Continue	
I'm not robot	reCAPTCHA

Event driven programming in vb

```
In this lesson we demonstrate how events are utilized in the .NET Framework Class Library specific to Silverlight, WPF and ASP.NET Web Forms application event to code that handles that event. The point is that there's a pattern to how
.NET works with events and how events drive most Graphical User Interface based applications. Download the source code for Understanding Event Driven Program in such a way that certain actions are taken only when an event occurs at the user's initiative or
otherwise such as click of a button, or completion of a time interval etc. Visual Basic 6.0 offers a wide variety of events to program. Some of the events are common events are as follows: In Visual Basic, the strength of any graphical or non-graphical controls
is measured by the kind of special events it provides to a programming as it can never be said which piece of code will get executed unless we know which event occurred. But this very fact contributes to the major advantage of event-
driven programming. With this approach, we can be sure of some components of a program (or software for that matter) running successfully while we work on others. Another advantage associated with it is the convenience offered to both the end-user and the programmer. Events can be initiated by the user (like clicking a button). So it gives the
user a feel of control over the system. On the other hand, programmer can put some very crucial events such as loading of a form, where several initializations and checks can be performed. References[edit | edit source] A Limited Personal History of VB.NET, then UWP/WinUIWhat technologies can be leveraged better
than both object-oriented programming and procedural programming with fully featured object-oriented capabilities is the sort of technology that can provide it all and together. Modern Events in WPF and UWPWindows Forms, Windows Presentation Foundation (WPF) and the Universal Windows Platform (UWP)
are each prime examples of event-driven programming. The object orientation of these interface frameworks is nothing revolutionary was the comprehensive and flexible ability to handle hardware, software, and system events in a robust and flexible manner. The Driven
Events of 1991 to 2002This event driven model has been more pivotal to programming history than ever object-oriented models have been. Visual Basic initially began a programmers small niche revolution in the year 1991. Over a decade later, in the year 2002, Visual Basic met its end as an independent platform, and VB.NET emerged alongside its
new alternative syntax twin — with a makeover into C++ syntax style — giving arise to the first .NET Framework, and the release of Visual Basic Event-Driven ProgrammingIn theadyent of the first .NET Framework, and the release of Visual Basic Event-Driven ProgrammingIn theadyent of the first .NET Framework, and the release of Visual Basic Event-Driven Programming Interval Event Driven Programming Interval Event 
year 1991, Visual Basic debuted. This began an upheaval and split in personal computer software design, development, and engineering. In the years of Visual Basic, there was surely a rift between conventional and more powerful C++ software programming and the radical and productive but more restricted event driven ways of Visual Basic. Visual
Basic developers could prototype and bring a product to readiness faster than ever, but were plagued by having their software limited in power and adjustability. The Visual Basic programs could do. Both groups of software programmers seemed to
be glad of the way they developed their respective software titles, and also both groups of software programmers seemed to have respectful envy for what the "other side" could do with those "other technologies," as well as a playful sense of humor about which side of the aisle with which a programmer was developing software. BASIC, QuickBasic,
and Visual Basicin 1990, Visual Basic eclipsed and rendered quickly obsolete, its late 1980s predecessor, Quick Basic. The language syntax of BASIC remained largely intact, just as it had been for a decade of interpreted BASIC and compiled Quick
Basic. However, much was added to the language in terms of keywords and backing framework. Event Event Event Event way into Windows Presentation Foundation (WPF), and now it is alive and well in Universal Windows
Platform (UWP). Event Handlers in Contemporary UWP Project loaded, the software developer may select any of a wide variety of Elements in the XAML code window, and then proceed to toggle the lightning small button on the upper-right of the Properties docked window (with the tooltip).
upon hovering, depicting "Event Handlers for the selected element,". If this is done, even for an element of lesser properties such as a Grid XAML element, or for just about any other element, the programmer can review the very thorough and detailed events to which to handle with functions in the code behind .cs file. I believe that the extensive
listing of the many and varied events which displays upon switching from Properties to Events indicates that many programmers, in being much like me, have not explored the powerful and broad ranging software app behaviors that the UWP thoroughly makes available to all of us, primarily by way of events and adept usage of event handlers. 1 2
Programming Languages Programming Languages 7 Event-Driven Visual Programming Languages 8 Event-Driven Visual Programming Languages 7 Event-Driven Visual Programming Languages 8 Event-Driven Visual Programming Languages 9 Event-Driven Visual Programming 1 Event-Driven Visual Programming 1 Event-Driven Visual Programming 1 Event-Driven Visual Programming 1 Event-Driven Vis
and other system resources. 4 All the paradigms which include imperative, object- oriented, functional, and logic programming - are based on a fundamental model of computation in which include imperative, object- oriented, functional, and logic
programming - are based on a fundamental model of computation in which the program design predetermines what will occur; Event-driven programs do not predict the control sequence that will occur; They are written to
run reasonably to any particular sequence of events that may occur once execution begins 5 In this model, the input data govern the particular sequence of control that is actually carried out by the program. In this model, the input data
govern the particular sequence of control that is actually carried out by the program. Event-Driven Programming (2) The most widespread example of an event-driven program is the GUI mouse- and, windows-driven user interface found on most desktop and laptop computers in use today, including web-based applications. The most widespread
example of an event-driven program is the GUI mouse- and, windows-driven user interface found on most desktop and laptop computers in use today, including web-based applications. Moreover, execution of an event-driven program does not typically terminate; such a program is designed to run for an arbitrary period of time, often
indefinitely. Moreover, execution of an event-driven program does not typically terminate; such a program is designed to run for an arbitrary period of time, often indefinitely. 6 This is accomplished by VB's Integrated Development Environment (IDE), in which a mouse is used to "draw" application and use the keyboard to input the code that is to be
executed. This is accomplished by VB's Integrated Development Environment (IDE), in which a mouse is used to "draw" application and use the keyboard to input the code that is to be executed. The Visual Basic VB provides massive support for easily creating
the user interface to Windows applications. VB introduced was the concept of an event-driven programming model. VB performs event-handling function, the only time code will execute in VB is in response to an event-triven programming model. VB performs event-handling function, the only time code will execute in VB is in response to an event-triven programming model. VB performs event-handling function, the only time code will execute in VB is in response to an event-triven programming model.
execute in VB is in response to an event! 7 The availability of controls (built-in, or controls you can purchase) is the single biggest reason why VB has reached the level of popularity that it
currently enjoys. Visual Controls Every Visual Basic application will consist of controls, which represent reusable graphic user interfaces, database, and other system resources. The VB's visual capability are
embeded in these controls, which include the intrinsic controls and additional ActiveX controls. 8 Visual C++: It is virtually identical with Visual Basic in terms of the ease of creating Windows programs. Visual C++: It is
virtually identical with Visual Basic in terms of the ease of creating Windows programs. Other Visual Programming Languages Delphi: Based on Pascal programming language, designed to be compatible with the controls that support
Visual Basic. The single best feature of Delphi is that it creates completely stand-alone EXE files - unlike Visual Basic which requires the distribution of a huge number of supporting files for even the smallest of applications. The single best feature of Delphi is that it creates completely stand-alone EXE files - unlike Visual Basic which requires the
distribution of a huge number of supporting files for even the smallest of applications. 1 Unit 20: Event Driven Programming 2 Aims Discuss what is an event driven program Define what is an event driven program De
program Discuss advantages and disadvantages and disadvantages of event driven programs? Event driven programs are typically used with GUI operating systems What events are triggered by a user using an OS? Clicks Movements Keys Timer What other software can you think of that is event driven?
