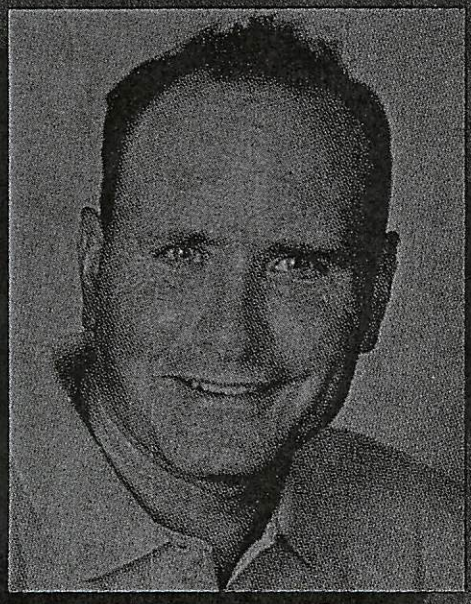
PROFESSIONAL

7/02
HALF PRICE BOOKS
$ 7.98

Visual Basic® Windows® CE Programming



Larry Roof

Programming Visual Basic for Windows CE to create professional applications for handheld computers

WROX

FANDANGO EXHIBIT 1069

Professional Visual Basic Windows® CE Programming

Larry Roof

Wrox Press Ltd.

Trademark Acknowledgements

Wrox has endeavored to provide trademark information about all the companies and products mentioned in this book by the appropriate use of capitals. However, Wrox cannot guarantee the accuracy of this information.

Credits

Author
Larry Roof

Development Editor
Dominic Shakeshaft

Editors
Craig Berry
Darren Gill

Index
Diane Brenner

Technical Reviewers
Arnold Cota
Mike Crowe
Daniel M. Laby M.D
Thomas Lewis
Scott Mason
William Moore
Boyd Nolan
Tihan Seale
Richard Veith

Cover/Design/Layout
Andrew Guillaume

Copy Edit
Alex Zoro
Barney Zoro

CHAPTER 2

Creating a Handheld Application

In this chapter I will introduce you to a process that you can use to create applications with Visual Basic for Windows CE. While this process is similar to ones you might have used to create applications with Visual Basic for Windows 95 or Windows NT, it does includes some steps unique to the handheld computer.

Through the course of this chapter I will walk you through this complete development process, from the creation of a new project, to the deployment of the finished application. Along the way you will learn how to use the key components of the VBCE Toolkit. If you are an inexperienced VB developer, I would recommend that you work through this chapter from beginning to end, using the step-by-step instructions as a guide to building this chapter's example program. On the other hand, an experienced VB developer may simply wish to skim through the chapter, focusing on the parts of the development process that are unique to CE applications. Either way, upon finishing this chapter you should be comfortable with:

- The process used to create applications using the VBCE Toolkit
- How to use the key components of the Toolkit

At the heart of this chapter is a nine-step process I use to create applications with the VBCE Toolkit. This will take you through everything from creating a new project, via testing in both the CE emulator and a handheld computer, to deploying your application using the Application Install Wizard. So let's get right at it. The completion of your first CE application is only a chapter away.

The CE Application Creation Process

As I have said, the process of creating applications for Windows CE has many similarities to the process used to create applications for Windows 95 or NT. Tasks such as creating the project, building the user interface and coding event procedures are identical between Visual Basic and the VBCE Toolkit. At the same time, however, some steps (like testing in the emulator and on the handheld computer) are unique to the CE environment. The focus of this chapter is, therefore, to detail each step in the development process and to provide you with step-by-step instructions so that you may become familiar with this process.

The nine-step process we will be using to create a Visual Basic for Windows CE application is as follows:

- Create a new project
- Set the project properties
- Construct the user interface
- Set the properties of the forms and controls

- Add code to specify what the application is to perform
- Test and debug the program in the CE emulator
- Test and debug the program on a handheld computer
- Create an installation program
- Test the installation program

Rather than providing a theoretical overview of each of these steps, we will instead look at each step as they are applied to an application that we will be constructing. This will, in fact, be the focus of the remainder of this chapter.

Time Manager – A Step-By-Step Example

In the demonstration of the CE application creation process we'll be building an example program, called Larry Roof's Time Manager. This application is a simple time billing program that allows you to track, categorize and bill your time; incorporating a timer that can be started, paused and stopped. This example will be enhanced as we proceed through the first part of this book to incorporate access to a CE database and a help file.

Step-By-Step Projects – The Recipe

As was detailed in the Introduction to this book, step-by-step projects will be presented in the following format:

- **Viewing the Completed Project:** I will start a project by showing you a completed example. This way you get to see what you are building before you begin.
- **List of Requirements:** Next I will provide you with a detailed list of the requirements for the application.
- **Designing the Application:** For each project we will then walk through the design, especially noting any special considerations for the CE environment.
- **Development of the Application:** The development of each application will be examined in detail so that you aren't left on your own to figure out how a program works.
- **Testing:** With each project you will learn testing techniques that you can apply to your own projects.
- **Deployment:** No CE application would be complete without an installation routine. Each of the applications presented in this book includes an installation program.

Three of these items; development of the application, testing and deployment are the nine-step application creation process itself.

Time Manager – The Completed Project

Before we start constructing the Time Manager application, let's take a look at the completed project. I use this approach because I believe that seeing the finished product aids in understanding discussions on its development.

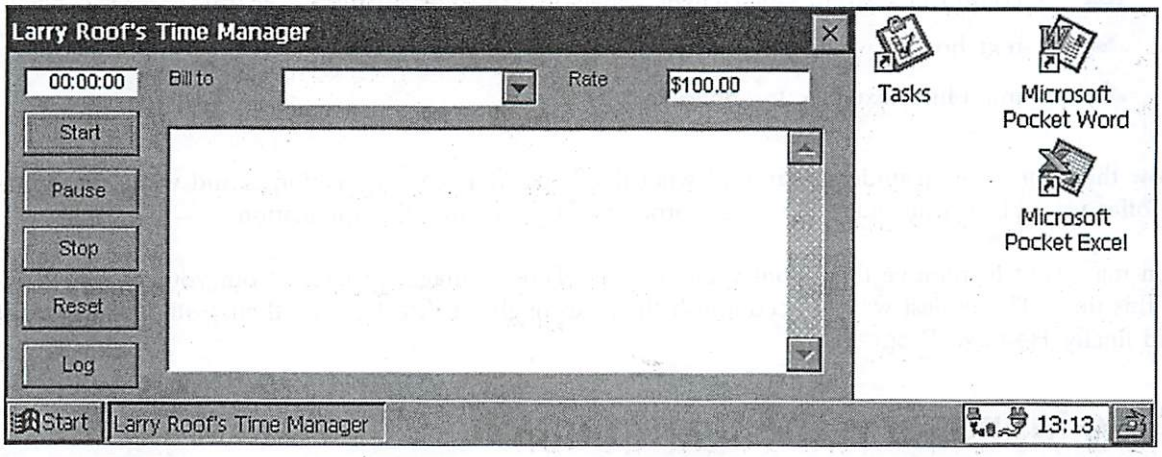
> **You must have already downloaded and run the setup for the software included with this book to perform this step.**

To install the initial version of my Time Manager on your handheld computer follow these steps:

- Establish a connection between your desktop and handheld computers
- Using the instructions laid out in Appendix A find the version of Larry Roof's Time Manger for Chapter 2
- Follow the installation directions

Using Time Manager

Once you have completed the installation of this example, run it on your handheld computer. The Time Manager interface should be displayed as shown in the screenshot below:



Test out the features of this application. While this version of my Time Manager provides minimal functionality it does support basic timing with the Timer control. To see this use the command buttons to start, pause, stop and reset the timer.

Time Manager – The Requirements

This initial version of my Time Manager application is primarily intended to demonstrate the creation of a simple application using the VBCE Toolkit. As such, the functionality it needs to provide is minimal. The only requirements for this version are to provide:

- A working timer that can be started, paused, stopped and reset
- Ability to select a client to bill for this time
- Ability to specify an hourly rate at which to bill
- Simple note taking capability

Time Manager – The Design

Version 1 of my Time Manager application has a fairly simple design. There is minimal functionality provided with this version. All that it will do is allow the user to time a session. It will not calculate or store billing information. Subsequent versions of the Time Manager will incorporate a CE database and a help file. The design of this version will thus be limited to:

- A single form
- Five command buttons for starting, pausing, stopping, resetting and logging time
- A combo box from which client names can be selected (we'll add the names in Chapter 7)
- A text box in which an hourly rate can be entered
- A multi-line text box for notes

Now that you have an understanding of what the Time Manager application is and what capabilities it has to offer we will turn our attention to the process of constructing the application.

You may want to remove this initial version of the Time Manager program from your handheld computer at this time. The easiest way to accomplish this is through the Start menu, then Settings, Control Panel and finally Remove Programs.

