Graphical User Interfaces, FLTK

9 Feb 2009
CMPT166
Dr. Sean Ho
Trinity Western University

Review last time: exceptions

- Constructor intializer list
  - Calling the superclass constructor
- Exceptions: throw, try/catch
  - Catching by name, getting auxiliary info
  - Re-throwing exceptions
  - Class hierarchies for exceptions
  - Standard exception classes

Standard exception classes

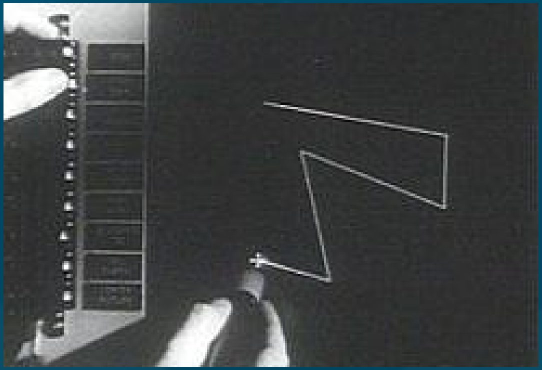
- Any object in C++ may be thrown, but the Standard C++ Library does include some standard exception classes for you to subclass:
    - **#include <stdexcept>**
- The superclass is exception; two subclasses include runtime_error and logic_error
    - **class Error : public runtime_error { ....**
- Constructors can take a string argument
    - Read it using the .what() method

What's on for today

- An excerpted history of GUIs
  - Sutherland, Engelbart, PARC, Apple, MS
- GUI toolkits
- Events and callbacks
- FLTK and Fluid
  - BankInterest example

Sutherland's SketchPad (1963)

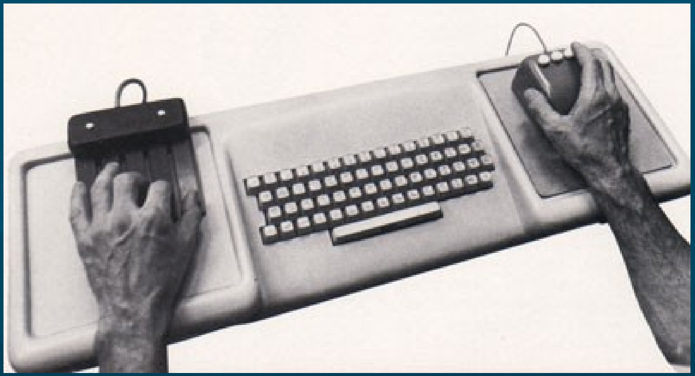
- Ivan Sutherland
  Ph.D. thesis at MIT
- Used light pen to directly manipulate graphical objects on screen
- Pioneer of computer-aided drafting (CAD):
  - Draw “master” diagram once
  - Instantiate multiple copies, tweak (OO design)
  - Constraint-based system (e.g., keep two lines at fixed angle)



Engelbart's NLS demo (1968)

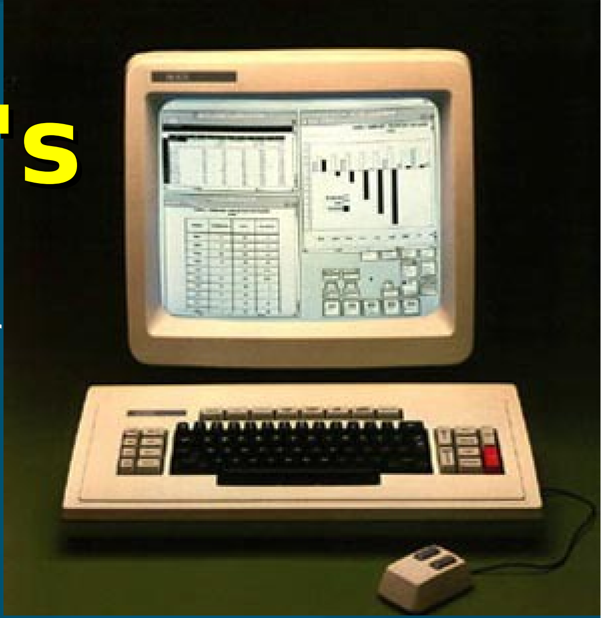
Douglas Englebart,
Stanford Research Inst.

