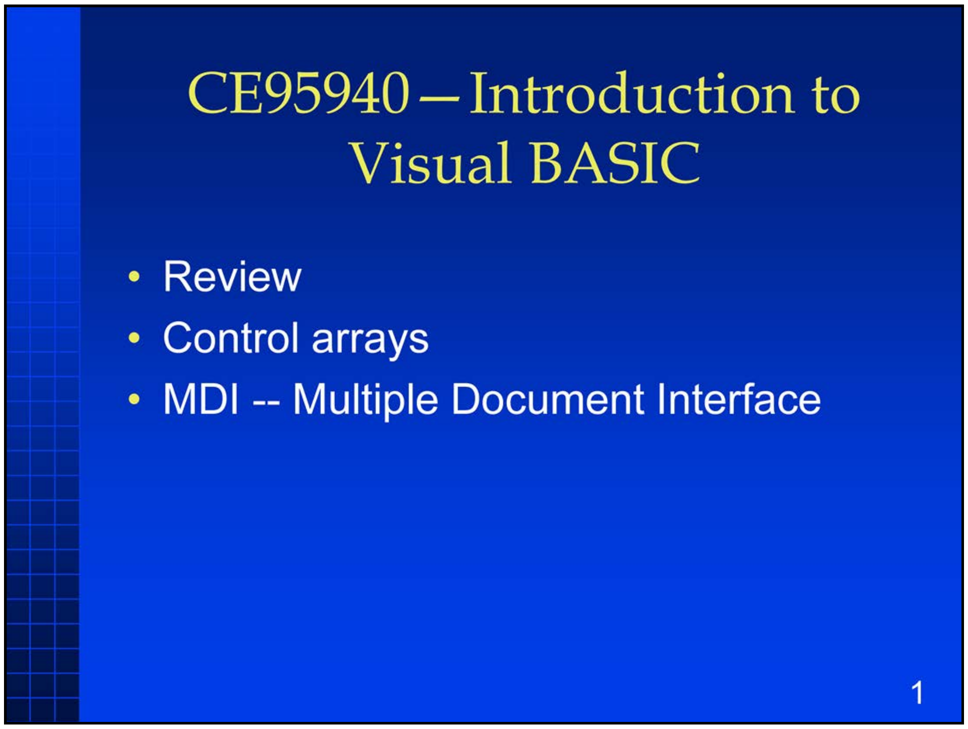
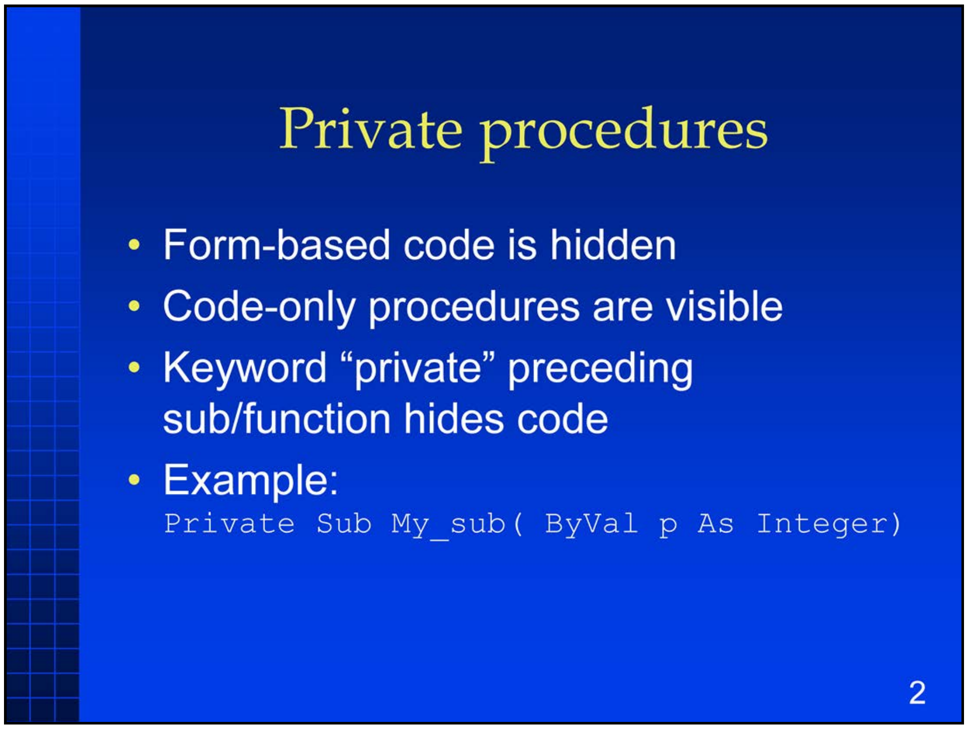


