In this class we will be using **Microsoft Visual Basic 6.0 and Windows XP** for our live-action videos. If you are using an earlier version of VB, you should not have a problem following these videos. VB.NET users will find significant differences in the physical appearance of the VB environment, however the programming topics we will discuss are relatively the same. The version of Windows you have does not matter. Most, if not all of these examples should be applicable regardless of what version of VB you're using.



Lesson 2. What is a program?

We have to start out with some basic definitions for what we're doing today. First, what exactly is a **program**? The dictionary defines a **program** as a set of coded instructions that enables a computer to perform a desired sequence of operations.

That's essentially all a program does. It performs a series of operations based on the instructions given to it by the computer programmer. That's you. Now in the old days, programmers actually had to program their stuff on "punch cards" which I'm sure was no fun. Then we had to type everything in on the keyboard line by line. That was no fun either. Then finally Windows came along, and made programming much, much easier.

Instead of typing in lines of code, Windows programs are made up of **objects**, like command buttons that you push, text boxes that you type information into, and menu bars that you can select functions from. Windows programs are not even easier to write, but they're easier to use.

Here's one of my favorite age-old windows programs: the Microsoft Calculator. It comes with Windows and it illustrates many of the common components that you'll find in Windows programs. Calculator has **command buttons** (those little buttons that you can push).

Edit V	ulacor iew Help	1				
					98.	
	Backs	pace	CE		C	
MC	7		9	1	sqrt	
MR	4	5	6	×	%	
MS	1	2	3	-	1/x	
M+	0	+/-		+	=	



📓 Calc	ulator				
Edit Vi	ew Help				987[
	Backs	pace	CE		C
MC	7	8	9	1	sqrt
MR	4	5	6	×	%
MS	1	2	3		1/x
M+	0	+/-		+	=

And we can put numbers in its **text box**, or we can type numbers right into the text box.

There's a **menu bar** that allows us to pull down menus.

📓 Cak	ulator				
Edit	iew Help	, 1			987.
Г	Scientific Digit gro	: uping	CE		С
MC	7	8	9	1	sqrt
MB	4	5	6	×	%
MS	1	2	3	-	1/x
M+	0	+/-		+	=



And of course we have our standard minimize, maximize, and close buttons.

So the objects in Windows programs: the command buttons, the text boxes make programming and using a program much, much easier.

Windows programs and all the objects in them generally have **properties**, **methods**, **and events**. **Properties** describe a characteristic of a program or an object. For example, a button might have a height, a width, and a color. Those all represent properties.

Windows programs also have **methods**. Methods are built-in functions that a program performs. For example, when we start calculator, it knows to pop a little zero in the window and wait for user input. That's a built in method.

And most Windows programs, if not all, have **events**. Windows programs are event-driven. That means they sit around waiting for the user to do something, generating an event. If we click on the number 9 button, that generates an event inside the program.

Calculator
Backspace CE C
MC 7 8 9 / sqrt
MR 4 5 6 * %
MS 1 2 3 - 1/x
M+ 0 +/ + =
· · · · · · · · · · · · · · · · · · ·

There's a little programming code underneath this button (that we'll see how to do today) that then puts a nine in the text box. That's called an event.

Properties, methods and events make up the core of Visual Basic programming.

So what is Visual Basic?

Visual Basic is a **programming environment**. It includes a **programming language**, and a **graphical user interface**. The programming language allows us to enter in commands that the computer will execute. The graphical user interface allows us to create forms and buttons and text boxes and then other things visually by clicking them and dragging them with a mouse. The combination of the two provides an extremely powerful programming environment.



Lesson 3. Our First Program

Now, lets start Visual Basic, and build our first program.

Here we are at the Windows desktop.



We're going to click on "Start", then "All Programs," and you should **see Microsoft Visual Studio 6.0 or Microsoft Visual Basic 6.0** (depending on what you installed)





And then you'll see **Microsoft Studio 6.0 Tools and Microsoft Visual Basic 6.0**. This is what we're going to use today. Go ahead and click on Microsoft Visual Basic 6.0.

The first window we get up is the new project window.



Now there are many ways to start new projects. A **project** in Visual Basic is essentially your program. For today, make sure **standard.exe** is selected and then click on open.

Standard EXE	ActiveX EXE	ActiveX DLL	ActiveX Control	VB Application Wizard	Ī
VB Wizard Manager	ActiveX Document Dll	Activex Document Exe	Addin	Data Project	
		P= 6		Open Cancel Help	
Don't show this	s dialog in the fu	iture			



Now we're sitting at **Project1** (a blank project) in Microsoft Visual Basic. You'll notice the Visual Basic interface is very similar to other Windows programs. We have our title bar across the top, our menu bar, and a tool bar. There's also a tool box going down the left hand side.

🐂 Project 1	l - Microsoft Visual Basic [design]	
<u>File E</u> dit <u>V</u>	jew Project Format Debug Run Query Diagram	<u>T</u> ools <u>A</u> dd-Ins <u>W</u> ind
🛛 😼 • 🍓	- Te 😂 🔒 🕺 🛍 🏙 🔛 🖂 🕨	II 🗉 💐 🗳 🔁
Coporal	🕿 Project1 - Form1 (Form)	Project - Project1
General		
N 🔛	🖣 Form1	🖃 🥵 Project1 (I
A abi		En Forms
	l	
•		Properties - Form
		Form1 Form
4 D 🔺		Alphabetic Catego
Ö 🗆		Appearance 1

In the middle we have a big blank form – this is where our program will go. On the far right we have our project window, a properties window, and a form layout window.

	Project Project
Project1 - Form1 (Form)	Project - Project A
a, Form1	
	🖃 🤧 Project1 (Project1)
	🖻 🚔 Eorms
	Form1 (Form1)
	1
	Properties - Form1
	ropercies - romit 🔼
	E
	Form1 Form
	Alababatta January al
	AIDDADEDC Categorized
	Caption
	Determined as the three three directions of the second
	Returns/sets the text displayed in an
	object's title bar or below an object's
	licon.
	100111
	Form Layout 🛛 🗙 🗙
•	



						ĺ												`						[-1			
		Ē	1	, I	Fe	DI	"Г	n	1																													_]	>	<			I	Γ	-
				:	:	:				:	:				:	:	:						:	:			:	:	:	:					:	:	:	:	:	2	:	:	:		- 1		I.	
																				,																									- 1			
			•	•	•	•	•			•	•	•			•	•	•	•	•			•	•	•			•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		- 1			
	•		•	•	•	•	•			•	•	•			•	•	•	•	•			•	•	•			•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		- 1			
	•		•	•	•	•	•			•	•	•		•	•	•	•	•	•			•	•	•			•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		- 1			
	•		•	•	•	•	•		•	•	•	•		•	•	•	•	•	•		•	•	•	•	• •	•	•	•	•	•	•			•	•	•	•	•	•	·	•	•	•		- 1			
	•		•	•	•	•	•			•	•			1	•	•	•	•	•		•	•	•	•	• •		•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		- 1			
	•		•	•	•	•	•		•	•	•	•	1	•	•	•	•	•	•		•	•	•	•			•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		- 1			
			•	•	•	•	•		•	•	•			1	•	•	•	•	•		•	•	•	•			•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		- 1		2.	
	1													1																							:								- 1			
_	1			:										1				Ń									:									:	:				:	:		le -	- 1		Li,	۳ı
-	1																	L	9																									E	- 1		E	
																		۰.	Ø,																										- 1		ll I	Εn
																																													- 1		12	
													Ι,																																- 1			
													Γ,							,							•									•	•								- 1			A
				•	•	•				•	•	•			•	•	•			,			•	•			•	•	•	•					•	•	•	•	•	•	•	•	•		- 1			_
	•		•	•	•	•	•			•	•	•			•	•	•	•	•	,		•	•	•			•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		- 1			I۸.
	•		•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		•	•	•	•	•		•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		- 1			iH
	•		•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		•	•	•	•	•		•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		- 1		Ē	-
	•		•	•	•	•	•			•	•	•		•	•	•	•	•	•	1		•	•	•			•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		- 1		1	Га
	•		•	•	•	•	•		1	•	•	•	1	•	•	•	•	•	•	1		•	•	•		1	•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		- 1			_
	•		•	•	•	•	•			•	•	•		•	•	•	•	•	•	1		•	•	•			•	•	•	•	•			•	•	•	•	•	•	•	•	•	•		- 11		F	Re
	•		•	•	•	•	•		•	•	•	•			•	•	•	•	•		•	•	•	•	• •	•	•	•	•	•		•	•	•	•	•	•	•	•	•	•	•	•		- 8			

Turn your attention for moment if you will, to this **Form1** window in the center.

Just like the calculator program that we looked at a minute ago, most Visual Basic applications will run on a form. In fact, most Visual Basic applications will have one or more forms. Here's calculator again.

		,	. 0				
					0.		Ш.,
	Backs	pace	CE		С		L.
MC	7	8	9	1	sqrt	L I	Pro
MB	4	5	6	×	%		For
MS	1	2	3	-	1/x		Alp
M+	0	+/-		+	=		Сар
	MC MR MS M+	MC 7 MR 4 MS 1 M+ 0	Backspace MC 7 8 MR 4 5 MS 1 2 M+ 0 +/-	Backspace CE MC 7 8 9 MR 4 5 6 MS 1 2 3 M+ 0 +/- .	Backspace CE MC 7 8 9 / MR 4 5 6 * MS 1 2 3 - M+ 0 +/- . +	Backspace CE C MC 7 8 9 / sqrt MR 4 5 6 * % MS 1 2 3 - 1/x M+ 0 +/- . + =	Backspace CE C MC 7 8 9 / sqrt MR 4 5 6 * % MS 1 2 3 - 1/x M+ 0 +/- . + =



Just picture in the background, behind all these buttons, the big blank empty form. This form is essentially our canvas and on it, we're going to paint different controls.

×		
General	Sproject1 - Form1 (Form)	
k 🔛	🗟 Form1	-
A abi		
ו-		
• •		
	•	•
ৰ চা হ		
Ö 🗆	ka	
8 <		
		-

In controls we have buttons and labels and text boxes and pictures and so on. Using these different controls we can create our own custom programs just like calculator.

Lets start by building our very first program by putting a button on this form and making this button give us a little message when the user clicks on it. Sounds pretty simple right? Let's take our mouse and find the **command button**.