Time Manager – The Construction

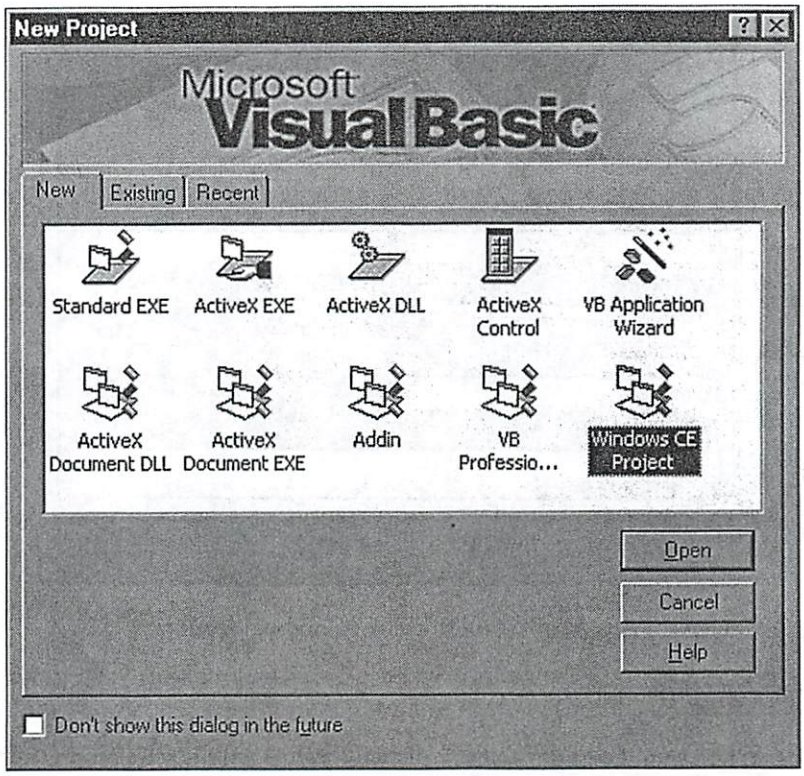
In the pages that follow I will lead you through all nine steps of the creation process using the Time Manager application as a demonstration. I would suggest that you work along with the instructions to create your own copy of the Time Manager.

> *The file Chapter 2 version of* `TimeManager.vbp` *in the source code contains a completed version of this project. Refer to Appendix A for instructions on downloading and installing the examples included in this book.*

Step 1 - Starting a New Project

This first step creates a new Windows CE project.

Start Visual Basic. The Visual Basic Integrated Design Environment, or IDE, will be displayed along with the New Project dialog as shown below:

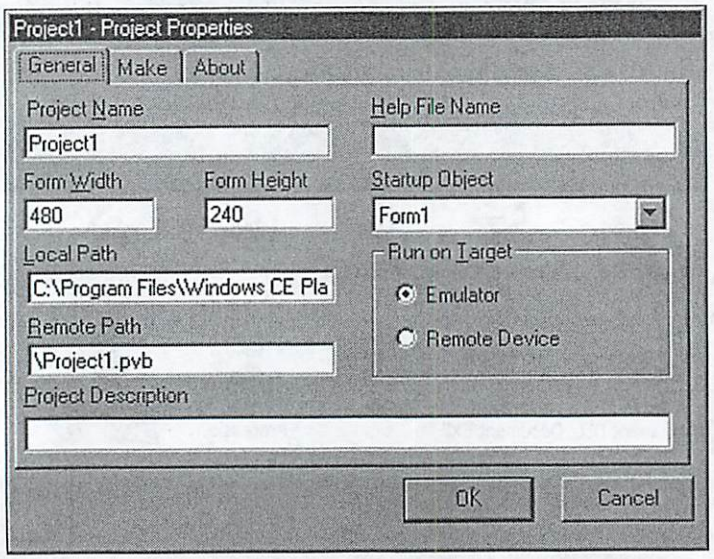


Select the Windows CE Project from the New Tab of this dialog. The template for this new project type was added to your system during the installation of the VBCE Toolkit.

Click the Open button to create the new Windows CE project. The default CE project will be loaded into the Visual Basic IDE.

Step 2 - Setting the Project Properties

The first thing that you want to do with a new project is to set its **Project Properties**. Much of this step will be new even to the experienced VB developer as it relates solely to CE applications. The Windows CE project template lends a hand here by automatically displaying the Project Properties dialog upon creation of a new Windows CE project. You can return to this dialog later by selecting the Project Properties menu item from the Project menu.



For our Time Manager example program we will use the following settings:

Set the Project Name to TimeManager.

Leave the default Form Width and Height values. Normally you would set these properties to match your needs. Typically, CE applications are run full screen, which means your form size should match the size of the display of your target handheld device. For newer handhelds this would typically be 640 x 240 but for older units this would be 480 x 240. We will see in Chapter 3 how you can adjust the form size at runtime to match the display of the system on which the application is running.

Make sure that you set the form size that you want in this opening dialog. The form size settings on the Project Properties dialog do not affect the size of the form after the project is created. If you want to change the size of the form later you must do so by setting the Width property of the form.

Modify the Local Path value by changing the filename at the end of the path from **`Project1.pvb`** to **`TimeManager.pvb`**. This is a work area where a copy of your project will be stored before being run from within the VB IDE.

Set the Remote Path value to **`\TimeManager.pvb`**. This is where the application will be sent to when downloaded to a connected handheld device.

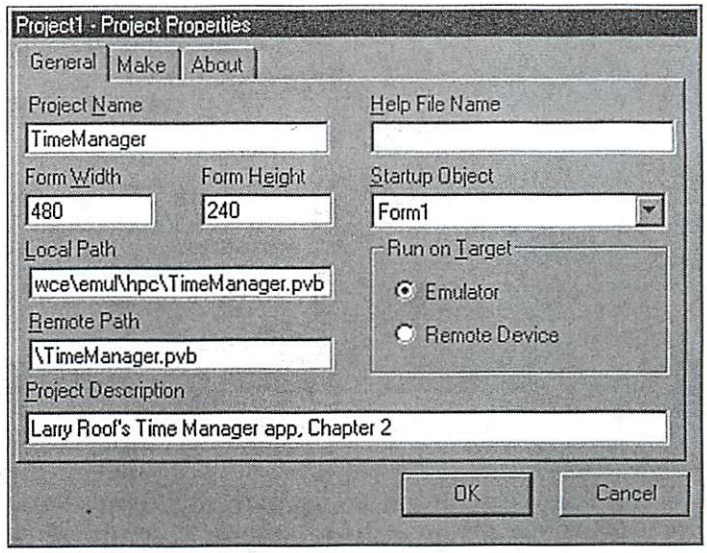
Set the Project Description to Larry Roof's Time Manager app, Chapter 2. This property has little use other than for the developer.

Leave the Help File Name empty. This property is not used by the VBCE Toolkit.

Leave the Startup Object as Form1. This setting is used to specify which form to load initially in multi-form applications.

Leave the Run on Target option box as Emulator. This configuration controls where your application will be run when it is executed from within the VB IDE. The options are either in the CE emulator or on a connected handheld device.

Your completed Project Properties dialog should now match this screenshot:

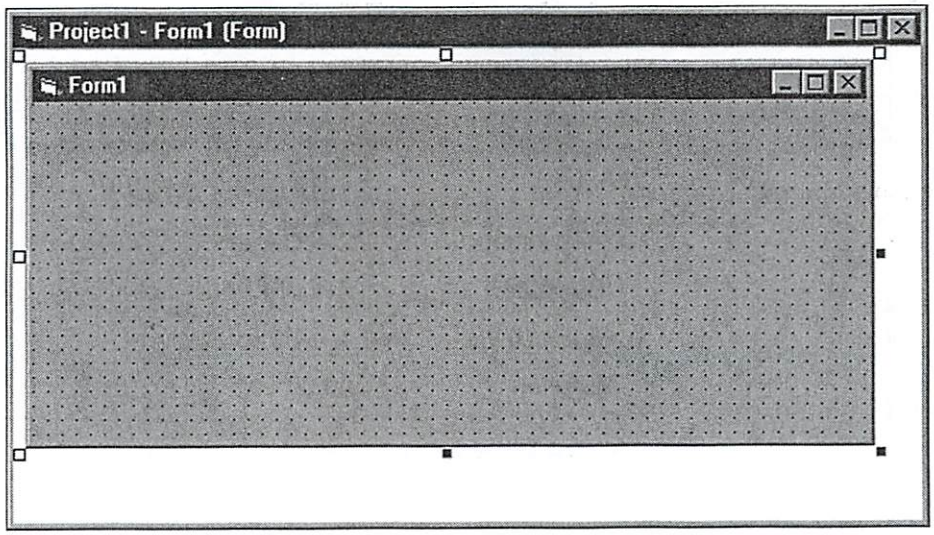


Click the OK button to close the Project Properties dialog.

