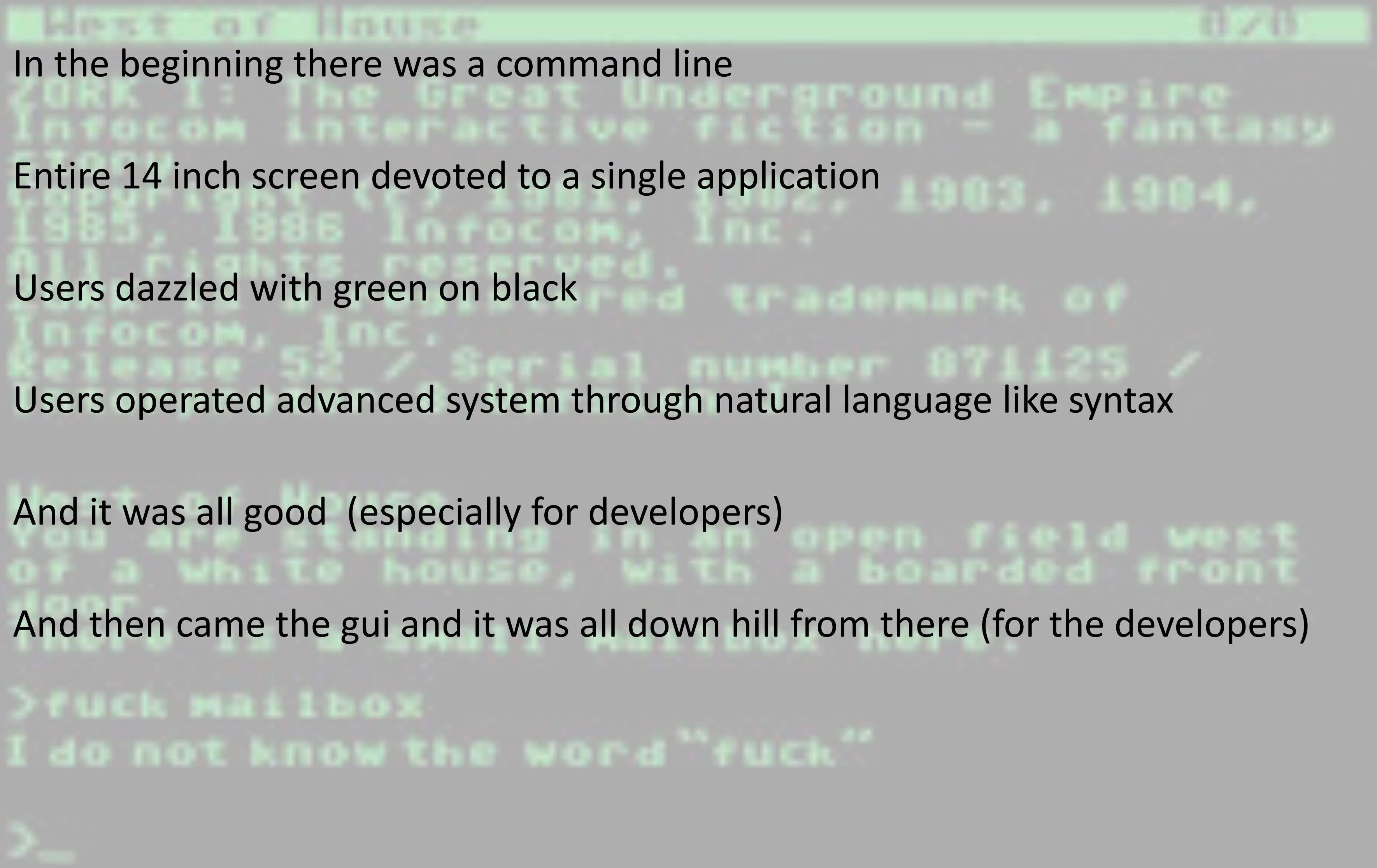
Interaktionsprogrammering TDDC73

Anders Fröberg
anders.froberg@liu.se

Outline

- Hopefully early lunch
- Historic perspective
  - Where are we coming from
  - Why do things look like they do
- Course outline



```
 West of House                               0/0
ZORK I: The Great Underground Empire
Infocom interactive fiction - a fantasy
story
Copyright (c) 1981, 1982, 1983, 1984,
1985, 1986 Infocom, Inc.
All rights reserved.
ZORK is a registered trademark of
Infocom, Inc.
Release 52 / Serial number 871125 /

West of House
You are standing in an open field west
of a white house, with a boarded front
door.
There is a small mailbox here.

>fuck mailbox
I do not know the word "fuck"

>_
```

In the beginning there was a command line

Entire 14 inch screen devoted to a single application

Users dazzled with green on black

Users operated advanced system through natural language like syntax

And it was all good (especially for developers)

And then came the gui and it was all down hill from there (for the developers)