×		
General	🐂 Project1 - Form1 (Form)	
🕨 🔛	🐂 Form1	
A abl		
í n		
		•
ৰ চা হা		



Go ahead and clic k on it and take your mouse over on the form. Notice that the mouse pointer has changed into a plus instead of a pointer icon. Go ahead now and click and drag to draw a little box.

×		
General	Project1 - Form1 (Form)	
k 🔛	S. Form1	ſ
A abl		
• •	· · · · · · · · · · · · · · · · · · ·	
		•
1 D 🔺		
Ö 🗆		
8 <		
	-	-

Let it go, and that will create a command button.

General	🔍 Project1 - Form1 (Form)
N 🔛	🖌 Form1
A abi	
	Command1
• •	
1 N N	
Ö 🗆	
⊳ 🗞	



The name of this button is **Command1**. In Visual Basic, all the different objects that you will create, whether they're buttons, or pictures or text boxes will all have a name. That name is one of their **properties**. In fact, you can see the name in the Properties window that says, "Properties for Command1".

orm)	
	Project1 (Project1) Forms Form1 (Form1)
indi	Properties - Command1
• • • • • • • • • • • • • • • • • • • •	Command1 CommandButton
	(Name) Command1
	BackColor 848000000F Cancel False
	Caption Returns/sets the text displayed in an
	Form Layout 🛛 🗙

Now that we have a button on our form, let's go ahead and run our program. In other words, we're going to start up our program and see what happens when we click on that button. To run your program, find the button that looks like a little triangle that points to the right.

• 🍓	• TE 😂 🖬 🐰 PB 🖻 🗛 🗠 🖓 🛯 🖌 🛛 😽 🖆 名 '	2
ral	S. Project1 - Form1 (Form)	Pro
	Form1	
abl		
	Command1	Pro
۲		Co
		Alj
ৰ ম		(N Ap
3		Ba
		Caj



Go ahead and click on that button now.

ie Found		
Command1	l⊋	
	-	

You can see our program is now running. Here it says **Form1** and **Command1**. Let's click on the button. You'll see that nothing's happening. Why is nothing happening? Well, we haven't told our program to do anything yet when the user clicks on that button. We haven't given this button an **event** yet. So let's see how we do that. Let's go ahead and close this program just like we would close any other Windows program. Notice that we're back in the Visual Basic Design Editor.

General	🗟, Project1 - Form1 (Form)
N	Form1
A abi	
	Command1
• •	
4 D 🔺	
Ö 🗆	
🗀 🖹	
R 🔨	
R 19	



In order to tell this button to do something when we click on it, we have to give it an event. We have to do some programming. So let's double click on this button. Take your mouse and double click on it. Notice a code window opens up.



This is called a **code window**. Here it says Private Sub Command1_Click(). What does all this mean? Well for today's class, you don't have to know what all this stuff means. All you really have to know is that everything that happens between the first line (*Private Sub Command1_Click()*) and the second line (*End Sub*) will run whenever you click on that button.

This is a **sub** or **subroutine** that will run when a user clicks on the Command1 Button. Everything that we type in the middle is going to go off when the user clicks on the Command1 Button.

The word **private** just means that the subroutine is private to this specific form.



Let's put an actual command in here and see the program do something. To give yourself some extra room, hit enter a couple of times on your keyboard. Now you have plenty of blank space. Click back up somewhere in the middle



Tab in with the tab key (press the tab key). That's going to move your little blinking cursor over to the right a little bit. It's just considered good programming form to indent your commands inside the private sub.





Do you have to do this? Do you have to indent your code inside the subroutine? No. You don't have to. It's just when you build more complex and more complicated programs – the code can sometimes get difficult to read, so it makes it easier to read if it's indented. But you don't have to do that if you don't want to. Now you're going to get your first command. Type this in:

msgbox

and then a space. Notice that a little yellow tool tip pops up.

• 💷	
⊾ [ab]	🗌 💭 Project1 - Form1 (Code)
	Command1 Click
•	Private Sub Command1_Click()
	msgbox
e a	As Vbl. agBox(Prompt , [Buttons As VbMsgBoxStyle = vbOK(As Vbl. agBoxResult
) 🗆	End Sub

Don't worry about that. It's just giving you some suggestions to tell you what goes next. Next, press open quotes, and then **Hello World** and then close quotes, and press enter.

• •		<u> 위유</u>
⊾ [ab]	💭 💭 Project1 - Form1 (Code)	
] 🗖	Command1 🔽 Click	•
•	Private Sub Command1_Click()	
	MsgBox "Hello World"	
의 <u>위</u>		
) 🗆	End Sub	



We've now entered in our first command. Essentially, **MsgBox** just pops up a message on the screen. And this specific message box is going to say "Hello World" whenever the user clicks on this button.

<u>D</u>ebug <u>R</u>un Diagram Add-Ins Window Edit View Project Format Query Tools Н 8 • 🍖 • 🛅 🖻 🔛 - ha 🛍 🚧 💐 😭 ÷ž \square II X ٨Ì Pr 🛢 Project1 - Form1 (Form) eral Start C <u>4</u> • - 🗆 🛛 🔄 Form1 Ε abl 🜉 Project1 - Form1 (Code) _ 🗆 × Command1 Click Ŧ • Pr Private Sub Command1 Click() Œ * 6 MsgBox "Hello World" A

Alright. Let's give it a try. Click on the start button and see what happens.

Here's our program. Our program is running now. Let's click on the button and see what we get.





And there it is! There's our message box that say's "Hello World." Click on the OK button. That will close the message box and put you back into your program. Congratulations! You've just written your first Visual Basic program. Give yourself a round of applause (just not too loud cause then the people around you will think there's something wrong with you.). Let's go ahead and close this. You'll return to the Visual Basic Editor. Let's go ahead and save our work. Let's click on the floppy disk button to save our program.

Save File As		? × "	oje
Save in: 🗁 VB98 💌 🗲 🛍 🕻	* 🎟 •		
🛅 Template 🛅 Wizards			orr
R		m	Fo Ca
File name: Form1	Sav	e	
Save as type: Form Files (*.frm)	Canc	el s	_
	Help	s s	the
		rm Layou	t

Now for some strange reason, the folks at Microsoft decided to save Visual Basic data files in the **VB98** folder. Instead, save your files in your "My Documents" folder.





Save File As		? × roje
Save in: 🗁 My Documents 💽 🗢 🛍	r 🕂 🔁	
📸 My Music		Forn
My Pictures		
My Webs		
New Folder		- Fo
		m
] Ca
File name: Form1	Оре	n 🖡
Save as type: Form Files (*.frm)	Cano	el s
	Help	s the
		rm Layout

Let's create a folder in here to store our VB stuff. Click on the "Create Folder" button.

And type in "VB Files" and then we can double click in our folder to go into our VB files folder

Save File As		? ×	roje
Save in: 🜔	My Documents 💽 🗢 🗈 💣 🎟 -		liect
过 My Music			Form
My Pictures			51
My Webs			
			- Fa
Ů			m
			Ca
File name:	Form1 Oper	n -	в
Save as type:	Form Files (*.frm)	el	s
	Help		s the
		m Layo	but



And now let's save the form. We'll give it a filename called, "**FirstProgram**." This will save the form as an "frm" file. Hit Save. And now we have to save the project file. The project file is like an overall master file that keeps all the information about this particular program in one place. Let's call this one our "**FirstProject**."

Save Project As		? ×	roje
Save in: 📴 VB Files		* 🏢 -	
			ject Form
			[*1. F
			- Ea
			m
			l ca
			100
File name: FirstProject		Save	9
Lines islest 14			s
Save as type: Project Files (*.vbp)	<u> </u>	Cancel	_
		Help	s the
1	-	Form Layo	ut

Then hit save. And there we go!

Now you see in just a few minutes, you were able to create your own program, run it, make it do some stuff, and then save that program. You are now a programmer!



Lesson 4. Properties

In this lesson, we're going to take a look at some of the properties of the objects we created in the last lesson. Here we are back at our Project1.



Now when we run this project, we can see we've got Form1 and Command1. Not very informative, is it? It would be nice if we could actually put some stuff on here. Like have Form1 say maybe, "First Project" or "First Program," and have our button say something like "Click Me." Command1 isn't very user friendly, is it? Let's close this.





Now to adjust these properties, let's go ahead and close the code window. Now we're back to our form window.

i.															_				_								-	-										-	-	_	_	1	н	B	ruj	ELL	-	1
	E	1	P	ra	j	20	ti	ŀ	- 1	Fc	or	'n	1		(F	0	r	m)																			J.	_		×		I			-8		f
Ī																					1						_	_												Р			н	L			·	P
Ш		E	٦,	F	o	۲N	ní	l								ļ	1																	_				×	¢	L			L	ſ	<u> </u>	3	Pr	o
							÷	i	÷	i	÷	÷	i	i	:	:	:	÷	÷	i	÷	i	i	:		i	Ĵ,	Ĵ	÷	÷	÷	i	:	:	:	:	:	:	:	L			L		•	<u> </u>	-	į
Ш		:	: :		:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	Ŀ			L				1.	
Ш		:	: :		:	:	:														ī.	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	Ŀ			L		•			
		:	: :		:	:	:				(Co	DN	nn	na	n	q.	1			L	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	Ŀ			L	ľ	Dron	orl	ie	5
Ш		:	: :		:	:	÷	h													1	:	:	:	:	:	:	÷	÷	÷	÷	÷	:	÷	÷	:	:	÷	:	Ŀ			L	i.	тор	GU		2
		•			•	•	÷	÷	÷	÷	÷	÷	÷	÷	•	•	•	÷	÷	÷	÷	÷	÷	•	•	•		•	÷	÷	÷	÷	•	÷	÷	÷	÷	÷	•	Ŀ			L	I	For	n1	Fo	or
							•	÷		÷		:							:	÷									•	•	•	÷		÷	•	•	•			Ŀ.			L		Alol	hab	eti	c
Ш		:			:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:		:	:	:	:	:	:	:	:	:	:	:	:	Ŀ			L					
Ш		:	: :		:	:	:	:	:	:	:	:	Ì	Ì	:	:	:	:	:	:	:	Ì	Ì	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	Ŀ			L		Bac	kCo	lor	•
																																								L.,					Bor	der	SEC	وار

Now here's the form, and here's the button. Let's start with the button. Let's click once on the button. Notice how we get these little dashes and dots around it.