Finish this step by saving your new project. This process is identical to how it would be done in Visual Basic. Under the File menu select Save Project. Save the form to the file **`TimeManager.frm`** and the project to the file **`TimeManager.vbp`**.

Step 3 - Constructing the User Interface

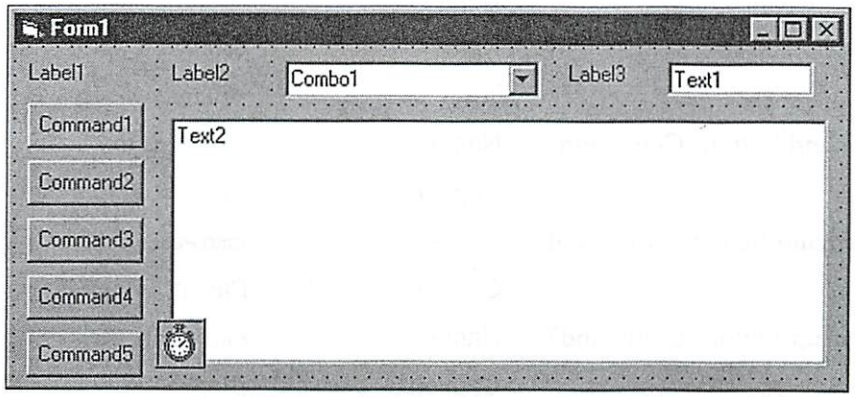
You now have a new project that is comprised of a single form – pretty much what you are used to. The primary difference with this project is that the form has been set to the size you specified in the Project Properties dialog:



In this step you will be constructing the user interface for the Time Manager application. The process you will use is just like what you would use for any other VB application.

If you closed it open the **`Form1`** form.

Add controls to the interface so that it looks like this:



The Visual Basic IDE does not prohibit you from adding controls to the toolbox that are not intended for use with Windows CE. However, it will stop you from placing a copy of the unsupported control onto a form.

Finish this step by saving your project.

Step 4 - Setting the Form and Control Properties

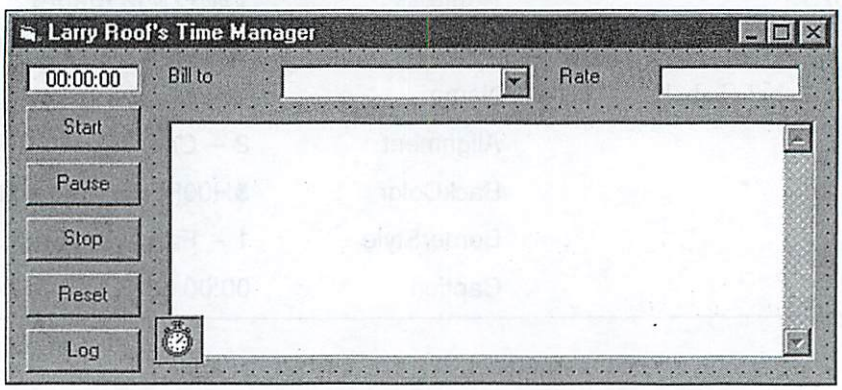
Now that you have added the controls to the interface of the Time Manager, let's turn our attention to configuration of the form and the control properties. Again, the process used here is identical to that used in Visual Basic.

Set the properties for the form and controls to match the following table:

Object	Property	Value
Form (Form1)	Name	**`frmTimeManager`**
	Caption	Larry Roof's Time Manager
Time display label (Label1)	Name	**`lblTimeDisplay`**
	Alignment	2 – Center
	BackColor	&H00FFFFFF& (white)
	BorderStyle	1 – Fixed Single
	Caption	00:00:00

Object	Property	Value
Bill to label (Label2)	Caption	Bill to:
Bill to combobox (Combo1)	Name	`cboBillTo`
	Text	<Blank>
Rate label (Label3)	Caption	Rate:
Rate textbox (Text1)	Name	`txtRate`
	Text	<Blank>
Start command button (Command 1)	Name	`cmdStart`
	Caption	Start
Pause command button (Command2)	Name	`cmdPause`
	Caption	Pause
Stop command button (Command3)	Name	`cmdStop`
	Caption	Stop
Reset command button (Command4)	Name	`cmdReset`
	Caption	Reset
Log command button (Command5)	Name	`cmdLog`
	Caption	Log
Note textbox (Text2)	Name	`txtNotes`
	MultiLine	True
	ScrollBars	2 – Vertical
	Text	<Blank>
Timer control (Timer1)	Name	`tmrClock`
	Enabled	False
	Interval	`500` (a half-second)

At this point your form should have an appearance similar to this one:



Finish this step by saving your project.

> **Normally you might test your interface out on a handheld device at this point, before proceeding any further with the development of your application. What you will find is what looks good to you on your large desktop monitor might not work as well on the targeted system. If possible involve an end user of your application in this evaluation process as well. As this is the first time through the process we will wait until Step 7 - Testing on a Handheld Computer to try out our interface on a handheld.**

Step 5 - Adding Your Coding

The next step in our development process is to add the code that will define how our application performs. For experienced Visual Basic programmers, this step is just the same as the process used to create desktop applications.

Add the following code to the General Declarations section of the form module:

```
Option Explicit

Const DEFAULTRATE = 100   ' The standard rate to charge clients
Dim mdtmElapsedTime       ' Used to track the time that has elapsed
Dim mdtmStartTime         ' Holds the time that the timer was started
```

> **If you are an experienced Visual Basic programmer, you will have to get used to working with the VBCE Toolkit's single data type – variant.**

Add the following code to the **Form_Load** event procedure that displays the default rate to bill clients:

```
Private Sub Form_Load()

' Initialize the controls on the form
  txtRate.Text = FormatCurrency(DEFAULTRATE)

End Sub
```

This procedure demonstrates the use of the VBCE's **FormatCurrency** *function. Included in the Toolkit language is a set of* **Format** *functions, which take the place of Visual Basic's more flexible* **Format** *function. You will see another of the* **Format** *functions in a moment.*

Add the following code to the **cmdStart_Click** event procedure, which handles the starting of the timer:

```
Private Sub cmdStart_Click()

' Adjust the time as needed to take care of the differences between
' "paused" and "stopped" modes.
  If (cmdStop.Enabled = False) Then
```

```
    mdtmElapsedTime = 0
  End If
  mdtmStartTime = Now

' Set the availability of buttons.
  cmdStart.Enabled = False
  cmdPause.Enabled = True
  cmdStop.Enabled = True
  cmdReset.Enabled = False
  cmdLog.Enabled = False

' Turn on the clock.
  tmrClock.Enabled = True

End Sub
```

The **If** statement at the beginning of this procedure is used to handle the differences in calculating time between starting from scratch and restarting the timer after it has been paused. We tell what mode we are currently in by checking the **Enabled** property of the **cmdStop** control. If it has a value of **False** we are stopped, a value of **True** means we are paused.

Add the following code to the **cmdStop_Click** event procedure, which handles the stopping of the timer:

```
Private Sub cmdStop_Click()

' Set the availability of buttons.
  cmdStart.Enabled = False
  cmdPause.Enabled = False
  cmdStop.Enabled = False
  cmdReset.Enabled = True
  cmdLog.Enabled = True

' Turn off the clock.
  tmrClock.Enabled = False

End Sub
```

Add the following code to the **cmdPause_Click** event procedure, which handles the pausing of the timer:

```
Private Sub cmdPause_Click()

' Set the availability of buttons.
  cmdStart.Enabled = True
  cmdPause.Enabled = False
  cmdStop.Enabled = True
  cmdReset.Enabled = False
  cmdLog.Enabled = False

' Save the time that has elapsed up to this point and stop the clock.
  mdtmElapsedTime = Now - mdtmStartTime + mdtmElapsedTime
  tmrClock.Enabled = False

End Sub
```

Add the following code to the **cmdReset_Click** event procedure, which handles resetting the controls on the form. This procedure would be used to clear the values from a previous billing session before starting a new session:

```
Private Sub cmdReset_Click()

' Reset all of the controls.
  cboBillTo.ListIndex = -1
  txtRate.Text = FormatCurrency(DEFAULTRATE)
  txtNotes.Text = ""
  lblTimeDisplay.Caption = "00:00:00"

' Set the availability of buttons.
  cmdStart.Enabled = True
  cmdPause.Enabled = False
  cmdStop.Enabled = False
  cmdReset.Enabled = False
  cmdLog.Enabled = False

End Sub
```

