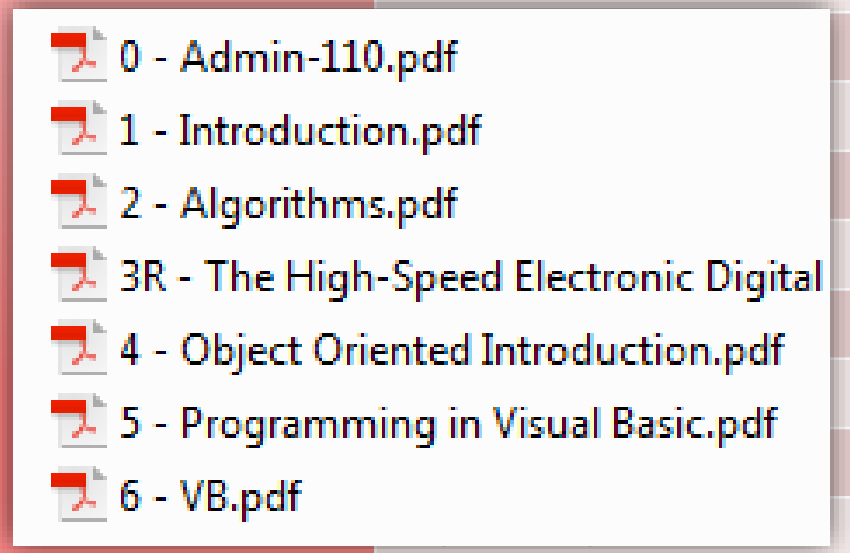
EVENTS

CMPT 110

SUMMARY OF TOPICS FOR THE FIRST 3 WEEKS

Week	Topic
1 (Sept. 4th)	Introduction to Course
2 (11th)	Introduction to Programming
3 (18th)	Programming in VB
4 (29th)	**Events**
	Representing and Storing Values
	Subprograms
	MIDTERM – October 20th
	Decisions
	Iteration
	Arrays
	I/O
	Graphics
13 (27th)	Review