ij.,	P	r	0	je	20	t	1		1	Fo	DI	10	n	1	(F	0	rr	n)															I	-		1	x	1	l		
i	1	. 1	Fo	DI	٢	n	1																									-			2	×	1				l	ľ	🖃 😼 Pro
			:						:	÷								:	:	÷	Ĵ													÷	÷	1	1				l	l	- Ē. 🕾
		•	•						•	•						•	•	•	•	•						 								•	•						н		
• •		•	•	•	•		-		•	·	•	•	•		-	•	•	•	•	·	•		-1	•	• •		• •	• •	• •	• •	• •	•	•	·	·	•					н		
• •		•	•	•	•										-							T,		•	•	• •	•	• •	•	• •	• •	•	•	·	·	•					н		∢
• •		•	•	•	•							~						ы				I.		•	•	•	•	• •	•	• •	• •	•	•	•	•	•					н	P	
• •		•	•	•	•	•						U	or	m	m	a	n					1	÷	•	•		•	• •	•	• •	• •	•	•	•	•	•					н		_
• •		•	•	•	•																	1	.'	•	•	•	•	• •	•	• •	• •	•	•	•	•	•					н		roperties
• •		•	•	•	•		Π		_	_	_	_	_	_	П		_	_	_	_	_		N		• •	• •	• •	• •	• •	• •	• •	•	•	•	•	•					н	12	-
		•	•	•	•				•	•	•	•	•			•	•	•	•	•	1		Ъ	ιč	•	• •	•	• •	•	• •	• •	•	•	•	•	•					н	1	Commany
• •		•	•	•	•			•	•	•	•	•	•			•	•	•	•	•	1	•	•	v	•	• •	•	• •	•	• •	• •	•	•	•	•	•					н		Comman
• •		•	•	•	•			•	•	•	•	•	•			•	•	•	•	•		•	•	•						• •	• •	•	•	•	•	•					н		
			1							1									1																						H		Alphabetic
			1							1																									1						H		



These dots are used to resize the button.

errojecti - rom	r (ronny	
🖹, Form1		🖃 😼 F
		 Ē.
:::::: :		 •
Co	mmand1	
::::: :		 Propert
		 Comma
		Comme
		 Alphabe
		 Miphabe
		(Name)
		Ann ann
		 Appear
		 PackCol

You can also move this button, by the way, by clicking on it and dragging it. But if you look over on the right, you'll see the properties window. And right now since we have Command1 selected or highlighted, it says the properties for Command1. The name of this button is Command1. Look for the **caption property**.

🖨 Form1		F	orms
	•••••		Eorm1 (FirstProgram
Comman	nd1 📑 🗄 🖓 🖓	Properties -	Command1
		Command1	CommandButton
		Alphabetic	Categorized
		(Name)	Command1
		Appearance	1 - 3D
		BackColor	8H8000000F
		: Cancel	False
		Caption	Coremand1
		: CausesValida	ation TrueS
		: Default	False
		DisabledPictu	ure (None)
		DownPicture	(None)



Caption is what actually shows up on the button, and it doesn't necessarily have to be the button's name. This is just one of the properties for the command button.

	🖃 🎰 Project1 (FirstProject.vl
🖹 Form1	E Forms
· · · · · · · · · · · · · · · · · · ·	Form1 (FirstProgram
Command1	Properties - Command1
· · · · · • • • • • • • • • • • • • • •	Command1 CommandButton
	Alphabetic Categorized
	(Name) Command1 🔺
	Appearance 1 - 3D
	BackColor 🗌 &H8000000F
	Cancel False
	Caption Command1
	CausesValidation True
	Default False
	DisabledPicture (None)
	DownPicture (None)

Let's click on the caption and change this to "Click Me."

, Projecti - Poniii (Ponii)	Project1 (FirstProject.	/
🚔, Form1	E Forms	-
	E Form1 (FirstProgra	m
Click Me	Properties - Command1	×
· · · · · · · · · · · · · · · · · · ·	Command1 CommandButton	-
	Alphabetic Categorized	
	(Name) Command1	
	Appearance 1 - 3D	
	BackColor 🛛 &H8000000F	
	Cancel False	
	Caption Click Me	
	CausesValidation True	
	Default False	
	DisabledPicture (None)	
	DownPicture (None)	•
p		



Notice the caption on the button changes as well. The button will say whatever you put in the caption property. That's how the caption property works. Now let's go ahead and run our program. Click on the start button, and look at that! We changed the button so it says, "Click Me" instead of Command1.



If you click on it, it will generate the "Hello World." We've changed one of the button's properties.

			Project - Pi
📮 Form1		_ 🗆 🗡	
Click Me	2		(3) Pro; (3) (1)
	FirstProject X		
	Hello World		
	ОК		



💐 Project1 - Form1 (Form)	Project 1 (FirstProject.vl 🔺
Click Me	Forms
	Form1 Form Alphabetic Categorized
	(Name) Form1 Appearance 1 - 3D AutoRedraw False BackColor 8448000000F BorderStyle 2 - Sizable Caption Form1 ClipControls True
•	DrawMode 13 - Copy Pen

Let's go ahead and close the program. Now what about the form itself? Click on the Form's title.

And notice that we have the properties for form one. Remember, almost every object in a Visual Basic program has its own properties. A button has its own set of properties. The form has its own set of properties.

🖷, Project1 - Form1 (Form)	Project1 (FirstProject.vl 🔺
Still Ma	Forms
	Properties - Form1 × Form1 Form
	Alphabetic Categorized
	(Name) Form1 Appearance 1 - 3D
	AutoRedraw False
	BorderStyle 2 - Sizable Caption Form1
	ClipControls True
	DrawMode 13 - Copy Pen 🗨



	1	P	r	oj	je	:c	t	1	-	F	- 0	r	n	n	1	(F	o	r	п	n)								Γ	- 🕒 📋	/I ▲	
				M	y	F	1	' s	t	F	Pr	0	g	1	a	n	n				:										Errms		
L	:	:	:	:	:	:					С	lio	sł	<	M	1e	•					:	•	:	:			:	:	F	roperties - Form1	>	<
	:	÷	:	÷	÷	:	ł	:	:			:	:	:			:	:	:			:		÷	:	:		:	:	[orm1 Form	•]
	:	:	:	:	:	:	:	:			•	:	:	:				:			:	:	•	:	:	:		:	:		Alphabetic Categorized		
	:	:	:	:	:	:	:	:	:			:	:	:			:	:	:		:	:	•	:	:	:		:	:		(Name) Form1		ſ
	:	÷	÷	÷	÷	÷	:						÷	:				÷			:	:		÷	:				÷		Appearance 1 - 3D		
	:	:	:	:	:	:	:	:	:		•	:	:	:			:	:	:		:	:		:	:	:		:	:		AutoRedraw False		
		:	:	:	÷	:	:	:					:	:				:			:	:		÷	:				:		BorderStyle 2 - Sizable		
	:	:	:	:	:	:	:	:	:			:	:	:				:	:		:	:		:	:	:		:	:		Caption y First Program		
		:	:	:	÷	:	:						:	:				:			:	:		÷	:				:		ClipControls True		
٥																								1							ControlB0X True DrawMode 13 - Copy Pen	-	

Now you can click on the caption for the form and make this say, "My First Program," for example.

Let's run our program now.

Statistics	₽ () P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P() P - () P - ()() P - ()() P - ()()()()()()()()()()

And there we go! It's a little more user-friendly now. It says, "My First Program" across the title bar. That's the caption property for the form. Let's close our program again.



	3.	F),	0	j	P	C	ł	1	-	F	0	r	m	1		(F	C	or	n	n))	I							Þ	š	P	roj	ect	L (F	irs	st	Pr	oj	ecl		d,
				~	1		F	ir	5	t	P	r	D	g	ra	a r	i											1	1]	2	9 F	orm	s orm	1 (íFi	irs	tPr	'nai	rai •	n I
	:	:	:	:							1	CI	ic	:k	h	d,	e					:	:					Pro	p)e	rt	tie	25	- Fo	rmi							
	:	:	:	:				7	•	•						•	•					:	:					Fo	r	m	1	F	orr	n								I
	:	:	:					:	:	:				•	:	:	:				•	:	:					A	lp	ha	ь	et	ic:	Cat	ego	riz	e	d Ì				
	:	:	:	:				:	:	:	:	:			:	:	:	:	:			:	:		: :			0	٧a	m	e))			Fo	orn	n1					4
	:	:	:	:				:	:	:	:			:	:	:	:	:			:	:	:					A	Ρļ)e	ar	'a	nce	;	1	- 3	3D	I				
	:	:	:	:		1		:	:	:	:	:		:	:	:	:	:	:		:	:	:					A	ut	:oF	Re	ed	Irav	v	Fa	als	е					
	:	:	:	:				:	:	:	:	:		:	:	:	:	:	:		:	:	:			:	:	В	ас	k	Ξo	olo	r] <u>}</u>	хн	180	000	00	N	
	:	:	:	:				:	:	:	:			•	:	:	:	:			•	:	:					B	ſ	P	al	et	te:	Sy	sten	n					ή	\$
	:	:	:	:				:	:	:	:			•	:	:	:	:			•	:	:					c		F		M	enu Vind	J Te>	t evt		-					
۵		•	•					•	•						•	•						•						D				Ai Ai	ctiv ctiv	e Tit e Bo	le B rder	ar	Te	exI	t			

Now there are many other properties for command buttons and forms. One of the other properties for example is back color.

Click over on the BackColor properties and you'll see a little drop down. Click on the drop down and it will open up a list of colors. Select the palette tab and you'll actually get a color palette.

🖣 Project1 - Form1 (Form)	Project1 (FirstProject.vl
🖣 My First Program	E Forms E Form1 (FirstProgram ▼
Click Me	Properties - Form1 🛛 🗙
	Form1 Form
	Alphabetic Categorized
	:::: (Name) Form1
	::::: Appearance 1 - 3D
	:::: AutoRedraw False
	::::: BackColor 0000F& 🗸
-	Bo Palette System



💐 Project1 - Form1 (Form)	Project 1 (FirstProject.vl 🔺
S. My First Program	Forms
Click Me	Properties - Form1 🛛 🗙
	Form1 Form
	Alphabetic Categorized
₽	(Name) Form1
	Appearance 1 - 3D
	AutoRedraw False
	BackColor 📃 &H00FFF 💌
	BorderStyle 2 - Sizable
	Caption My First Program
	ClipControls True
• • • • • • • • • • • • • • • • • • • •	ControlBox True
-	DrawMode 13 - Copy Pen

You can click one of these different colors like light blue.