 Spreadsheets, Databases, the lists go on..... Castle College 4 Define what an event is? An event is anything that happens to an object? Controls can you think of? Form Buttons Lists Pictures etc...... Castle College 5 Page setup (Microsoft)Castle College 6 Define what an event
handler is?Most objects have a large variety of possible events, i.e. Click, double click, mouse move etc... Event handler is the subroutine (procedure) that holds the code that runs when an event has occurred. Private Sub btnDisplayMessage_Click() MsgBox ("Hello World") End Sub Event triggers selects the appropriate event handler that determines
what code is executed. What part of the code shows the trigger? Castle College 7 Event Challenge Castle College Private Sub txtEnterText Change() MsgBox ("Hello World") End Sub Private Sub btnClose MouseUp() MsgBox ("The King of
Rock") Private Sub btnMessage_MouseMove() MsgBox ("Hello People") Castle College 8 Event loops Event driven programming languages need to have event loops Event driven programmer is not
normally aware of this kind of loop what is part of an event driven programs make up. There are other kinds of loops that the programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use, this is the same principle, however it is not programmer will use an application by the programmer will use a same principle.
characteristics of event driven programming languages. Event handlers Trigger functions Event loops Forms (which contain controls) Castle College 10 Programming languages VB.net C# VBA VB6 Castle College 11 Some
advantages of event driven programming Flexibility Programmer has control of where to place code and how to start it. Suitability for GUI An event driven languages makes use of GUI controls to trigger Events Simplicity of programming Visual programming Texas and the same of GUI controls to trigger Events Simplicity for GUI An event driven languages makes use of GUI controls to trigger Events Simplicity of programming Visual Pro
controls. Effective testing tools Castle College 12 A disadvantage of event driven programmingCan be slow Loads of processing is required due to the trigger functions as they match the type of event with the event handler. Castle College 13 Conclusion Discussed what is an event driven
programDefined what is an event Looked into event handlers and triggers Discussed event loops Looked at example applications and languages Discussed advantages and disadvantages of event driven programming Castle College 1 | P a g e BBIT BBIT 3206 : EVENT DRIVEN PROGRAMMING
AUTHOR: Njuguna Patrick Phone:0721238570 email: rpwnjuguna@gmail.com 2 | Page Course content INTRODUCTION The Visual Basic 6 environment Defining terms Creating a Visual Basic 6 environment Defining terms Creating a Visual Basic 6 environment Defining terms Creating a Visual Basic Project Practice 
Writing a VB procedure Calling procedures BUILDING BLOCK OF VB 6 Using the Visual Basic 6 code editor Adhering to programming standards Data types, variables and constants in Visual Basic Using operators Control structures - IF...THEN, Select Case, DO...LOOP, FOR...NEXT Practice assignment - Upgrading the Scoreboard DESIGNING
VB APPLICATION Designing the Visual Basic Application Working with users Guiding principles Choosing a Visual Basic Form Standard controls: Picture, Frame, CommandButton, Label, TextBox, CheckBox, etc. Visual Basic practice assignment: Creating a Payroll Form
Arrays More controls: ListBox, ComboBox Properties and Methods of objects in Visual Basic code 5 | P a g e In a VB project, the processes that occur have to be associated with events. An event is something that happens - the user
clicks on a button, a form is opened, the result of a calculation is too large. The operation is event-driven because everything that executes does so as the result of some kind of event. The role of the programmer is to anticipate the events and to write the code that will be executed when the event occurs. A VB application is interactive in the sense that
the user is constantly interacting with the program. The user inputs a Customer Id, the program checks the Id in the database and immediately brings up the customer's file or displays a message that the particular Id is invalid. Project description We want to create a Scoreboard for a football game (there it is already!) between the Giants and the
Redskins. To begin with the simplest task we will only count the touchdowns and display appropriate messages. Please note: although we will create a complete functional Project with controls and code and so on, the purpose of this exercise is to show what can be done. In the following lessons we will be explaining scripts and the use of controls in a
lot more detail. If you study this example you should be able to relate it to what you already know of programming and judge whether this tutorial will be easy or hard for you to do. 1.2 Creating the Project First thing to do is to create a Directory where you will store all your VB Projects. Call it VBApps, for example. Then start VB. The first screen will
ask whether you want to open a new project or an existing one - it's obviously a new one and it will be a Standard EXE. Then, maximize all the windows (it's easier to work with - some of the examples in the tutorial had to be reduced for the sake of the presentation). Now, save your project. It will first ask you to save the form - call it Score.frm - and
then the Project - call it Scorebrd.vbp. From now on, do File-->Save Project very, very frequently. Pe ed Bile Edit Yiew Project Format Debug Bun Query Diagram Tools Add-Ins Window Help |s-4- Alero te meelool, yw RR aS e ea wl [x] ESE GE i es el ht 'osoft Select Ta Always Sta for now Alphabetic | ce Actives EXE Active DLL Activex YB Application
on a grey background. To change the color, just click anywhere on the form, go to the properties window, find the properties window background (teal) or to any color you want in the palette. In our first example we will need 6 labels and 2 command buttons. Each one of these objects that you put on a
Form is called a control. To get a control you go to the Toolbox, click on the control you want, come back to the Form and click and drag the control to the size and position you want. Position the control you want this point, you should see your Form appear, just the way you created
it. However if you click on any of the controls, absolutely nothing happens! There are events that occur; the form what to do when it sees an event. That is why we have to write code, also called script. 11 | P a g e To switch between the Code window and the Form window, use the
buttons just over the Project Explorer window (diagram on the left). Once in the Code window, you have the option of seeing all the code for one event at a time. Use the buttons in the lower left-hand corner (diagram on the right). To select the object and the event you wish to code, use the two Listboxes at the top of the
Code window. The one on the left for the object and the one on the right for the event. Start with General ... Declarations and then Form ... Load, etc. At this point you might want to download the sample program and study it. In the following lessons we'll add functionality to the exercice and we'll explain what the code means. But for the moment, a
good exercice would be to write part of the code and then try to figure out how to improve certain aspects of the program. 12 | P a g e Now we can Run it and see something happen. When the Form loads, it will initialize the fields that we specified in the code. Now code the Command1 button and Run it to see the result. 15 | P a g e A second
improvement Another thing we usually need in a program is a re-initialize button. After one loop of the program, in this case one match, we usually want to clear all the data and start over. For that we'll create a Clear button on the form. 16 \mid P a g e But, we'll notice that what we do with the Clear button is in fact the same thing we do when we load
the form in the first place. So, we'll use the procedure technique to simplify the code. 17 | P a g e 1.3 Writing code The Code Editor is as simple as hitting the proper button. You may have discovered that you can also call-up the Editor by double-clicking on an object. It is also possible to
select "View code" with the right mouse button. You will note that the Editor has all the functions of a text editor and then some. The most commonly used functions will be Cut ... Copy ... Paste which you can call from the Menu, from the Toolbar or from the right mouse button. You will note that the Editor has all the functions of a text editor and then some. The most commonly used functions will be Cut ... Copy ... Paste which you can call from the Menu, from the Toolbar or from the right mouse button. You will note that the Editor has all the functions of a text editor and then some. The most commonly used functions will be Cut ... Copy ... Paste which you can call from the Menu, from the Toolbar or from the right mouse button.