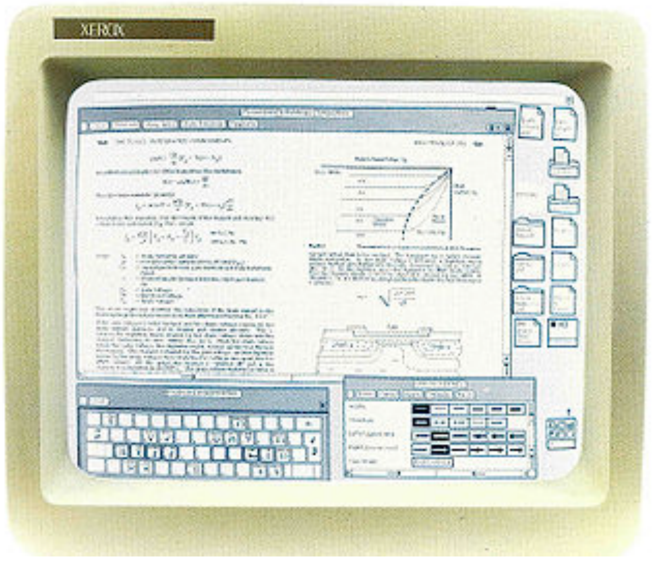
..and then came the GUI

..and then came the GUI

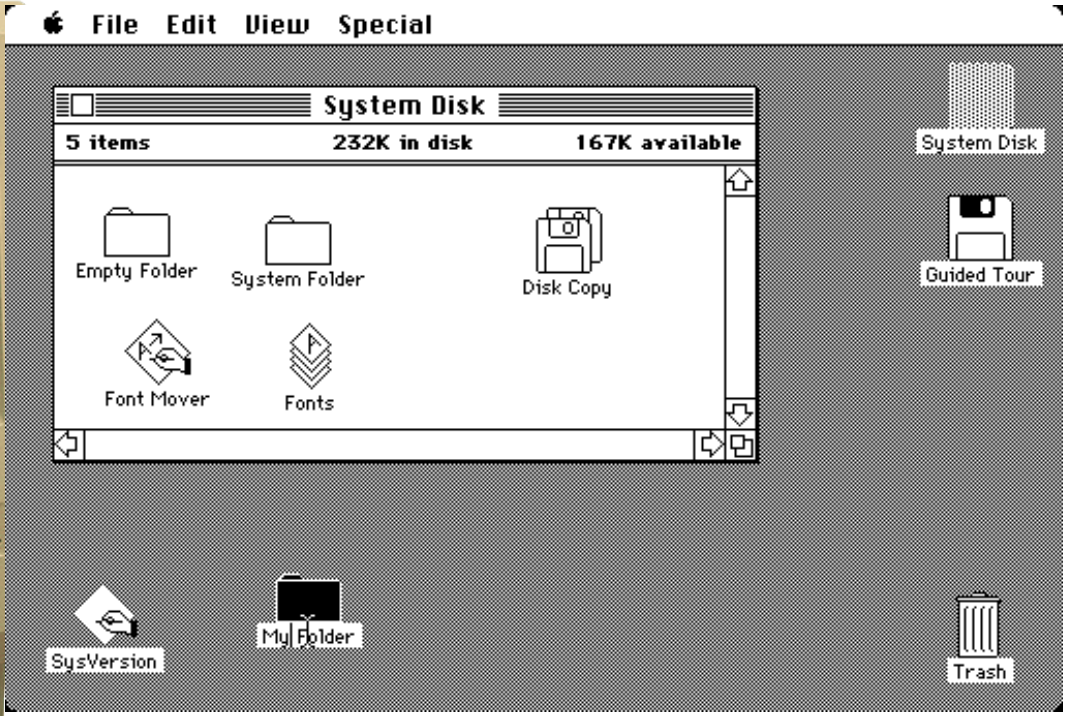
..and then came the GUI

..and then came the GUI

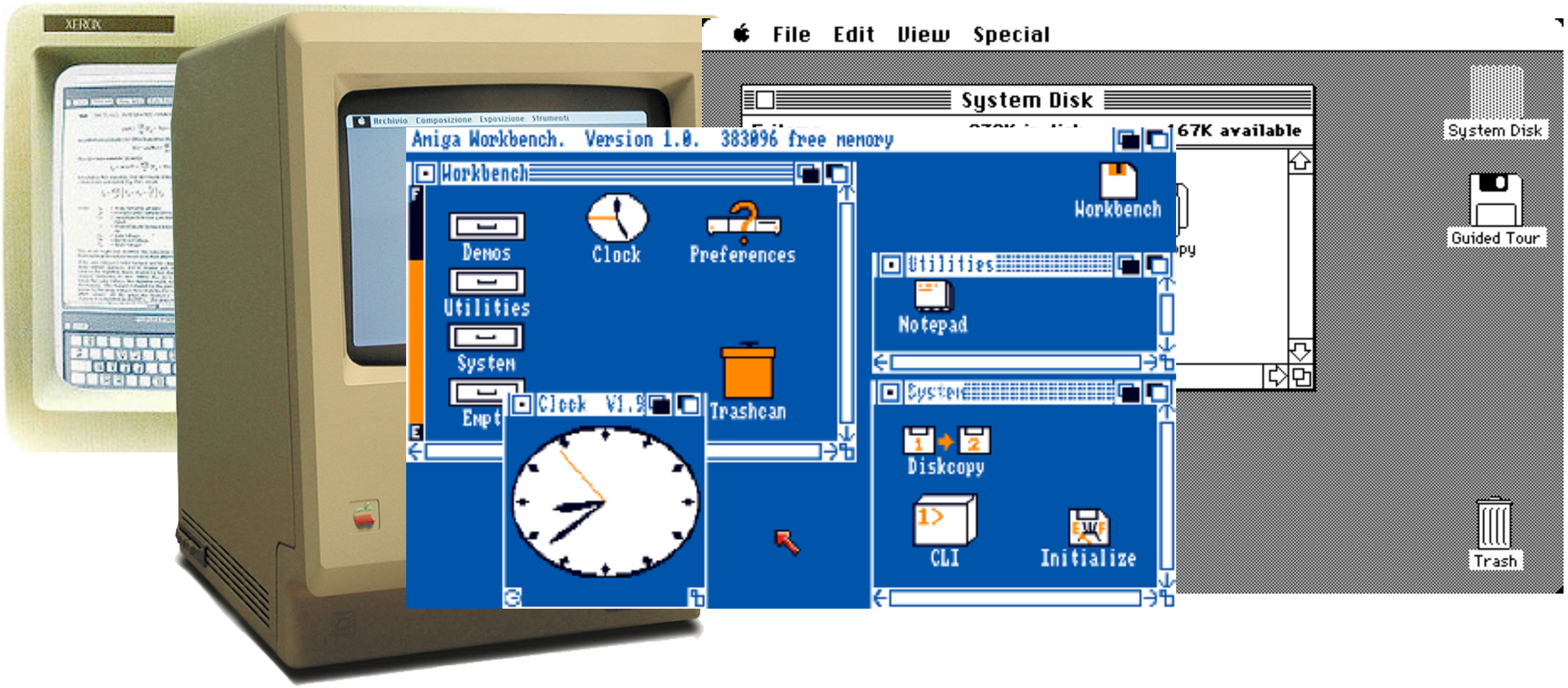
..and then came the GUI



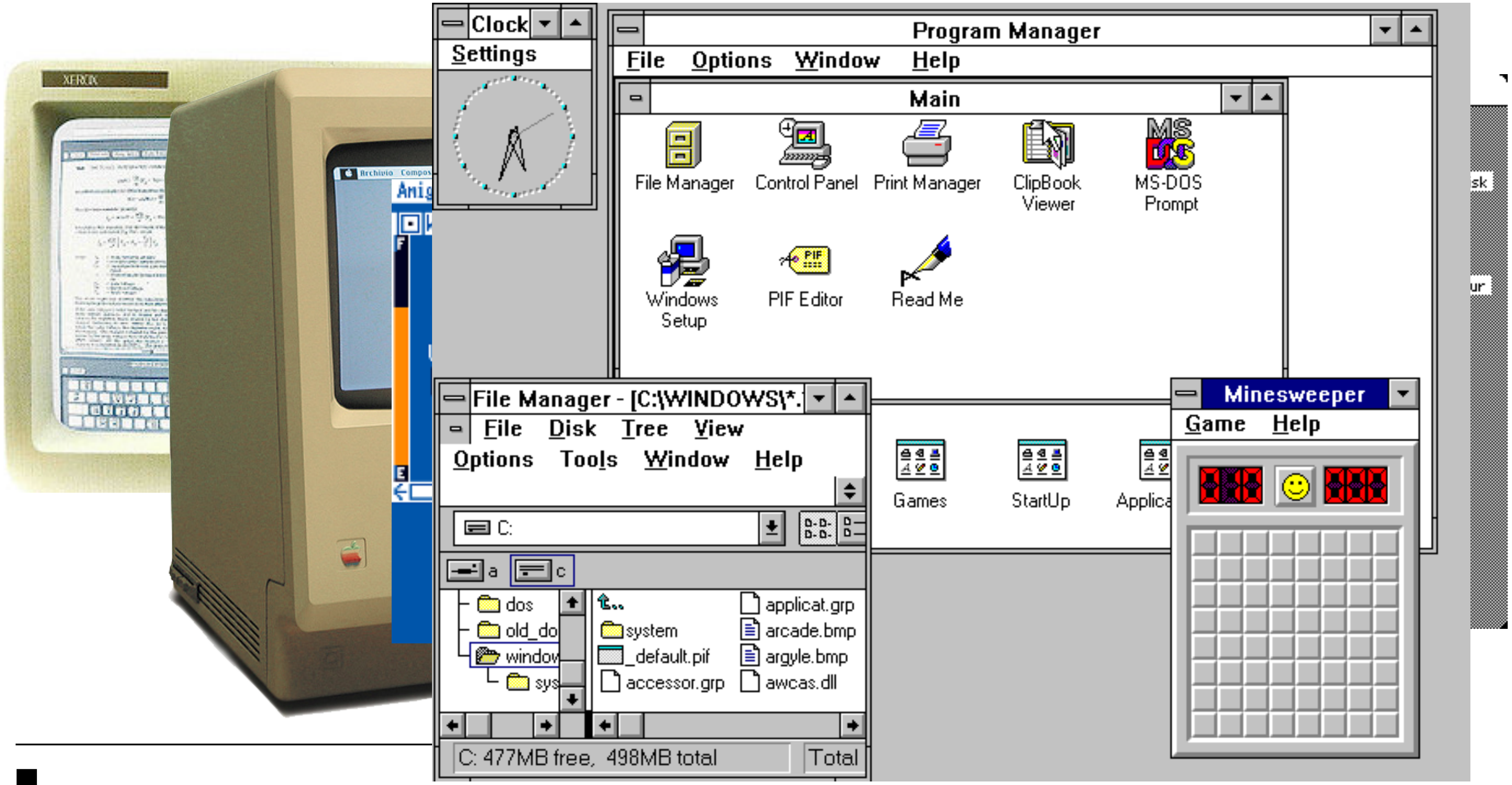
..and then came the GUI



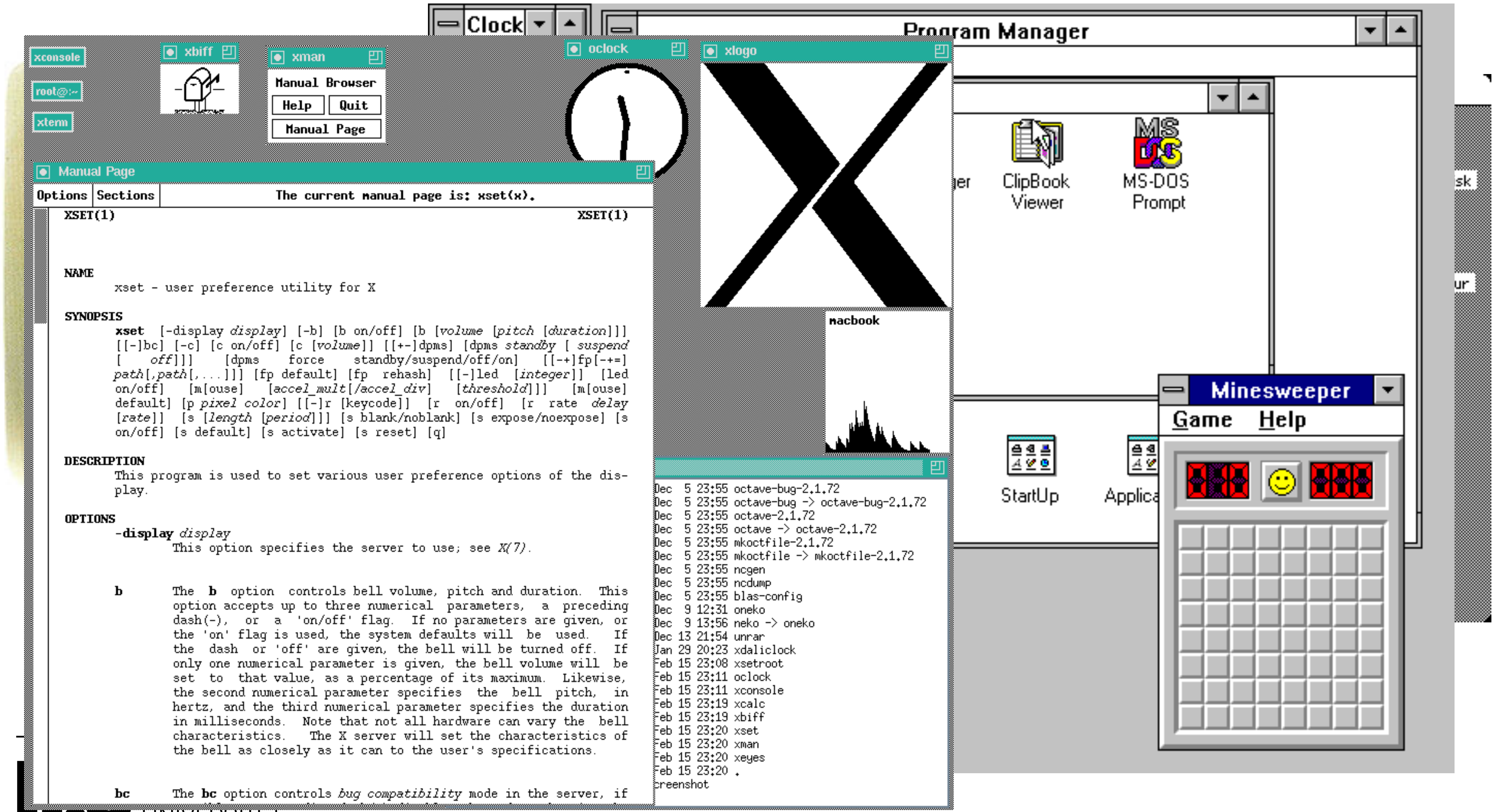
..and then came the GUI



..and then came the GUI



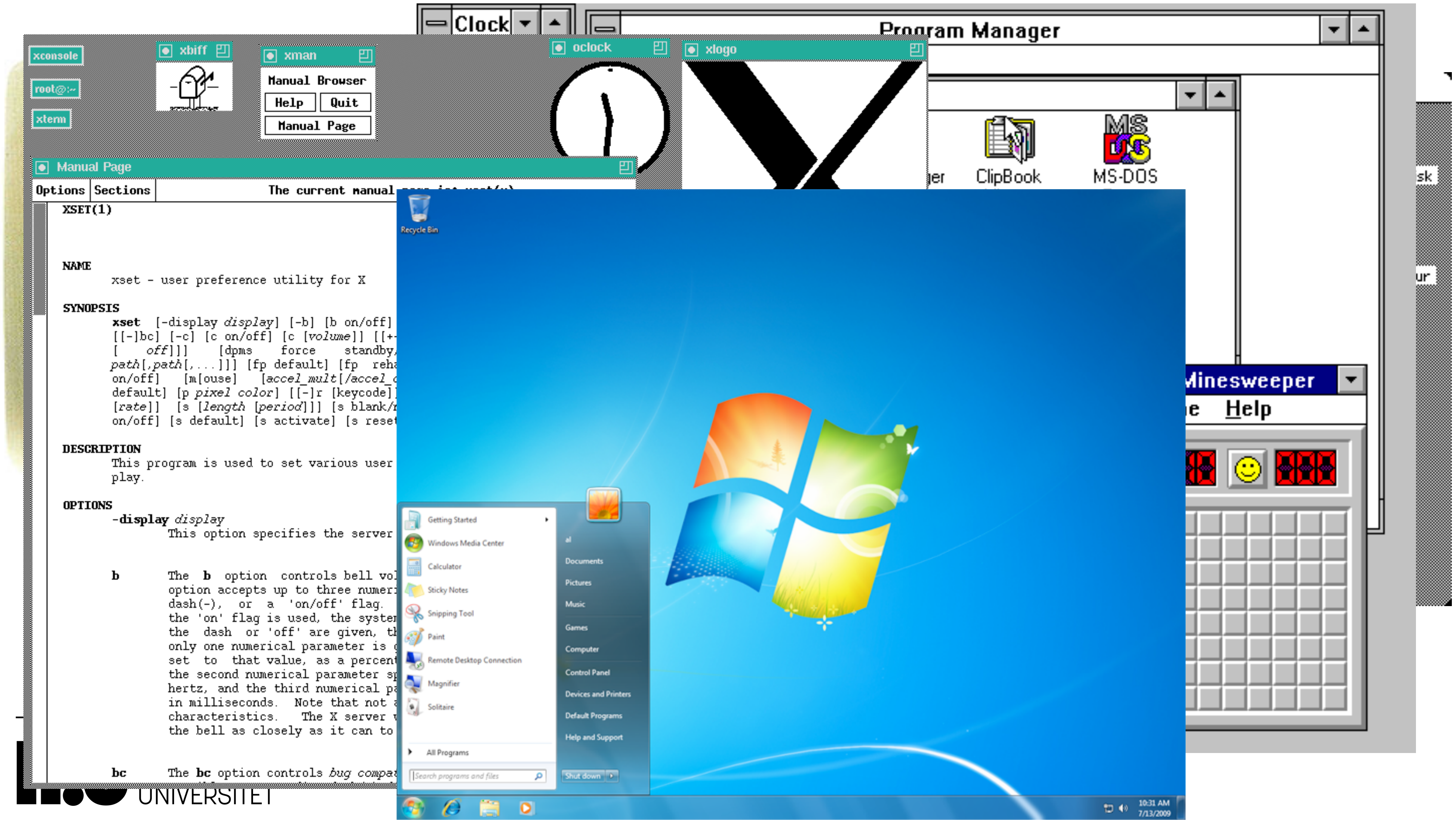
..and then came the GUI



```
The current manual page is: xset(x).

XSET(1)                                                    XSET(1)


NAME
       xset - user preference utility for X

SYNOPSIS
       xset  [-display display] [-b] [b on/off] [b [volume [pitch [duration]]]
       [[-]bc] [-c] [c on/off] [c [volume]] [[+-]dpms] [dpms standby [ suspend
       [    off]]]    [dpms    force    standby/suspend/off/on]    [[-+]fp[-+=]
       path[,path[,...]]] [fp default] [fp  rehash]  [[-]led  [integer]]  [led
       on/off]   [m[ouse]   [accel_mult[/accel_div]   [threshold]]]   [m[ouse]
       default] [p pixel color] [[-]r [keycode]]  [r  on/off]  [r  rate  delay
       [rate]]  [s [length [period]]] [s blank/noblank] [s expose/noexpose] [s
       on/off] [s default] [s activate] [s reset] [q]

DESCRIPTION
       This program is used to set various user preference options of the dis-
       play.

OPTIONS
       -display display
              This option specifies the server to use; see X(7).


       b      The  b  option  controls bell volume, pitch and duration.  This
              option accepts up to three numerical  parameters,  a  preceding
              dash(-),  or  a  'on/off' flag.  If no parameters are given, or
              the 'on' flag is used, the system defaults will  be  used.   If
              the  dash  or 'off' are given, the bell will be turned off.  If
              only one numerical parameter is given, the bell volume will  be
              set  to  that value, as a percentage of its maximum.  Likewise,
              the second numerical parameter specifies  the  bell  pitch,  in
              hertz, and the third numerical parameter specifies the duration
              in milliseconds.  Note that not all hardware can vary the  bell
              characteristics.   The X server will set the characteristics of
              the bell as closely as it can to the user's specifications.


       bc     The bc option controls bug compatibility mode in the server, if
```