- NLS (oNLine System) innovations:
  - Mouse
  - Windowing system
  - Collaborative document editing with email, IM, and video conferencing
  - Hyperlinks
  - Chording keyboard

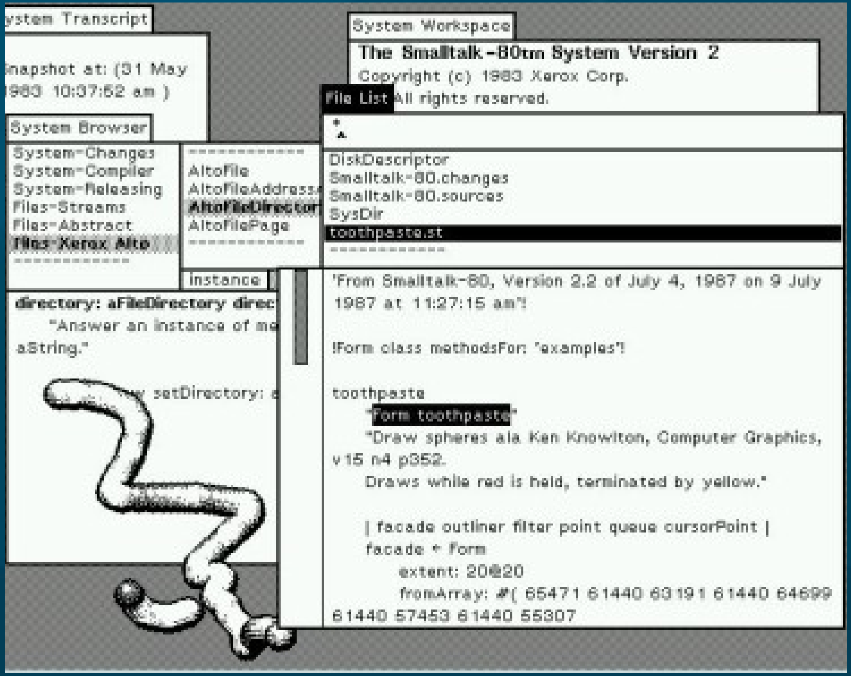


Xerox PARC in the 1970's

Smalltalk
on the Star



- Xerox Palo Alto:
  - Towards “paperless office”
  - Microcomputers: Alto (1973), Star (1981)
  - WIMP model: windows, icons, menus, pointer
  - Desktop
  - Smalltalk (1974):
    - Pure OO language
    - Integrated graphical development and runtime environment



```
napshot at: (31 May
983 10:37:52 am )
```

```
System-Changes
System-Compiler
System-Releasing
Files-Streams
Files-Abstract
Files-Xerox Alto
-------------
```

```
-------------
AltoFile
AltoFileAddress
AltoFileDirector
AltoFilePage
-------------
```

```
directory: aFileDirectory direc
    "Answer an instance of me
aString."

            setDirectory: a
```

```
The Smalltalk-80tm System Version 2
Copyright (c) 1983 Xerox Corp.
All rights reserved.
```

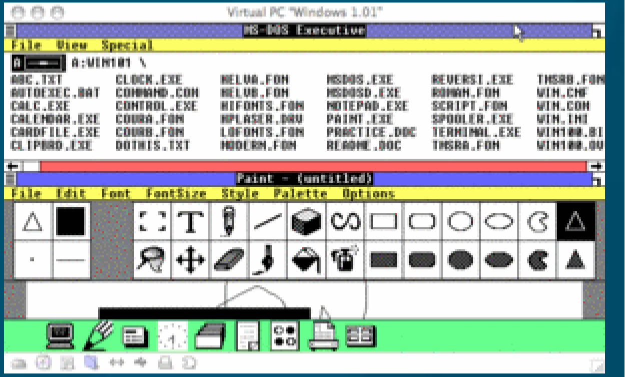
```
DiskDescriptor
Smalltalk-80.changes
Smalltalk-80.sources
SysDir
toothpaste.st
------------
```

```
'From Smalltalk-80, Version 2.2 of July 4, 1987 on 9 July
1987 at 11:27:15 am'!

!Form class methodsFor: 'examples'!

toothpaste
    "Form toothpaste"
    "Draw spheres ala Ken Knowlton, Computer Graphics,
v15 n4 p352.
    Draws while red is held, terminated by yellow."

    | facade outliner filter point queue cursorPoint |
    facade ← Form
        extent: 20@20
        fromArray: #( 65471 61440 63191 61440 64699
61440 57453 61440 55307
```

Apple in the 1980's

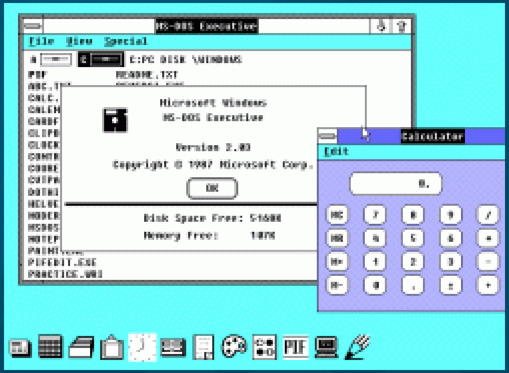
- Lisa (1983):
  - Drag-and-drop
  - Double-click to open/run
- Macintosh (1984):
  - Much cheaper ($2,495 vs. >$10k)
  - Accessible to the public
  - Mass-marketing ad campaign during SuperBowl and 1984 Olympics in L.A.

