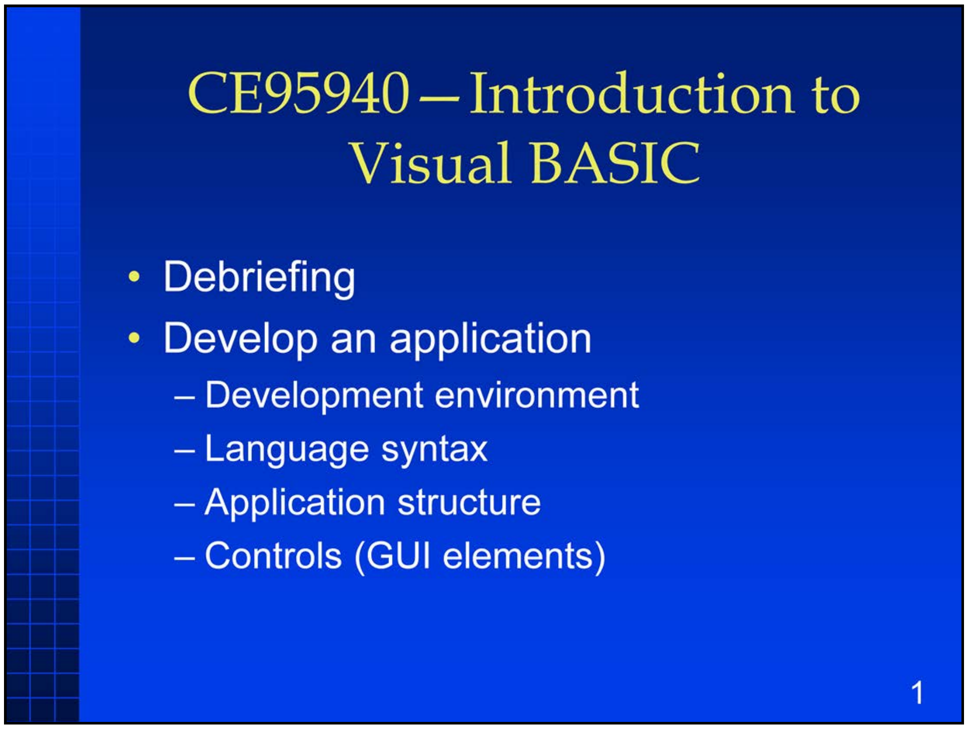


# CE95940 – Introduction to Visual BASIC

- Debriefing
- Develop an application
  - Development environment
  - Language syntax
  - Application structure
  - Controls (GUI elements)

1

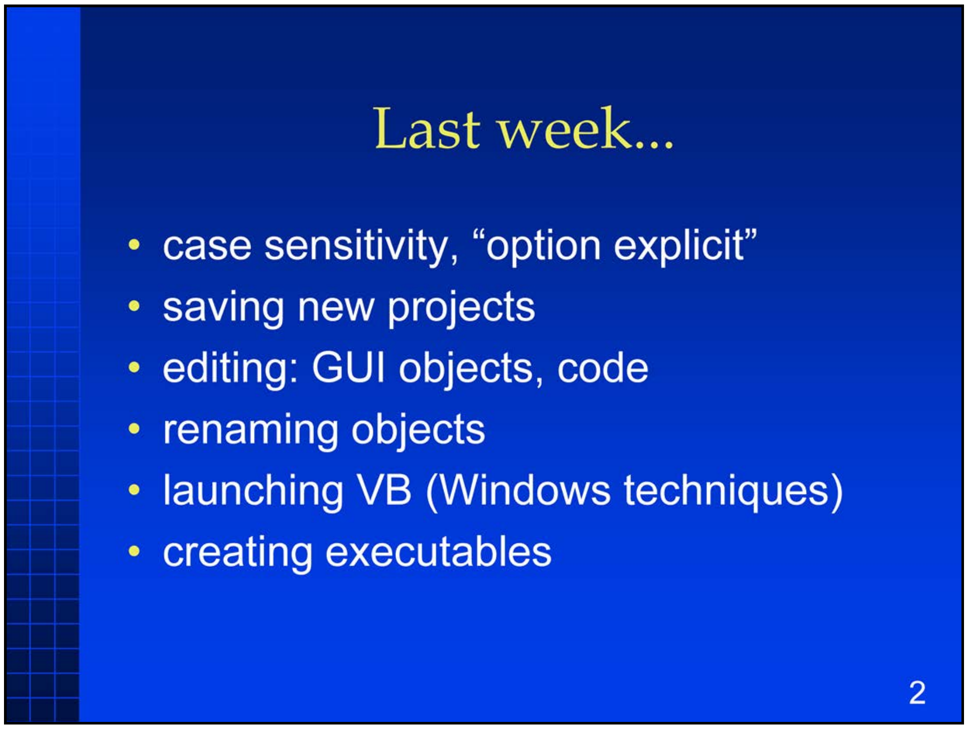
Review common questions and problems from last week

More detailed view of IDE, language, application and controls -- develop an application and cover these areas

Monday section: typo in Watstar doc: “m” not “k”

Shape object -- opaque background

Both sections: customize Windows, set “save settings”; toolbar is MS Office; startup group