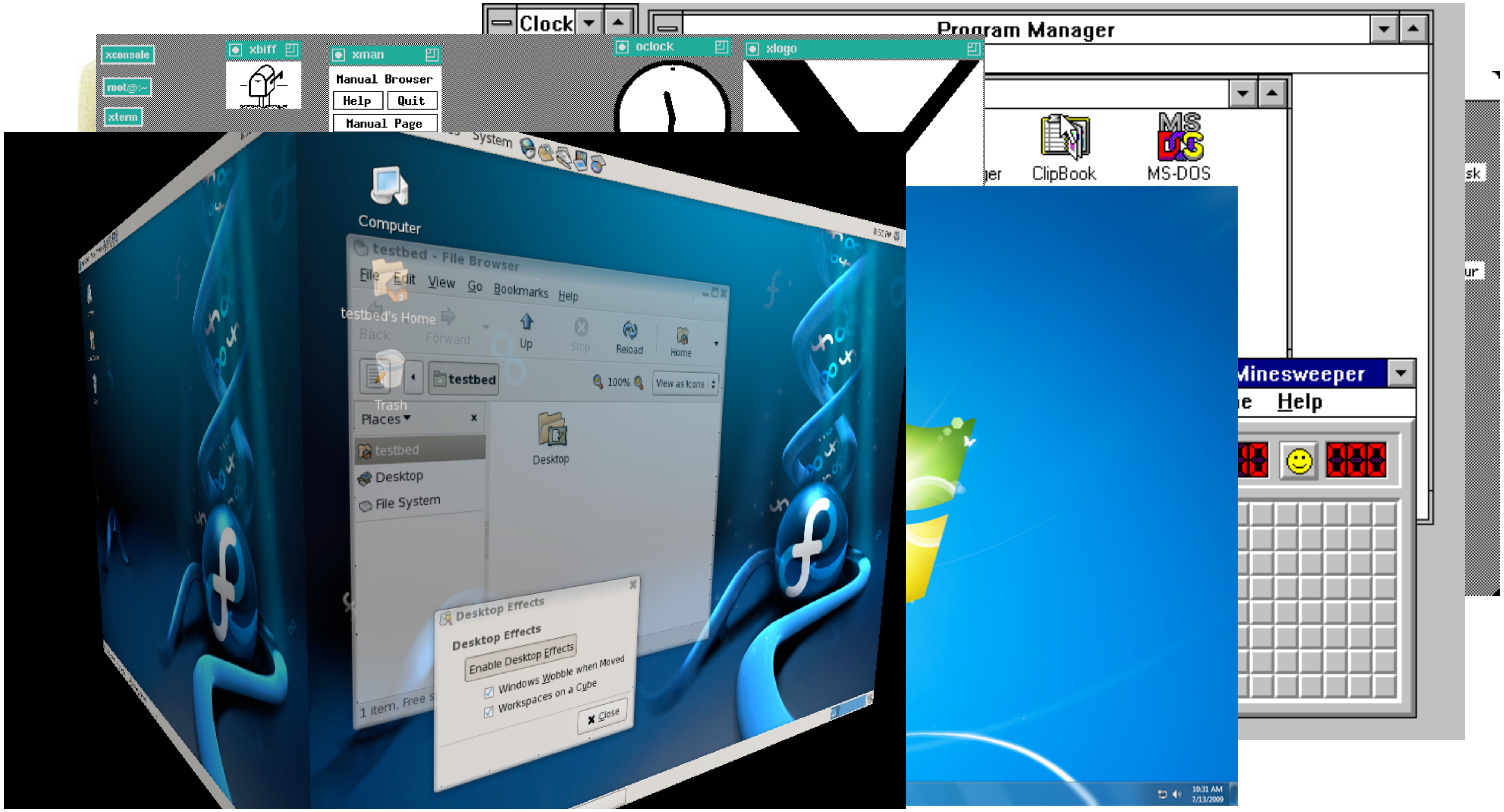
```
Dec  5 23:55 octave-bug-2.1.72
Dec  5 23:55 octave-bug -> octave-bug-2.1.72
Dec  5 23:55 octave-2.1.72
Dec  5 23:55 octave -> octave-2.1.72
Dec  5 23:55 mkoctfile-2.1.72
Dec  5 23:55 mkoctfile -> mkoctfile-2.1.72
Dec  5 23:55 ncgen
Dec  5 23:55 ncdump
Dec  5 23:55 blas-config
Dec  9 12:31 oneko
Dec  9 13:56 neko -> oneko
Dec 13 21:54 unrar
Jan 29 20:23 xdaliclock
Feb 15 23:08 xsetroot
Feb 15 23:11 oclock
Feb 15 23:11 xconsole
Feb 15 23:19 xcalc
Feb 15 23:19 xbiff
Feb 15 23:20 xset
Feb 15 23:20 xman
Feb 15 23:20 xeyes
Feb 15 23:20 .
creenshot
```

..and then came the GUI



```
Manual Page
Options  Sections            The current manual
XSET(1)


NAME
       xset - user preference utility for X

SYNOPSIS
       xset  [-display display] [-b] [b on/off]
       [[-]bc] [-c] [c on/off] [c [volume]] [[+-
       [    off]]]   [dpms    force    standby,
       path[,path[,...]]] [fp default] [fp  reha
       on/off]   [m[ouse]   [accel_mult[/accel_
       default] [p pixel color] [[-]r [keycode]
       [rate]]  [s [length [period]]] [s blank/
       on/off] [s default] [s activate] [s reset

DESCRIPTION
       This program is used to set various user
       play.

OPTIONS
       -display display
              This option specifies the server


       b      The  b  option  controls bell vol
              option accepts up to three numeri
              dash(-),  or  a  'on/off' flag.
              the 'on' flag is used, the syster
              the  dash  or 'off' are given, th
              only one numerical parameter is
              set  to  that value, as a percent
              the second numerical parameter sp
              hertz, and the third numerical pa
              in milliseconds.  Note that not a
              characteristics.   The X server v
              the bell as closely as it can to


       bc     The bc option controls bug compat
```

..and then came the GUI



1:14 PM
07/12 | WINNIPEG, 23°C
SAT Jul 12
TOMORROW, JUL 13
Meeting
10 AM – 12 PM
MON, JUL 14
Seth dr apt
2:30–3:30 PM
Dr Martinez
Karate
Google
Email
Travel
People
Office
Finance
Facebook
Settings
Clock
oclock
xlogo
xconsole
xterm
9:00 AM
Categories
#Woman
#Man
#Baby
#Travel
#Tech
#Food&Drink
Places
testbed
Desktop
File System
Trash
Up
Stop
Reload
Home
View as Icons
Desktop Effects
Enable Desktop Effects
Windows Wobble when Moved
Workspaces on a Cube
Close

Categories
Screen Mirroring
IPHONE
Music
Desktop Effects
Enable Desktop Effects
Windows Wobble when Moved
Workspaces on a Cube
Close
Places
testbed
Desktop
File System
Trash
Reload
Home
Clock
Office
Settings
August
70°F
9:00 AM
1:14 PM

Command line ->GUI

- New problem arises for developers
- Complex interaction structure
  - Elaborate graphics
  - Multiple interaction mechanisms
  - Non-linear command structure

Solutions

- Iterative methods
- Software Development Kits (SDKs)
  - Swing
  - Motif
  - .Net Forms
  - React React-Native
  - Flutter
  - Vue
  - Svelte
- GUI builders
- Specialized GUI languages

Some stats

- Already in ’94 almost all Unix contained a GUI
- Half to code is GUI code
- Time spend equal all other parts together
- Benefit Tools
  - Reduces code by 83%
  - Time spend reduced by a factor 4 (Building GUI)

So what is this UI thing

Wikipedia

The user interface (also known as human computer interface or man-machine interface (MMI)) is the aggregate of means by which people—the users—interact with the system—a particular machine, device, computer program or other complex tool. The user interface provides means of:

- Input, allowing the users to manipulate a system
- Output, allowing the system to indicate the effects of the users' manipulation.

Wiktionary

Noun

user interface (plural user interfaces)

1. (countable) The part of a software application that a user sees and interacts with.