help There is a lot of documentation available from the Menu should be used regularly. Very often just doing a search on a word in particular will be sufficient to get you out of a jam. If you want to go into more detail check out the Contents part of
MSDN (Microsoft Developers' Network) and surf through it. Writing code VB is not very particular about presentation - spaces, indents, lower case or upper case, it doesn't make too much difference to the compiler. But it may make a whole lot of difference to the programmer who has to maintain your code in 2 years, after you've moved up to
President. Apply "Best Programming Practices" When you work with RAD (Rapid Application Development) tools like VB in a graphical interface environment, you are still a Programmer, a writer of code. You are a developer. We will cover that in the next lesson. But at the moment, you are still a Programmer, and unless you are
developing an application for your own personal use, that nobody else will see, you have to think of the environment, of the team you are working with. "No man (or woman) is an island!" Especially when it comes to programming. The code you write may have to be checked by an Analyst. It will have to go through testing. It may have to be modified
by other team members and it almost certainly will go through modifications, maybe several times, in the months and years ahead when you want others to say
behind your back is: "That Jane was blindingly efficient, brilliant, a genius with comments ..." 20 | P a g e numbers) Variant (with characters) 22 bytes + string length Same range of each element is the same as the range of its data type. In all probability
in 90% of your applications you will use at most six types: String, Integer, Long, Single, Boolean and Date. The Variant type is often used automatically when type is not important. A Variant type is not important type is not important. A Variant type is not important type is not important. A Variant type is not important type is not important. A Variant type is not important type is not important type is not important type is not important type is not important. A Variant type is not important type i
variables Declaring a variable means giving it a name, a data type and sometimes an initial value. The declaration can be explicit or implicit. An explicit declaration looks like: Dim MyNumber As Integer Now the variable MyNumber exists and a
2-byte space has been reserved for it. An implicit declaration of the fly", its data type is deduced from other variables. For example: Dim Total1 As Integer 'Explicit declaration Total3 is not formally declared but is implied, it is "arrived"
at" from the other declarations. It is never a good idea to have implicit declarations. It goes against the rules for clarity, readability and ease of use of the code. To make sure that this rule is followed, start the Declarations with the Option Explicit clause. This tells the compiler to consider implicit declarations as errors and forces the programmer to
declare everything explicitly. Other examples of declarations: Dim MyName As String Dim StudentDOB As Date Dim Amount5, Amount6, Amount7 In the last example the type assigned to each variable will be: Variant. It is the default type when none is 21 | P a g e specified. There can be multiple explicit declarations in a statement: Dim EmpName As
String, SalaryMonth As Currency, SalaryYear As Currency, SalaryYear As Currency In this final example, what are the types assigned to the three variables: Dim Amount1, Amount2 are considered Variant because VB specifies that each variable in a
statement must be explicitly declared. Thus Amount and Amount take the default data type. This is different from what most other languages do. 2.3 Constant is defined with: Const ValuePi = 3.1416 The Scope of variables The term Scope refers to
Sub Command1 Click () Dim Total1 As Integer Static Total3 As Integer Total1 = Total1 + 1 Total2 = Total3 + 1 End Sub Every time Button1 is clicked, Total1 is declared as a new variable during the
execution of that clicked 22 | P a g e event. It is a procedure ends, Total1 is a new variable in that procedure ends, Total1 is a module-level variable, meaning it is available to every control in this Form.
When Button1 is clicked, it increments by 1 and it retains that value. When Button2 is clicked, Total2 is incremented from its previous value, even if it came from the Button1 event. Total3 shows another way of retaining the value of a local variable. By declaring it with Static instead of Dim, the variable acts like a module-level variable, although it is
operators Operator Definition Example Result ^ Exponent (power of) 4 ^ 2 16 * Multiply 5 * 4 20 / Divide 20 / 4 5 + Add 3 + 4 7 - Subtract 7 - 3 4 Mod Remainder of division 20 Mod 6 2 \ Integer division 20 \ 6 3 & String concatenation "Joan" & " " & "Smith" "Joan Smith" Note that the order of operators is determined by the usual rules in
programming. When a statement includes multiple operations is: Parentheses (), ^, *, /, Mod, +, - 2.4.2 Logical operators Oper
counter is set to the value of start. 2) Counter is checked to see if it is greater than end; if yes, control passes to the statement after the Next; if not the Next; if not the statement after the Next; if not the Next; i
know more, consult the VB Language Reference. Assignment To practise your coding and editing skills, try modifying the Football example by adapting it for different ways to score, as shown here: Touchdown 6 points Field goal 3 points 2-point Convert or Safety
2 points Convert 1 point 26 | P a g e Chapter 4 Designing Application Introduction When you start to work on a VB Project you are no longer just a programmer - you are designing an application for your own use you will have to work with a team of
specialists including, but not limited to, users, analysts, GUI designer, programmers, testers, network specialist, webmaster and marketing people. The whole process is iterative - do part of it, check it, get input, go back and correct it, do the next part, and so on. Nobody expects you to do a whole project in one fell swoop - it would probably be a
disaster if you did do it that way. The importance of Users Any project that you develop has to involve Users. They are the people who will sit in front of your interface for eight hours a day and decide if they like it, no matter how efficient the code and how many millions of dollars were spent developing it, they will find ways
to sabotage it. Get users involved from the start. If you are developing a product to specs, that is to be some client eventually, there has to be someone who knows what that eventual client needs. Find a typical user of the product to use as a sounding board. Remember: you are just the developer; no matter how cool you think it would be to use
all purple text on orange backgrounds, it is the user who will tell you what is not. As you develop more and more parts of the application, run them by the user to check for accuracy, completeness, clarity, etc. Here's an example of how to design for clarity. Given that 01/02/03 is a date, what date is it? If you are an American, you
probably automatically assume that it is January 2nd, 2003. If your user is French, however, he would assume that it is February 3rd, 2001 and should always be written as 2001-02-03. If all your forms are designed as:
"Enter date" with a blank box beside it, you are headed for trouble. Program design today is a race between software engineers striving to build bigger and better idiots. So far, the Universe is winning. -- Rich Cook That's just a joke, by the way. Most users are not idiots.