## Last week...

- case sensitivity, “option explicit”
- saving new projects
- editing: GUI objects, code
- renaming objects
- launching VB (Windows techniques)
- creating executables

minor points:

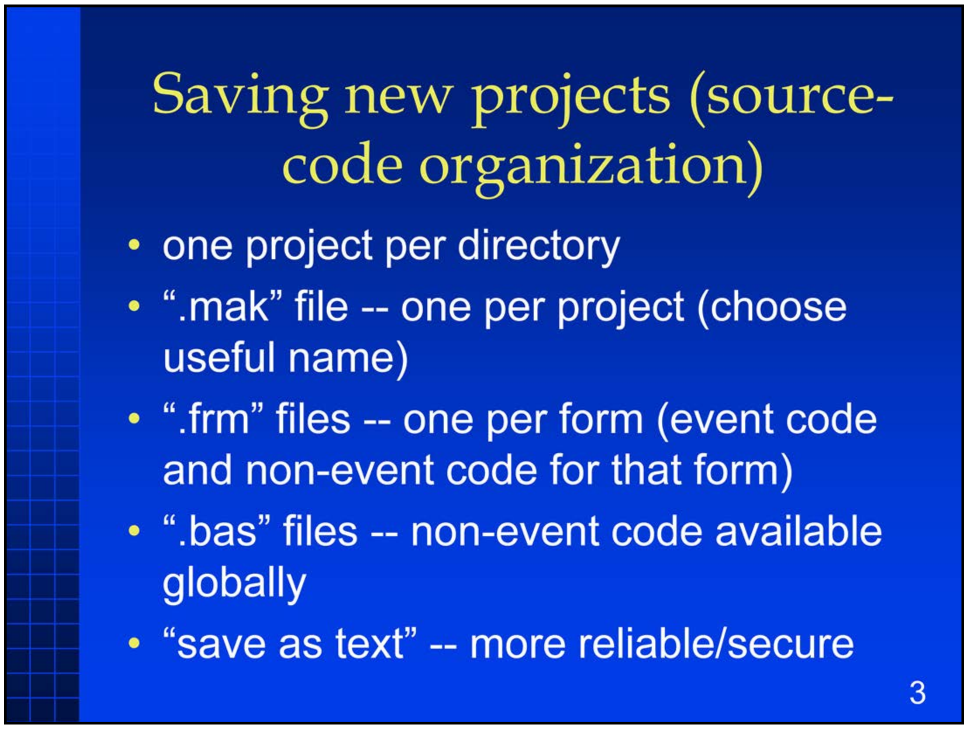
not case sensitive (editor will change references)

function names changes to upper-case

generated automatically when “Require Variable Declaration” is on (set to yes)

can be removed

can be added by hand



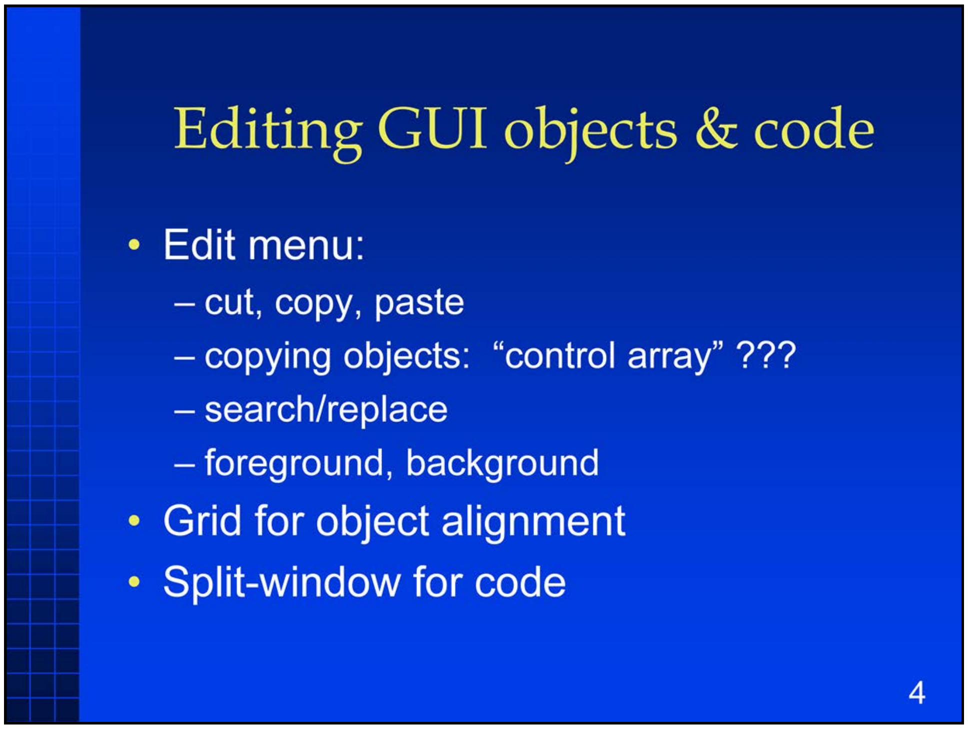
## Saving new projects (source-code organization)