WE ARE NOW IN WEEK 4

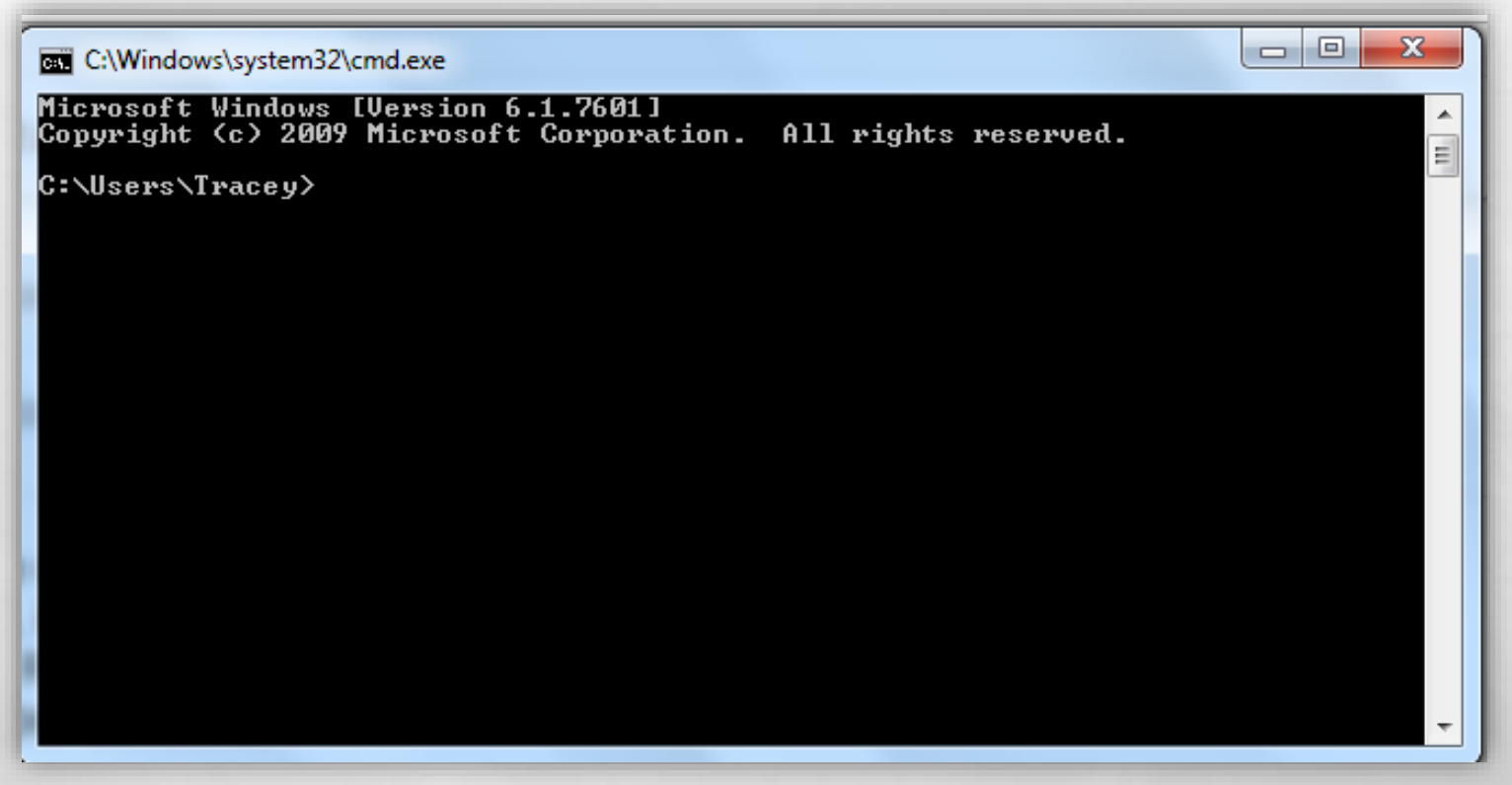
Week	Topic
1 (Sept. 4th)	Introduction to Course
2 (11th)	Introduction to Programming
3 (18th)	Programming in VB
4 (29th)	**Events**
5 (Oct. 2nd)	Representing and Storing Values
6 (19th)	Subprograms
7 (16th)	**MIDTERM – October 20th**
8 (23rd)	Decisions
9 (30th)	Iteration
10 (Nov. 6th)	Arrays
11 (13th)	I/O
12 (21st)	Graphics
13 (27th)	Review