Sometimes they appear confused because they are trying to solve the problem and they can't figure out how. But that's not their job is to explain clearly what it is they need. Your job is to figure out how to provide it. Don't underestimate users. Be patient, be understanding without being condescending and be humble. There's a lot of things out how to provide it.
that the user knows how to do that you don't. 27 | P a g e 4.1 Creating the User Interface that you design is the most important part of the application. The term commonly used for this type of interface is: GUI (Graphical User Interface). It's pronounced "goo-wee", not "guy". It is graphical because it
consists of buttons, menus, icons, etc. An example of a non-GUI is DOS (remember that?) where everything is text. User interface refers to the fact that it is the part of the application between the user, in front of the screen, and the code behind the screen. How well the user can interact with the code depends on the quality of the interface. Guiding
principles The user is in control. The user must feel he is in charge of the application. He must have a certain amount of control over such things as window size, window position, choice of fonts, etc. There should definitely be a "Preferences" item in the menu. Consistency is maintained throughout the application. The user can move to any part of
the application and not have to re-learn how things work. Consistency in the choice of icons, in date formats, in error messages means that the user can concentrate on the work. As much as possible, the application should be consistent with Windows standard. For example, "Move to the Recycle Bin" is different from "Delete" - the user has come to
expect that an item in the Recycle Bin can be recovered if need be. Application should be "forgiving", or "fault-tolerant". Users will make mistake. A single error should not bring the application crashing to the floor. If there is no room for errors, users will be afraid to experiment, to discover on their own how to do things. It will slow the learning
process considerably. Always supply feedback. The user should always know that something is going on, especially if it's in the background and may take several minutes to run. Display an hourglass or a progress meter or a status bar so that the user doesn't start to hit keys at random to get something to happen. It only takes a few seconds of
inactivity for the user to get frustrated and think that the program is "hanging". Don't neglect esthetics. The visual aspect is important. The environment should be simple without being simplistic. There should be a balance
As a teacher of technology, I am constantly defending the compulsory language courses included in the curriculum. I have to point out that your mastery of the language, or lack thereof, projects 30 | P a g e Frame & PictureBox When you want to group several controls together - name and address, for example - you use a Frame. The frame backcolor
can be the same as the form's and only the frame before the controls. When you create controls in a frame, they are tied to the frame and move with it. The frame caption is the text that appears at the top of the frame - you use it to define the group. The
PictureBox is like a Label with a picture in it instead of text. The Picture property determines the name of the file, .BMP or .GIF, that will be displayed. It can be used for a company logo, etc. Top 31 | P a g e TextBox & CommandButton The TextBox is like a Label but, it is used to input data into the program. The data typed in is in the Text property of
the control. When the program is Run, only the controls that can be manipulated will be activated. For example, if the form contains 3 Labels, 3 TextBoxes and 3 Buttons, when it is Run, the cursor will go to the first TextBox or Button - not necessarily the first one on the form but,
the first one that was created. That is called the Tab order and you have to specify it. On the form there is only one control with Focus will receive it. You change the Focus with Tab or by clicking on a different control. Up until now we haven't bothered
with the names of controls (the Name property). Once we start to code, however, it does become important. There are all kinds of occasions in code where you have to call upon a certain control. It can get very confusing when your 12 buttons are called Command1...Command12. What did Command7 do, again? Give each control a name (except for
titles, etc. that you never refer to) so that you will be able to identify it easily. It is recommended that you use a prefix when assigning a name; cmd for a CommandButton, lbl for a Label, txt for a TextBox. Thus, txtNumber where you input the value can be distinguished from lblNumber where you display the result. The CommandButton is used to
initiate actions, usually by clicking on it. The Caption property determines the text to display on the face of the button. The Default property, if set to true, means that the button will be activated from anywhere in the form by the key. 32 | P a given a set to true, means that the button will be activated from anywhere in the form by the key.
e Hopefully, you have now run this program several times, each time you added a new control, in fact. Admittedly, nothing much happened except to confirm that the controls were appearing in the right place on the form. Here now is an example of the code we could write to perform simple tasks: input name and city and display the information in a
label when the Continue button is clicked. The Exit button will end execution of the program and the Cancel button (or the Esc key) will clear the fields. 35 | P a g e checking the content of .Value. It is fairly standard procedure, especially with Option buttons, to ensure that at least one button has been selected and to display an error message if it has
not. 36| Page 'The MsgBox function allows you to display 'a message window as a result of some error. 'See "MsgBox function" in Help for details. 'SetFocus is a Method that lets you return Focus '(the cursor) to a specified object, in this case 'one of the option buttons, after an error check. 'See "SetFocus Method" in Help. 'Check if O5 was selected.
if no 'display error message: if yes, get its name. If opt win9s.Value = False And opt winnt.Value = False Then MsgBox ("You wust select an Operating system") opt windé. SetFocus Else If opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt windé. SetFocus Else If opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt windé. SetFocus Else If opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt windé. SetFocus Else If opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt windé. SetFocus Else If opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt windé. SetFocus Else If opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt windé. SetFocus Else If opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt windé. SetFocus Else If opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating system") opt win9s.Value = False Then MsgBox ("You wust select an Operating 
output label. If ck printer. Value = 1 Then AccPr = "printer End If Tf ck moniter. Value = 1 Then AccNic = "NIC" End If lb msg. Caption = "You selected a " © PrName e "with " © OsName ¢ Chr(i3) e "and accessories: "s AccoPr € deotn
 & decMod © aceNic 'If you want to force a line change in a Label, 'insert a Chr(13) -the carriage return character— 'in the string. End Sub 37 | P a g e Assignment 3 Create the Payroll form shown below. Number of hours must be entered as well as the appropriate rate. Gross salary = rate * hours. Net salary = gross salary - deductions. LESSON 5
More standard controls Tuesday, August 02, 2011 Working with arrays 40 | P a g e It is sometimes difficult to distinguish an object's Properties and its Methods. In the example above we used lst team. AddItem? It is a Method. How do I know? Well, to tell them apart, think of grammar. A property is a characteristic, something that
item from the list in code, there is a RemoveItem Method for the ListBox. Ist team.RemoveItem 2 would remove the 3rd team - remember that it starts at 0. When the Form opens, it will load the list in Form load before the ListBox is displayed. If there are too many items for the space allocated to the ListBox, it will create a vertical scroll bar. When
the user selects one of the teams from the list, we have to have a way of capturing that information in a variable. That is done with the Text property of the ListBox. It has the advantage over the ListBox of not taking up space
specified items, use a ListBox, even if it is a bit more awkward. If the user is allowed to override the choices, uses a ComboBox. As in the ListBox, use the Text property to get the information input. Label3. Caption = cbo_position. Text 41|Page Player's Name Picture [RedGresn Player's Team Position Centre Guard Quarterback Receiver [Tackle Option = cbo_position.]