- one project per directory
- ".mak" file -- one per project (choose useful name)
- ".frm" files -- one per form (event code and non-event code for that form)
- ".bas" files -- non-event code available globally
- "save as text" -- more reliable/secure

namespace problems; default names

for now, only one form per project

text-save format; exportable, recoverable



## Editing GUI objects & code

- Edit menu:
  - cut, copy, paste
  - copying objects: “control array” ???
  - search/replace
  - foreground, background
- Grid for object alignment
- Split-window for code

4

edit menu for both code & objects

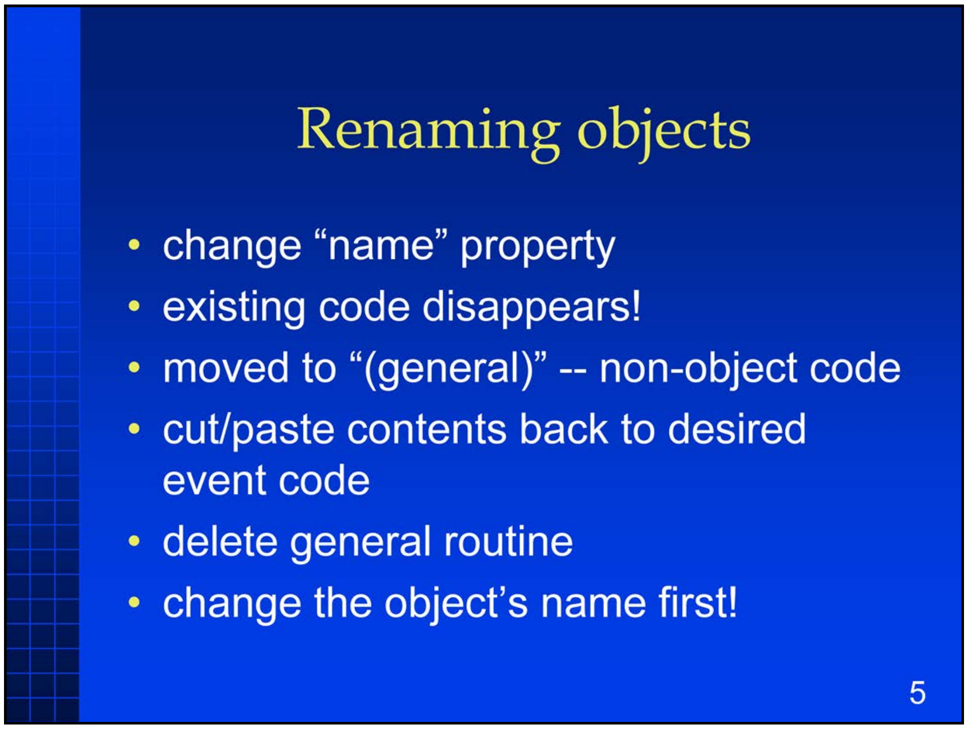
standard operations: build groups of objects with lasso, ctrl-click

move delete duplicate=copy+paste

fg/bg: some can do it, some cannot

grid eases placement, controllable via environment options (bottom of list)

split code window, cut/paste between



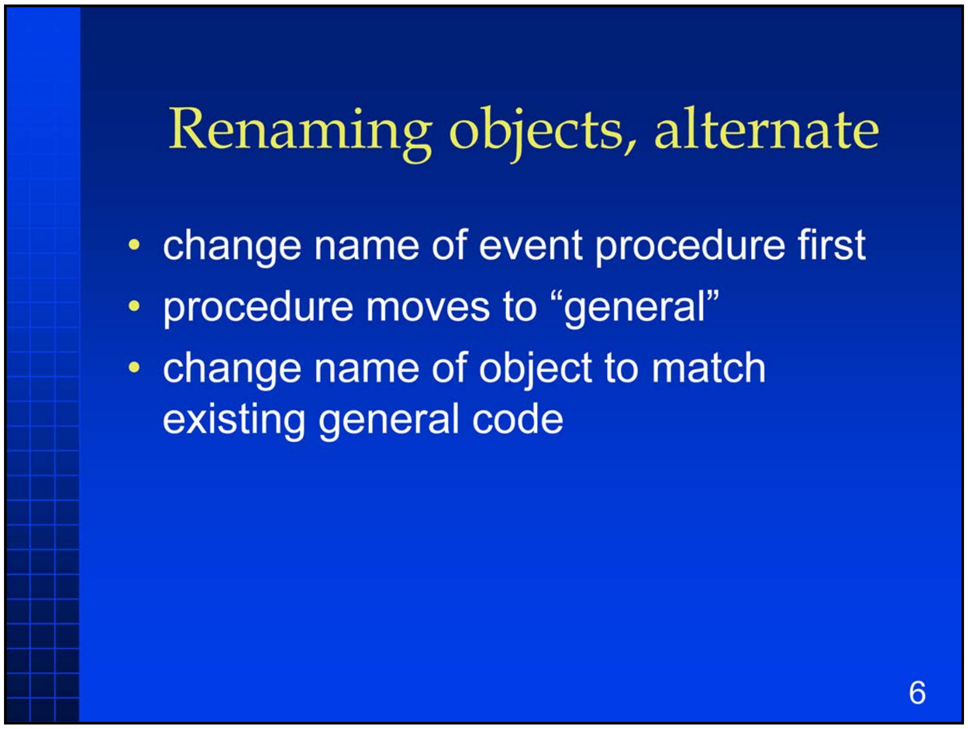
## Renaming objects

- change “name” property
- existing code disappears!
- moved to “(general)” -- non-object code
- cut/paste contents back to desired event code
- delete general routine
- change the object’s name first!