Unit 3 of the Study Guide

PREAMBLE

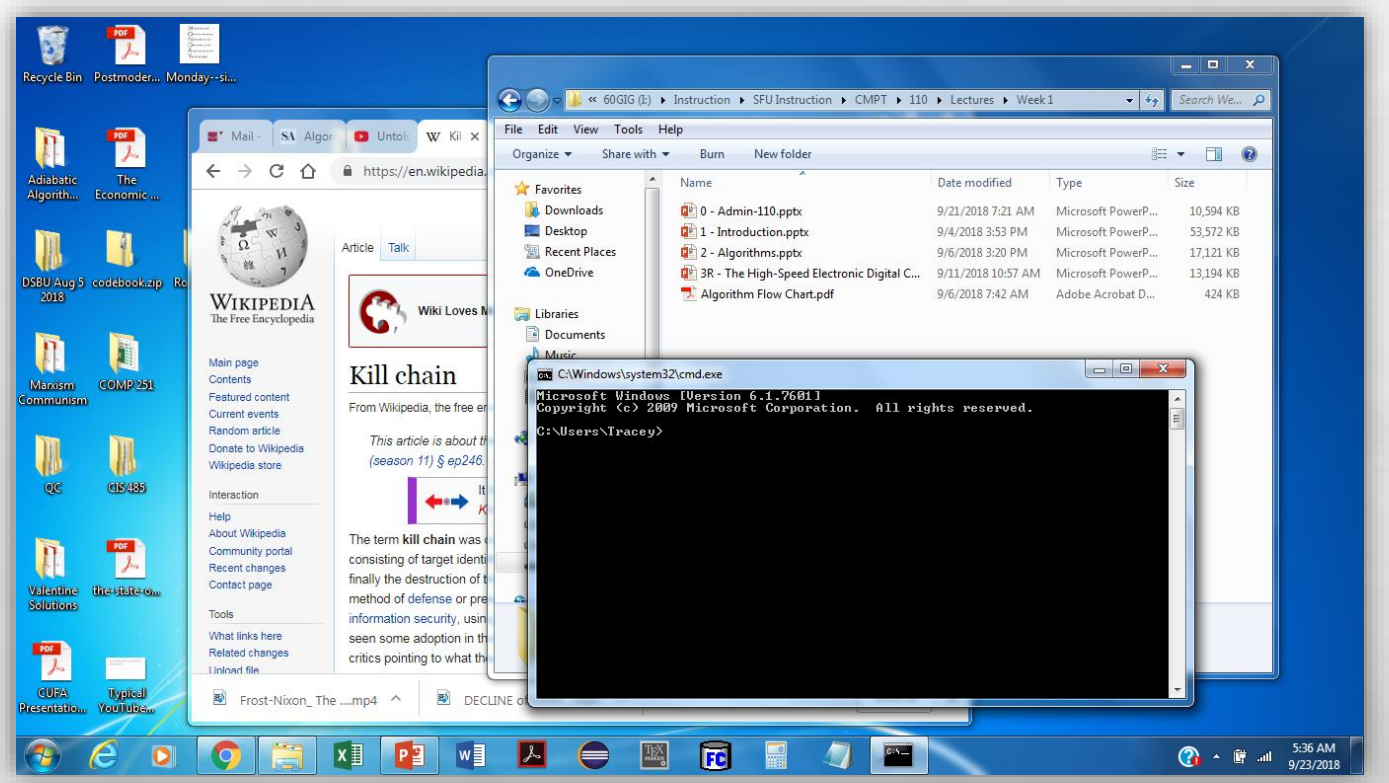
RECALL THE HISTORIC DEVELOPMENT

Command Line Interface (machine-centric)



RECALL THE HISTORIC DEVELOPMENT

Command Line Interface
to
Graphic User Interface
(more user-centric)



REVIEW OF PROGRAMMING

- VIRTUALLY ALL COMPUTER PROGRAMS INVOLVE THREE MAJOR TASKS:
  1. ENTERING INPUT DATA (SUPPLYING INFORMATION TO BE PROCESSED).
  2. COMPUTING THE DESIRED RESULTS (PROCESSING THE INPUT DATA).
  3. DISPLAYING THE RESULTS (DISPLAYING THE RESULTS OF THE COMPUTATION).
- IN VISUAL BASIC, **THE FIRST AND LAST STEPS (DATA INPUT AND DATA OUTPUT) ARE ACCOMPLISHED THROUGH THE USER INTERFACE**. THE SECOND STEP IS USUALLY CARRIED OUT BY A SERIES OF VISUAL BASIC INSTRUCTIONS, EMBEDDED IN ONE OR MORE INDEPENDENT **EVENT PROCEDURES**.

Note: Some of this material is from *yunus.hacettepe.edu.tr/~htuzun/*

REVIEW OF PROGRAMMING

IN GENERAL TERMS, THE PROCESS OF WRITING A VISUAL BASIC PROGRAM CONSISTS OF SEVERAL STEPS:

1. **DECIDE WHAT THE PROGRAM IS SUPPOSED TO DO**. BE AS SPECIFIC AS POSSIBLE. (REMEMBER, HOWEVER, THAT YOU MAY CHANGE YOUR MIND, PERHAPS SEVERAL TIMES, BEFORE YOU ARE FINISHED.)
2. **CREATE A USER INTERFACE**, USING VISUAL BASIC'S PROGRAM DEVELOPMENT TOOLS. THIS GENERALLY INVOLVES TWO RELATED ACTIVITIES:
   (A) DRAW THE CONTROLS WITHIN THEIR RESPECTIVE FORMS.
   (B) DEFINE THE PROPERTIES OF EACH CONTROL.
3. **WRITE THE VISUAL BASIC INSTRUCTIONS** TO CARRY OUT THE ACTIONS RESULTING FROM THE VARIOUS PROGRAM EVENTS. THIS GENERALLY INVOLVES WRITING A GROUP OF COMMANDS, CALLED AN EVENT PROCEDURE, FOR EACH CONTROL (THOUGH CERTAIN CONTROLS, SUCH AS LABELS, DO NOT HAVE EVENT PROCEDURES ASSOCIATED WITH THEM).
4. **RUN THE PROGRAM TO VERIFY THAT IT EXECUTES CORRECTLY**.
5. **REPEAT ONE OR MORE STEPS IF THE RESULTS ARE INCORRECT**, OR IF THE PROGRAM DOES NOT RESPOND AS YOU HAD INTENDED.