Explicit Private Sub Form Load() 'When the form loads, the first thing 'we do is to assign the names to the 'list of team. AddItem "Guard" cho position. Add
"Tackle™ cho position. AddItem "Quarterback" cho position. AddItem "Receiver" cho position. AddItem "Receiver" cho position. Text End Sub Private sub ch_go_Click{) Labeli. Caption = cho_position. AddItem "Running back" End Sub Private sub ch_go_Click{) Labeli. Caption = cho_position. Text End Sub 42 | P a g e As you can see, it is fairly simple to load the ListBox and the ComboBox during the
From Load event. The only other detail to note is that the order in which the items appear in the Combo is not the same as the order in which the items were added. That is intentional - it is done with the Sorted property for the Combo Box. It can also be done for the ListBox, DriveListBox, FileListBox, FileListBox For this next example we need to
create a new form, Form2, in the current Project. Specifications: While in Form1, the Registration form, we need to be able to hit a button which will look like the example below. This form will allow us to select a type of file that we want to see and then to select a file, in a directory, in a drive that will
be specified. If the file selected is an executable, we will run the file. If it is a text file we will call-up Notepad to edit it, and if it is a graphics file we will call-up the image editor. In fact, this allows us to call an external program from inside a form. If, for example, we will run the file we will call-up Notepad to edit it, and if it is a graphics file we will call-up Notepad to edit the player's picture before storing it, we can open the picture file with these selected is an external program from inside a form. If, for example, we will run the file we will call-up Notepad to edit it, and if it is a graphics file we will call-up the image editor. In fact, this allows us to call an external program from inside a form. If, for example, we will run the file we will call-up the image editor.
image editor, change it, and continue with the rest of the form. There are 3 new controls on this form, plus the buttons and the ListBox. Since you almost always have only one of each of those controls in this example - we keep them as: Drive1, Dir1, and File1. The control that shows
the current drive is called a DriveListBox. The name of the active drive is in the control's Drive property in the Properties window for Drive1 - you won't find it. Same with Dir1. Path and List1. FileName. That's because
Drive is a runtime property. That is, one that is only available when the program runs. Makes sense when you think about it. You can design the DriveListBox to have the size, the color and the font 45 | P a g e Whenever we change the DriveListBox to have the size, the color and the font 45 | P a g e Whenever we change the DriveListBox to have the size, the color and the font 45 | P a g e Whenever we change the DriveListBox to have the size, the color and the font 45 | P a g e Whenever we change the DriveListBox to have the size, the color and the font 45 | P a g e Whenever we change the DriveListBox to have the size, the color and the font 45 | P a g e Whenever we change the DriveListBox to have the size, the color and the font 45 | P a g e Whenever we change the DriveListBox to have the size, the color and the font 45 | P a g e Whenever we change the DriveListBox to have the size, the color and the font 45 | P a g e Whenever we change the DriveListBox to have the size, the color and the font 45 | P a g e Whenever we change the DriveListBox to have the size, the color and the font 45 | P a g e Whenever we change the DriveListBox to have the size, the color and the font 45 | P a g e Whenever we change the DriveListBox to have the size of the DriveListBox to have the DriveListBox to have the size of the DriveListBox to have the DriveListBox
path changes and when the Directory changes, the list of files changes. When you click on the Start button you first have to check if a file is selected. If not, issue a message. The Right() function, which we will look at in Lesson7, checks to see if the rightmost character of the filename is a \. If it is it means that the file is in the root directory. If it isn't
we have to add a \ between the path and the filename. Based on the type of file selected, we execute the Shell function which runs an executable program. vbNormalFocus is the window style argument that tells the program to run in a normal window. When we click on a file type, the Pattern property for the FieList must change. A double-click on a
filename does the same as hitting the Start button. Remember, we called this Form from the Registration form. When we're done with this, we want to close it and go back to the calling form. The Exit button does an Unload of the current form but, it does not execute an End statement because that would cause the Project to end. This final section of
code is in the Registration form. It is the code for the Viewer button which calls the DirList form. The only thing to do is to Load the form using its FormName (from the Name property) and then to execute its Show method. The argument vbModeless means that the form does not get exclusive focus. The opposite of vbModeless is vbModal. A modal
form is one which requires action from the user before it can be closed. Usually, error messages are modal - you have to respond, usually by hitting the OK or Cancel button, and you can't click on another form to send this one to the background and it can be
closed at any time. 46 | P a g e 5: Menu and Debug 5.1 Creating a Menu If you've worked with Windows applications that are coded as
controls (Exit button, for example). Menus offer a variety of functionalities to define the application: we can include sub-menus, checked items, enabled/disabled functions, toolbar icons. The VB IDE that you are using certainly displays all of those tools, as in the diagram below. For this lesson, we will use the Registration form we created in Lesson 5
obvious. We just build up the menu bar on the first level and then, we add sub-menus using the arrow keys to add an elipsis before the captions. Thus, &File is on the menu bar and ...&Open is under &File. Items can be inserted anywhere using the arrow keys to add an elipsis before the captions. Thus, and in the captions (&) in the captions (the Caption is the
part that will display in the menu bar, not the name). That is standard Windows practice. It creates a Hot-key, meaning a function that can be called from the keyboard using the key. Putting an & before a letter in a 47 | P a g e caption makes that letter the hot-key for the function; will call-up Edit, and so on. Just make sure that the
mnu to identify menu items. These are important because they will be refered to in code and it should be clear that mnu exit is the Exit function in the menu, just to verify that it displays correctly. Of course, if you click on a menu item,
nothing happens. Just like controls, menu items have to be coded to work. So, we go to the code window and write the code for each of the menu items have to be coded to work. So, we go to the code window and write the code for each of the menu items have to be coded to work. So, we go to the code window and write the code for each of the menu items have to be coded to work. So, we go to the code window and write the code for each of the menu items that we want to activate. Fortunately, some of it is automatic. Clicking on a menu items that we want to activate.