Notice how we've changed the back color or the background color for the form. If you scroll down the properties list, you'll see all kinds of different properties. There's enable, fonts, ForeColor, icon, and more. One of the other properties you can change is the name of the object itself. For example, if we click on our command button, we can actually change the name of this button from Command1 to something else.

💐 Project1 - Form1 (Form)	□ 🕞 🔤 🔤	(FirstProject.vl 🔺
🐂 My First Program	Forms	rm1 (FirstProgram
Click Me	Properties - Con	nmand1 🛛 🔀
	Command1 Com	imandButton 📃 🖃
	Alphabetic Cate	gorized
	(Name)	Command1 🗾
	Appearance	1 - 3D
· · · · · · · · · · · · · · · · · · ·	BackColor	8H8000000F
	Cancel	False
	Caption	Click Me
	CausesValidation	True
	Default	False
	DisabledPicture	(None)
	DownPicture	(None)



	Projec	t - Project: B
🖹 My First Program		Project1 (
Click Me		- 😁 Forms
Ŕ		

Let's change this button's name to "HelloButton." Let's run our program again.

Notice our button says, "Click Me." Click on it. But wait a minute – something's not right. Our button's not working anymore. We just broke our button! Let's close our program and figure out what happened.

Back in our Visual Basic Design Editor, let's double click our button and check its programming code.

E HelloButton	- Click	-
Private S	ub Command1_Cli	ick()
MsgBo	x "Hello World"	,
End Sub		
Private S	ub HelloButton	Click()
End Sub		
	42	



Okay, we double clicked and now Visual Basic put us down in a new private sub! What's going on?! Well you have to be careful if you change the name of an object. We changed the name of the button. We renamed it from Command1 to HelloButton. When you do that, any programming code that happens to be in that object is lost. It's still there as you can see, but Visual Basic loses track of it. All you have to do, is take the code on top, cut it out...

		~		
• 🔁		2		Ĵ
×				P٢
eral				E
A 2		<u>,</u> F	Project1 - Form1 (Code)	Ē
	Ę	(G	eneral) - Command1_Click -	1
abl			Projector Carlo Common da California	
			Private sub commandi_click()	Ľ
_			MagBox "Hello World"	Pr
•				H
三周			End Sub 📉	ρ
				7
1			Private Sub HelloButton_Click()	Å
			End Sub	B
_			End Sub	c
	:			C
				LC.

and then paste it below.

 - **	- 1	
eral		Paste
		Project1 - Form1 (Code)
abl		HelloButton Click
		Private Sub Command1_Click()
•		End Sub
ন স		Private Sub HelloButton_Click()
=		MsgBox "Hello World"
Ē		End Sub



And there we go. And we can delete the extra lines of code by simply highlighting them, and hitting delete on our keyboard.

So keep in mind that if you change the name of your button, you've got to change your code too.

Now we've made our first program and we've learned how to change some of the properties of the different objects in our program: our command button and our form's properties.



Lesson 5. Text Boxes and Labels

In this lesson, we're going to learn how to make our program a little more functional with some new controls. Specifically, we're going to learn about **text boxes, and labels**.



Here we are, back in our good ol' little program. Let's now add some additional **controls** to this program. Let's add a **label** and a **text box**. A **label** is just used for displaying some information on your form. You can put words in there, numbers... whatever kind of text you want to have. A **text box** is used to actually store data. Let's see how these work. First, let's move the little "Click Me" button down a little bit. Click on it and hold the mouse button down and drag it down.

×		
General	S Project 1 - Form 1 (Form)	V
k 🔛		
	🖷, My First Program	
	Click Me	
• •		
ৰ চ ম		
Ö 🗆		
🗀 🗈		
🔊 🔨		
		_



	6	×																	
	Gen	eral		🐂 Pro	ojecti	L - F	orm	1 (Fa	rm)									- 🗆 >	C
(\geq 1		in, N	1y Fir	st P	rogr	am								_ [l ×	1	
	$ \mathbf{A} $	labi				:::						11	:::		11		:::		
\langle	EXA.	Z. I		:::								::	: : :	::	::		: : :		
										• •	·ι·	• •	• • •	• •	• •				
	_	-								·	T. i	<u></u>	<u> </u>						
		•		111		:::			111	2	295	x 37	75		11				
		_		:::		:::						::	:::	::	::		:::		
										• •	• • •	• •	• • •	• •	• •	• • •			
		A	1	1.1.1								• •		• •					
	▲ ►	5					or					11							
	. <u>#</u> .			111			Clic	k Me		11		11	:::	::	11		:::		
	0			:::						::		::	:::	::	::		:::		
	~									• •	• • •	• •	• • •	• •	• •	• • •			
	-			:::		:::				::		11	:::		11		:::		
	R.	~		:::		: : :						::		::	::				
		लाम			_	_	_	_	_		_	_	_	_	_	_	_	-	

Now grab a label. Click on the "A." Then come on over to your form and click and draw a box.

Let your mouse go. And now we have an object that says "Label1."

×	
	🖹 Project1 - Form1 (Form)
	S My First Program
A abl	
	Label1
• •	
4 D	Click Me
Ö 🗆	
🗀 🖹	
N 🔨	



, Project1 - Form1 (Form)	Project 1 Project 1	(FirstProject.v ; prm1 (EirstProgra	vI ▲
Enter Your Name:	Properties - La	oel1	×
19	Label1 Label		-
	Alphabetic Cat	egorized	
	(Name)	Label1	
	Alignment	0 - Left Justify	
Click Me	Appearance AutoSize	1 - 3D False	
· · · · · · · · · · · · · · · · · · ·	BackColor	0000008H&	F
	BackStyle	1 - Opaque	
	BorderStyle	0 - None	
	Caption	ter Your Name:	
	DataField 4	5	•

In the properties for Label1, change the caption so that it says, "Enter Your Name" and then a colon.

That's basically just a message that you want to give to the user. Now notice how the background of the label is gray. That's because it's specified in the properties box. We could change the back color of this label to match the background color of the form. But then if we change the form again, we've got to change the label. So an easier thing is to change it from opaque to transparent.

, Project1 - Form1 (Form)	Project1 (FirstProject.v Project 1 (FirstProgram Forms Form1 (FirstProgram	
Enter Your Name:	Properties - Label1	×
	Label1 Label	-
	Alphabetic Categorized	
	(Name) Label1	
	Alignment 0 - Left Justify	
	Appearance 1 - 3D	
	AutoSize False	
· · · · · · · · · · · · · · · · · · ·	BackColor 🗌 &H8000000F	
	BackStyle 👌 - Transpar 💌	
	BorderStyle 0 - None	
	Caption Enter YouÑNam	1
	DataField	•


Notice how that happened. The BackStyle changes from opaque to transparent and now the background of the label is transparent. So if we change the color of the form again, we don't have to keep changing all our labels.

🐂 Project1 - Form1 (Form)		Proj	ect1 (Forms	(FirstProject.	vl 🔺
🗟, My First Program			L For	m1 (FirstProgra	
Enter Your Nama:		Properties	- Labe	el1	X
•		Label1 Lab	el		-
2295 × 255		Alphabetic	Cate	gorized	
		(Name)		Label1	
		Alignment		0 - Left Justify	
····		Appearance	,	1 - 3D	
Ulick Me		AutoSize		False	
· · · · · · · · · · · · · · · · · · ·		BackColor		🔲 &H80000001	F
		BackStyle		0 - Transpar 💌	
		BorderStyle		0 - None	
		Caption		Enter Your Nam	μ
• • • • • • • • • • • • • • • • • • • •	-	DataField			•
					_

The label is a little big. So grab the box from the bottom and shrink it just a little bit. Let's also resize it this way.

a, Project1 - Form1 (Form)	
🐂 My First Program 📃 🗖	
Enter Your Name: 🕂 🔸	
	Propertie
	Label1 L
	Alphabet
	(Name)
Click Me	Alignmen
	Appeara
	AutoSize
	BackColo
	BackStyle
	BorderSt
	Caption
	DataField



Let's put a text box next to it so that the user can type in their name. Click the "ab" text boxbutton on our toolbox. Then draw a box out on our form.

General	🖷. Project1 - Form1 (Form)	
A labl	🐂 My First Program	
	Enter Your Name: Text1	

There's a text box. Text boxes have a property called, "Text." By default, it says "Text1." Let's highlight it and delete it. That way our text box will start blank.

		rm1 (FirstPron
Ducinght Court/Court	Properties - Tex	t1
Projecti - Formii (Formi)	Text1 TextBox	
🐂 My First Program 📃 🗖	Alphabetic Cate	gorized
	ScrollBars	0 - None
	TabIndex	2
	TabStop	True
	Tag	
	Text	
	ToolTipText	
	Тор	240
Click Me	Visible	True
	WhatsThisHelpID	0
	<u> </u>	1
	Text Returns/sets the t	ext contained
	Cower Lawouk	



Also, scroll up to the very top of the properties list and while we're at it, let's change the name of the box so that instead of saying "Text1," let's say it says, "YourName."

		∎ram
shi Eaumi (Eaum)	Properties - Text1	×
	Text1 TextBox	-
First Program	Alphabetic Categorized	
	(Name) YourName	
	Alignment 0 - Left Justi	fy
· · · · · · · · · · · · · · · · · · ·	Appearance 🗼 1 - 3D	
	BackColor 8H80000	005
	BorderStyle 1 - Fixed Sing	gle
	CausesValidation True	
· · · · · · · · · · · · · · · · · · ·	DataField	
Click Me	DataFormat	_
	DataMember	
	i	
	(Name)	
	Returns the name used in code	to
	Form Layout	×

Now this is the actual name of the box itself. The box (the object itself) is now called, "YourName." It's important that you don't put any spaces between "Your" and "Name." So let's see what we've got so far. Click on the start button.

• 12 0	¥ 🖬 🐰	• 6 4	K) CH	⇒ II	 Note <li< th=""><th>김 😽 🛠 </th><th></th></li<>	김 😽 🛠	
						Project	
	💐 My Firs	st Program			_ 🗆 🗵		
	Entern	Your Name:]	□ ≫	P
		\searrow					
		Click Me					



Notice how we have our label, "Enter Your Name:," and a text box. You can click in this text box and type in your name. And you can click on the "Click Me" button if you want to, and it still says "Hello World".