Microsoft Windows (1980's)


Virtual PC "Windows 1.01"
MS-DOS Executive
File View Special
Paint - (untitled)
File Edit Font FontSize Style Palette Options

```
A:WIN101 \
ABC.TXT        CLOCK.EXE      HELVA.FON      MSDOS.EXE      REVERSI.EXE    TMSRB.FON
AUTOEXEC.BAT   COMMAND.COM    HELVB.FON      MSDOSD.EXE     ROMAN.FON      WIN.CNF
CALC.EXE       CONTROL.EXE    HIFONTS.FON    NOTEPAD.EXE    SCRIPT.FON     WIN.COM
CALENDAR.EXE   COURA.FON      HPLASER.DRV    PAINT.EXE      SPOOLER.EXE    WIN.INI
CARDFILE.EXE   COURB.FON      LOFONTS.FON    PRACTICE.DOC   TERMINAL.EXE   WIN100.BI
CLIPBRD.EXE    DOTHIS.TXT     MODERN.FON     README.DOC     TMSRA.FON      WIN100.OV
```

- Windows 1.0 (1985):
  - Mostly character-based graphics
  - Tiled windows
  - Popularity dwarfed by Mac

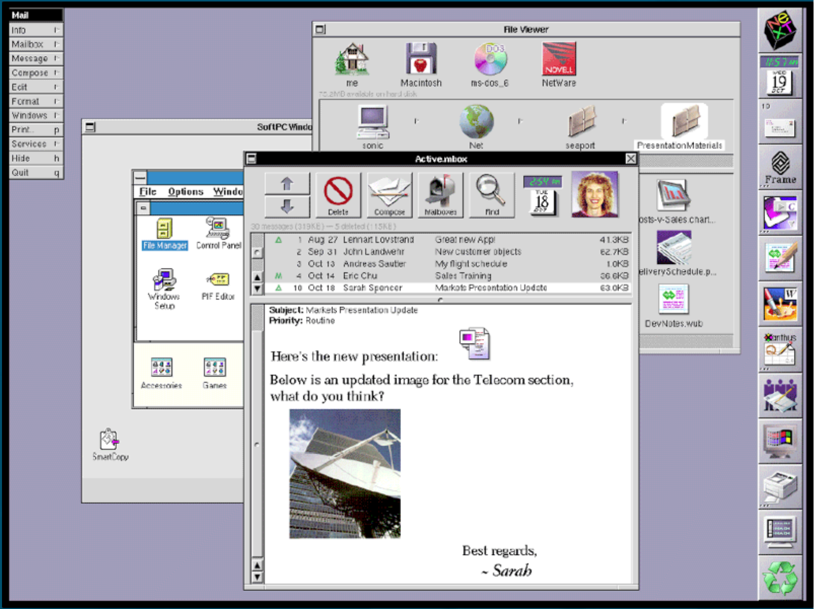


```
C:PC DISK \WINDOWS
PIF            README.TXT
PIFEDIT.EXE
PRACTICE.WRI

Microsoft Windows
MS-DOS Executive
Version 2.03
Copyright © 1987 Microsoft Corp.
Disk Space Free: 51680
Memory Free:     107K
```

- Windows 2.0 (1987):
  - Overlapping windows
  - Apple sues MS over “look and feel” (loses)
- Windows 3.11 (1992), Win95:
  - Looks pretty; wildly popular

Other GUI environments

- GEM (Digital Research) for Atari (1985)
- Amiga Workbench (1985)
- NeXTstep (Steve Jobs) (1988)
  - Pretty, but CPU-intensive
- OS/2 (IBM) (1988):
  - competed with Windows
- Unix X10 (1984), X11 (1987)
  - Network transparency (Xwin32)
  - Multiple libraries on top: Athena, Motif/CDE, OpenLook, KDE/Qt, Gnome/gtk, FLTK



```
30 messages (319KB) — 5 deleted (115KB)
△  1 Aug 27 Lennart Lovstrand  Great new App!              41.3KB
   2 Sep 31 John Landwehr      New customer objects         62.7KB
   3 Oct 13 Andreas Sauter     My flight schedule            1.0KB
M  4 Oct 14 Eric Chu           Sales Training               36.6KB
△ 10 Oct 18 Sarah Spencer      Markets Presentation Update  63.0KB
```

Subject: Markets Presentation Update
Priority: Routine

Here's the new presentation:

Below is an updated image for the Telecom section, what do you think?