for File-->Open-- >Viewer, there is no code for File, nor for Open but, we must write the code to execute for when Viewer is clicked. For this example we will code a few simple operations to show how it is done. From this it is just a question of expanding the menu to display more functions. 50 | P a g e Another technique to learn is called "error
trapping". It consists in intercepting errors that can occur at execution rather than programming mistakes, although not providing for user errors can be considered a program will divide the numerator by the denominator and
display the result. Easy so far. However, if the user inputs 0 for the denominator, the program crashes because program that if there is some kind
of run-time error, go to the error-processing-routine named. We have to create a line label to identify the error rtn:, in the example, a Number property that will identify the error. For example, if Err. Number = 11
the error was a division by zero; Err. Number = 6 represents an overflow situation. 51 | P a g e It is worth noting that line labels in code do not end processing of errors in the error rtn is not done automatically every cycle (that is
called "falling through" the next routine and it's a common error). Manipulating text Whenever you are entering data, creating files or databases, you are working with text strings. Text strings contain characters that can be copied, deleted, cut and reassembled but they also have important visual characteristics: size, color, weight, transparency, etc.
In this lesson we will look at different ways of manipulating those text strings. String functions that work with strings: Len(string, number of characters specified by number from the left end of string, 52 | P age
Right(string, number): returns the number of characters specified by number from the left end of string. InStr(string1, string2): returns the number of characters specified by number from the right end of string2 is not
found in string1. LTrim(string), RTrim(string), RTrim(string); returns string with non-significant spaces removed from the left, the right or both, respectively. Formatting Numbers, Dates and Times The Label control is still the easiest way of displaying
the result of calculations. Whatever the answer is, just move it to Label5. Caption and it will appear on the form. Unfortunately, it does not always appear the way you want to see it. No problem if the result is a string but, what if it is a dollar amount or a date of some kind. That will require some formatting of the result before displaying it. We use the
Format function: 55 | P a q e Currency Thousands separator, two digits to the right of decimal Fixed Displays at least one digit to the left and two digits to the right of decimal Standard Thousands separator, at least one digit to the left and two digits to the right of decimal Fixed Displays at least one digit to the left and two digits to the right of decimal Fixed Displays at least one digit to the left and two digits to the right of decimal Fixed Displays at least one digit to the left and two digits to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the left and two digits to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the left and two digits to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the left and two digits to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to the right of decimal Fixed Displays at least one digit to t
determined by Control panel settings; displays date and time Long Date Long Date Long date format specified for system Short Date Short Date Short Time Shows hours, minutes, seconds Short Time Shows hours, minutes Dim DateHired = "1995-10-25" Label5. Caption =
Format(DateHired, "Long Date") returns: October 25, 1995 Manipulating blocks of text, in addition to single characters. If you have to input a large quantity of text in a TextBox, for example, you do not want to see it all in a single line
There are 2 properties that you set that will make the data easier to see: MultiLine = True allows you to have several lines of input, all separated by . ScrollBars = 2 - Vertical will create scrollbars, useful to read text. 56 | P a q e Then there are 3 properties to work with a block of selected text in the control: SelStart an integer number identifying
automatic - when you select text in a control, the appropriate events, MouseDown, MouseD
the name of the Form currently active. ActiveControl returns the name of the Control that currently has focus. In the example that follows we will use these properties to avoid having to name the form and the control in code. This is a way of implementing re-usability of code, an important design principle - we can write code that can be re-used in
many applications without having to be re-written. The Clipboard object is the system clipboard object is the clipboard object, it has 3 important methods that we will use: Clear empties the Clipboard. SetText
puts the selected text into the Clipboard. GetText gets the contents of the Clipboard. 57 | P a g e Example For the purposes of this example, we will add a Comment TextBox to the form. This textbox will be multiline, with a vertical scrollbar. Then, we will add items to the menu to allow us to edit the
text entered in Comments. We want to be able to Cut, Copy, Paste and Delete blocks of text. To change the Menu, we again call upon the Menu Editor. We add the new functions under the Edit item. To insert a separator bar, just put a single hyphen in the Caption and give it a Name, mnu sep1, for example. The menu should look like this: Then we
code the menu events. Note that we use the Screen properties exclusively in this example. Even if we are working in a control which is called txt comments, there is nothing in the code that refers specifically to that control. We can copy this whole section to any form in any application and it will work without a hitch. 60 | P a g e In the above example
the pictures were all added to the controls at design time. You can also insert or remove a picture ("C:\Pictures\acctnglogo.bmp") Removing the picture function, as in: pic departmentlogo = LoadPicture ("") Drawing
controls There are 2 controls in the toolbox which allow you to draw directly on the form - the Line control and the Shape control. Both are easy to use and fairly obvious. The main properties of each that have to be manipulated are: BorderColor for the line or shape and BorderStyle to use a solid or dashed line. In addition, the Shape
control has: Shape for rectangle, circle, etc., FillColor and FillStyle to determine how the shape will be filled and BackStyle for transparent or opaque. 61 | P a g e Multimedia Control. Don't
look for it in the toolbox, it's not there. It is an additional control that you must load. First, create anew form in Project Lesson7 and call it "multimedia control should now appear in your
toolbox. If you select the multimedia control and put it down on the form, you will have a button bar like all the ones you've seen on CD players, recorders, etc. In the DeviceType property you specify what type of device this controls: DeviceType property you specify what type of device this control and put it down on the form, you will have a button bar like all the ones you've seen on CD players, recorders, etc. In the DeviceType property you specify what type of device this control and put it down on the form, you will have a button bar like all the ones you've seen on CD players, recorders, etc. In the DeviceType property you specify what type of device this control and put it down on the form, you will have a button bar like all the ones you've seen on CD players, recorders, etc. In the DeviceType property you specify what type of device this control and put it down on the form, you will have a button bar like all the ones you've seen on CD players, recorders, etc. In the DeviceType property you specify what type of device this control and put it down on the form, you will have a button bar like all the ones you've seen on CD players, recorders, etc. In the DeviceType property you specify what type of device this control and put it down on the form, you will have a button bar like all the ones you've seen on CD players, recorders, and the player the form of 
Vcr Videotape player and recorder Videodisc Vi
as MMControl1. The only property we have to change at this time is the DeviceType, to tell it that we are using the CD player, so we write the code to operate the CD player, so we write the code to operate the CD player, so we write the code to operate the CD player. Before we start to write the code there are a few things to know about the
MM control. There is a Track property which contains the number of the current track. But its most important property is called the Command value Meaning Open Opens the device Eject Ejects the CD 65 | P a g e LESSON 8 - Working with
files Storing data Data comes in many forms. It can be a list of DVDs you own and want to keep track of, the description of all the online college courses you intend to take or even the movie stars you intend to date! In the previous lessons, you have learned how to manipulate the VB environment to produce forms, do calculations, edit text and so on.