reminder about structure of code window:  list of objects, list of events

best idea is to change object-name first, before adding any code

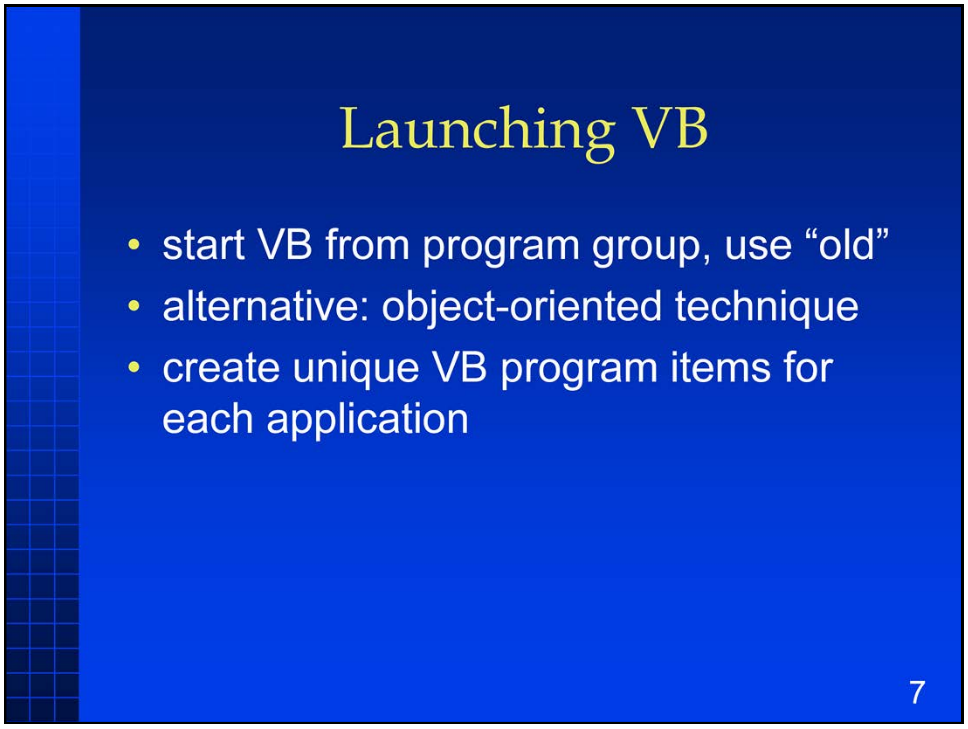
changing the general routine name to look like an event routine creates a “duplicate name” error



## Renaming objects, alternate

- change name of event procedure first
- procedure moves to “general”
- change name of object to match existing general code

name the object after the code, not the code after the object



## Launching VB

- start VB from program group, use “old”
- alternative: object-oriented technique
- create unique VB program items for each application

7

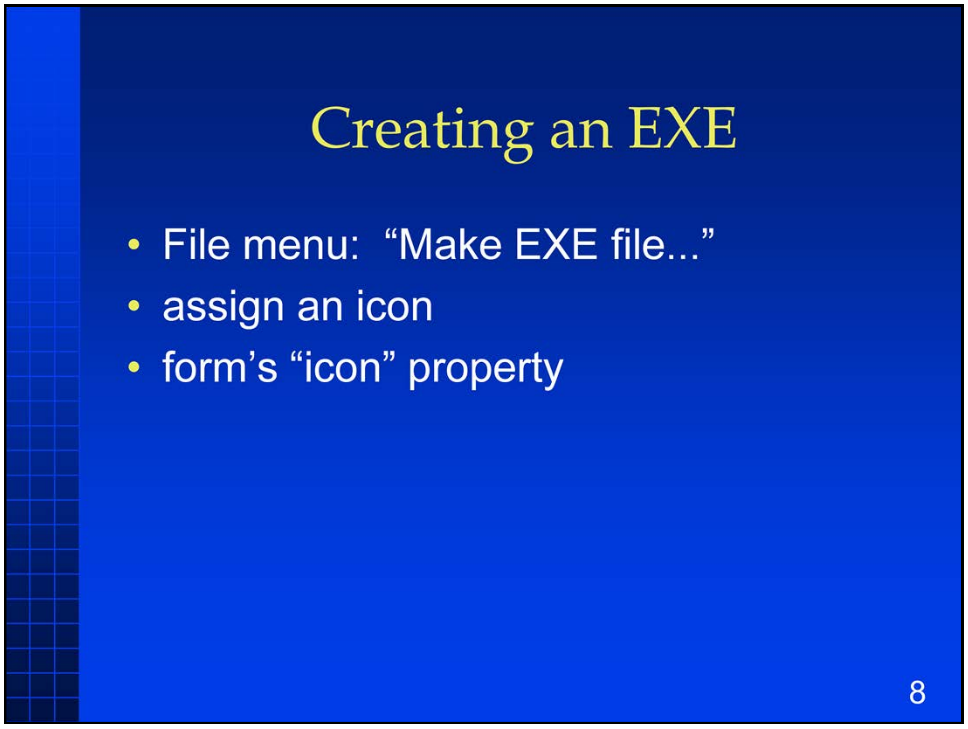
object technique from OS/2 and others, opposite to the “document” model where application takes over