GUI for programmers

- Two main components
  - User(s)
  - System/Program/Application/Hardware
- Two pipes of information
  - Input from the User(s)
  - Output to the User(s)

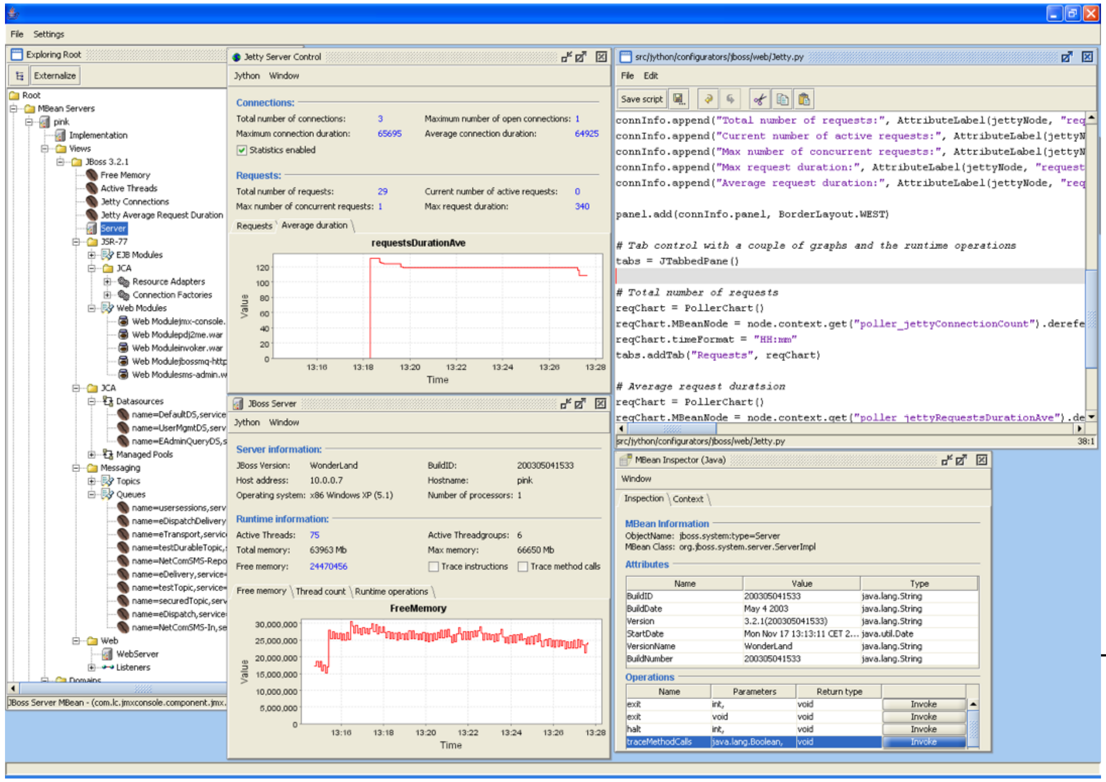
GUI for programmers

- Two main components
  - User(s)
  - System/Program/Application/Hardware
- Two pipes of information
  - Input from the User(s)
  - Output to the User(s)

Focus

What constitutes a (G)UI

- The worlds smallest GUI ○



Connections:
Total number of connections: 3 Maximum number of open connections: 1
Maximum connection duration: 65695 Average connection duration: 64925
[✓] Statistics enabled

Requests:
Total number of requests: 29 Current number of active requests: 0
Max number of concurrent requests: 1 Max request duration: 340

Requests | Average duration

```
connInfo.append("Total number of requests:", AttributeLabel(jettyNode, "req
connInfo.append("Current number of active requests:", AttributeLabel(jettyN
connInfo.append("Max number of concurrent requests:", AttributeLabel(jettyN
connInfo.append("Max request duration:", AttributeLabel(jettyNode, "request
connInfo.append("Average request duration:", AttributeLabel(jettyNode, "req

panel.add(connInfo.panel, BorderLayout.WEST)

# Tab control with a couple of graphs and the runtime operations
tabs = JTabbedPane()

# Total number of requests
reqChart = PollerChart()
reqChart.MBeanNode = node.context.get("poller_jettyConnectionCount").derefe
reqChart.timeFormat = "HH:mm"
tabs.addTab("Requests", reqChart)

# Average request duratsion
reqChart = PollerChart()
reqChart.MBeanNode = node.context.get("poller_jettyRequestsDurationAve").de
```

Server information:
JBoss Version: WonderLand BuildID: 200305041533
Host address: 10.0.0.7 Hostname: pink
Operating system: x86 Windows XP (5.1) Number of processors: 1

Runtime information:
Active Threads: 75 Active Threadgroups: 6
Total memory: 63963 Mb Max memory: 66650 Mb
Free memory: 24470456 [ ] Trace instructions [ ] Trace method calls

Free memory | Thread count | Runtime operations

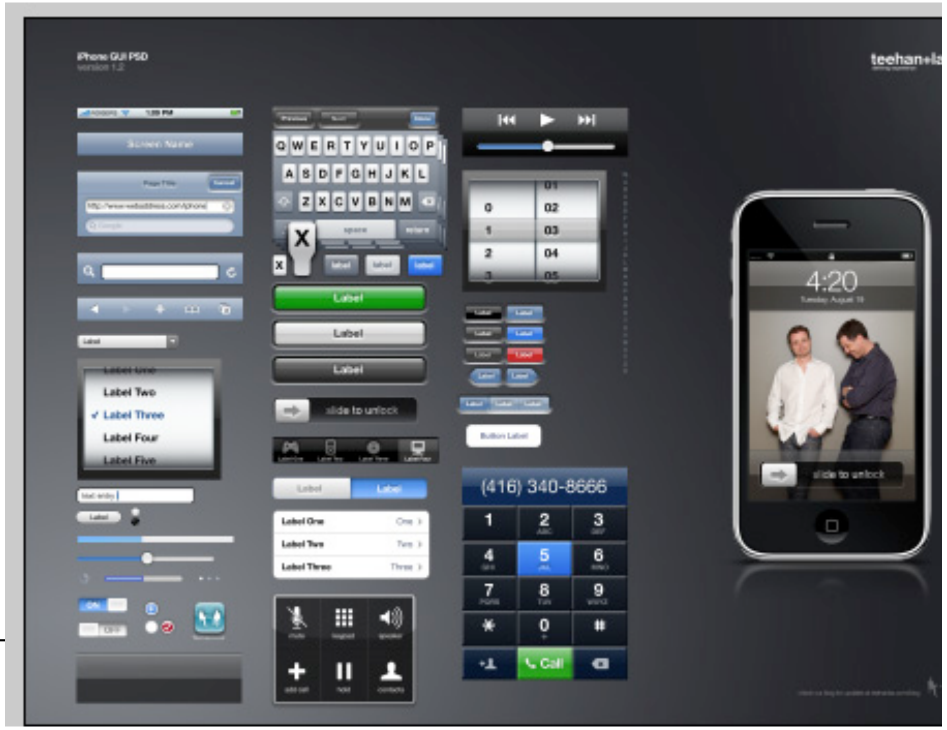
MBean Information
ObjectName: jboss.system:type=Server
MBean Class: org.jboss.system.server.ServerImpl

Attributes

Name	Value	Type
BuildID	200305041533	java.lang.String
BuildDate	May 4 2003	java.lang.String
Version	3.2.1(200305041533)	java.lang.String
StartDate	Mon Nov 17 13:13:11 CET 2...	java.util.Date
VersionName	WonderLand	java.lang.String
BuildNumber	200305041533	java.lang.String

Operations

Name	Parameters	Return type	
exit	int,	void	Invoke
exit	void	void	Invoke
halt	int,	void	Invoke
traceMethodCalls	java.lang.Boolean,	void	Invoke