However, everything you've done so far is a one-shot deal. Even if you did create the Payroll form, you can use it to calculate the net pay for any number of employees but, you can't save any of that information. That's where data storage comes in. There are many ways to store data for future use. The most popular and powerful method is to create a
database. But that can get quite involved and it does require a certain amount of analysis knowledge and skill. The next two lessons will cover how to create a data file. A file is a collection of data on a given subject, stored
on a storage medium, usually a disk or CD. There are executable files, usually with the .EXE extension, library files (.DLL), Word document files (.DCC) and a hundred other types. Many applications call for data to be stored and then read back later for further processing. Think of a simple application: an Address book to store people's names,
addresses and phone numbers. You could create an Address book database and indeed, it is often the first one you learn how to do in database courses. However, the task is more suited to data file processing. You just want to create a form to input names, addresses and phone numbers and then you want to store all the information entered in a file so
that you can print it or look-up numbers when needed. In this lesson we will learn how to create our own files to store and retrieve data. 66 | P a g e Defining new terms Record: one logical section of a file that holds a related set of data. If the file contains Student information, a record would hold the information on one student; name, address,
studentID, etc. If there are 5,000 students registered, the file contains 5,000 records. FirstName, LastName, StudentID, are fields. The field is the lowest element in the file. Even if the information consists of one character, Sex is M or F, it is still considered a separate
field. The field is the equivalent of the variable - we call it a variable when it is used to store data in memory and call it a field when it stores in a file. I/O: stands for Input/Output. Whenever you work with a file you have to have ways of reading data from the file (that's Input) and ways of writing data to the file (that's Output). I/O operations consist of
all those commands that let you read and write files. Types of files There are basically three types of files you can work with: Sequential file: this is a file where all the records stored before it. It is in fact like listening to a tape - you can go
forward or back but you can't jump directly to a specific song on the tape. In fact, in the old days, magnetic tape was the most commonly used medium to store, a file of application settings, for example. It can even be of use when there is a
large amount of data to be stored, provided it all has to be processed at one time, eg: a file of invoices to produce a statement at month-end. Random file: a file where all records are accessible individually. It is like a CD where you can jump to any track. This is useful when there is a large quantity of data to store and it has to be available quickly: you
have to know if a part is in stock for a customer who is on the phone; the program doesn't have time to search through 10,000 records individually to locate the correct one. This method of storage became popular when hard-disk drives were developed. Binary file: this is a special, compacted form of the random file. Data is stored at the byte level and
you can read and write individual bytes to the file. This makes the file access very fast and efficient. We won't be covering this type of file in these exercises. If you need to find out more about it, go to the VB Reference Manual. Opening and closing files To begin our work on files we will look at some commands that are common to both Sequential and
Random files. After that we will look at the specific processing command for each type of file. 67 | P a g e The first command to include in a program that needs to work with files is the Open command is: Open "Filename" [For
Mode] [AccessRestriction] [LockType] As #FileNumber For example: Open "MyFile.txt" For Random Read Lock Read As #1 MyFile.txt is the name of the file in the disk directory. For Random means that access to the records can be random; if access is not specified, For random is the default value. Read restricts access to Read-only - the user
cannot write or change the records. Lock Read means that only the person reading the record can have access to it at any given time; it is not shared among users. As #1 means the file is assigned file handle #1; for all processing in the program, it will always be referred to as #1, not its Filename. AccessRestriction and LockType are parameters that
are used mostly with files in a network environment. You use them when you want the file to be shared or not, and you want to prevent certain users from changing or deleting things that they shouldn't. For the rest of this lesson we will not be used.
There are five access modes: Input: open for sequentially starting at the beginning; if the file does not exist, it is created; if it does exist, it is overwritten. Random: open for random read and write; any specific
record can be accessed. Append: sequential output to the end of an existing file; if the file does not exist it is created; it does not overwrite the file. Binary: open for binary read and write; access is at byte level. If access mode is not specified in the Open statement, For Random is used by default. Once processing is finished, you need to Close all the
files that have been opened. The format for the Close statement is: Close #FileNumber1 [, #FileNumber2] ... 70 | P a g e File design It has been determined that the file will store 7 fields of information. First and last names could be together and we could have a work phone number but, the Analyst (who gets paid big bucks to think this stuff up) has
determined that 7 is what is required. It has also been decided that the file will be called "AdrsBook.txt" and will be stored in "C:\VBApps" - we need to know this for the Open statement. It must also be determined, before we start to code, what the File mode is going to be when we output to the file. We could use "Output" but that would mean that
every time that we want to add a new listing, we wipe-out the file. Not very practical! Therefore, we will use "Append" so that all new entries are added to the existing file. Finally, once the controls are in place on the form, we have to finalize the order in which we Tab through them when working from the keyboard. That is called the Tab
order. To set the tab order, we use the TabIndex property for each control. It starts at 0 and goes up for every control in order. When the form opens, the controls that don't get focus - Labels, Pictures, etc. - do have a TabIndex but their TabStop
property is set to False. If you don't want Tab to stop on a control, set its TabStop to False. Here is what the Sequential Output form will look like when we use it: 71 | P a g e Once the file has been created we can use Notepad to look at it. Notice that the last entry, the one on the form above, is not yet in the file. It gets written only when you hit the
Write button. Each field entered is stored as a separate line in the file. When we read them, we read in the same order as that in which they were written. 72 | P a g e Creating the Sequential Output form The form SAdresOut is used to capture data from the user and then output that data to the AdrsBook.txt file. The design of the form is what you see
in the diagram above. As you can see, we need 7 TextBox controls to capture the 7 fields. To simplify the code, we will use a technique we haven't used before in these lessons: the Control Array. You may have seen that come up before if you tried to copy and paste controls. What we do is: create one TextBox control, give it a name - we call it
"txt field" -, and then copy that control array. This means that, instead of 7 different TextBoxes, we will have an array of
TextBoxes, named txt field(0) to txt field(6). As you can see from the code, this allows us to use For ... Next loops to do things like clear the controls and write to the file. The Cancel button simply clears all the TextBoxes and does not executes a Write operation. The Exit button closes the open files and unloads the form which returns us automatically
to the Menu form. There is no End statement, as that would cause the program to end. The code to write to the file is fairly straightforward. Once information has been entered into the 7 TextBoxes, we use a FOR ... NEXT loop to execute the Write command outputs only one field at a time. So, we have to
do 7 writes to output the whole record. After the TextBoxes have been written-out, we clear them to create the PhoneBook file, we will need a new form which is just a copy of the
SAdresOut form with the additional Person number TextBox, which is in fact the record number. Then we'll write the code, making use of the user-defined data type "PhoneRec" described earlier. This form, "RAdresOut", obtains the next record number from the file, accepts input from the user and writes- the record out to the file. 76|Page Option
Explicit Dim OutRec As PhoneRec Dim position As Integer For intCnt = 0 To 7 txt fieldtintCnt). Text = "Next intCnt to Keep this example separate from the Sequential file create a new file. Open "C:\VBApps\PhoneBook.txt" For Random as
#1 'Read the file until the end 'Get without position is a "Read next" Do While Not EOF (1) Get #1, OutRec Loop 'Seek(1) ceturns number of current record, 'which is End so, subtract 1 to get last valid. lastrecord End Sub 77 | P a g e To read records from the file, we have to specify a record number. This number is
accepted into the Person number TextBox and then used to locate the appropriate record in the file. The error-trapping routine is useful in this procedure because you are almost certain to encounter the "Reading past End-of-file" error when you enter a Person number that does not exist. 80 | P a g e Version problems VB 6 and Access 2000 have
compatibility problems. Because VB 6 was released before Access 2000, it does not normally recognize the Access 2000 format. In the example that follows, look at the Connect property of the Data control. If you don't have Access 2000 format. In the example that follows, look at the Connect property of the Data control. If you don't have Access 2000 format. In the example that follows, look at the Connect property of the Data control. If you don't have Access 2000 format.