Now, wouldn't it be nice if that box could say, "Hello (your name)" instead of "Hello World"? Let's see if we can do that. First, let's see if we can get the message box to just say your name. Let's close our program.

The name of the text box is "YourName." So instead of having the message box print, "Hello World" on the screen, let's have it print whatever is in the "YourName" box. Double-click on our "Click Me" button and that will open up our code window. Let's change "Hello World" so it says "YourName."



Let's see if we get what we want. Click on the run button. Type in your name and hit the "Click Me" and wait a minute!

	• 1	📽 🖬 👗 🏘 🛍 🚧 👂 🖓 🕞 🖬 📲 👹 🔮 🚭 😤	5
		🖨 My First Program	Pro:
He	Project elloBut	Enter Your Name: Richard	
	Pri		
	End	Click Me FirstProject X YourName	



It say's "YourName" in it – why is that? Hmm... let's take another look at our code. Close the program. Our code says, "YourName" in quotes – oh wait a minute! That's what's happening! The message box command is taking the word "YourName" and printing it in the box, "YourName."



If we get rid of the quotes, now what will happen is MsgBox will now print the **value** that is in the YourName box. Let's see how that works. Click on the start button again. Type in your name. Hit the "Click Me" button. Ah - there we go!

	Proje	🖿 My First Program 📃 🗆 🗶	
Н	elloBu	Enter Your Name: Richard	
	Pr En		
		Richard	

Now the message box command is 'message boxing' your name. What is your name equal to right now? It's equal to your name. And that's how we can use the message box command and a command button to display information on the screen – whatever information a user has typed in. Let's close our program.

"YourName" (without the quotes) is now a **value**. If it's inside some quotes, it's just going to print the word "YourName" in the box. Now, it's going to send the value for YourName to the message box function instead.



Now we're still only half-way there. Because if you recall, we want this button to say, "Hello (and then your name)." Well here's how you do it. Right in front of YourName, type in open quotes, Hello, close quotes and space. Let's see if this works. Let's run our program.



Whoa – what's going on here? Compile error, expected end of statement. **Compile error** means that there is a syntax error in our program somewhere. A **syntax error** means that essentially, we didn't write a line of code correctly. **Expected end of statement** means that Visual Basic expected the statement to end after the word, "Hello." Something's not quite right. Let's click on OK to close this error message. Notice how MsgBox "Hello" is in red. If Visual Basic finds any syntax errors in our code, it's going to highlight them in red.

What's happening here is that we've got some text in a string of characters. A **string** is just a fancy word for some text inside of quotes.





We need some way to put them together. Well, here's a special command for you: Use the ampersand (&) to put together "Hello" & YourName and then send them both to MsgBox. Let's see how it works. Let's click on the run button. Type in your name and click on the "Click Me" button.

🖨 My First Program	
Enter Your Name:	Richard
	FirstProject X
Click Ke	HelloRichard
	ОК

Now it says, "Hello(and then your name)." It took the text "Hello, " and concatenated (that's just a fancy word for putting two text strings together) your name – the value from our box. But notice there's no space between them. How do we get a space in there? Let's close our program and stick a little space between the letter "o" and the closing quote.





Remember, whatever you put inside those quotes, gets put on the screen. If you want to put a comma in there, you can put a comma in there too.

۴.	- 1	🖻 🖬	X Ba	🛍 🏘	n ca	•		8	r -	8,
× al	_								- 1	Pro:
		🗾 Proje	ct1 - For	m1 (Cod	e)					
abl	Ľ	HelloB	utton		Click	[믴	١.,
		Pr	ivate	Sub He	lloButt	on_0	lick	:()		Pro
•			MsgB	Box "He	110 " 6	; Υοι	ırNan	e		For
না		En	d Sub							Alp (Na Ap

Let's run it. Type in Joe this time. Click on "Click Me."

· 6. • 1	📽 🖬 👃 🍋 🗠 🖂 🖌 🖬 🖷 🛃 🗠 🗠 🖌 🖬 🖷 🛃	8
	🖻 My First Program	Pro
Project	Enter Your Name: Joe	
Priv		
End	Click Me FirstProject X Hello, Joe	

Now it says "Hello, Joe" with a space!

So in this lesson we learned a lot. We learned how to put a label and a text box on the screen. We learned how to concatenate two text strings together. So our program's starting to get a little more functional!



Lesson 6. Simple Calculator

Let's take what we've learned in the last couple of lessons, and put them together to build a program that actually does something useful. Let's build our own simple calculator.

If you already have your Project1 open, let's go ahead and close it. Then click file and then click on New Project.

Project1 - Microsoft	Visual Basic [de	esign]			
Eile Edit View Project	Format Debug	Run Qu	įery Dįagram	Iools A	dd-Ins Wind
New Project	Ctrl+N Ctrl+O	44 ·	n 🛛 🕨	11 10	* 6 2
Add Project					

If you haven't saved your changes to the last project you just had open, it's going to ask you if you want to save changes to the following files: our "vbp" our project file, and our "frm", our form file. Go ahead and click yes – that will save those.

Microsoft ¥isual Basic	×
Save changes to the following files?	Yes
FirstProject.vbp FirstProgram.frm	No
	Cancel

And now we're prompted for the new project. We're going to create a standard "exe" (a standard executable program). Let's click on OK.





And here we are with a blank, new form and a blank new project. Here's what we're going to do for this project. We're going to build a simple calculator. We want, two text boxes where the user can enter in two numbers – a command button that they can then clic k on – and then a third text box that will have the answer in it.

B	• 🖪 🕞 🖪 % 🕫 📭 🏘 🗠 😒 🕨 II	• X • • •
General	🖷 Project1 - Form1 (Form)	
R 🔛	🐂 Form1 📃	
A		
A abi		
The second se		
V ()		
	Π	

So let's start out by putting a text box on the form.

Image: Second	Image: Second secon	eneral	🖷 Project1 - Form1 (For	m)	_ 0
abl Text1	abl Text1		🖷. Form1		- 🗆 ×
		[ab]			
□ · · · · · · · · · · · · · · · · · · ·	c	api	I ext1		
с — — — — — — — — — — — — — — — — — — —	с С				
G	۰ ۲				
e	С С	1.000			
•		~			
		(•			

Change the name of the text box to "FirstNumber."

•		•			•		*	•	•		+ +	+ +	•		•					•			+ +	Properties - Text1	X
+ + +		• • •	• • •			* * *		• • •				T T T	* * *	•	•	T T	• •			• • •			• • •	Text1 TextBox	•
			* * *					* * *		+ + +			• • •	• • •	+ + +		* * *				* * *		* * *	Alphabetic Categorized	
															•	• • •							• • •	(Name) FirstNumber	•
		•		•	•	•	*	•						•		•	•	•				•	•	Alignment 👋 0 - Left Justify	
		•	•	•	•	+	*	•	+		•	•	•		•	•	•	•	+	•	+	+	•	Appearance 1 - 3D -	
	ł	•		+ +	•	•	*	•			1	•		•	•	•		•	•	+	-		•	BackColor 8H80000005	
		•		+ +					+	+ +	+ +		+ +	•	•	•	•							BorderStyle 1 - Fixed Single	
a logicity	-	•	• •	*	•				•	+ +	•	•	•	•	•	•	•	•	•	•		*	•	CausesValidation True	
_																								Dacameid	111



Scroll down and find text and get rid of it.

Forms	
Properties - FirstNumber	I
Alphabetic Categorized	-
RightToLeft False ScrollBars 0 - None	
TabIndex 0 TabStop True	
Tag Text	
ToolTipText K Top 120	
Visible True	•
Text	

Now we need to make two more of these guys. We need to make one more for a number to go into and then we need to make one more for the answer. So let's just try copying and pasting this thing. Let's click on the text box and hit **copy** and then **paste**.





Hold on a second! Vis ual Basic is saying, you already have a control named "FirstNumber." Do you want to create a **control array**? Hm. We haven't talked about control arrays yet, so for today, let's just say **no**.

ľ		1	0	m	m	1																						1		l c		E	x			Г	8	b	ŝ
P	-		-			-	_	_	-																					÷						I.	-	9	<u>_</u>
L					Ν					-		00	-	18		•								• •		۲	۲	•	• •		٠							. 6	9
L				- 24	hi	6				-						• •		1				1		• •				•	• •		۲	۲							
5					- 1	٥									•	• •								• •					• •				8						
	•	•3								-					•	• •								• •					• •			1							
	- 5		•	• •	1	1.5	1	82		1	10		3		•	• •				۲				• •					• •							н.			
•	1	• `		1	1	0.5						1				•	1	1	1					•	1					1	1					-	-	-	ŝ
1	•		•		0		0	1	0	10	1	1		* 0	• •	•	0	1	1				* 1	•	1	1			• •	1	1	1	1			1.13	70	DE	
1	۰.	•	• •			1	0.3		0.0					* 0	•	• •	0.3						*	• •					• •		1	3	1				Concession of the local division of the loca		1
•	•	•					0.0		0.0							• •	0.0							• •							1					15			2
•	۰.	•		• •												• •								• •					• •								1e	хC	1
•		•		• •																				• •				-								Re:			
		•													-																						Ale	sh:	1
12	23	•													0																						mil	JE H	Î
							1		1		1											1.1	1				1				100								

We should get our text box. Let's click on it and drag it underneath the first one.



Go to properties and change the name of the second text box to "SecondNumber."

1	+		*	+	*	+	*	6	• •	£)	1.1			-	- *			*	÷.,		
Č,		•	•	•	•	•										*	•	•			
		•		÷		•	•	0				8	-	-		÷		+		-	8
į			1		1	•		Ľ,				ž			*		•	*			
				-																	
	÷			*	*	•	•	-	•	•		8							•		
ł		÷	•	•	•	•	•		•	Ċ,		Č,					•	•	•		
ł									•		Ū.									3	8
						18								85				10			
												8					-				
					•					8		8									8
1		3	1	1	33							8			3						
				-	•	•				-						•		-			
ŝ			10	1									115		15	1	3				3
ŝ				8								8									
-						•															
1	-	1	1	1	20	1							1		*	*	*	*		10	



And notice the benefit of copying and pasting...