Best regards,
~ *Sarah*

NeXTstep

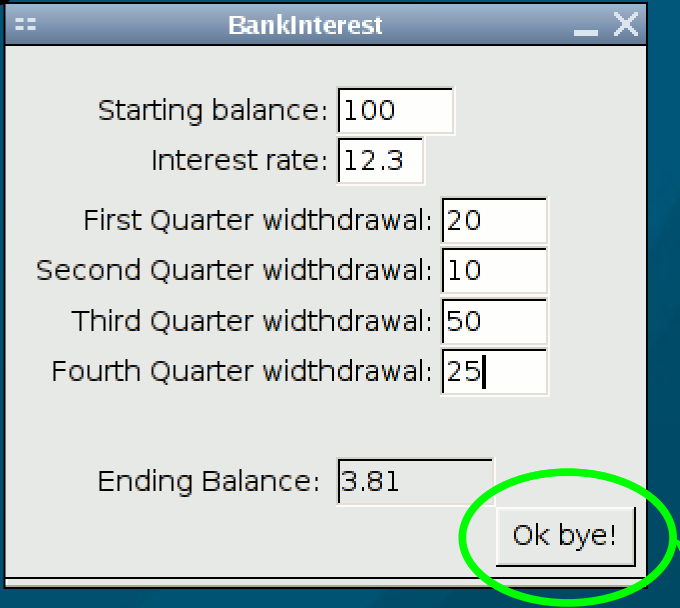
OS environment vs. toolkit

- In the past, the only GUI was what was provided by the operating system
- Now, we can write programs that link to various GUI toolkits:
  - Libraries that provide a way to build a GUI program
  - Menus/windows that look just like Windows:
    - Link with MFC or Visual Basic or .NET
  - Other options: FLTK, Qt, wxWidgets, gtk, ...
    - Cross-platform: can run on Linux, Mac, etc.

Compiling with GUI toolkits

- Libraries provide GUI components as objects
  - Windows (Fl_Window), menus, tabs, etc.
  - Widgets: buttons, textboxes (Fl_Input), sliders, scrollbars, dials, etc.
- Link your program with the toolkit library
  - Static linking: libfltk.a
    - Needed objects are bundled into the executable
  - Dynamic linking: libfltk.dll.a / libfltk.so
    - Need separate shared library
  - FLTK-1 libs: fltk, fltk_gl, fltk_images, fltk_forms

Events and callbacks

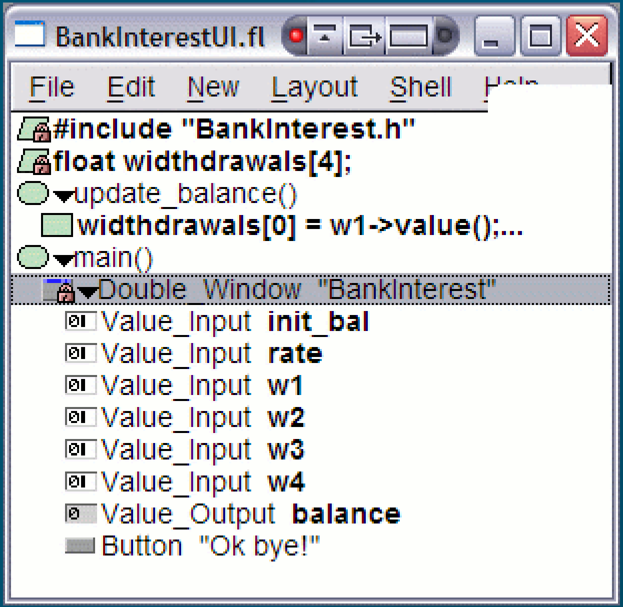


BankInterest

Starting balance:	100
Interest rate:	12.3
First Quarter widthdrawal:	20
Second Quarter widthdrawal:	10
Third Quarter widthdrawal:	50
Fourth Quarter widthdrawal:	25
Ending Balance:	3.81

- An event is a user action:
  - Click a button
  - Fill in a text box
  - Press a key
  - Move the mouse
- A callback is a procedure invoked by an event:
  - Close the window when user clicks “Ok bye!”
  - Draw a circle where the user clicks the mouse