Add the following code to the **cmdLog_Click** event procedure, which after the chapter on CE databases will handle the saving of information from the current billing session. Presently, it does nothing more than set buttons:

```
Private Sub cmdLog_Click()

' Set the availability of buttons.
  cmdStart.Enabled = False
  cmdPause.Enabled = False
  cmdStop.Enabled = False
  cmdReset.Enabled = True
  cmdLog.Enabled = False

End Sub
```

Add the following code to the **tmrClock_Timer** event procedure, which handles the calculation and display of the running time:

```
Private Sub tmrClock_Timer()

  Dim dtmCurrentTime
  Dim strTempString
  Dim strTimeString

' Update the time display. The two format strings are used to provide the
' leading "00:" effect that I wanted for the display.
  dtmCurrentTime = Now
  strTimeString = FormatDateTime(dtmCurrentTime - mdtmStartTime + _
              mdtmElapsedTime, vbShortTime)
  strTempString = FormatDateTime(dtmCurrentTime - mdtmStartTime + _
              mdtmElapsedTime, vbLongTime)
' Test to see which time format is being used.
  If Right(strTempString, 1) = "M" Then
```

```
        strTimeString = strTimeString & Mid(strTempString, _
                Len(strTempString) - 5, 3)
    Else
        strTimeString = strTimeString & Mid(strTempString, _
                Len(strTempString) - 2, 3)
       lblTimeDisplay.Caption = strTimeString
    End If

End Sub
```

You will notice that I have included an **If** statement which tests to see what the result of formatting with **vbLongTime** is. This is because there are two basic ways of formatting the date. In the US, for example, the **vbLongTime** constant adds an AM or PM onto the end of the time. In Europe, however, these additional characters are not there. Thus in order for my program to work correctly all around the world I needed to check the result of the formatting with **vbLongTime**.

Finish this step by saving your project.

> **Until you are familiar with the differences between the Visual Basic and the VBCE languages I would suggest that you incrementally test your code. Since the VB IDE does not prohibit you from using statements and keywords that are not part of the Toolkit language, it won't be until you run your program that you will become aware of these problems. Read on to find out how to test your code.**

Step 6 - Testing in the CE Emulator

Finally, you are ready to test your application. Most of what you have done up to this point has been nearly identical to the steps you would have taken in creating a normal Visual Basic application. Here, you will be introduced to something new – testing in the CE emulator.

The Windows CE emulator is an application that runs on your desktop machine and emulates an operating CE machine. What it provides to you, as a developer, is an environment in which you can quickly test your CE applications. Also, because of this CE emulator, you do not actually need to have a handheld computer to build applications for a handheld computer.

The CE emulator is for all practical purposes a functional CE machine sitting on your desktop. It has a registry that can be configured, supports the CE object store for database functionality (more on this in Chapter 7) and allows you to add and use ActiveX controls (more on this in Chapter 4).

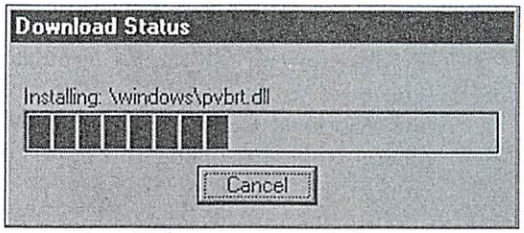
The general rule of thumb is to do the bulk of your testing using the CE emulator. It is simply quicker and easier to accomplish than testing on a connected handheld computer. Then once you have your program running as desired in the emulator you can switch your testing to a handheld computer for final adjustments.

Follow this process to test your application in the CE emulator:

First you need to configure your project to run in the CE emulator. Under the Project menu select TimeManager Properties (This should be the last menu item under the Project menu). On the Project Properties dialog confirm that the Emulator option button from the Run on Target configuration is selected.

Where your application will run during testing is controlled by the Run on Target setting that is located on the Project Properties dialog.

Before you can use the CE emulator to test for the first time you need to download the VBCE Toolkit runtime files. Under the Windows CE menu, select Download Runtime Files. A download status dialog will be displayed which will inform you of the download progress.



This step should not need to be performed again for future testing unless the emulator's supporting files are deleted or corrupted. You will be prompted to perform this step if you have not already done so.

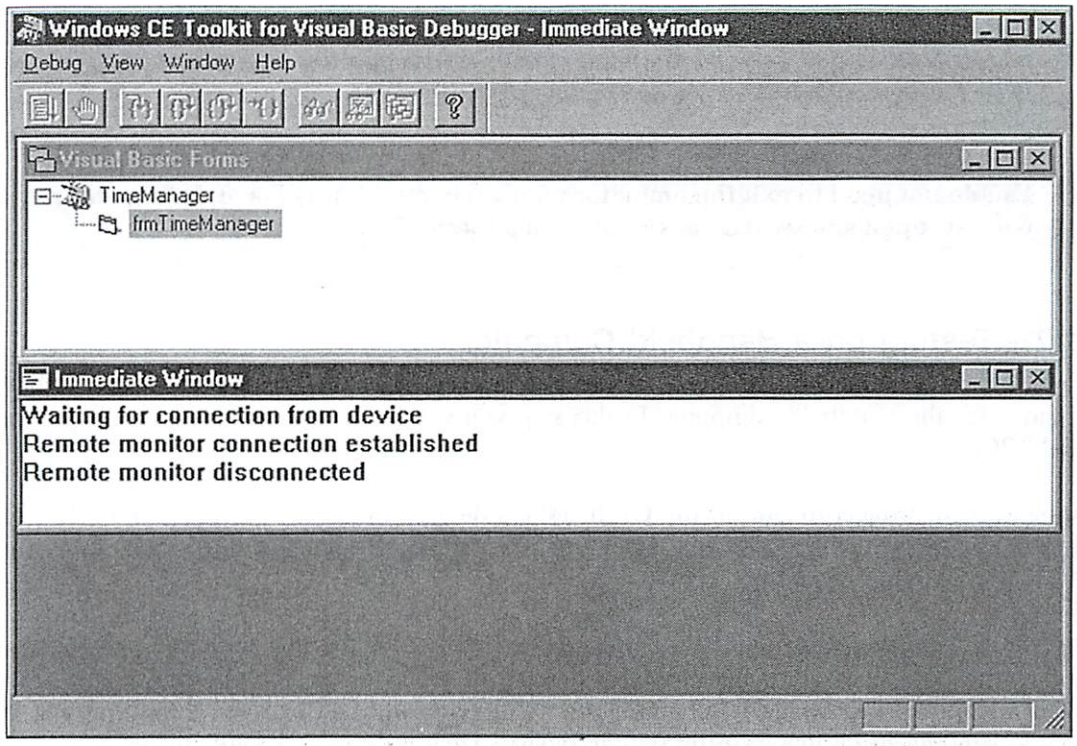
The Windows CE emulator will be displayed as shown in the screenshot below:

While the CE emulator is automatically started when you run a program from within the VB IDE, it can also be manually started when VB is not running. Using the Start menu find the Windows CE Platform SDK folder and select Desktop Handheld PC Emulation.

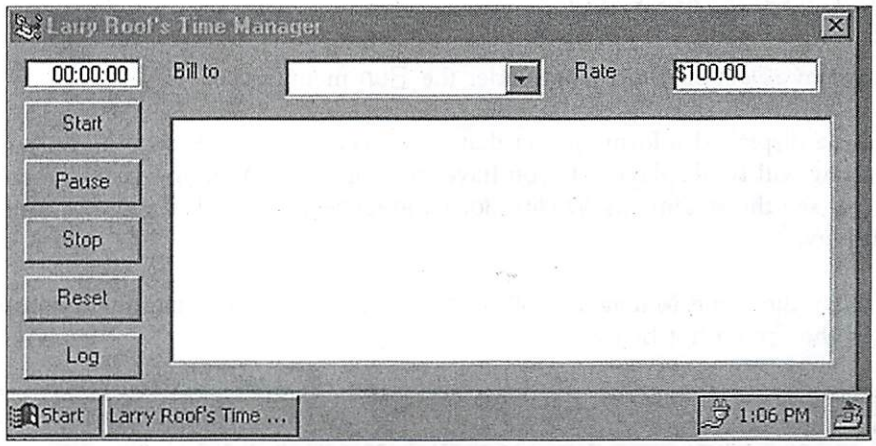
To run the Time Manager application in the CE emulator select Start from under the Run menu. You may be warned that there is no startup form. To correct this problem set the Startup Object under the Project Properties Dialog.

A step is performed automatically for you when you select to Start your program. The Make project step is included into the Start process. Make project is responsible for placing a copy of your `.pvb` file in the location specified by the Project Properties dialog.

The Debugger Window will be displayed. If you have the Immediate Window visible within the Debugger Window you will see the statements Waiting for connection from device and Remote monitor connection established displayed.



The Time Manager application will be run in the emulator. An example of this is shown in the screenshot below:



Try out the application. Click on the Start command button. The time display should start to increment. Test the other buttons as well. To exit the test, click on the X in the upper right corner of the window. You will be asked if you want to exit the debugger. Select Yes.