CE95940 – Introduction to Visual BASIC

- Review
- Control arrays
- MDI -- Multiple Document Interface

Review common questions and problems from last week



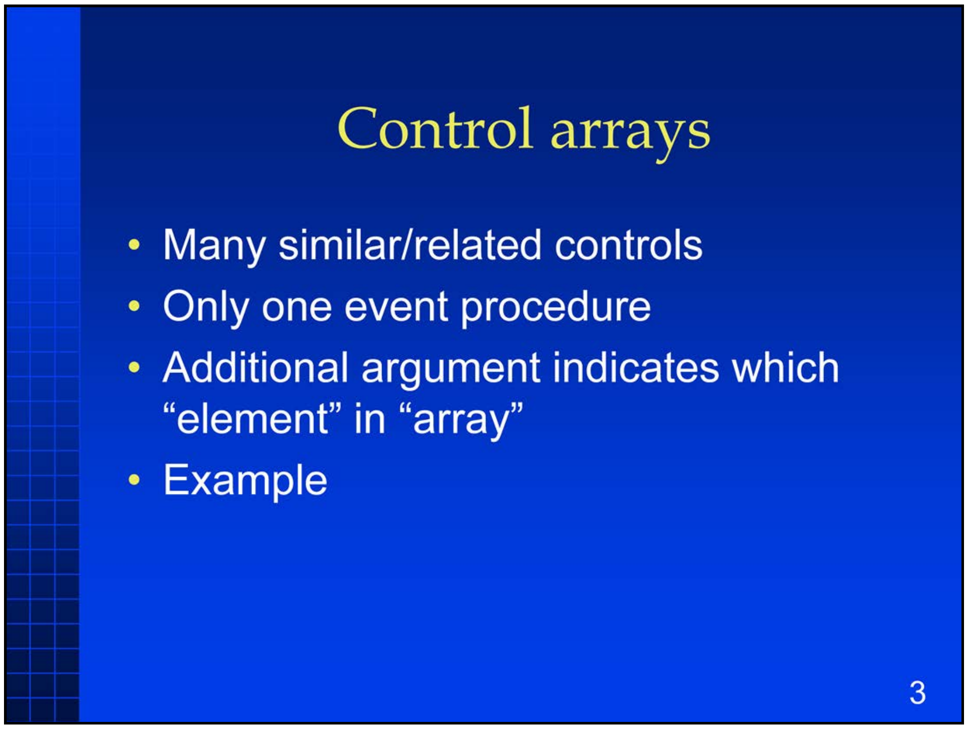
Private procedures

- Form-based code is hidden
- Code-only procedures are visible
- Keyword “private” preceding sub/function hides code
- Example:

```
Private Sub My_sub( ByVal p As Integer)
```

2

Clarification of scope of procedures: code in form modules is indeed hidden as stated; however, code in code modules (e.g. global) is visible: can be hidden with “private” keyword



Control arrays

- Many similar/related controls
- Only one event procedure
- Additional argument indicates which "element" in "array"
- Example