the entities are the certral idea, not the application code (application is an attribute of the object)

makes *lots* of program items

Windows lacks support -- multiple instance often disallowed (eg VB)

pesonal taste

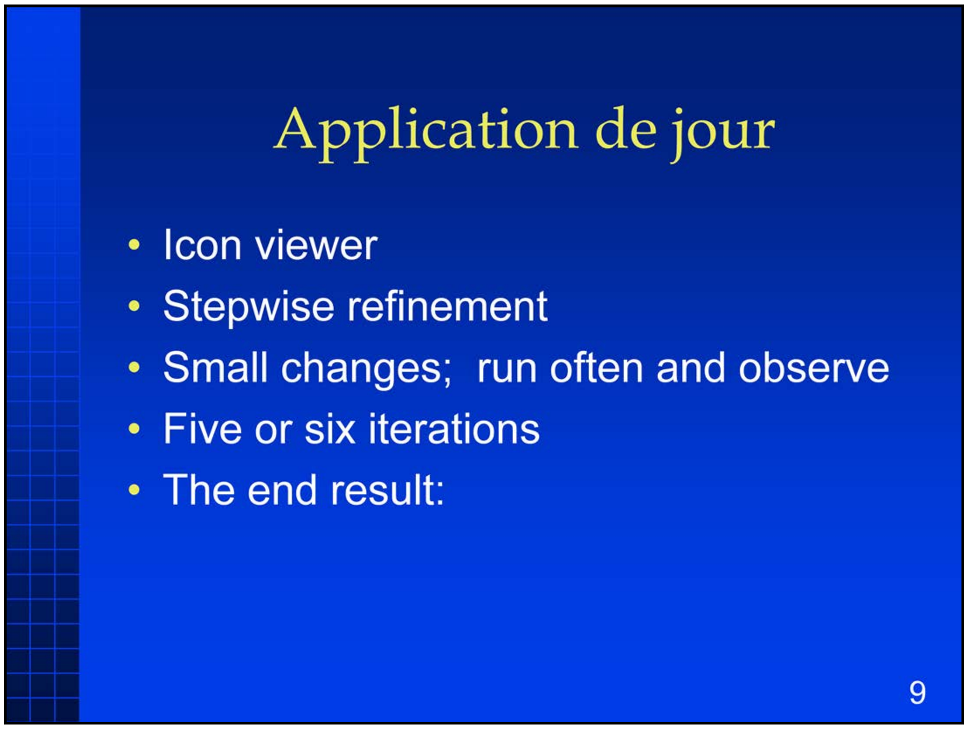


## Creating an EXE

- File menu: “Make EXE file...”
- assign an icon
- form’s “icon” property

8

EXEs are small -- required VB runtime DLL (freely distributable)
may also require other DLLs: in general, not a trivial problem
topic for later discussion (Setup Wizard)



## Application de jour

- Icon viewer
- Stepwise refinement
- Small changes;  run often and observe
- Five or six iterations
- The end result:

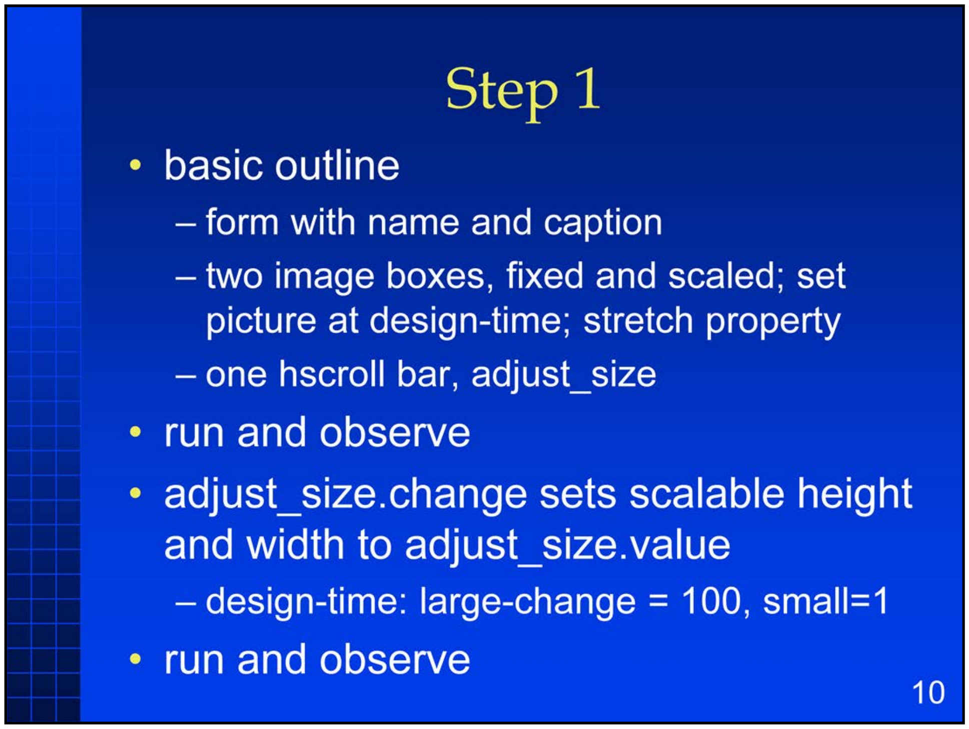
9

run the program (iconview.exe); use \apps\vb\icons\elements\earth.ico

shutdown ppt

The purpose of this demo is to show some of the kinds of issues involved with good event-driven UI programs (eg consistency, code organization, user’s perspective).  This particular app is not the point, not are the details about controls used.