			1			-							-	-								-		Propercies - SecondNumber	-
		•		•	•					•			•	•							•	•	•	SecondNumber TextBox	
	-	•		-	*	•	•	•	•	• •	•	•	•	•				1		ŝ	•	•	*	and the second se	_
•									*	1	1			1								Ċ.		Alphabetic Categorized	
				1										2	1									- I and and I	
				•		•	•	•	+	-	•	•		•	•	•	•	1		ŝ	•	•	•	RightToLeft False	
				•		*	•	8	1	*	*	*	•	*			1	3	3	1	•	1	*		-
						2					2			1										ScrolBars 0 - None	
							•		2			•	•				•	8	8					TabIndex 1	
						-		2	1	1				•								Č,		TabStop True	- 10
				•			•	•		•	8	81	-	•	+	-					•	•	•	inductop inde	
•							1	•				*		1		1	1							Tag	
					2	2	•	•	•		•	•	8						2	3	-			Text	
	-			-		-	1	-	-		-	-	-	-	-	1	-		1	1		1		ToolTipText K	-
																								Top 600	
																								Visible True	-

The text field here is blank because we copied the other one! It will copy almost all of the properties except of course its name. Let's make one more copy.

We'll drag this guy down on the bottom. That's where our answer will go.

ieneral	🖷 Project1 - Form1 (Form)
	S. Form1
abl	
c	



And let's name this guy, "Answer."

1	1	•	•	•	•	•	•		3		•	•	•	• •		•	•		• •	1	
	;	:	:	:	:	:	1					:	:			:	:			Properties - Te	xt1
•		•	•	•	•	•	•	• •			•	•	•	• •	•	•	•	•	• •	Statistical and statistical and statistical statistica	
	:	:	:	:	:	:	:				:	:	:		:	:	:		: :	Text1 TextBox	1
																				1	
	•	•	•	•	•	•	•	•	8	•	•	•	•	• •	•	•	•	•	• •	Alphabetic Ca	enorized
3		1	1	•	•	1			8			•				1				100	.ogonzou j
	÷	•	÷	÷	÷	÷	÷				•	÷	•		•	÷	•			(Name)	Answer
		:	:	:	:	:	:				:	:	:		:	:	:		: :	Alignment	0 - Left Justi
	•	•	•	•	•	•	•	• •			•	•	•	• •	•	•	•	•	• •	ringrimorie	o corebase
	1	•	•	•	•	•	•	•		•	•	•	•	• •	•	•	•	•	• •	Appearance	1 - 3D
	•		:	:	:	•	:				:	:	:		:	:	:			BackColor	R 8H80000
•		•	•	•	•	•	•	• •		•	•	•	•	• •	•	•	•	•	• •		
				•		•	•	• •		•	•		•	• •	•		•	•	• •	BorderStyle	1 - Fixed Sind

So now we have "FirstNumber," "SecondNumber," and "Answer."

1000	
N 🔝	🖌 Form1 📃 🗖 🗙
A Last	· · · · · · · · · · · · · · · · · · ·
A Jabi	
-vv	· · · · · · · · · · · · · · · · · · ·
10.000	
	I
	· · · · · · · · · · · · · · · · · · ·
an 🗎	
<u> </u>	
#3 D	

Now we need a command button in the middle here that we can click on. And add these two values up top and put the answer down at the bottom. So lets click on the command button. Draw a button in the middle.

k 🔝	🖼 Form1
A 177	· · · · · · · · · · · · · · · · · · ·
A abi	*****
	· · · · · · · · · · · · · · · · · · ·
•	
	· · · · · · · · · · · · · · · · · · ·
	Command1 📑 Command1
. 🔺	· · · · · · · · · · · · · · · · · · ·
비	·····
	•••••



Let's change its properties. Let's give it a good name. Let's call it, "CalculateButton." That will be the name of the button. Let's change its caption to "Calculate."

alculateButton	oı CommandButton	ategorized	CalculateButton	1 - 3D	8H800000F	False	Calculate	True	False	(None)
Properties - Ca	CalculateButto	Alphabetic Cat	(Name)	Appearance	BackColor	Cancel	Caption	CausesValidatio	Default	DisabledPicture
		ate								
	. 📺	l.		<u></u>	::::::					
		Calcu								
Γ	No.22	Calcu								
:		(= Calcu	:	· · · · · · · · · · · · · · · · · · ·	:	• • • • • • • • • • • • • • • • • • • •	: : : : : : : : : : : : : : : : : : : :		• • • • • • • • • • • • • • • • • • • •	
		Calcu		· · · · · · · · · · · · · · · · · · ·					• • • • • • • • • • • • • • • • • • • •	
		 Calcu							• • • • • • • • • • • • • • • • • • • •	

And now, let's go ahead and actually run our program. Type in a value up top, like "5" and then "7" and hit "Calculate."

5	-	6
7	_	
Calculate		
	_	

Wait a minute! We forgot to put programming in our button, right? The program just doesn't *know* what we want to do! We have to actually tell it what to do. So let's put some programming in there.



Let's close our program and click on the "Calculate" button to open up the code window. Here we are inside the CalculateButton_Click(). This is a private subroutine which means that only this form can use it. Whatever text we type in between the Private Sub and the End Sub will run whenever the user clicks on CalculateButton on the form.



And now let's think, "What do we want to do with this program?" We've got two textboxes, "FirstNumber" and "SecondNumber." We want to then add their values together and put the answer in the answer text box. So here's how we're going to type it in. We're going to say, "Answer = FirstNumber + SecondNumber."

	Project1 - Form1 (Code)	
	CalculateButton Click	1
1	Private Sub CalculateButton_Click()	Prope
2	Answer = FirstNumber + SecondNum	Answ
3		Alpha
	End Sub	
)	т	Pass
		Right

Basically, you do the math backwards. The name of the control goes first, "Answer." In this case, the "Answer" text box is going to be set equal to the FirstNumber text box plus the SecondNumber text box. Let's see how this works now. Let's go ahead and run our program. Click on the start button.



Type in "10." Tab down and type in "25." Then click on your "Calculate" button. And... wait a minute. That's not quite right! We've got 1025. Hm.. What happened? Well essentially, Visual Basic is going to treat those text boxes as *text* – not numbers. So anything you type in there, is going to be treated like text. We could type in the top text box, "Richard" and "Rost" in the bottom text box, and then hit "Calculate." That's going to add those two things together. It's going to treat them like *text*.

	🗟 Form1 📃 🔍	Proje
1	10	
Calculate	25	Ē
Priv	Calculate	
End	1025	

So what we have to first do, is we have to tell Visual Basic, "Hey! Treat these as *numbers* – not as text. Give me the **value** of the numbers in these boxes and don't treat this stuff as text." So how do we do that?

Well, let's close our program to get back inside of our code. Let's click right in front of "FirstNumber" and type in the word, **Val**. Then open parenthesis. Then click at the end of FirstNumber and put in a close parenthesis. That's the **Val function**. Basically it's going to take that FirstNumber field and convert it to a number. Let's do the same thing for SecondNumber.

roject1 - Form1 (Code)		
liculateButton	Click	•
Private Sub Calculat	teButton_Click()	
Answer = Val(Fin	rstNumber) + Val(SecondI	Number)
End Sub	I	

So "Answer" is going to be set equal to the **value** of the first number **plus** the value of the second number. Let's go ahead now and run our program.



	🗟, Form1		
Form1 (10	11	×
rtton	15	-	
te Sub			
nswer	Calculate	R	
ub	25		

Let's type in "10" and then "15" and hit "Calculate."

Oh – It's a beautiful thing! Again, give yourself a little round of applause.

You can see how the **Val** function took the text in the text box and converted it to a number. Let's go ahead and close our program and close our code window. And now we've made our first simple calculator!



Lesson 7. More Calculator

In the last lesson, we made a simple calculator, but all our simple calculator can do is add two numbers together. In this lesson, we're going to make it capable of subtraction, multiplication, and division. Now before we go too far, we forgot to save our program in our last lesson, didn't we? And that can be a problem – especially if the power goes out, or someone comes along and spills coffee on your keyboard! Let's go ahead and save our program by clicking on the **save** button.

🐂 Projec	t1 - Mi	crosoft	¥isual I	Basic [d	esign]		222266			
<u>File E</u> dit	⊻iew	Project	Format	Debug	<u>R</u> un	Query	Djagram	<u>T</u> ools	<u>A</u> dd-Ins	<u>W</u> ind
5 - 1	j - 1	3 😂		, 🖻 (2 /4	K)	CH 🕨		- 💐 🛙	r -
×			joch1	Earmal (Easter					
General			jecci -		(runii)					

Notice we're still in our VB Files folder.

Save File As	? ×
Savein: 🗁 VB Files 💽 🗲 🛍 (* 💷 -
3. FirstProgram	
File name: SimpleCalculator	Save
Save as type: Form Files (*.frm)	Cancel
	11-1-1



Let's save the form. Let call this "SimpleCalculator" and hit save. We can also save the project as "SimpleCalculator as a "vbp" file. It's okay to have the form and the project named the same thing. Let's change the "Calculate" button so the caption says "add." We'll make some more buttons that subtract, multiply, and divide. Click on the "Calculate" button and move to its properties.

🐂 Form1	🖃 👘 Forms 📃
	Form1 (SimpleCalculator
	Properties - CalculateButton
	CalculateButton CommandButton
Calculate	Alphabetic Categorized
	(Name) CalculateButton
	Appearance 1-3D 🕅
	BackColor 🛛 &H8000000F&
	Cancel False
	Caption Calculate
	CausesValidation True
	Default False
	DisabledPicture (None)
	DownPicture (None)

We can leave the name alone. **CalculateButton** is fine – we don't have to worry about changing it to "SumButton." But let's change "Caption" to "Add."

🖷, Form1	🚊 📲 Forms 🔺
	Properties - CalculateButton 🛛 🗙
	CalculateButton CommandButton
Add •	Alphabetic Categorized
..	(Name) CalculateButton
	Appearance 1 - 3D
	BackColor 🛛 &H8000000F&
	Cancel False
	Caption Add
	CausesValidation True
	Default False
	DisabledPicture (None)
	DownPicture (None)



Now the button says "Add." In fact, let's make it a little smaller. Now let's add another button to subtract. We can just copy and paste the "Add" button. Slide it down next to the "Add" button.

F <u>o</u> rmat <u>D</u> ebug <u>R</u> un Q <u>u</u> ery Diagram <u>T</u> ools <u>A</u> dd-Ins <u>W</u> ir	ndow <u>H</u> elp
■ 2 10 12 14 10 11 = 20 12 10 1	2 19 🛠 🔁 🔊
🖷, Project1 - Form1 (Form)	Project - Project1
🛋 Form1	E Sorms
Add	Form1 (Simpl
	Properties - Command1
	Command1 CommandButt
Add	Alphabetic Categorized
	(Name) Commanc
	Appearance 1 - 3D
	BackColor 📃 &H800
	Cancel False
	Caption Add

Let's change this one's name to "SubtractButton."