To exit the emulator, select Suspend under the Start menu on its interface.

Your version of the Time Manager program is now resident in the CE emulator. To run it again at a later time simply select Run *from under the* Start *menu on the emulator, then enter* `TimeManager.pvb` *in the Run dialog* Open: *combo box and click the* OK *button.*

> **You do not need to exit the emulator after each test. Simply leave the emulator window open and switch back to the Visual Basic IDE.**

Step 7 - Testing on a Handheld Computer

Well, while the CE emulator is pretty cool, the whole purpose of this book is to show you how to create applications for the handheld computer. In this step you will learn how to run your CE program on your handheld PC.

To configure your project to run on the CE handheld device, under the Project menu select TimeManager Properties. On the Project Properties dialog confirm that the Remote Device option button from the Run on Target configuration is selected.

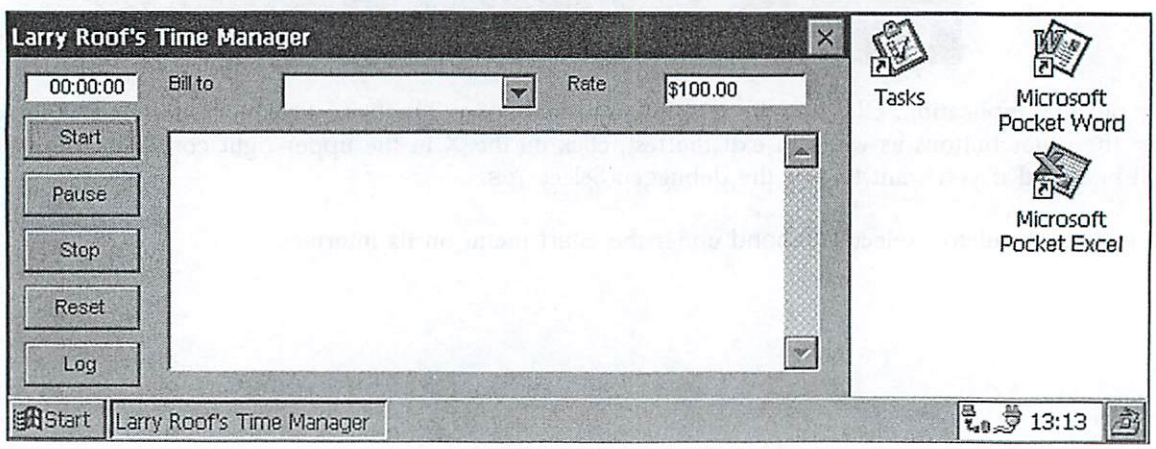
Before you can use the connected handheld computer to test for the first time you need to download the VBCE runtime files. If you forget to perform this step it will be done automatically for you. Under the Windows CE menu, select Download Runtime Files. A message will be displayed informing you that a connection is about to be made to the remote device. Then a download status dialog will be displayed which will inform you of the download progress.

This step should not need to be performed again for future testing unless the handheld computer is hard reset or the supporting files are deleted or corrupted.

Run your project by selecting Start from under the Run menu.

A message will be displayed informing you that a connection is about to be made to a remote device. The Debugger Window will be displayed. If you have the Immediate Window visible within the Debugger Window you will see the statements Waiting for connection from device and Remote monitor connection established displayed.

After a short delay the Time Manager application will be run on your handheld computer. An example of this is shown in the screenshot below:



Try out the application. Click on the Start command button. The time display should start to increment. Test the other buttons as well. To exit the test, click on the X in the upper right corner of the window. On your desktop computer you will be asked if you want to exit the debugger. Select Yes.

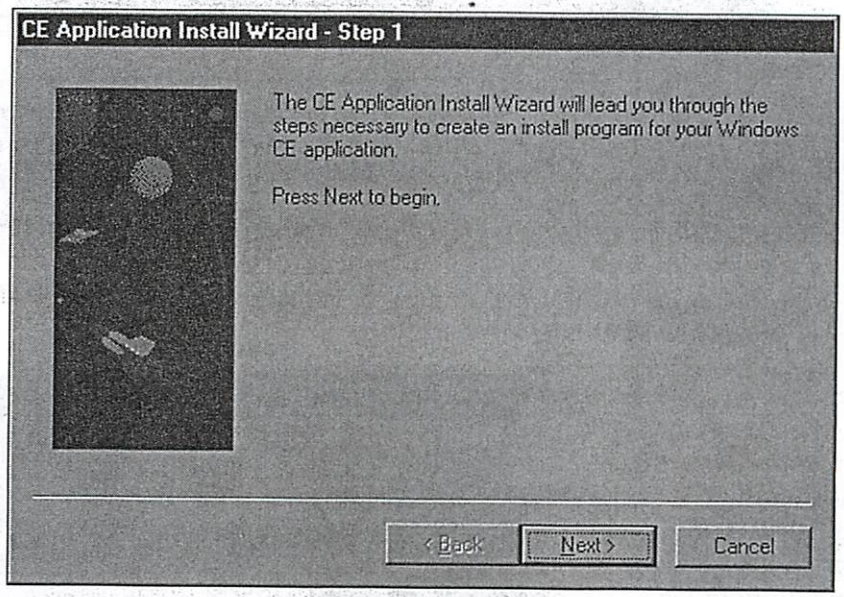
The screenshot above demonstrates a problem that you may encounter in your CE applications, where the form does not make complete use of the display. In this example the form will only take up approximately 75 percent of a 640 x 240 display. You will learn how you can work around this problem in the next chapter.

Your version of the Time Manager program is now resident on your handheld computer. To run it again at a later time simply select Run from under the Start menu on the handheld, then enter `TimeManager.pvb` in the Run dialog Open: combo box and click the OK button.

Step 8 - Creating an Installation Program

Now that we have a completed application, the next step is to create an installation program. This program will handle the downloading of your application and its supporting files to a handheld device. In this step you will be led through the Application Install Wizard that is included with the VBCE toolkit.

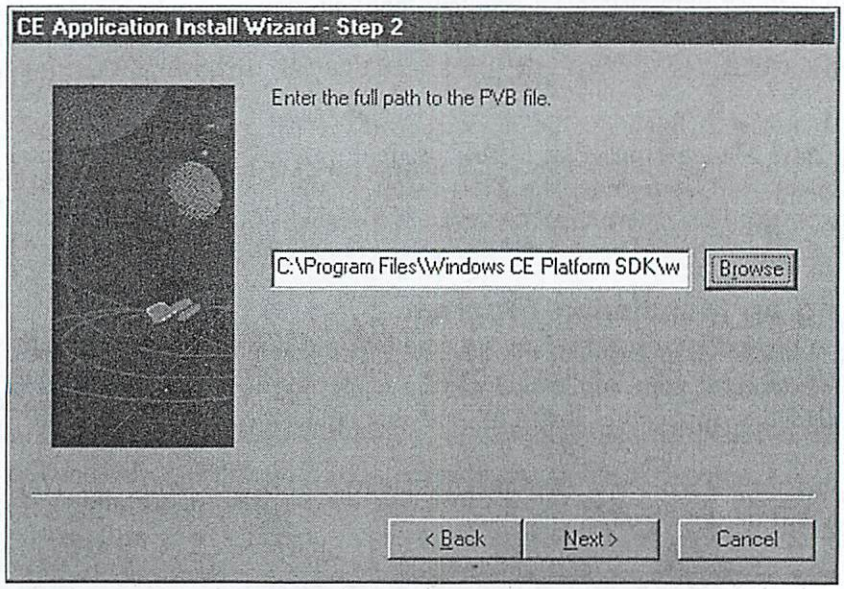
From within the Visual Basic IDE select Application Install Wizard from under the Windows CE menu. The following dialog will be displayed:



The CE Application Install Wizard is similar to the Application Setup Wizard that is included with Visual Basic. Through a series of dialogs, you define what should be included with your application, where and how it should be installed and any additional files to include with the installation.

> *The CE Application Install Wizard can also be run from outside of the Visual Basic IDE. From the* Start *menu navigate through your folders until you find the* Windows CE *folder and select* Application Install Wizard.