3

Usage: simplify code, eg calculator, toolbar, anywhere lots of similar controls needed

create a button; rename; cut & paste a few: note prompt for control array, note index property

reference example lect6\1

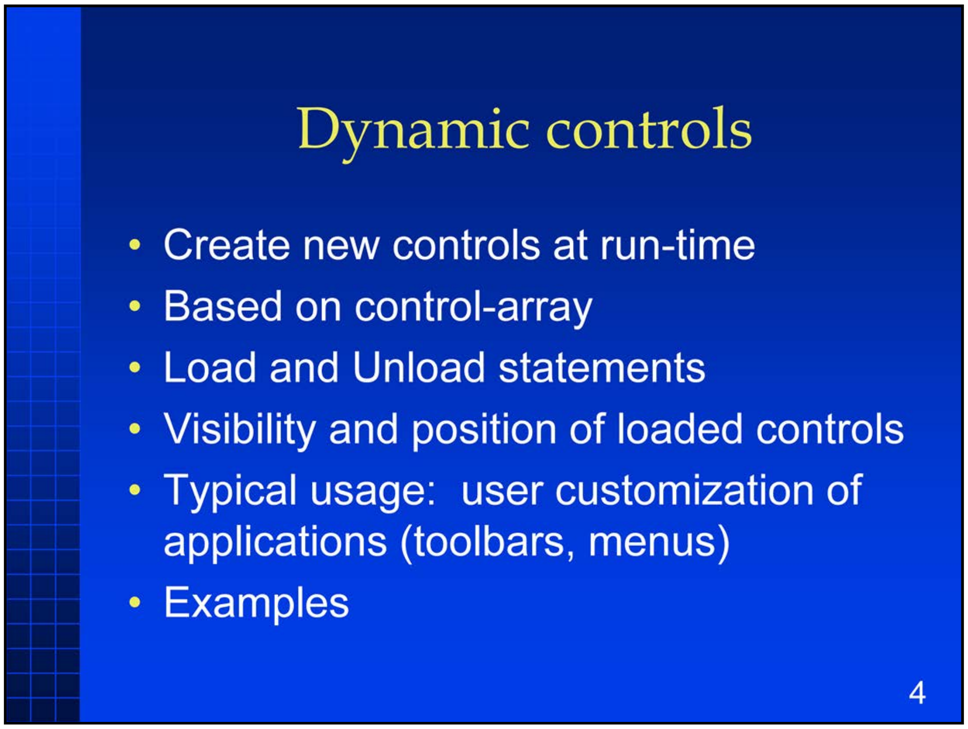
typical click event:

```
   select case index
    case 0
     ...
    case 1
     ...
  end select
```

<<<

possible trick in this case chr$(asc("a") + index)

>>>



Dynamic controls

- Create new controls at run-time
- Based on control-array
- Load and Unload statements
- Visibility and position of loaded controls
- Typical usage: user customization of applications (toolbars, menus)
- Examples

given a control array, can add elements

create simple example (create single control as array, dyn-create one more): note visibility and position problem; have to move it and make it visible, change caption etc.