Sector Form1		Forms
	Properties	- Command1 🛛 🛛 🗙
	Command	l CommandButton 📃
Add •	Add Alphabetic	Categorized
	(Name)	SubtractButton
	Appearance	e 1 - 3D
::::	BackColor	8H800000F&
	Cancel	False
	Caption	Add 5
	CausesValid	ation True
	Default	False
	DisabledPict	ure (None)
	DownPicture	e (None) 🚽



S. Form1		EB Forms	rm1 (SimpleCalculati	• or, ▼
		Properties - Sub	tractButton	X
		SubtractButton	CommandButton	-
Add S	Subtract	Alphabetic Cate	egorized	
		(Name)	SubtractButton	
		Appearance	1 - 3D	
11111		BackColor	8H800000F&	
		Cancel	False	
		Caption	Subtract	
		CausesValidat්ගිා	True	
		Default	False	
		DisabledPicture	(None)	
		DownPicture	(None)	-1
		1	l	

And we'll put the caption "Subtract" on it.

Now let's see its code. Double click on the subtract button.

	roject1 - Form1 (Code)
S	btractButton 🔽 Click
	Private Sub CalculateButton_Click()
	Answer = Val(FirstNumber) + Val(SecondNumber)
	End Sub
	Private Sub SubtractButton_Click()
	End Sub
	I
=	



Here we are inside of our code window again. Press the enter key a couple of times. This time its going to be very simply "Answer equals the value of the first number **minus** the value of the second number." Could we have copied and pasted that and just changed the operation? Certainly.

5	Project1 - Form1 (Co	ode)		[
S	ubtractButton	•	Click		ŀ
	Private Sub (CalculateButt	on_Cli	ck()	
	Answer = Val(FirstNumber) + Val(SecondNumber)				
	End Sub				
	Private Sub SubtractButton_Click()				
	Answer = Val(FirstNumber) - Val(SecondNumber)				
	End Sub		I		
=	≣◀				-

Let's go ahead and try it. Let's run our program.

• 🖬 📂	. X B C A > ~ .	▶ ■ ■ ◎ 曾 4 17 12
	🚔 Form1	
1 - Form1 ((5	IX
utton	7	-
ate Sub		
Answer :	Add Subtract	
Sub	12	-
ate Sub		
Answer :		
Sulo		



Type in five and seven and click add. We get twelve. Click subtract. We get negative two.

• 🛅 💣 🗖	1 X B B A 10 00	▶ II ■ 💐 🖆 🕾 😤 🖡
	i, Form1	
1 - Form1 ((5	IX
Button	7	•
vate Sub		
Answer :	Add	
Sub	-2	-
vate Sub		
Answer :		
Sub		

Perfect. Our add and subtract buttons are working perfectly! Let's add two more buttons real quick. Let's add multiply and divide. Close the program and we're going to close the code window for now. Let's copy and paste two more buttons.

• *	· 🛅 😅 🖬 🐰	10日間は いう	C≊ ▶ II		9
× eral	🖷, Project1 - Form	1 (Form)			×
	💐 Form1		L		
abl					
۰					
	Add				
1		·····		· · · · · · · · · · · · · · · · · · ·	



Let's change the properties of these buttons now. The third button's name is going to be "MultiplyButton"

			Form	is Form1 (SimpleCalculator.fi	rm)
	::::::::	:::::::::::	Properties - M	ultiplyButton	[
Subtract	Multiply	Subtract	Alphabetic Ca	CommandButton	•
			(Name) Appearance BackColor Cancel Caption CausesValidatio Default DisabledPicture DownPicture	MultiplyButton 1 - 3D B4H8000000F& False Multiply True False (None) (None)	
			Caption		

And its caption will be multiply. We'll click on the next button. This one will be "DivideButton," with a caption of "Divide."

			Forms	rm1 (SimpleCalculator.fr	m)
_ : : : : :			Properties - Divi	deButton	2
Subtract	Multiply	Divide	DivideButton Co Alphabetic Cate	ommandButton gorized	•
			(Name) Appearance BackColor Cancel CausesValidation Default DisabledPicture DownPicture	DivideButton 1 - 3D 8H8000000F& False Divide True False (None) (None)	
			Caption		



Now let's double click on these buttons to adjust their code.

۲I		<u> </u>
	Private Sub CalculateButton_Click()	
	Answer = Val(FirstNumber) + Val(SecondNumber)	
	End Sub	
	Private Sub MultiplyButton_Click()	
	End Sub	_
	Private Sub SubtractButton_Click()	
	Answer = Val(FirstNumber) - Val(SecondNumber)	
	End Sub	

We've double clicked on the multiply button. Notice that we're between the other two buttons. That's because Visual Basic arranges the subroutines alphabetically. Notice we've got CalculateButton then MultiplyButton, then SubtractButton. But the code for these is going to be real easy. All we'll do is highlight the top one in the CalculateButton subroutine, copy it with Control + C and then paste it with Control + V in the MultiplyButton subroutine.

We use these keyboard tricks all the time.

Copy Cut
Paste
Save

Become familiar with these keyboard shortcuts.



So anyhow we've pasted the code and now we'll change the plus operation to a multiplication operation, which is that asterisk (Shift + 8).

```
Private Sub MultiplyButton_Click()
Answer = Val(FirstNumber) * Val(SecondNumber)
End Sub
```

Now as you can see both your code window and your form window on the screen, all you have to do to flip between them is simply click on the form window. Now double click on "Divide" here.

		· · · · · · · · · · · ·	· · · · · · · · · · · · · ·	-
Add	Subtract	Multiply	Divide	
· · · · ·			6/	pndNumber)
· · · · ·	· · · · ·			

That will put you in a new private sub back in the code window. Now all you have to do is paste (hit Control + V). That code that we copied earlier was still in the Windows clipboard. Division is a forward slash.

Di	videButton 🔽 Click	•
	Private Sub DivideButton_Click()	
	Answer = Val(FirstNumber) / Val(SecondNumber)	
	End Sub	



The value of FirstNumber divided by the value of SecondNumber. Let's go ahead and save our program and run it. Let's type in fifty and ten. Let's add them.

					'
. • 🖬 🖄) 🖬 🕹 🖻 🖻	M 000	→ II	• 😽 🖻	名 🍯 🛠
	🗃 Form1			_ 🗆 ×]
	50				
iject1 - Fori	10	_			
leButton	[·-				•
Private	Add	Subtract	Multiply	Divide	
Answ					
End Sub	60				
Private					
1 nsw					

Subtract them.

/			-		
. • 🖬 📂	e X B C	M n a	> II -	8 🖻	김 🐮 🛠
_					
	, Form1			_ 🗆 🗵	
	50				
oject1 - For	10	_			
leButton	1.0				-
Private	Add	Subtract M	ultiply	Divide	
Answ)
End Sub	40				
Private					
4 nswer					J



Multiply.

. • T #	- * 🖬 %	ħ Ē	M N	CH	- 	•	6	** 🛠
	Form1							
	50							
oject1 - Fori	10		_					
JeButton	10							-
Private		Add	Subtract	Mult	iply	Divide		
Ansv							-).	
End Sub	50	0					-	
Private								
ມກອນຄ								

And Divide.

		· - · ·			—	- ·
. • TB M	🖓 🖬 🕺 🛍 I	B 🗛 🗠	Си 🕨	II 🔳 💐	83	5 🛠
	💐 Form1					
	50					
iject1 - Fori	10					IJŇ
JeButton						•
Private	Add	Subtract	Multiply	Ditide]	
Answ						
End Sub	5					
Private						
) ມກອນສ						



Beautiful! Let's change the ten to zero. Then click Add, Subject, Multiply, then Divide. Uh -oh!

1	>	
. • 🖬 🖆	2 🖬 🗟 🏘 🗠 😁 🕨 🔳 🔳 😻 🖆 🕾 🤔 🏸	
	Microsoft Visual Basic	
ject1 - Fori	Run-time error '11':	
leButton	Division by zero	
Private		
	\mathbf{k}	
Ansv	ř	
End Sub	Continue End Debug Help	
Private		
d ກອນອ		

There's an error! This is not a compile error. This is not a syntax error in our code. Our code is fine. What we have here is a run-time error. There's two kinds of errors you're going to get. **Syntax errors** are generally compile-time errors. In other words, when the program gets put together before it actually runs, then Visual Basic program looks at it and says, "Hey! This isn't right. I can't run your program. Something's wrong here." Those are the easy ones to find.

Then you've got your **run-time errors**. Run-time errors are the nasty ones. These are errors in the logic of your code. They don't usually show up until you start running your program. And sometimes they're buried deep so they don't even show up right away. But this is an easy one to fix. It say's "Division by zero.'

Basically, you've violated one of math's rules. You can't divide by zero. So what can we do? Well we can either End the program or we can Debug the program. Let's go ahead and click on the Debug button.

🧓 P	🖉 Project1 - Form1 (Code)					
Di	videButton Click	ŀ				
	<pre>Private Sub DivideButton_Click()</pre>					
⇔	Answer = Val(FirstNumber) / Val(SecondNumber)					
	End Sub					
	Private Sub MultiplyButton_Click()					
	Answer = Val(FirstNumber) * Val(SecondNumber)					



Essentially it brings us into the code that generated the error. If you hold your mouse point over any of these you can see the value of what these different variables are.

	🖉 Project1 - Form1 (Code)					
Di	videButton	- CI	lick	<u>.</u>		
	Private Sub DivideButt	on_Cli	ick()			
⇔	Answer = Val(First	Number	r) /	Val(SecondNumber)		
	End Sub					
	Private Sub MultiplyBu	tton_C	Click	:()		
	Answer = Val(First	Number	r) *	Val(SecondNumber)		

What we're going to do at this point, is we're going to stop our program. Our program is technically still running.

Jер Р	🖉 Project1 - Form1 (Code)					
Di	videBut	tton Click				
	Priv	vate Sub DivideButton_Click()				
⇔		Answer = Val(FirstNumber) / Val(SecondNumber)				
	End	Sub				

Click on the End button (up top), and that will stop the program. Now there's two ways we can fix this error. The first way is to simply have Visual Basic ignore any errors and continue moving. Click in the DivideButton subroutine and press enter to get a blank line (and then tab in). Type "**on error resume next**".