Reinforces concepts of seamless design etc.



## Step 1

- basic outline
  - form with name and caption
  - two image boxes, fixed and scaled; set picture at design-time; stretch property
  - one hscroll bar, adjust_size
- run and observe
- adjust_size.change sets scalable height and width to adjust_size.value
  - design-time: large-change = 100, small=1
- run and observe

10

create a working directory somewhere

launch VB

make form roughly square, use properties to observe

set borders around images

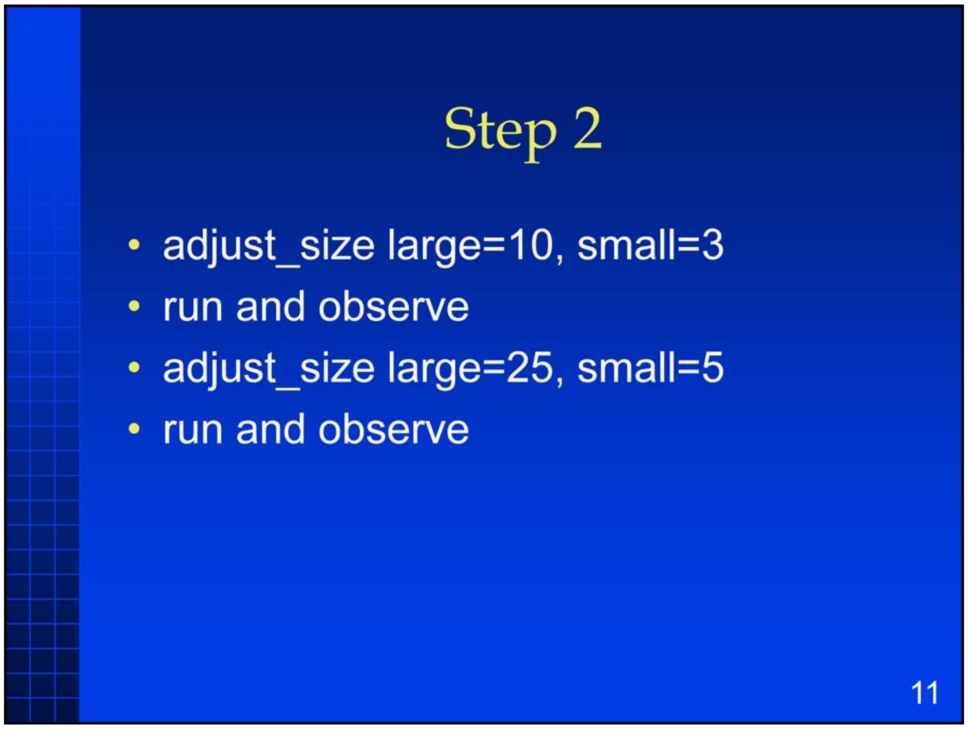
stretch property = true for scaled image

during first save, set project name as “iconview” (iconview.mak)

explain scrollbar value: just a “visual counter”, button indocate where in range the current value is; explain large and small

shutdown VB, user filemanager to create VB program item in group

restart



## Step 2

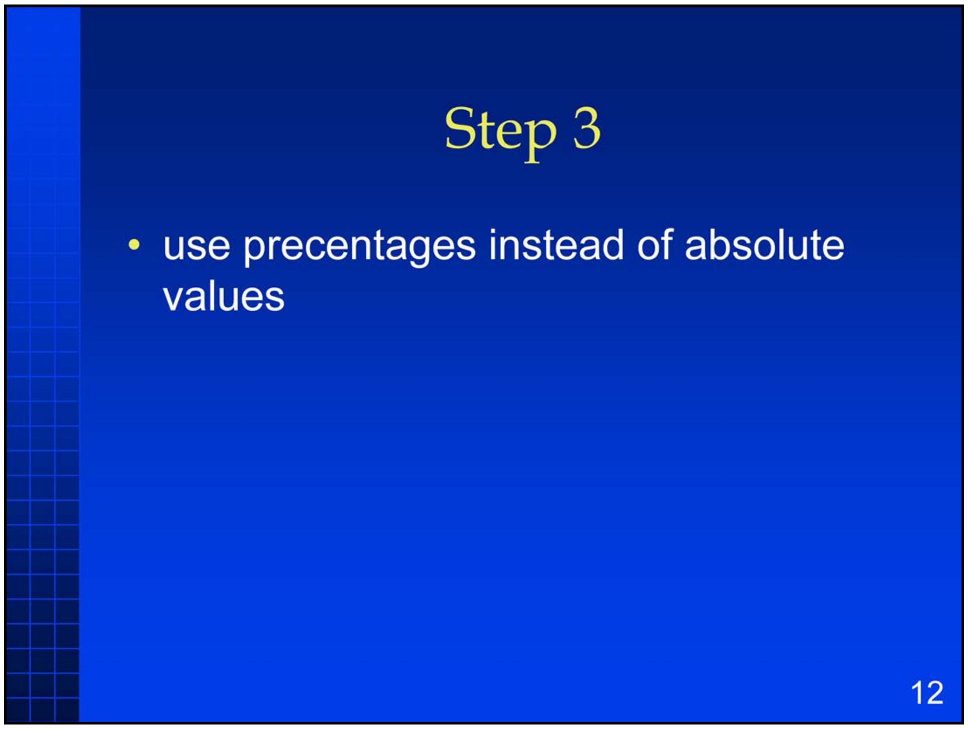
- adjust_size large=10, small=3
- run and observe
- adjust_size large=25, small=5
- run and observe