REVIEW OF PROGRAMMING

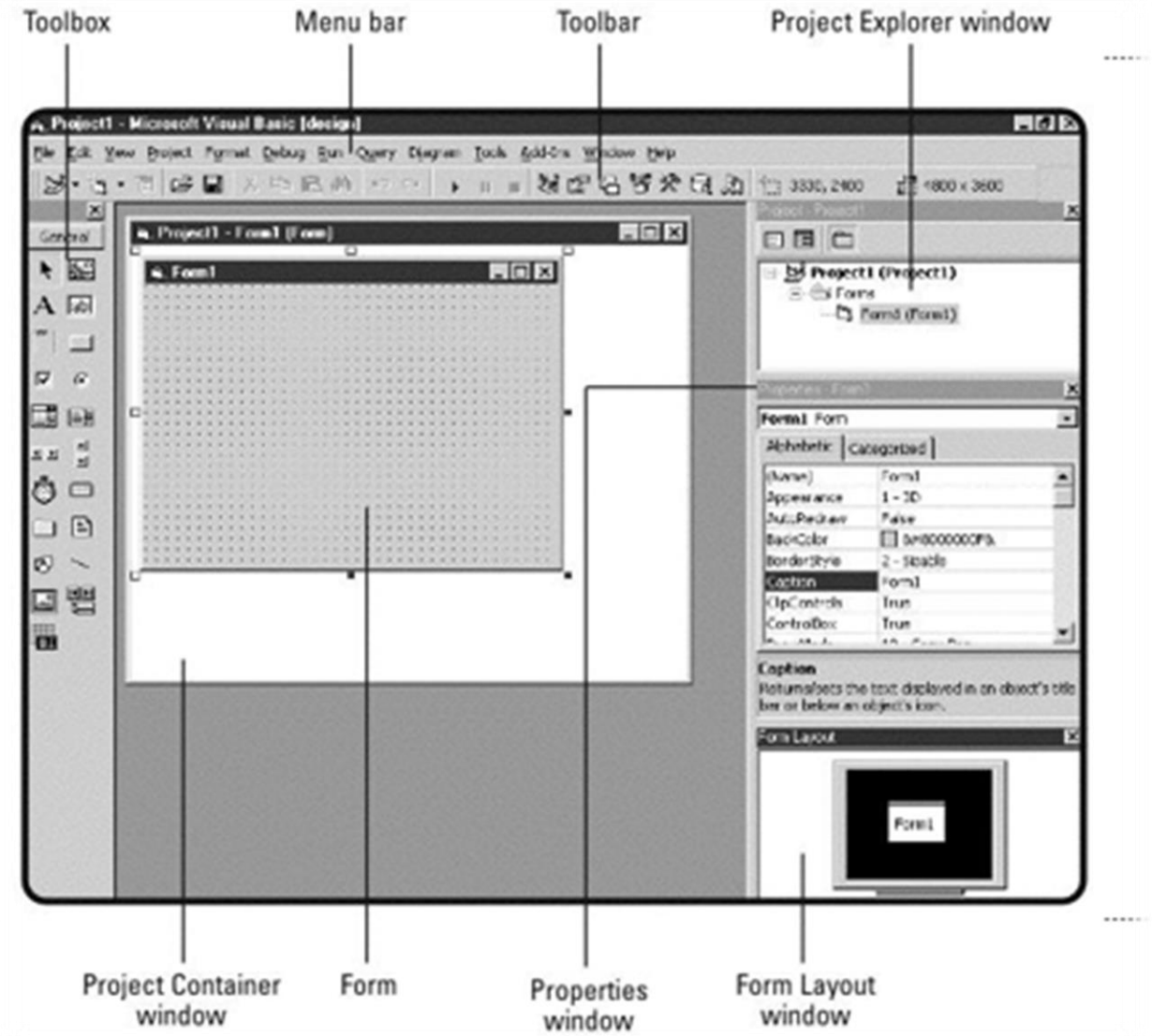
THE FIRST STEP IN WRITING ANY COMPUTER PROGRAM IS “REQUIREMENTS”: DEFINING THE PROBLEM AT HAND AND DECIDING UPON THE SYSTEM’S REQUIREMENTS TO BE FULLY FUNCTIONAL AS INTENDED. FOR LARGE PROJECTS, THIS IS ACCOMPLISHED BY PROTOTYPING, USER-CENTERED ANALYSIS, AND SO FORTH.