	ا لے	Project1 - Form1 (Code))		
	Di	ivideButton	▼ Cli	ck	
c		Private Sub Div I on error re Answer = Va	ideButton_Cli sume next l(FirstNumber	ck()) /	Val (SecondNumber)
		End Sub			



What does that do? Basically it says, if you encounter an error while running this subroutine, ignore it and move on to the next line. In this case, there is no next line. But it will basically not generate an error. Let's see how that works. Let's run the program. Type in two and zero and hit divide.

• 📜 💕	
	🛎 Form1 📃 🗆 🗙
tt1 - Form1 Autton ivate Su	
On Err Answer	Add Subtract Multiply Digge
d Sub	
ivate Su	
Answer	
d Sub	

Nothing happens! Like I said, it was easy. But there's nothing telling the user, "Hey you can't divide by zero!" So it would be nice if we could generate our own little error message. Let's stop the program.

Now there are a couple of ways to do this. Let's get rid of this "On Error Resume Next" and let's put in a couple of extra blank lines. What we'd like to do is we'd like to handle this error ourselves. And the way we're going to do that is we're going to look at the denominator (the second number). And we're going to see if it's equal to zero before we try dividing it. If it's zero, we'll pop up a message box that says, "Hey you can't divide by zero!" It it's not zero, we'll let it do the math.

In order for this to work, we have to make Visual Basic smart. We have to make Visual Basic be able to look at the value and make a decision either way as far as what to do. If this is zero, pop up the error message and get out of the subroutine. How do we do it?



Well we're going to use a new function called, "if." We're going to use an "**if-then statement**." And here's how it works.

Type in: "if the value of second number equals zero, then MsgBox 'you can't divide by zero', exit sub, end if." Like this:

📕 Project1 - Form1 (C	ode)		
DivideButton	•	Click	
Private Sub	DivideButton	Click()	
If Val(S MsgB Exit end if	econdNumber) ox "You can' Sub	= O Then t divide by	zero."
Answer =] Val(FirstNu	mber) / Val	(SecondNumber)
End Sub			

What is all this? Let's take a look at it line-by-line.

"If" (means that we're going to check a value) the value of the second number is equal to zero, then do some stuff. We're going to do all the stuff between the "if" statement and the "end if" statement. You can put any number of lines that you want in there. What's the stuff? Well, MsgBox, you know. Can't divide by zero – that's easy. Exit sub means "get out of Dodge." That means exit this sub – you're done. Don't continue on below. Let's save our program and run it to see what we get. Enter in five and zero and then click "Divide."

	🛢 Form1 📃 🔍	
oject1 - F ideButton	5	×
Privat)	0	
If	Add Subtract Multiply Divide	
En	SimpleCalculator X	
An;	You can't divide by zero.	≘r)
End Sul	ОК	
Desirente	Cub MultiplePutters a company	



Look at that. Isn't that nice? Hit Ok and let's try something with an actual value. Replace zero with two and hit "Divide."



We get 2.5. Let's go ahead and close our program.

Now you know how to make your Visual Basic program "smart" with an "if" statement.



Lesson 8. Compiling

In this lesson, we're going to see how to compile our Visual Basic program into a stand-alone executable file.

If we want to run our simple calculator program, we have to come inside the Visual Basic Editor and then run the program. But that's not very useful in our day-to-day life. We don't want to have to stop and run Visual Basic just to run this program. So it would be nice if we have some way to compile this program up into an executable file. In other words, a program file that we can run just from inside Windows. And we can do that!

First, make sure you save your project. Hit the save button. Click on file and move down to "Make SimpleCalculator.exe."

Save Project Save Project As					
Save SimpleCalculator.f Save SimpleCalculator.f	rm Ctrl+S rm <u>A</u> s				
Save Selection Save C <u>h</u> ange Script		ubtract	Multiply	Divide	ŀ.
Print Print Setup	Ctrl+P				
Ma <u>k</u> e SimpleCalculator.e Make Project <u>G</u> roup	exe	_			
1\\VB Files\SimpleC 2\My D\VB Files\Fir	alculator.vbp stProject.vbp	_			
E <u>x</u> it	Alt+Q				



Make Projec	t						? ×	
Save in: 📔) VB Files	•] ←	£	Ċ	•		
File name:	SimpleCalculator					OK		
						Canc	el	
						Help		
						Ontion	s	

This will compile your program up into an executable program. It'll ask you for a name, such as **SimpleCalculator**. Hit Ok.

You'll hear Windows do some churning and bubbling. And when it's all done, you'll have an executable program. But where is it? Let's go look for it. Let's minimize our Visual Basic Editor for a moment and go to Start and then to My Documents. Click it.

	707 Run
All Programs 🕨	😚 Windows Security
	🖉 Log Off 🔟 Disconnect
🍂 Styrt 🐁 Project1 - Microsoft Visu	J


Double click the VB Files folder.



Here's our SimpleCalculator (that's our program), SimpleCalculator Visual Basic Form File, and SimpleCalculator Visual Basic Project file. How do you tell the difference between these guys?





Click on "Tools" and then "Folder Options..." Go to the "View" tab. Scroll down and then uncheck, "Hide extensions for known file types."

 Display file size information in folder tips Display simple folder view in Explorer's Folders list Display the contents of system folders Display the full path in the address bar Display the full path in the title bar Do not cache thumbnails Hidden files and folders
 Display simple folder view in Explorer's Folders list Display the contents of system folders Display the full path in the address bar Display the full path in the title bar Do not cache thumbnails Hidden files and folders
 Display the contents of system folders Display the full path in the address bar Display the full path in the title bar Do not cache thumbnails Hidden files and folders
 Display the full path in the address bar Display the full path in the title bar Do not cache thumbnails Hidden files and folders
 Display the full path in the title bar Do not cache thumbnails Hidden files and folders
Do not cache thumbnails
🛅 Hidden files and folders 🛁
Do not show hidden files and folders
O Show hidden files and folders
Hide extensions for known file types
Wide protected operating system files (Becommended)
Launch folder windows in a separate process
Restore Defaults
OK Cancel Apply

Click on OK. Now you can actually see what these are. Let's find the "exe" file and double click on it.





And there we go! Our program runs right from My Computer or Explorer, or wherever you happen to be in Windows. It's now its own standalone Windows application. You don't have to load up Visual Basic to run your program.

Visual Basic Project						
📑 Rena 🕻	, Form1	N		_ 🗆 🗡		
🔯 Move		_ 13				
Copy					ipace	
🚳 Publi Web						
😥 E-ma		11	- 1	- 1		
🗙 Delei	Add	Subtract M	ultiply Dir	/ide		
Other Pl						
🕒 My D						
🛅 Shar						
🧧 My Con	npacor	~~~			•	
🍕 My Net	work Places	• •				

You can right click on the executable file and drag it to the desktop and create a shortcut to it.





So now your simple calculator is on your desktop. And we can click on it once to rename it. Anytime we want to run this, we can double click on it here on the desktop.



Now one of the things that students always ask me in my classroom is, "Can you share your program?" "Can I take that 'exe' program that we just created and give it to a buddy on a floppy disk and say, 'Here, you can run this.'?"

The answer is – it depends. If he has any other Visual Basic programs on his machine, chances are he has all the necessary files. You may have to give him some additional files, some "dll's" and some other control files that he might not have on his computer – again, depending on what's on his machine right now.



🔝 ActiveX Control Test Container Microsoft Visual Studio 6.0 Tools ы 0 🍓 🛛 API Text Viewer 🐁 Microsoft Visual Basic 6.0 🔤 DataObject Viewer 👮 DDE Spy Places 🔩 Depends 😇 DocFile Viewer 🥵 Error Lookup xes OLE Client Test (**OLE Server Test** brt. OLE Tools 0LE View Package & Deployment Wizard - 60 ら Š. **Process Viewer ROT Viewer**

If you look in Microsoft Visual studio 6.0 Tools, you will find a Package & Deployment Wizard

This guy basically takes your program and packages it up with all the necessary files that you need to distribute your program. Feel free to run through this Package & Deployment Wizard if you'd like to. You only need to bother with it if you want to share your Visual Basic program with other people. You can try giving them your "exe" file first and see if it works, but if not, you might need to run through this.

Keep in mind that since you have Visual Basic on your computer, you don't need to do that, you can run just the "exe" because you have all the necessary files on your machine.



Lesson 9. Review

Let's review what we've learned in today's class.

- We learned about Visual Basic and about Windows programs in general.
- We created a couple of different programs.
- We created a simple program to display a message on the screen.
- We built a simple calculator.
- We learned basic commands such as "msgbox".
- We learned about different **controls**, **command buttons**, **text boxes**, and **labels**.
- We learned about decision-making, the "**if-then statement**" for example.
- Some real basic **error handling**.
- The "on error resume next statement".
- We learned about **compiling** your program into a standalone executable so you can run it anywhere you happen to be in Windows.
- We talked briefly about **distributing** your program to other users.

Tell us what you think. Log on to www.599cd.com/Survey and take a short survey about this course.

RICK'S NOTE: I really do enjoy getting surveys from you! Make sure you visit the web page above and fill out the survey for this class. Let me know if I've moved too fast, and whether or not I covered material that was helpful to you!

Take your **skills check** quiz at **www.599cd.com/Test**. If you pass, you can print out a Certificate of Completion.

What's next? Visit www.599cd.com for our complete list of courses.

Need Help? Visit www.599cd.com/TechHelp for technical assistance.

Make sure you're on our Mailing List. Go to www.599cd.com/MailingList for details.

Contact Us. If you have any questions, go to **www.599cd.com/Contact** for information on how you can contact us by phone, email, or live online chat.

Don't forget to visit our <u>User Message Forums</u> online at: **www.599cd.com/Forums**. You can chat with our instructors, other users, and even Richard too. You can ask us all of your questions, get answers, and tell us what you thought of our class.



This course, handbook, videos, and other materials are **copyright** 2002 - 2006 by Amicron Computing. All rights reserved. No portion of this course, handbook, videos, or other course materials may be reproduced, copied, edited, or otherwise distributed without the express written permission of Amicron Computing. Amicron Computing shall not be held liable for any errors or omissions in this document.

This document may **not** be used as part of a training course without express, written permission from Amicron Computing and the purchase of an **Instructional License**. For details, contact:

Amicron Computing PO Box 1308 Amherst NY 14226 USA www.599cd.com