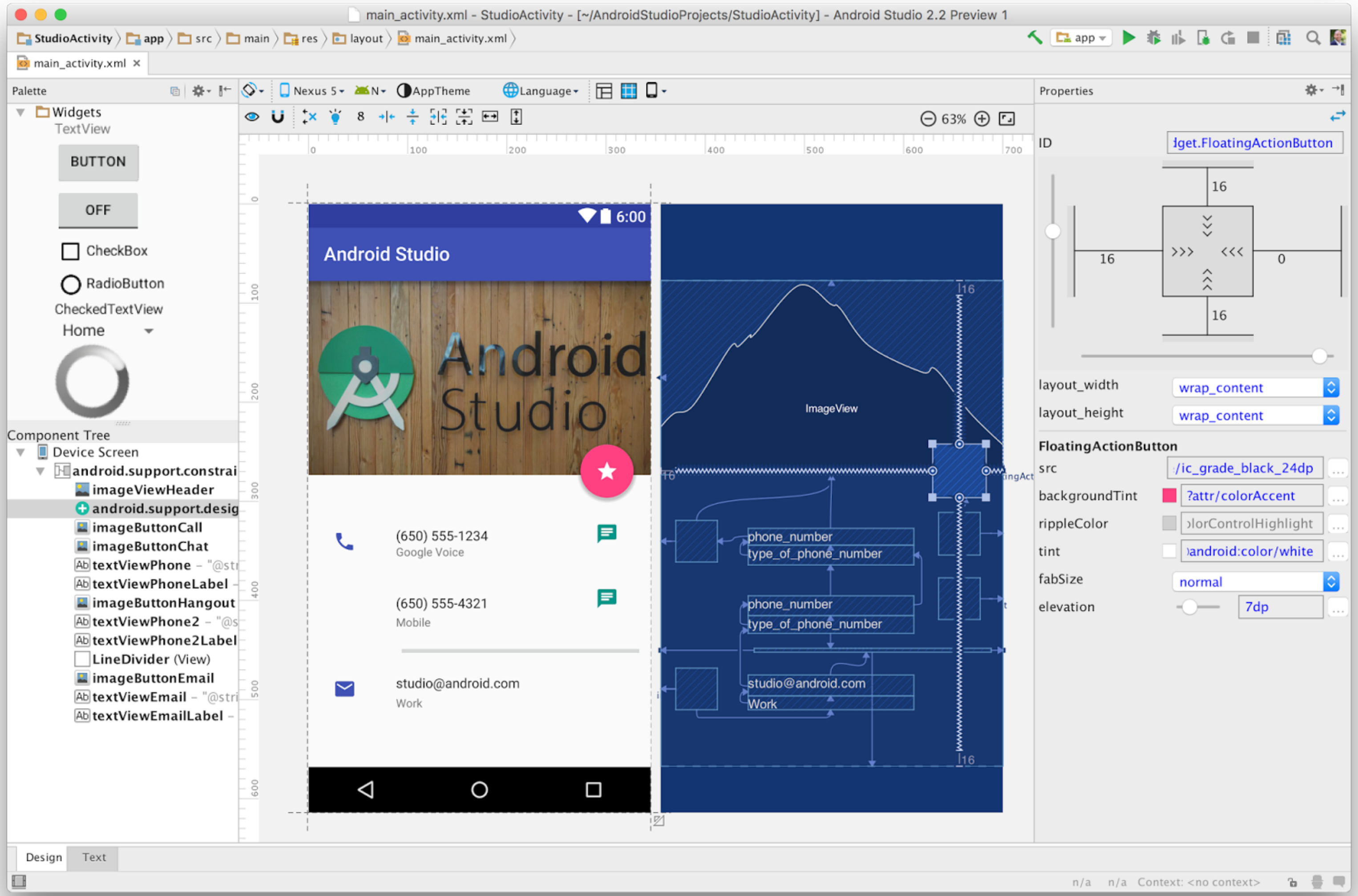
A GUI for a programmer

```
public void
paint(graphics g){
g.moveTo(100,100);
g.setColor(“ffffff”);
g.drawlineTo(200,300);
}
```

Spans from low level to very high level

```
val intent = Intent()
intent.type = "image/*"
intent.action = Intent.ACTION_GET_CONTENT
startActivityForResult(
        Intent.createChooser(intent, "Select Picture"), PICK_IMAGE);
```

...and even visual



main_activity.xml - StudioActivity - [~/AndroidStudioProjects/StudioActivity] - Android Studio 2.2 Preview 1

StudioActivity 〉 app 〉 src 〉 main 〉 res 〉 layout 〉 main_activity.xml 〉

Component Tree
- Device Screen
  - android.support.constrai
    - imageViewHeader
    - android.support.desig
    - imageButtonCall
    - imageButtonChat
    - textViewPhone – "@str
    - textViewPhoneLabel –
    - imageButtonHangout
    - textViewPhone2 – "@s
    - textViewPhone2Label
    - LineDivider (View)
    - imageButtonEmail
    - textViewEmail – "@stri
    - textViewEmailLabel –

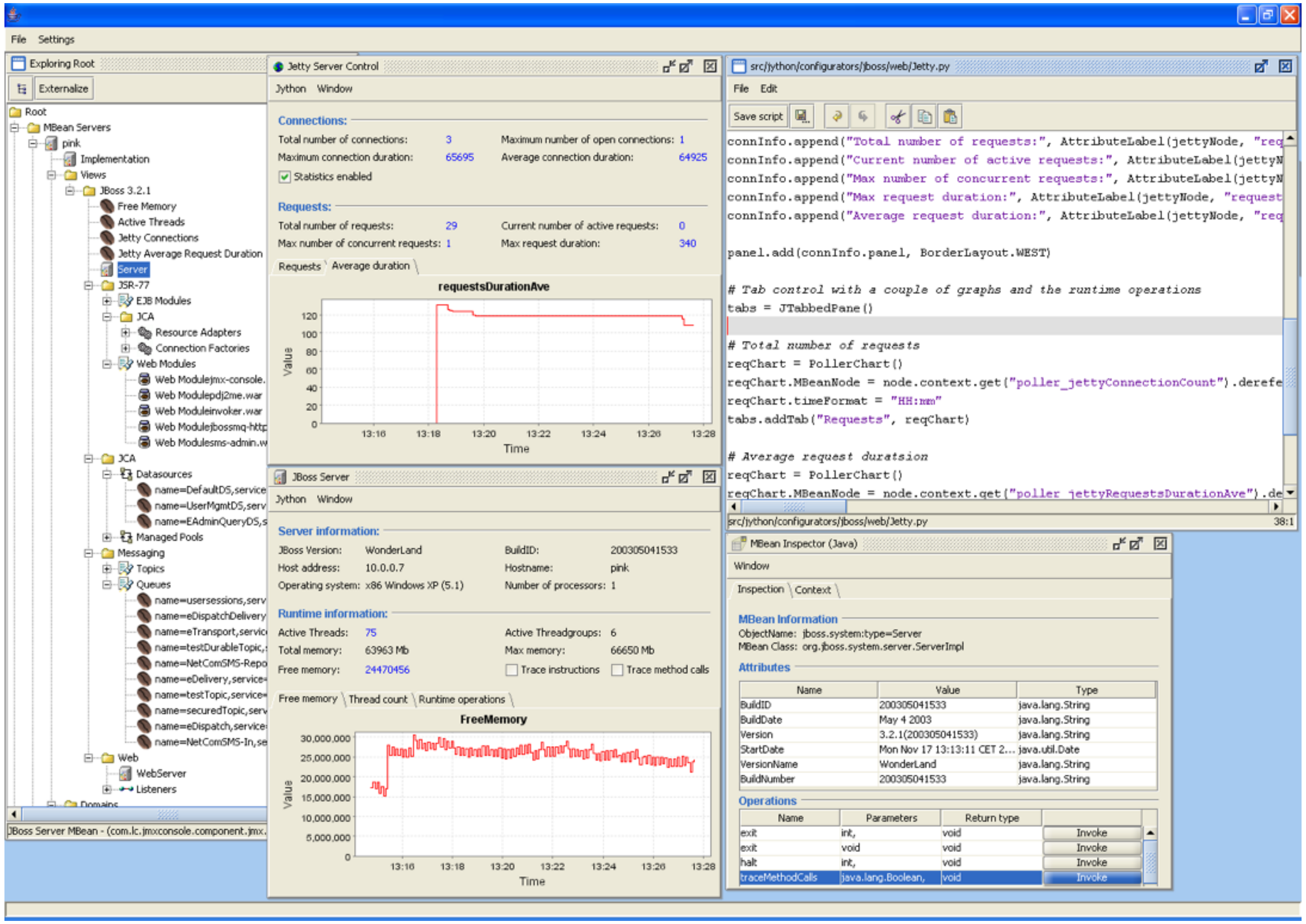
Properties

ID	ıget.FloatingActionButton
layout_width	wrap_content
layout_height	wrap_content

FloatingActionButton

src	/ic_grade_black_24dp
backgroundTint	?attr/colorAccent
rippleColor	ɔlorControlHighlight
tint	ıandroid:color/white
fabSize	normal
elevation	7dp

GUI for the User



```
Root
  MBean Servers
    pink
      Implementation
      Views
        JBoss 3.2.1
          Free Memory
          Active Threads
          Jetty Connections
          Jetty Average Request Duration
          Server
          JSR-77
            EJB Modules
            JCA
              Resource Adapters
              Connection Factories
            Web Modules
              Web Modulejmx-console.
              Web Modulepdj2me.war
              Web Moduleinvoker.war
              Web Modulejbossmq-http
              Web Modulesms-admin.w
          JCA
            Datasources
              name=DefaultDS,service
              name=UserMgmtDS,serv
              name=EAdminQueryDS,s
            Managed Pools
          Messaging
            Topics
            Queues
              name=usersessions,serv
              name=eDispatchDelivery
              name=eTransport,servic
              name=testDurableTopic,:
              name=NetComSMS-Repo
              name=eDelivery,service=
              name=testTopic,service=
              name=securedTopic,serv
              name=eDispatch,service
              name=NetComSMS-In,se
          Web
            WebServer
            Listeners
      Domains
JBoss Server MBean - (com.lc.jmxconsole.component.jmx.
```

Connections:

Total number of connections:	3	Maximum number of open connections:	1
Maximum connection duration:	65695	Average connection duration:	64925

[✓] Statistics enabled

Requests:

Total number of requests:	29	Current number of active requests:	0
Max number of concurrent requests:	1	Max request duration:	340

Server information:

JBoss Version:	WonderLand	BuildID:	200305041533
Host address:	10.0.0.7	Hostname:	pink
Operating system:	x86 Windows XP (5.1)	Number of processors:	1