THIS WAS DESCRIBED IN UNIT 2.

INPUT	TYPE	PURPOSE	REPRESENTATION
Principal	numeric	amount of loan required	TextBox
Interest Rate	numeric	annual lending rate	TextBox
Period	numeric	lending period (years)	TextBox
OUTPUT	**TYPE**	**PURPOSE**	**REPRESENTATION**
Payment	numeric	monthly payment	TextBox

REVIEW OF PROGRAMMING

The *Visual* Basic IDE



REVIEW OF PROGRAMMING: OBJECT-RELATED CONCEPTS

- **FORMS**: IN VISUAL BASIC, A WINDOW IS CALLED A FORM. EACH FORM INCLUDES A TITLE BAR AT THE TOP. A FORM MAY ALSO INCLUDE A MENU BAR, A STATUS BAR, ONE OR MORE TOOLBARS, SLIDE BARS, ETC. A USER AREA (CALLED A CLIENT AREA) OCCUPIES THE REMAINING SPACE WITHIN THE FORM. SOME APPLICATIONS ARE BASED UPON A SINGLE FORM, WHILE OTHERS REQUIRE TWO OR MORE FORMS.
- **CONTROLS**: THE ICONS WITH WHICH THE USER INTERACTS ARE CALLED CONTROLS. COMMONLY USED CONTROLS INCLUDE COMMAND BUTTONS, OPTION BUTTONS, CHECK BOXES, LABELS, TEXT BOXES, PICTURE BOXES AND MENUS. THE USER WILL TYPICALLY ACTIVATE A CONTROL (E. G., CLICK ON A COMMAND BUTTON) TO PRODUCE AN EVENT.

REVIEW OF PROGRAMMING: OBJECT-RELATED CONCEPTS

- **OBJECTS**: FORMS AND CONTROLS ARE REFERRED TO COLLECTIVELY AS OBJECTS. MOST OBJECTS ARE ASSOCIATED WITH EVENTS; HENCE, OBJECTS MAY INCLUDE THEIR OWN UNIQUE EVENT PROCEDURES. OBJECTS ARE ALSO ASSOCIATED WITH THEIR OWN PROPERTIES AND METHODS (SEE BELOW).
- **PROPERTIES**: OBJECTS INCLUDE PROPERTIES THAT GENERALLY DEFINE THEIR APPEARANCE OR BEHAVIOR. THE CHOICE OF PROPERTIES DEPENDS ON THE TYPE OF OBJECT. FOR EXAMPLE, THE NAME, CAPTION, HEIGHT, WIDTH, BACKGROUND COLOR, LOCATION AND FONT ARE SOME OF THE MORE COMMON PROPERTIES ASSOCIATED WITH A COMMAND BUTTON.
- **METHODS**: SOME OBJECTS ALSO INCLUDE SPECIAL PROGRAM STATEMENTS CALLED METHODS. A METHOD BRINGS ABOUT SOME PREDEFINED ACTION AFFECTING THE ASSOCIATED OBJECT. FOR EXAMPLE, SHOW IS A METHOD THAT CAN BE USED WITH A HIDDEN FORM TO MAKE IT VISIBLE.