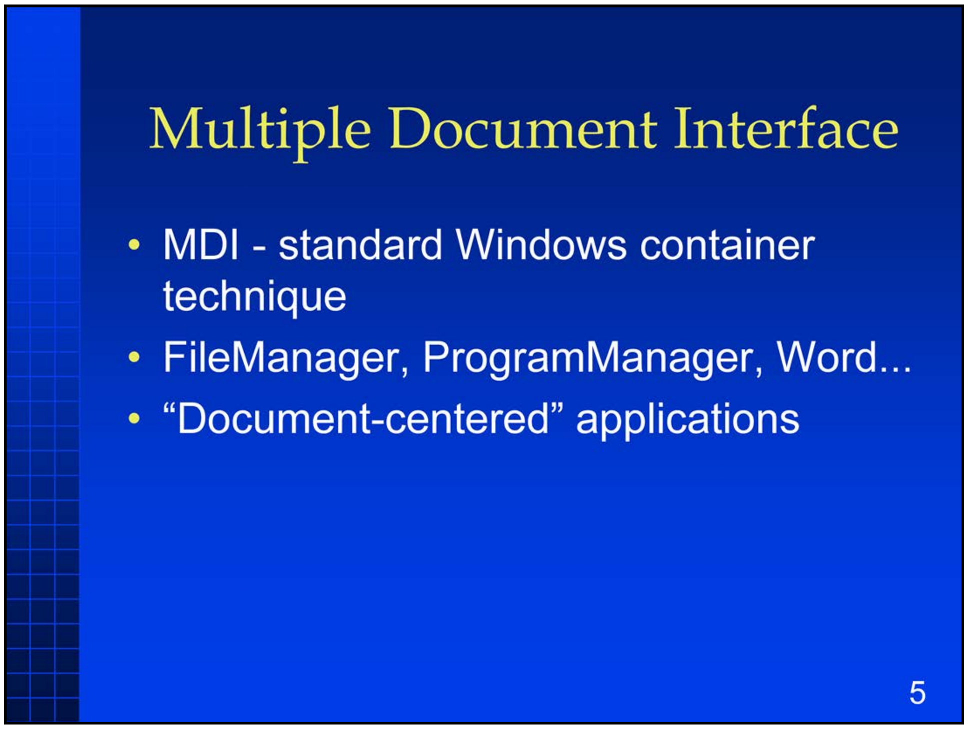
Example 6\2: similar to previous: make some buttons in a control array, add another button; remember 0-origin and related issues

note load control-array element: (click event for create button)

```
 Load Alphabet_button(B_count)
Alphabet_button(B_count).Visible = True
Alphabet_button(B_count).Top = Alphabet_button(B_count).Top + 500
Alphabet_button(B_count).Left = Alphabet_button(B_count).Left + (B_count - 5) *
   Alphabet_button(B_count).Width
 B_count = B_count + 1
```

Example 6\3 removing: use unload : can only unload dynamically-created ones (click event for remove)

Example 6.4 dynamic menus: can add items or entire menus (add code to add entire menu, handwave details)



Multiple Document Interface

- MDI - standard Windows container technique
- FileManager, ProgramManager, Word...
- “Document-centered” applications

MDI is standard windows technique for creating form that contains other forms, use parent-child terminology

Used is applications where document/object abstraction is required: e.g. Word has one child per document, many can be open

MS terms: document-centred: pseudo-concurrent access to similar documents

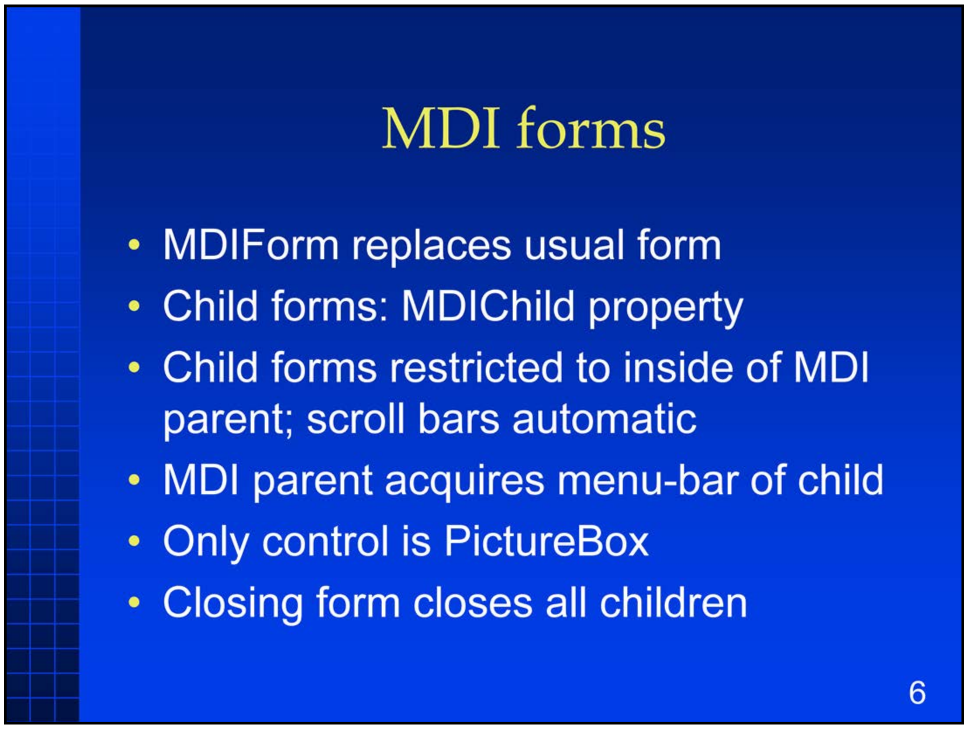
VB: only one container per project (MDIForm), other forms can be marked as contained or not (called “child forms”)