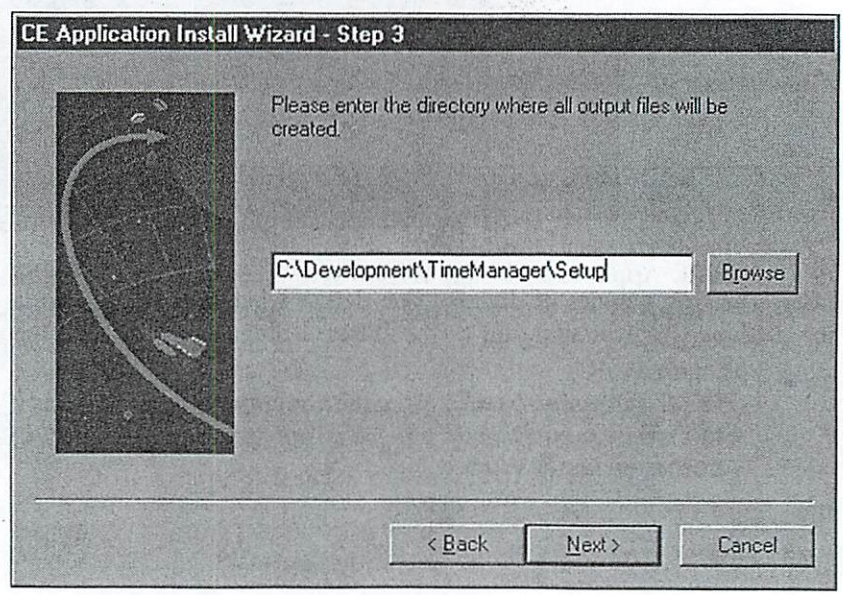
The second dialog in the Application Install Wizard prompts you to enter the path to where you have created the `.pvb` file for your program. Click the Browse button to display the file dialog. It should be set to the folder you specified to use to create the project file in the Project Properties dialog. If not, you will have to navigate to that folder.



Make sure that your project file is up to date before creating the installation. Unlike the wizard used by Visual Basic to create installations, this wizard does not give you the option to rebuild your program during the process of generating the installation kit.

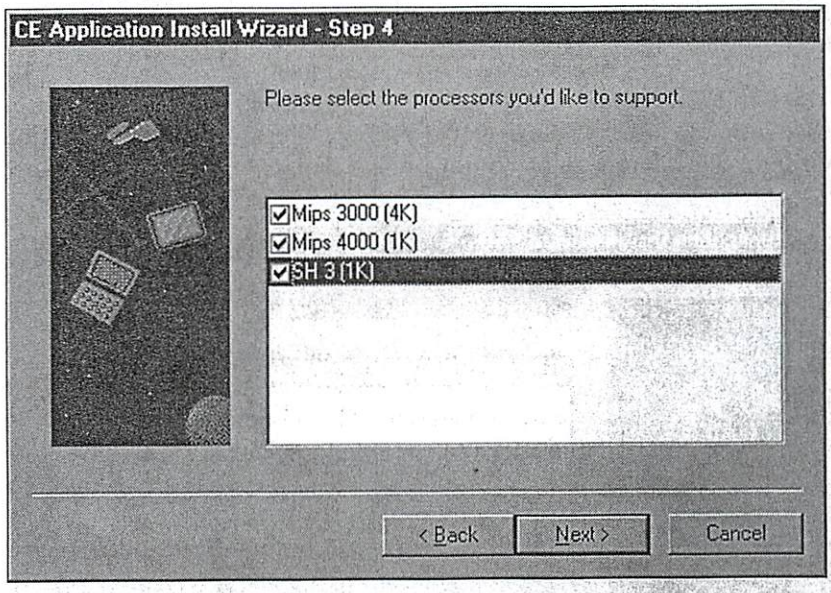
To do this, while you are within the VB IDE with your project loaded, select Make TimeManager... from under the File menu to generate a `.pvb` file.

The third dialog in the CE Application Install Wizard prompts you for a path on your desktop computer to use to store the installation files. I prefer to create a subfolder under the folder where your project is stored. For example:



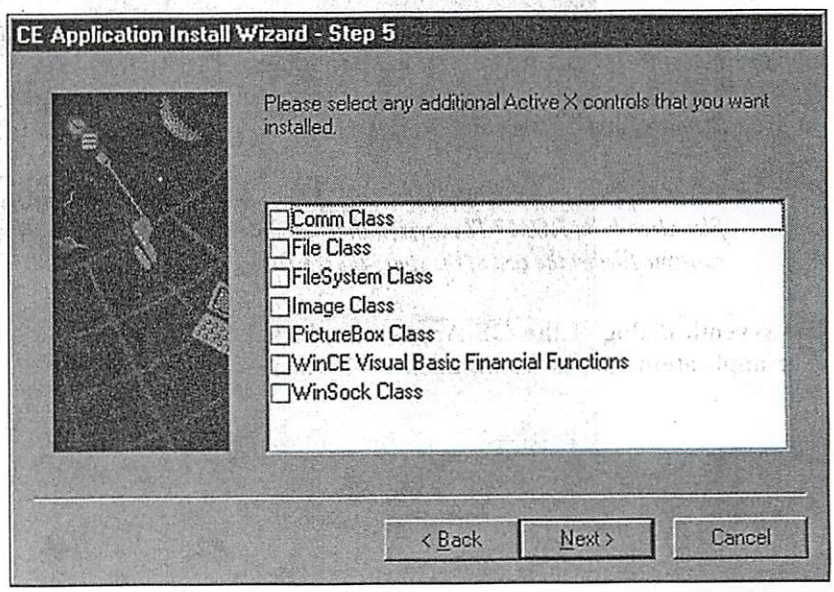
The CE Application Install Wizard functions similar to the Visual Basic Application Setup Wizard configured to produce its output to a directory. In both cases a set of files are generated to the specified directory and are used to install the application from there.

The fourth dialog in the CE Application Install Wizard is used to select the target platforms for which to create installation kits. While your project file is the same for all three platforms, the Pocket Visual Basic runtime files are not.



Until you are comfortable with understanding the purpose of each of the files and folders generated by the installation kit and also know which files to include for each platform, you may wish to generate three separate installations, one for each platform.

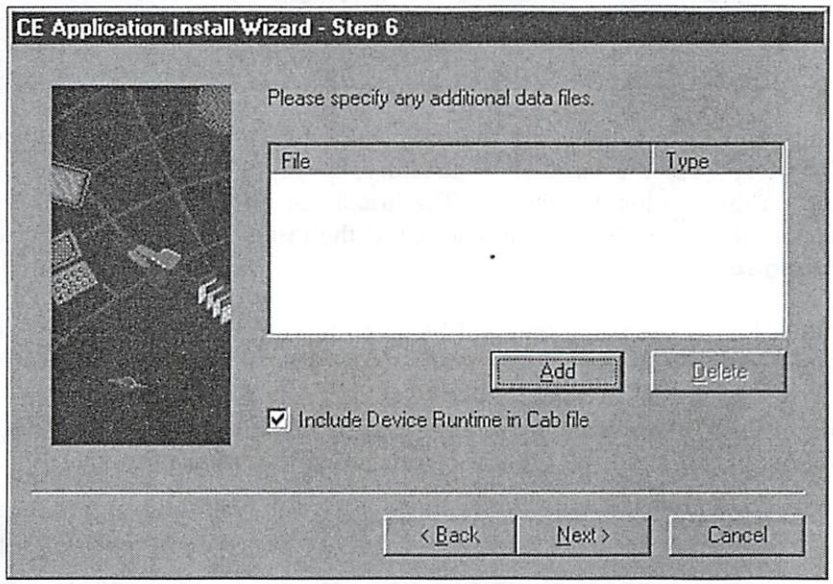
The fifth dialog of the CE Application Install Wizard is used to specify any ActiveX controls that you have used within your application and as such should be included in the installation kit.



Unlike the Visual Basic Application Setup Wizard that checks your project file to determine what ActiveX controls your project makes use of, the CE Application Install Wizard requires you to specify what controls to include.

> **The ActiveX controls shown on the screenshot above include only those controls that were shipped with the Visual Basic for Windows CE Toolkit. Any third party or self-written controls that your application uses will have to be included using the next dialog.**

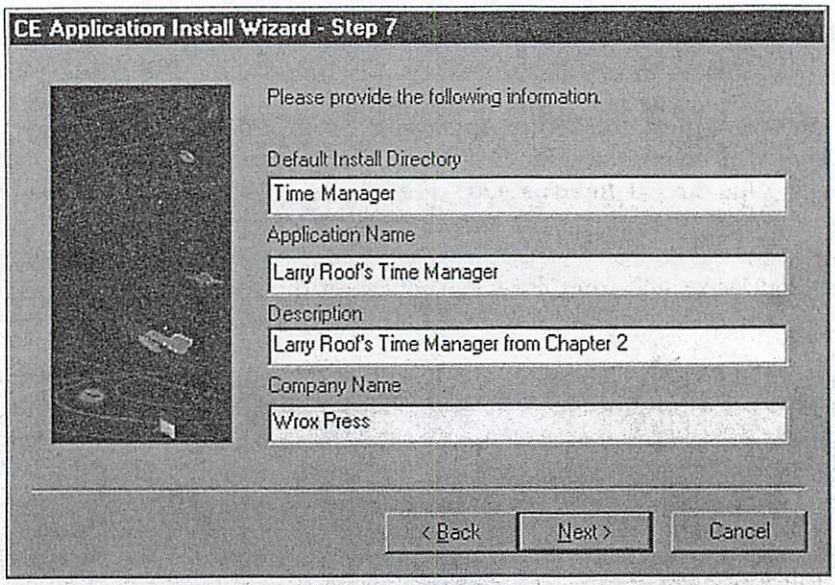
The sixth dialog of the CE Application Install Wizard allows you to include files other than your project file, standard ActiveX controls and the VBCE Toolkit runtime files, with the installation kit. You may use this feature to include help files, data files or any other files that are used with your application.