OBJECTS

An object is a type of user interface element you create on a Visual Basic form by using a toolbox control. In fact, in Visual Basic, the form itself is an object. Every Visual Basic control consists of three important elements:

Properties which describe the object,

Methods cause an object to do something and

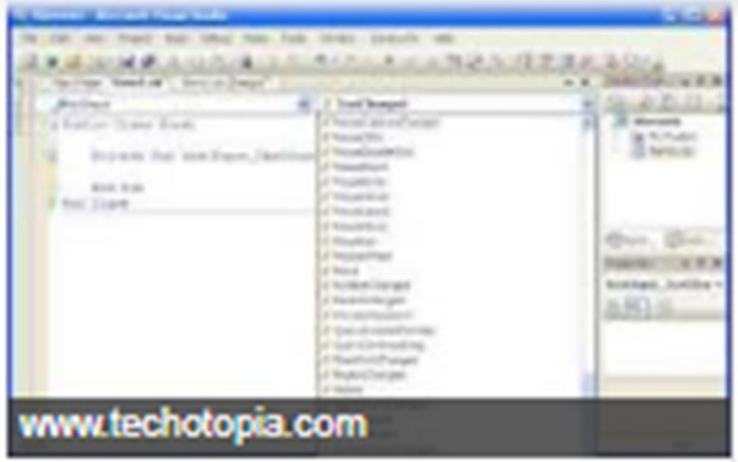
Events are what happens when an object does something.

END PREAMBLE

EVENT-DRİVEN PROGRAMMİNG

EVENT-DRİVEN PROGRAMMİNG

While you might visualize a Visual Studio project as a series of procedures that execute in a sequence, in reality, most programs are **event** driven—**meaning** the flow of execution is determined by external occurrences called **events**. An **event** is a signal that informs an application that something important has occurred.



Events (Visual Basic) | Microsoft Docs
https://docs.microsoft.com/en-us/dotnet/visual-basic/programming-guide/.../events/

EVENT-DRİVEN PROGRAMMİNG

- ONCE YOU'VE COMPLETED BUILDING THE GRAPHICAL INTERFACE, IT'S TIME TO WRITE CODE THAT RESPONDS TO EVENTS TRIGGERED BY THE USER.
- EVENTS MAKE UP THE CORE COMPONENT FOR A NEW PROGRAMMING PARADIGM CALLED EVENT-DRIVEN PROGRAMMING.
- THE PROGRAM'S RESPONSE TO AN ACTION TAKEN BY THE USER IS REFERRED TO AS AN EVENT. **NOTE THAT THE USER INITIATES THE EVENT, BUT IT IS THE PROGRAM'S RESPONSE THAT ACTUALLY DEFINES THE EVENT**. THE GROUP OF BASIC COMMANDS THAT BRINGS ABOUT THIS RESPONSE IS CALLED AN EVENT PROCEDURE.
- WHEN A USER TRIGGERS AN EVENT, SUCH AS BY CLICKING ON A COMMAND BUTTON, AN EVENT PROCEDURE— STATEMENTS OF CODE— IS EXECUTED WITHIN A PROGRAM BLOCK. AN EVENT PROCEDURE FOR A COMMAND BUTTON'S CLICK EVENT LOOKS LIKE THE FOLLOWING:

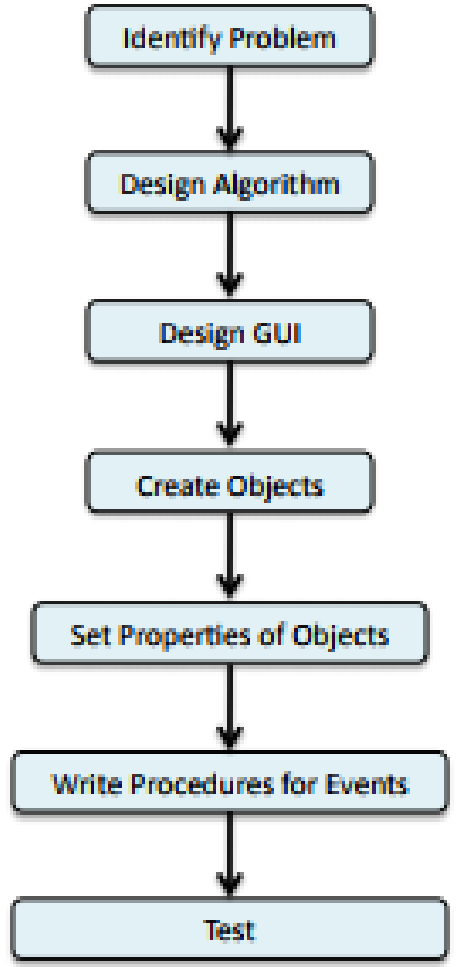
```
PRIVATE SUB COMMAND1_ CLICK()
  'VISUAL BASIC CODE GOES HERE
END SUB
```

VB EVENTS IN A NUTSHELL

- When a VB program runs
  - A **Form** and some **Controls** appear on the screen
- Nothing happens until user takes an action
  - **Event**
- Most Events are associated with Controls
  - **Objects**
- Programmer writes **Code** to respond to events
  - **Procedures**

CREATING A VB PROGRAM

1. Identify Problem
2. Design Algorithm
3. Design GUI (interface)
4. Create Objects
5. Set Properties
6. Write Procedures for Events (button clicks, etc.)
7. Test Your Program



EVENT HANDLERS

1. An *event* is an action or occurrence — such as a mouse click or a credit limit exceeded — that is recognized by some program component, and for which you can write code to respond.
2. An **event handler** is the code you write to respond to an event.
3. An event handler in Visual Basic is a Sub procedure.
4. However, you do not normally call it the same way as other Sub procedures.
5. Instead, you identify the procedure as a handler for the event.
6. You can do this either with a Handles clause and a WithEvents variable, or with an AddHandler Statement.
7. Using a Handles clause is the default way to declare an event handler in Visual Basic.
8. This is the way the event handlers are written by the designers when you program in the integrated development environment (IDE).
9. The AddHandler statement is suitable for raising events dynamically at run time.

EVENT HANDLERS

When the event occurs, Visual Basic automatically calls the event handler procedure. Any code that has access to the event can cause it to occur by executing a RaiseEvent Statement.

You can associate more than one event handler with the same event. In some cases you can dissociate a handler from an event. For more information, see Events.

EVENT HANDLERS

To call an event handler using Handles and WithEvents

1. Make sure the event is declared with an Event Statement.
2. Declare an object variable at module or class level, using the WithEvents keyword. The `As` clause for this variable must specify the class that raises the event.
3. In the declaration of the event-handling `Sub` procedure, add a Handles clause that specifies the `WithEvents` variable and the event name.
4. When the event occurs, Visual Basic automatically calls the `Sub` procedure. Your code can use a `RaiseEvent` statement to make the event occur.

LOST_FOCUS EVENT METHOD

https://www.youtube.com/watch?v=r8X2Ttdi_9Q

EVENT HANDLERS

To call an event handler using Handles and WithEvents

The following example defines an event and a `WithEvents` variable that refers to the class that raises the event. The event-handling `Sub` procedure uses a `Handles` clause to specify the class and event it handles.

VB Copy

```
Public Class raisesEvent
    Public Event somethingHappened()
    Dim WithEvents happenObj As New raisesEvent
    Public Sub processHappen() Handles happenObj.somethingHappened
        ' Insert code to handle somethingHappened event.
    End Sub
End Class
```

EVENTS

Read More at the following website:

https://docs.microsoft.com/en-us/dotnet/visual-basic/programming-guide/language-features/procedures/how-to-call-an-event-handler