large=10 and small=3 not enough

large=25 and small=5 seem ok

notice that large images goes way off screen

consider min and max properties; observe default values



## Step 3

- use precentages instead of absolute values

could set limits at designtime (form width less left edge of image) but what if layout changes?

in fact, prefer not to use absolute # at all, use sliders to express % of allowable space

change min and max to 0, 100; maybe change small & large

in load event (initialization for form):
```
  compute allowable space:  vert = form.height - scaled.top
                            horz = form.width - scaled.left
```

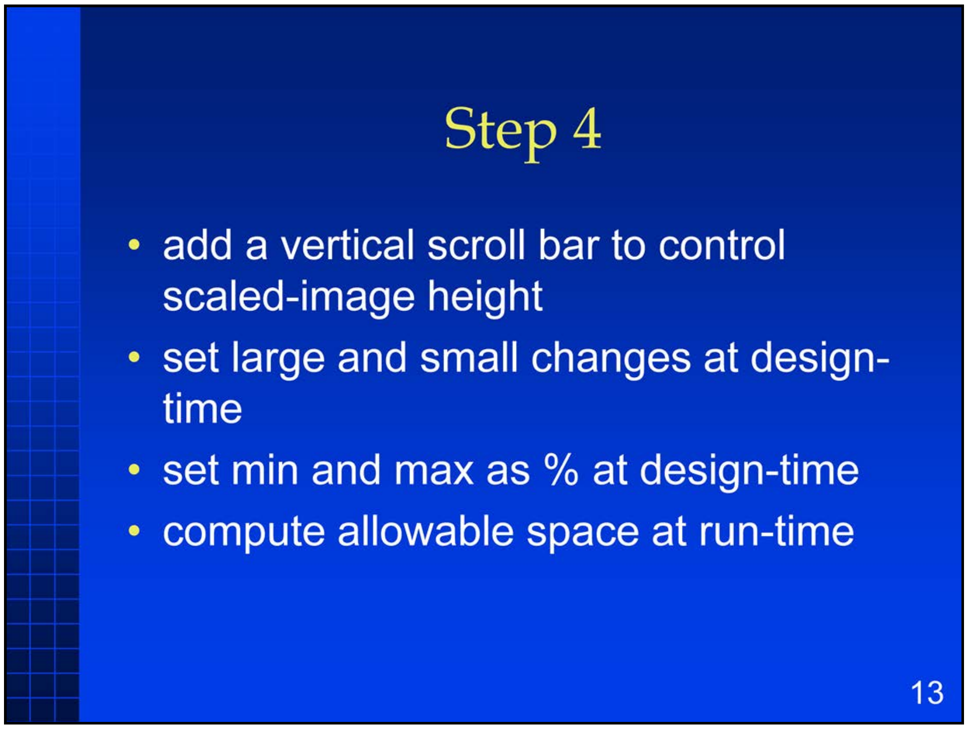
change event now sets height, width as:
```
     height = allowable_vert * (change_value/100)
```

run and observe -- not quite right (goes a little off the edge of the form)

use form.scalewidth (interior space, not including borders, title)

observe proportion in scroll bar changes

startup, consistency problems -- set value at design-time, adjust min value to slightly bigger that 0