The Include Device Runtime in Cab file checkbox allows you to control whether or not the Toolkit runtime files are included with the installation kit. If you are developing an application to be used with devices that already have the Toolkit runtime files installed, you would clear this checkbox so that the runtime files would not be included. This would minimize the size of the installation kit.

> *What happens if you download a copy of the VBCE Toolkit runtime files to a device that has the runtime files already on ROM? The version that you download will be used. This allows you to update the runtime files at the cost of the space the runtime files use in RAM.*

The seventh dialog of the CE Application Install Wizard is used to specify information on where and how your application should be installed on the handheld computer.

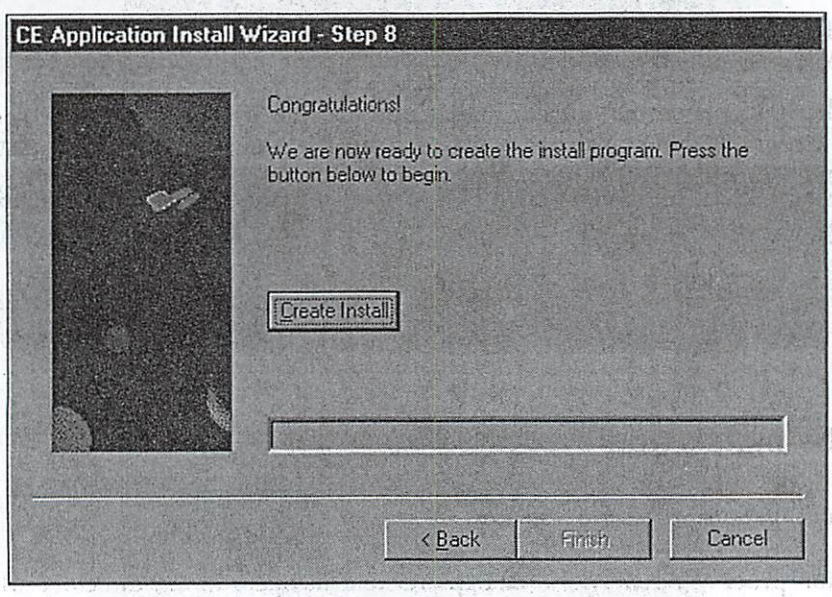


The two important items in this dialog are the installation directory and the application name.

Default Install Directory - For the installation directory, specify the folder that you would like your application placed on the handheld computer. The installation process will append **`Program Files\`** to the front of whatever you enter. In the example below the installation directory will thus be **`Program Files\Time Manager`**.

Application Name – This is the name that will be given to your application when it's installed on the handheld computer.

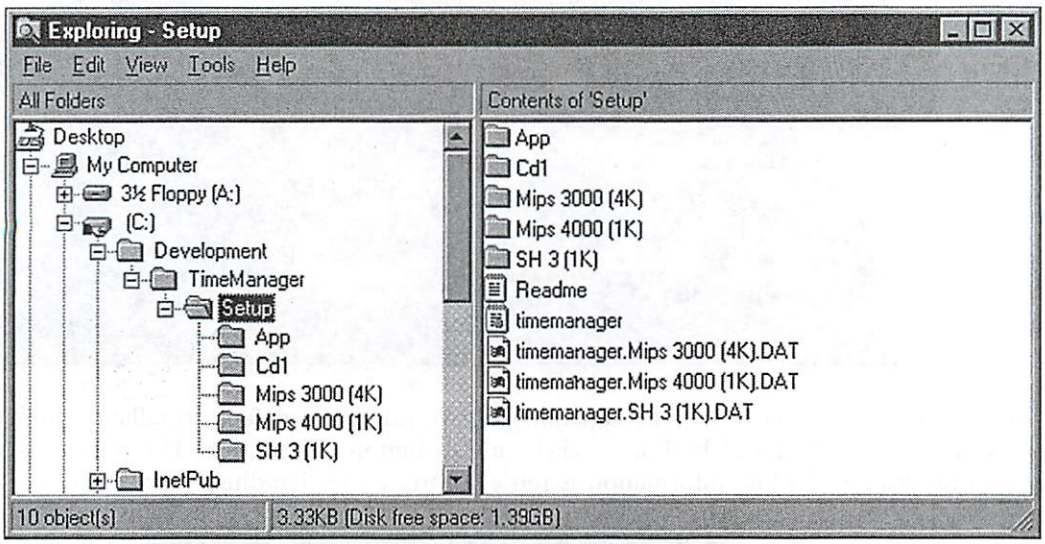
The eighth dialog of the CE Application Install Wizard is nothing more than a "Okay, let's do it". Click the Create Install button to begin the process of generating the installation kit.



The Finish command button will be enabled to signify the completion of the process of generating the installation kit.

The CE Application Install Wizard, unlike the Application Setup Wizard included with Visual Basic, does not provide the capability of saving the settings that you use to create an installation kit. Instead, it provides instructions in a file named `Readme.txt` for rebuilding the `.cab` files used by the installation. This file can be found in the directory in which the setup was created, as shown in the figure below.

An example of the output generated from the CE Application Install Wizard can be seen in the screenshot below:



The folders generated by this process are:

- `App` Folder – Contains the project file for your application. In the case of this example that is the file `TimeManager.pvb`.
- `Cd1` Folder – Contains the VBCE Toolkit runtime files (if they were specified to be included), the setup program, and initialization file used by the setup program and `CAB` files for each target platform you specified to include with the installation kit.
- `Mips 3000`, `Mips 4000`, `SH 3` Folders – Contains processor specific files for each target platform you specified to include with the installation kit.

The contents of the `Setup` *folder can be copied to a diskette or compressed into a zip file for distribution.*

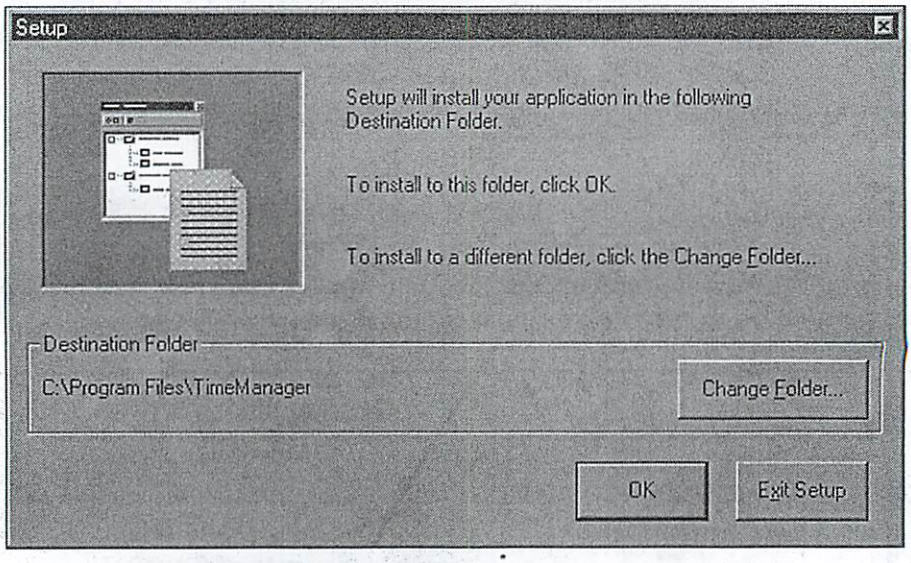
Step 9 - Test the Installation Program

With the completed installation in hand, you are ready to test just how well it works. In this step you will be led through the installation process to see how the kit performs.

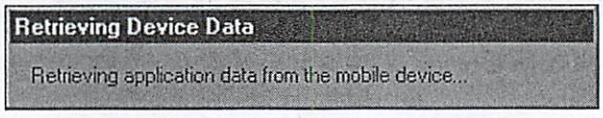
To start the installation process, open Windows Explorer and navigate to the directory where you created the installation kit.

In this directory you will see a folder named `Cd1`. Enter that directory.