Also, read the PDF emailed to you, which is also located at the following website:

https://www.tutorialspoint.com/vb.net/vb.net_basic_syntax.htm

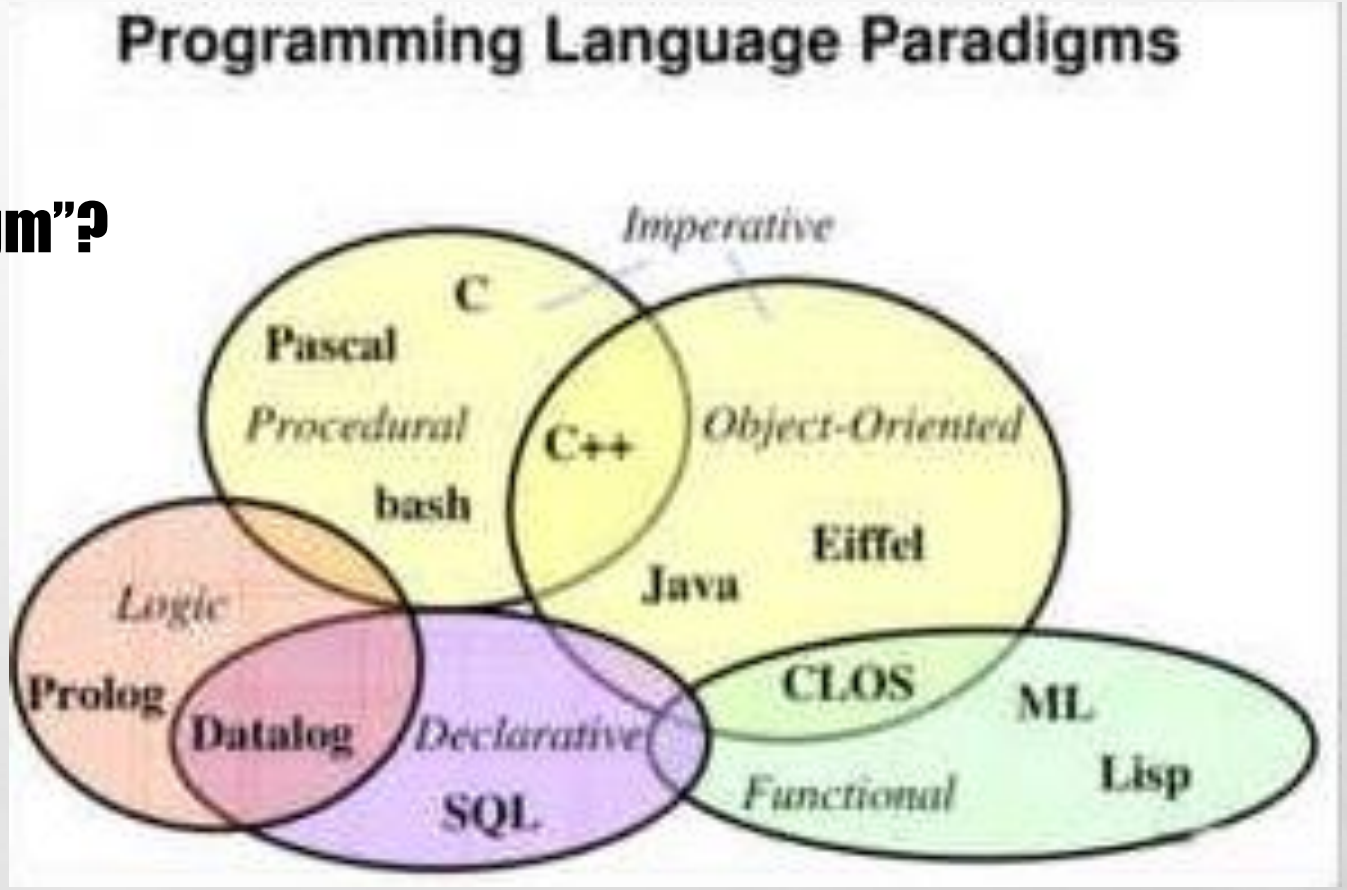
SUMMARY

Event Driven Paradigm

- The flow of the program is determined by events, such as or user actions (mouse clicks, key presses) or messages from other programs
- Is widely used in graphical user interfaces
- Is clearly divided down to two sections:
  - the first is event selection (or event detection)
  - the second is event handling.
- Limitations → Sometimes leading programmers to create error prone, difficult to extend and excessively complex application code

POST-SCRIPT

Is it really a programming "paradigm"?




PRESENTATION TERMINATED