Using Fluid

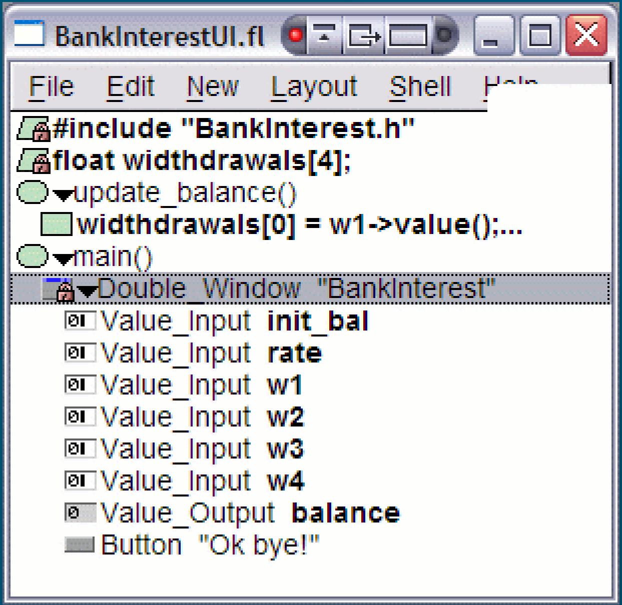


```
#include "BankInterest.h"
float widthdrawals[4];
update_balance()
  widthdrawals[0] = w1->value();...
main()
  Double_Window "BankInterest"
    Value_Input init_bal
    Value_Input rate
    Value_Input w1
    Value_Input w2
    Value_Input w3
    Value_Input w4
    Value_Output balance
    Button "Ok bye!"
```

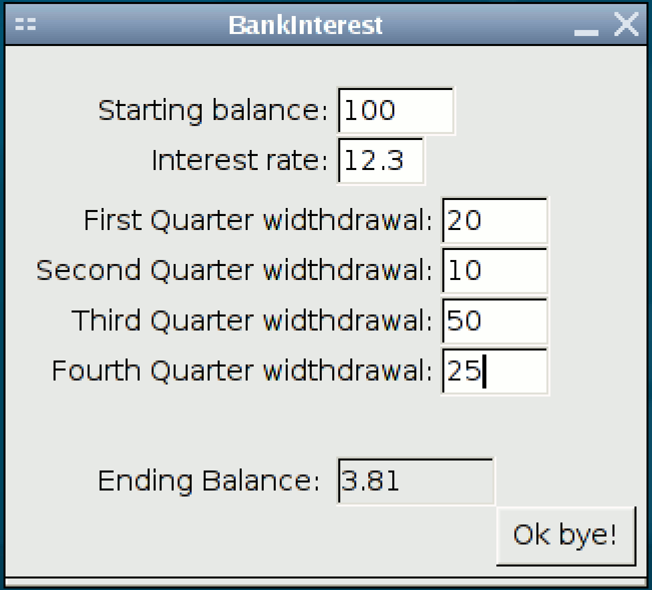
- Fluid is FLTK's interactive GUI designer
  - Drag and drop widgets
  - Write code blocks / callbacks
- Saves *.fl Fluid files; exports *.cxx/*.h code
- Compile and link this code into your program
- It is possible to write a whole program in Fluid
- But better to separate GUI from main program logic: form vs. function
  - Akin to HTML/CSS vs. PHP/ASP/JS

FLTK example: BankInterest

- BankInterestUI:
  - Just the user interface
  - Get values from the widgets
  - Minimal program logic
    - But I did choose to put main() here
- BankInterest:
  - Main program functionality
  - Provides functions invoked by UI callbacks
    - calc_balance()



```
#include "BankInterest.h"
float widthdrawals[4];
update_balance()
  widthdrawals[0] = w1->value();...
main()
  Double_Window "BankInterest"
    Value_Input init_bal
    Value_Input rate
    Value_Input w1
    Value_Input w2
    Value_Input w3
    Value_Input w4
    Value_Output balance
    Button "Ok bye!"
```



Starting balance:	100
Interest rate:	12.3
First Quarter widthdrawal:	20
Second Quarter widthdrawal:	10
Third Quarter widthdrawal:	50
Fourth Quarter widthdrawal:	25
Ending Balance:	3.81

Fluid and C++ program design

- Two ways of structuring your FLTK program:
- BankInterest example: main() in Fluid
    - #include separate file for core logic
- CubeView example: main() in separate C++ file
    - Fluid file: defines CubeViewUI class
        - which contains an Fl_Window
        - which contains a CubeView
            - which is a subclass of Fl_Gl_Window
    - CubeMain.cxx:
        - main() instantiates a CubeViewUI