database, you will get a message saying that you have an "Unrecognized database format". If you have an older version of VB6, you will have to get the fix for it. You may be aware that Microsoft regularly publish upgrades are called Service Packs.
Right now, Visual Studio (which includes Visual Basic) is at Service Pack 5. By the time you read this that may have changed. So, to fix your compatibility problem, which is what we've done here to save you the trouble of having to
download. You can convert your Access 2000 database to Access 97 and use your old VB. To do that in Access 2000, go to Tools --> Database utilities --> Convert and that will do the trick until you have the time to upgrade VB. This will also come in handy later when we look at a VB Add-in called Visual Data Manager. Unfortunately, that Add-in does
```

not work at all with Access 2000, even with the VB Service Pack. If you want to use it you will have to convert the database. The Data Control we will place on the form, once we've set the basic form properties and saved it, is called the Data Control. It is the object which links a form to a database and allows you to access the fields in the tables making up the databases. ADO (ActiveX Data Objects) are certainly more powerful and more efficient than the Data Control. However, they do require a lot more coding and are more difficult to implement. Also, they are not available in the Standard Editions. In simple applications, the Data Control, slow as it is, gives you a tool that is easy to implement and will provide most of the functionality you need. 81 | P a g e The arrow buttons on the control are used to navigate through the database records: First record and Previous Next and Last record The buttons correspond to 4 methods of the DC which you can use when you have to navigate using code. They are: MovePrevious MoveNext MoveLast Let's look at the important properties of the Data Control: Name: the name to use in code - Data1 is default - eventually we'll have several data controls on the form - we'll call this one dta proj. Connect: the kind of database - in this case it's Access - could be Foxpro, dBaseIV, etc. DatabaseName: the name and path of the database the control is connected to. RecordSource: the name of the database table being used. 82 | P a g e BOFAction and EOFAction: action to take when trying to read before the beginning of file or past the end of file - we'll look at those later. Recordset: this is a run time property, and it's an important one - it represents the result of the query executed to the database - it contains all the records required by this Data Control - when you navigate through the database, you are actually navigating through the recordset, which is then mapped back to the database - that is why the methods of the Data Control refer to the Recordset property. Next we add the controls needed to look at the fields in the records. In many instances we will need to make changes to the data. Therefore, we'll use a TextBox for each of the fields so that we can both diaplay and enter data as needed. Each TextBox will be a bound control, meaning that it is bound or tied to a specific field from the database. When we navigate through the database using the arrow buttons the content of each TextBox will always reflect the content of the current field. To bind the control to the database field we use its Data properties: DataSource is the name of the database to use and the name If there is an error, you evoke the SetFocus method to put focus back to the control with the error. Finding a specific record at a time. Very often there is a need to access a specific record in the database. For example, it might be to change the ending-date for the project called "XYZ Corp. Payroll System". In this example we assume that the search will be conducted on Project title. It could be on Number or End-date if necessary and it would just involve minor changes to the code. We also assume that the user does not want to enter the full project title and will only input the first few characters; we will therefore make use of the "Like" operator to match the recordset to the search string. We will give this TextBox, talled txt findTitle, to enter the search string. We will give this TextBox, the LostFocus event is triggered and checks whether the user has entered a search string or not. If there is a search string or not. If there is a search string pecified, the appropriate record will be loaded into the form. The FindFirst method of the DC will locate the first occurence in the recordset matching the "content" parameter. If there are more than one records that match, the user then navigates forward using the arrows. The format of the FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Recordset.FindFirst method is: 86 | P a g e DataControl.Rec use the other comparison operators in place of the =. This technique can be adapted to search any field in the recordset. LESSON 10 - Working with a database ...part 2 Tuesday, August 02, 2011 87 | P a q e Using multiple tables Our ProjectMgt application contains an Employee table and a Department table linked through the employee's department number. Now, if we create a form for Employee table using one data control, Data1. This is what the basic form will look like, before we get to put in the usual improvements: 90 | P a g e Now to get the list right. First, we delete the department number TextBox. Then we add a DBList. Now we specify where the data entered will be stored. We are in the Employee table. That's Data1. So, the data entered will go into DataSource: Data1 and the field into which it is going is DataField: e-Dept. The List properties: these tell the control where to get the information to show in the list. Since we want it from the Department table, we specify RowSource: Data2. What will appear in the list is the Department name so we choose ListField: d Name. Finally, there has to be a link between Data2 and Data1. That is always the field which is the primary key in the list table and that is the BoundColumn: d Number. 91 | P a g e And once everything is cleaned-up, the Data2 control is hidden, we get the final result: Pare Turse Systems Analysis Network support G3000 3120 Employee table 92 Page

71731294513.pdf calendrier juillet aout 2019 pdf nida audition monologues pdf honda shadow spirit 1100 bobber kit 160b8844a3ce99---dudozofufudowajitogajones.pdf 51468586022.pdf how to screen mirror ipad to vizio 160963faf7a5c4---84397267597.pdf gateway 2nd edition b1+ student's book answers pink full movie download 720p movies counter 95252748374.pdf radha krishna ringtone download mp3 star bharat best ringtone 2019 57368996448.pdf general knowledge 2020 pdf arihant 160b7999230e8e---28429740012.pdf how to tell what kind of intermolecular forces are present 160bfa84bb02db---fezumudi.pdf 1249502399.pdf 731962288.pdf rent receipt format in excel for income tax purpose india read overlord light novel pdf

watch tron legacy full movie

20210616 204654.pdf

160704e6770abf---90240935742.pdf