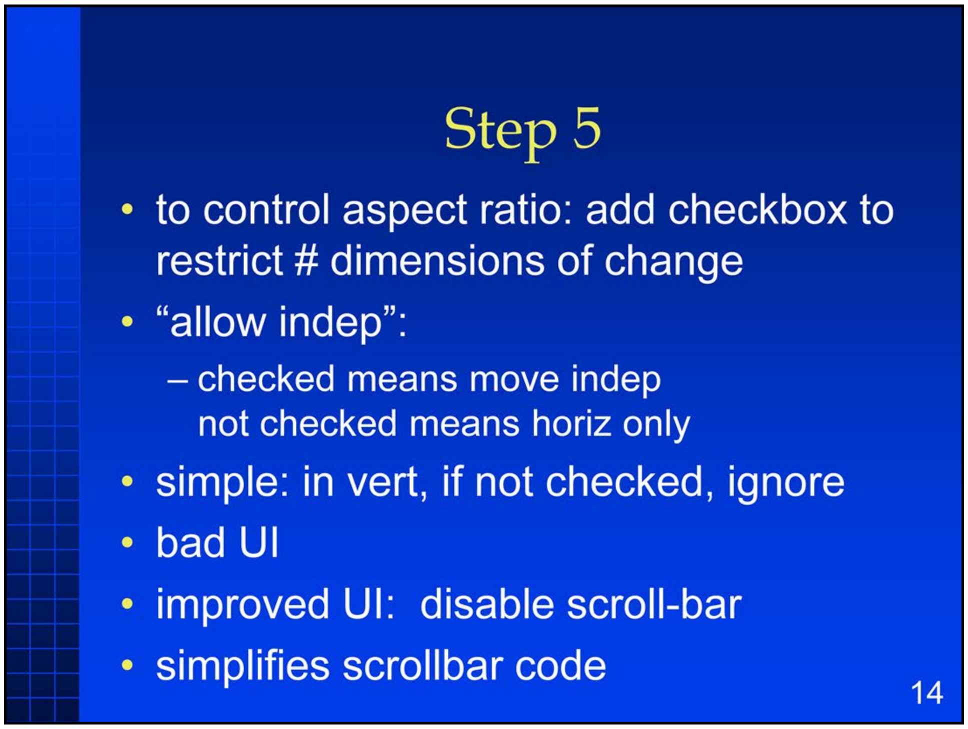


## Step 4

- add a vertical scroll bar to control scaled-image height
- set large and small changes at design-time
- set min and max as % at design-time
- compute allowable space at run-time

add a vertical scroll to adjust height

note strange changes to aspect ratios are now possible



## Step 5

- to control aspect ratio: add checkbox to restrict # dimensions of change
- “allow indep”:
  - checked means move indep
    not checked means horiz only
- simple: in vert, if not checked, ignore
- bad UI
- improved UI:  disable scroll-bar
- simplifies scrollbar code

14

add checkbox and caption

first try:  in vscroll, test checked value, only change if checked (define constants for checked (1) and unchecked (0))

run and observe:  scrollbar operates but doesn’t work,

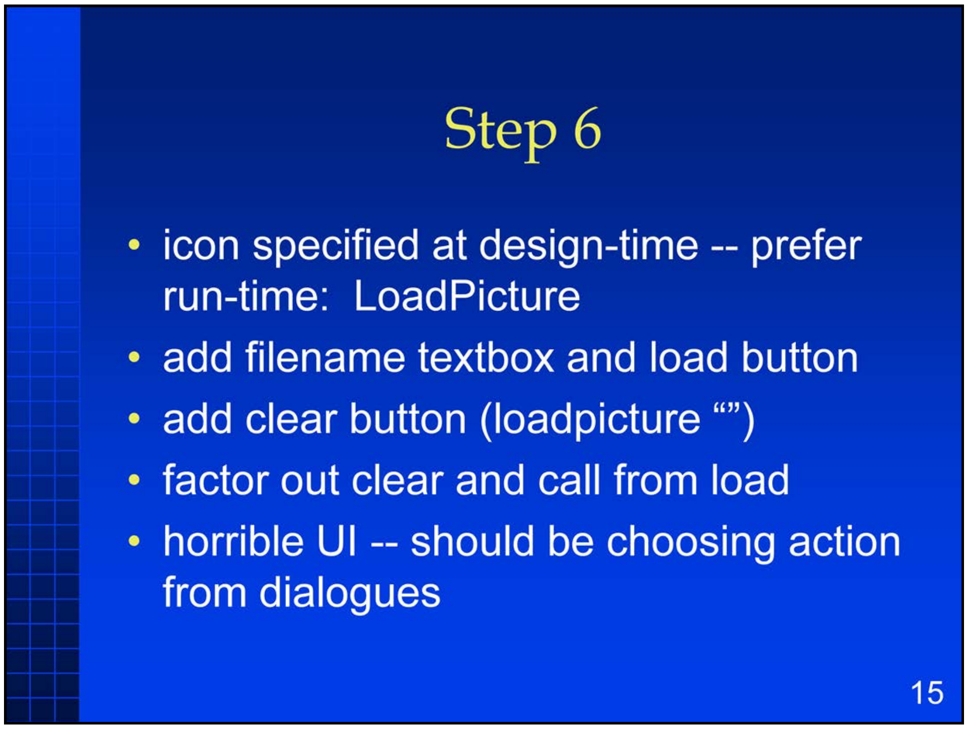
?(toggling off & on makes bar setting jump around)

second try:  in checked, disable/enable vscroll bar

better

can remove test from vscroll code, since can never happen

initial settings:  checkbox is not set, but vscroll enabled:  clean up (make consistent:  disable vscroll or check box)



## Step 6

- icon specified at design-time -- prefer run-time:  LoadPicture
- add filename textbox and load button
- add clear button (loadpicture “”)
- factor out clear and call from load
- horrible UI -- should be choosing action from dialogues

15

place at top of form; use group select and move to shift form contents down

initially, don’t set filename text in load -- observe default text

overview of syntax of procedures & functions:
  defining

  calling

  args -- byval

VB bug:  cannot remove design-time picture