Container support minimization & maximization of children within the container

contrast against multiple-form applications

Opinion -- I’m not sure what the value is; if you want two objects open, invoke the application twice (relates back to object-based view of user interface discussed before)

However, many windows applications seem to use it, so have brief overview.



MDI forms

- MDIForm replaces usual form
- Child forms: MDIChild property
- Child forms restricted to inside of MDI parent; scroll bars automatic
- MDI parent acquires menu-bar of child
- Only control is PictureBox
- Closing form closes all children

Simple example

new project

delete form; new MDI form

another two forms; set MDIchild; set distinguishing features

run -- set entry point; nothing happend

not shown by default, add shows to mdi-load; play for a bit (min, max); note scroll bars

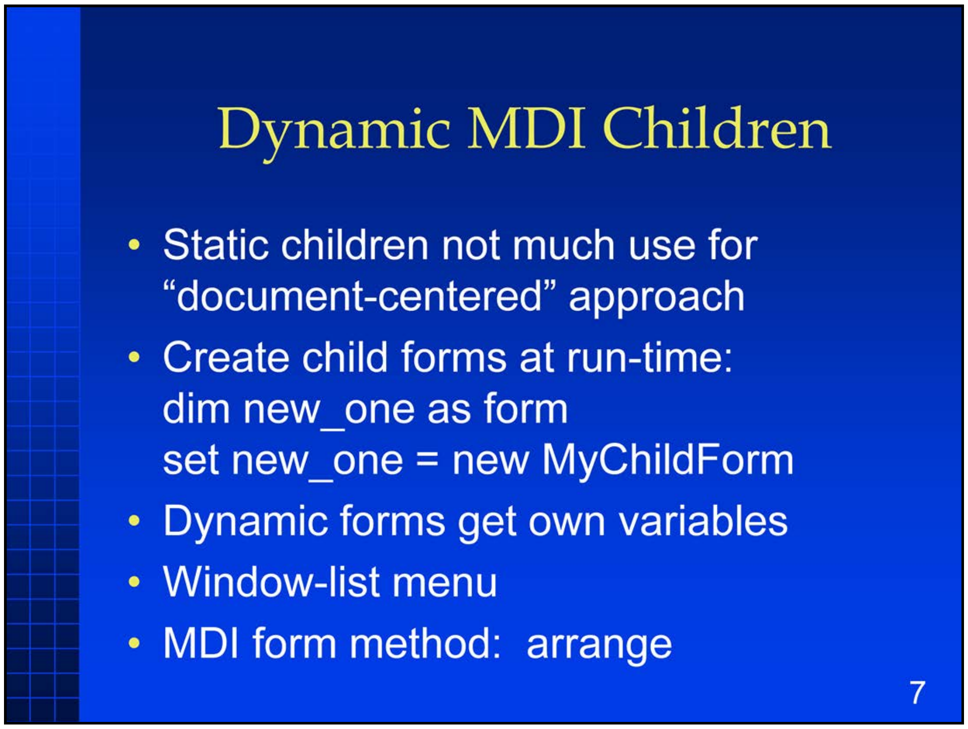
Demo program lect5\5: (cleaned up version)

Next: add menus to children -- big feature of MDI forms is that the MDI form acquires the menu of the active child (thinking being that only one menu-bar per application?)