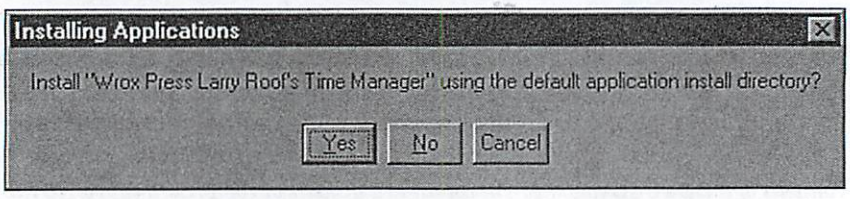
In the `Cd1` directory you will find the setup program, `Setup.exe`, that will perform the installation. Run the program and you will see the following dialog:



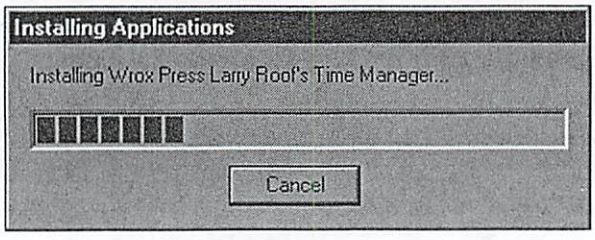
This first dialog provides you with the capability of overriding the default installation folder. For our purposes the default folder will be fine so click the OK button to continue. The following informational dialog will be displayed while information is retrieved from your handheld device:



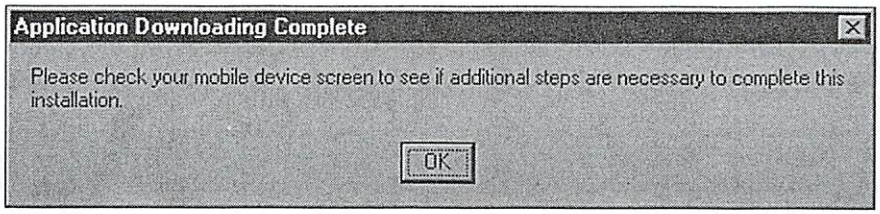
Upon completion of the retrieval of application data from your handheld device, the following dialog will be displayed which gives you one last chance to change where your application will be installed. Again, for our purposes, the default folder is fine. Simply click the Yes button.



The installation process will begin. A progress dialog, as shown below, will be displayed:

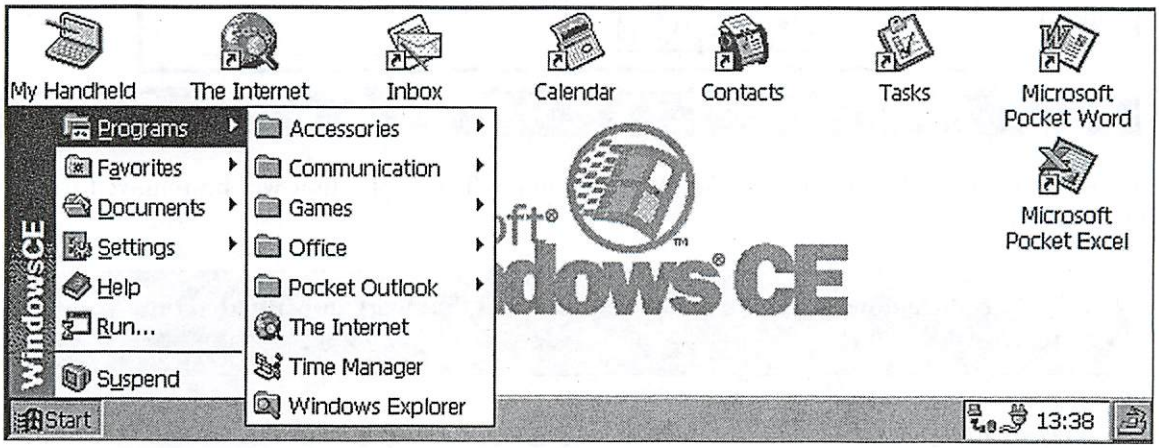


Upon completion of the installation, the following dialog will be displayed. It reminds you to check on the targeted handheld device for any additional steps that may need to be performed:

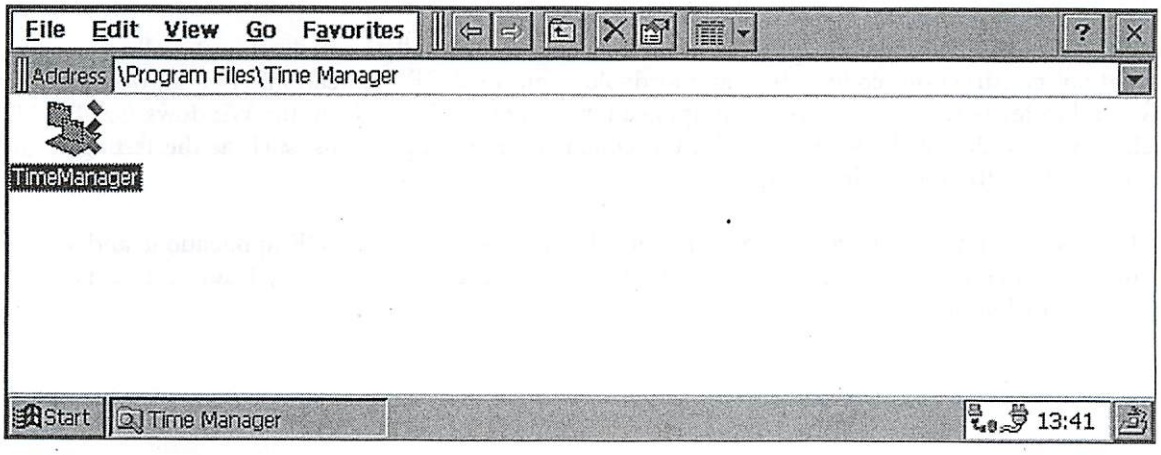


What you most typically will encounter on the handheld device is confirmation to overwrite files that already are resident on that device.

Next you will confirm the successful installation of the Time Manager program. The first step is to check if Time Manager has been added to the Start menu as shown in the screenshot below:

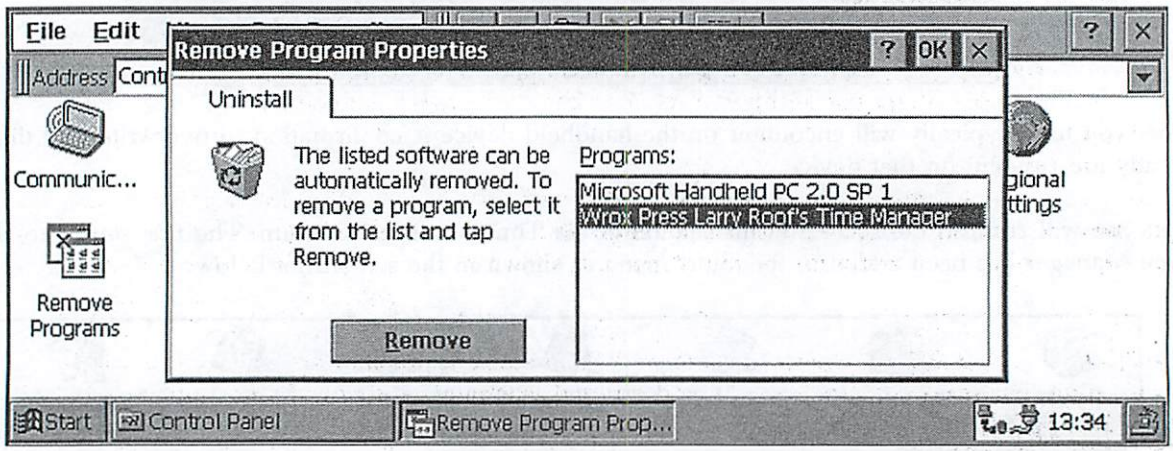


Next open up Windows Explorer on your handheld computer and navigate to the folder where you specified the installation to be placed. You should find the file **`TimeManager.pvb`** located in this folder as shown in the screenshot below:



Start the application to confirm a successful installation. Try out its features. Close the application after you have finished.

The last step of this confirmation process is to verify that the uninstall process will work with our application. Under the Start menu select Settings, then Control Panel. From the Control Panel interface start Remove Programs. An example of the Remove Program interface is shown in the screenshot below.



Select your program and then click on the Remove button. The application will be removed from your handheld device.

This procedure does not remove any runtime files that were downloaded onto your system.

If you are developing CE applications that will be deployed on a variety of platforms make sure to test the installation and the performance, on all of the targeted platforms.

Summary

That's it – the complete creation process, from conception to installation. I have just walked you through the creation, construction, coding, testing and deployment of a CE application. As I showed you, there are many similarities between developing an application using the Visual Basic for Windows CE Toolkit and straight Visual Basic. At the same time, VBCE contains some unique steps, such as the testing in the CE emulator and on the handheld computer.

Hopefully, at this point you are comfortable with the process of creating CE applications and we can turn our attention to common techniques that you will use to construct your own CE apps. That is the focus of the chapters to follow.