Runtime information:

Active Threads:	75	Active Threadgroups:	6
Total memory:	63963 Mb	Max memory:	66650 Mb
Free memory:	24470456	☐ Trace instructions	☐ Trace method calls

src/jython/configurators/jboss/web/Jetty.py

```
connInfo.append("Total number of requests:", AttributeLabel(jettyNode, "req
connInfo.append("Current number of active requests:", AttributeLabel(jettyN
connInfo.append("Max number of concurrent requests:", AttributeLabel(jettyN
connInfo.append("Max request duration:", AttributeLabel(jettyNode, "request
connInfo.append("Average request duration:", AttributeLabel(jettyNode, "req

panel.add(connInfo.panel, BorderLayout.WEST)

# Tab control with a couple of graphs and the runtime operations
tabs = JTabbedPane()

# Total number of requests
reqChart = PollerChart()
reqChart.MBeanNode = node.context.get("poller_jettyConnectionCount").derefe
reqChart.timeFormat = "HH:mm"
tabs.addTab("Requests", reqChart)

# Average request duratsion
reqChart = PollerChart()
reqChart.MBeanNode = node.context.get("poller_jettyRequestsDurationAve").de
```

src/jython/configurators/jboss/web/Jetty.py 38:1

MBean Information

ObjectName: jboss.system:type=Server
MBean Class: org.jboss.system.server.ServerImpl

Attributes

Name	Value	Type
BuildID	200305041533	java.lang.String
BuildDate	May 4 2003	java.lang.String
Version	3.2.1(200305041533)	java.lang.String
StartDate	Mon Nov 17 13:13:11 CET 2...	java.util.Date
VersionName	WonderLand	java.lang.String
BuildNumber	200305041533	java.lang.String

Operations

Name	Parameters	Return type	
exit	int,	void	Invoke
exit	void	void	Invoke
halt	int,	void	Invoke
traceMethodCalls	java.lang.Boolean,	void	Invoke

And GUI for You in this course

Android/Java

```
protected void onDraw(Canvas canvas) {
  super.onDraw(canvas);
  // Draw the shadow
  canvas.drawOval( mShadowBounds, mShadowPaint );

  canvas.drawText(mData.get(mCurrentItem).mLabel, mTextX, mTextY, mTextPaint);
```

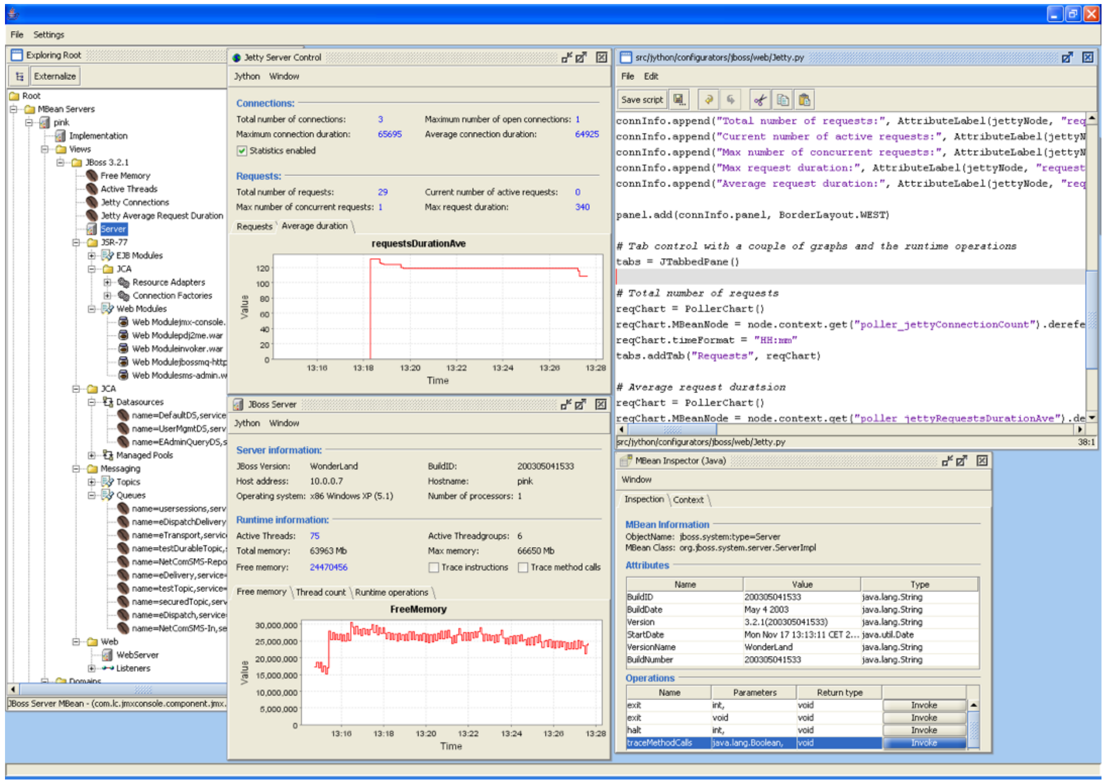
Spans from low level to very high level

Android/Kotlin

```
val intent = Intent()
intent.type = "image/*"
intent.action = Intent.ACTION_GET_CONTENT
startActivityForResult(
      Intent.createChooser(intent, "Select Picture"), PICK_IMAGE);
```

How do we get from

What user wants and needs (user/system requirements)



Connections:

Total number of connections: 3 Maximum number of open connections: 1
Maximum connection duration: 65695 Average connection duration: 64925
[✓] Statistics enabled

Requests:

Total number of requests: 29 Current number of active requests: 0
Max number of concurrent requests: 1 Max request duration: 340

Server information:

JBoss Version: WonderLand BuildID: 200305041533
Host address: 10.0.0.7 Hostname: pink
Operating system: x86 Windows XP (5.1) Number of processors: 1

Runtime information:

Active Threads: 75 Active Threadgroups: 6
Total memory: 63963 Mb Max memory: 66650 Mb
Free memory: 24470456 ☐ Trace instructions ☐ Trace method calls

```
connInfo.append("Total number of requests:", AttributeLabel(jettyNode, "req
connInfo.append("Current number of active requests:", AttributeLabel(jettyN
connInfo.append("Max number of concurrent requests:", AttributeLabel(jettyN
connInfo.append("Max request duration:", AttributeLabel(jettyNode, "request
connInfo.append("Average request duration:", AttributeLabel(jettyNode, "req

panel.add(connInfo.panel, BorderLayout.WEST)

# Tab control with a couple of graphs and the runtime operations
tabs = JTabbedPane()

# Total number of requests
reqChart = PollerChart()
reqChart.MBeanNode = node.context.get("poller_jettyConnectionCount").derefe
reqChart.timeFormat = "HH:mm"
tabs.addTab("Requests", reqChart)

# Average request duratsion
reqChart = PollerChart()
reqChart.MBeanNode = node.context.get("poller_jettyRequestsDurationAve").de
```

src/jython/configurators/jboss/web/Jetty.py 38:1

MBean Information

ObjectName: jboss.system:type=Server
MBean Class: org.jboss.system.server.ServerImpl

Attributes

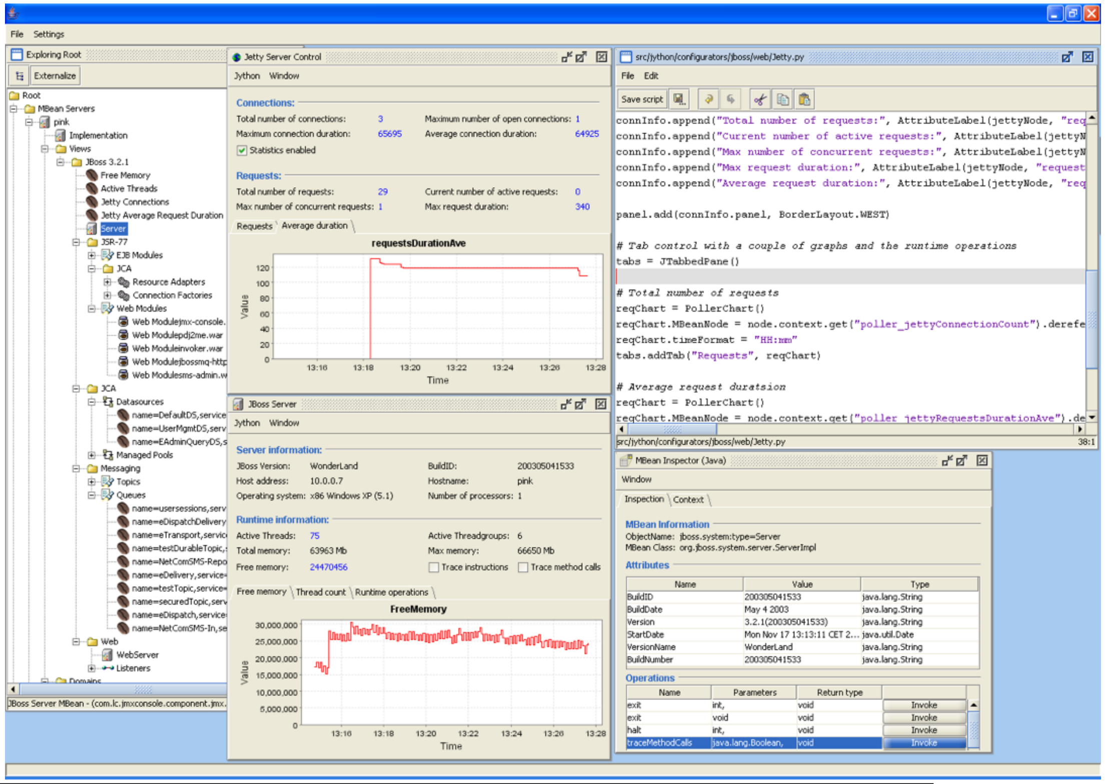
Name	Value	Type
BuildID	200305041533	java.lang.String
BuildDate	May 4 2003	java.lang.String
Version	3.2.1(200305041533)	java.lang.String
StartDate	Mon Nov 17 13:13:11 CET 2...	java.util.Date
VersionName	WonderLand	java.lang.String
BuildNumber	200305041533	java.lang.String