MDI forms are containers -- only control is Picturebox across top of window or bottom (align property); used for toolbar and status

can add controls inside PictureBox.

closing the MDI form closes all children: each child get a queryunload event



Dynamic MDI Children

- Static children not much use for "document-centered" approach
- Create child forms at run-time:
  dim new_one as form
  set new_one = new MyChildForm
- Dynamic forms get own variables
- Window-list menu
- MDI form method: arrange

7

not much use to have a fixed number of child forms -- want to be able to create them dynamically -- corresponds to a "new" menu item

create a form that will act as a prototype for the document, then create instances dynamically

example 6\6

MDI parent with one menu item: new -> look at code

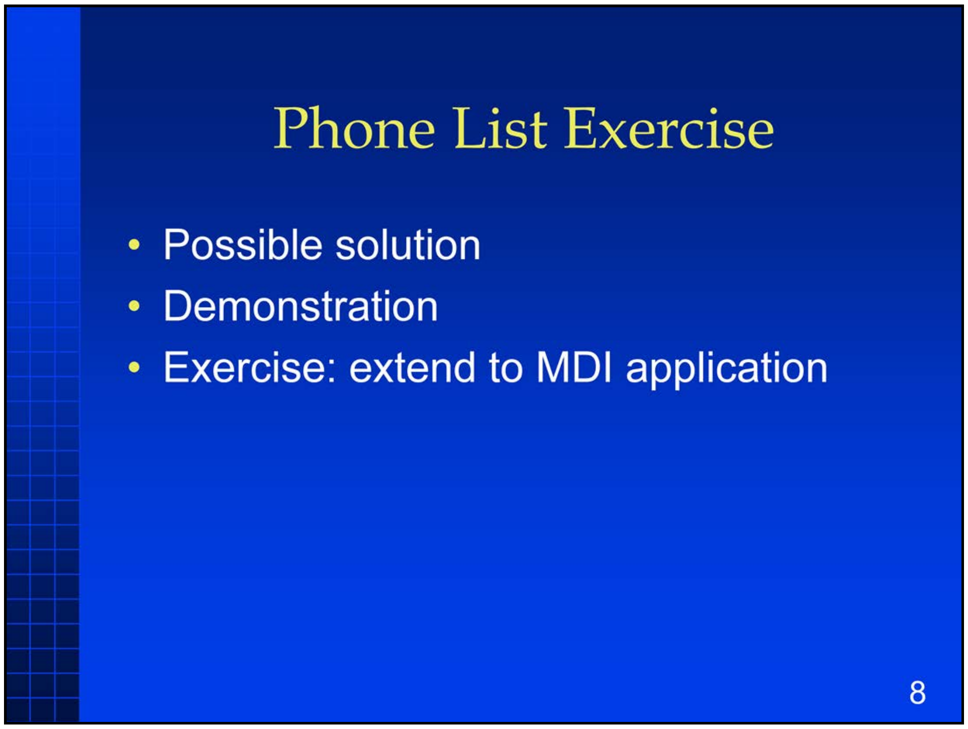
child prototype has new and window-list menus -- therefore each instance of child has same

NewChild: creates new form, keeps track of form variable in array (not used here but generally reasonable idea) (eg closeall)

each instances of child has own set of variables

with all these windows, need housekeeping: add "window-list" automatically (look at menu-design for child)

use arrange method of MDIform (0=cascade, tile_h=1, tile_v=2, arr_icon=3); add arrange tile to menu



Phone List Exercise

- Possible solution
- Demonstration
- Exercise: extend to MDI application

8

show working program:demos\lect6\7

basic method:

read file into memory, build array & list

keep visible parts consistent at all times: change both together, ignore file

do updates/changes etc in memory, use modal form to allow cancelling (null name means cancelled)

at end-of-program, erase file then recreate(dump memory to file); ignore null names (had been deleted)