Operations

Name	Parameters	Return type	
exit	int,	void	Invoke
exit	void	void	Invoke
halt	int,	void	Invoke
traceMethodCalls	java.lang.Boolean,	void	Invoke

How do we get from

```
protected void onDraw(Canvas canvas) {
  super.onDraw(canvas);
  // Draw the shadow
  canvas.drawOval( mShadowBounds, mShadowPaint );

  canvas.drawText(mData.get(mCurrentItem).mLabel, mTextX, mTextY, mTextPaint);
```



```
Root
  MBean Servers
    pink
      Implementation
      Views
        JBoss 3.2.1
          Free Memory
          Active Threads
          Jetty Connections
          Jetty Average Request Duration
          Server
        JSR-77
          EJB Modules
          JCA
            Resource Adapters
            Connection Factories
          Web Modules
            Web Modulejmx-console.
            Web Modulepdj2me.war
            Web Moduleinvoker.war
            Web Modulejbossmq-http
            Web Modulesms-admin.w
        JCA
          Datasources
            name=DefaultDS,service
            name=UserMgmtDS,serv
            name=EAdminQueryDS,s
          Managed Pools
        Messaging
          Topics
          Queues
            name=usersessions,serv
            name=eDispatchDelivery
            name=eTransport,servic
            name=testDurableTopic,
            name=NetComSMS-Repo
            name=eDelivery,service=
            name=testTopic,service=
            name=securedTopic,serv
            name=eDispatch,service
            name=NetComSMS-In,se
        Web
          WebServer
          Listeners
      Domains
JBoss Server MBean - (com.lc.jmxconsole.component.jmx.
```

Connections:

Total number of connections:	3	Maximum number of open connections:	1
Maximum connection duration:	65695	Average connection duration:	64925

[✓] Statistics enabled

Requests:

Total number of requests:	29	Current number of active requests:	0
Max number of concurrent requests:	1	Max request duration:	340

Server information:

JBoss Version:	WonderLand	BuildID:	200305041533
Host address:	10.0.0.7	Hostname:	pink
Operating system:	x86 Windows XP (5.1)	Number of processors:	1

Runtime information:

Active Threads:	75	Active Threadgroups:	6
Total memory:	63963 Mb	Max memory:	66650 Mb
Free memory:	24470456	☐ Trace instructions	☐ Trace method calls

```
connInfo.append("Total number of requests:", AttributeLabel(jettyNode, "req
connInfo.append("Current number of active requests:", AttributeLabel(jettyN
connInfo.append("Max number of concurrent requests:", AttributeLabel(jettyN
connInfo.append("Max request duration:", AttributeLabel(jettyNode, "request
connInfo.append("Average request duration:", AttributeLabel(jettyNode, "req

panel.add(connInfo.panel, BorderLayout.WEST)

# Tab control with a couple of graphs and the runtime operations
tabs = JTabbedPane()

# Total number of requests
reqChart = PollerChart()
reqChart.MBeanNode = node.context.get("poller_jettyConnectionCount").derefe
reqChart.timeFormat = "HH:mm"
tabs.addTab("Requests", reqChart)

# Average request duratsion
reqChart = PollerChart()
reqChart.MBeanNode = node.context.get("poller_jettyRequestsDurationAve").de
```

src/jython/configurators/jboss/web/Jetty.py 38:1

MBean Information

ObjectName: jboss.system:type=Server
MBean Class: org.jboss.system.server.ServerImpl

Attributes

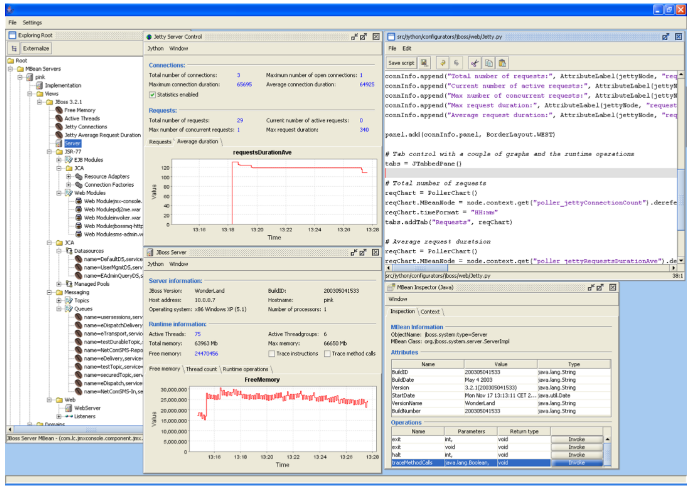
Name	Value	Type
BuildID	200305041533	java.lang.String
BuildDate	May 4 2003	java.lang.String
Version	3.2.1(200305041533)	java.lang.String
StartDate	Mon Nov 17 13:13:11 CET 2...	java.util.Date
VersionName	WonderLand	java.lang.String
BuildNumber	200305041533	java.lang.String

Operations

Name	Parameters	Return type	
exit	int,	void	Invoke
exit	void	void	Invoke
halt	int,	void	Invoke
traceMethodCalls	java.lang.Boolean,	void	Invoke

Solution

SDKs
for (gui-)code:
How to build it



Connections:
Total number of connections: 3 Maximum number of open connections: 1
Maximum connection duration: 65695 Average connection duration: 64925
Statistics enabled

Requests:
Total number of requests: 29 Current number of active requests: 0
Max number of concurrent requests: 1 Max request duration: 340

Server information:
JBoss Version: WonderLand BuildID: 200305041533
Host address: 10.0.0.7 Hostname: pink
Operating system: x86 Windows XP (5.1) Number of processors: 1

Runtime information:
Active Threads: 75 Active Threadgroups: 6
Total memory: 63963 Mb Max memory: 66650 Mb
Free memory: 24470456 Trace instructions Trace method calls

```
connInfo.append("Total number of requests:", AttributeLabel(jettyNode, "req
connInfo.append("Current number of active requests:", AttributeLabel(jettyN
connInfo.append("Max number of concurrent requests:", AttributeLabel(jettyN
connInfo.append("Max request duration:", AttributeLabel(jettyNode, "request
connInfo.append("Average request duration:", AttributeLabel(jettyNode, "req

panel.add(connInfo.panel, BorderLayout.WEST)

# Tab control with a couple of graphs and the runtime operations
tabs = JTabbedPane()

# Total number of requests
reqChart = PollerChart()
reqChart.MBeanNode = node.context.get("poller_jettyConnectionCount").derefe
reqChart.timeFormat = "HH:mm"
tabs.addTab("Requests", reqChart)

# Average request duratsion
reqChart = PollerChart()
reqChart.MBeanNode = node.context.get("poller_jettyRequestsDurationAve").de
```

src/jython/configurators/jboss/web/Jetty.py 38:1

MBean Information
ObjectName: jboss.system:type=Server
MBean Class: org.jboss.system.server.ServerImpl

Attributes

Name	Value	Type
BuildID	200305041533	java.lang.String
BuildDate	May 4 2003	java.lang.String
Version	3.2.1(200305041533)	java.lang.String
StartDate	Mon Nov 17 13:13:11 CET 2...	java.util.Date
VersionName	WonderLand	java.lang.String
BuildNumber	200305041533	java.lang.String

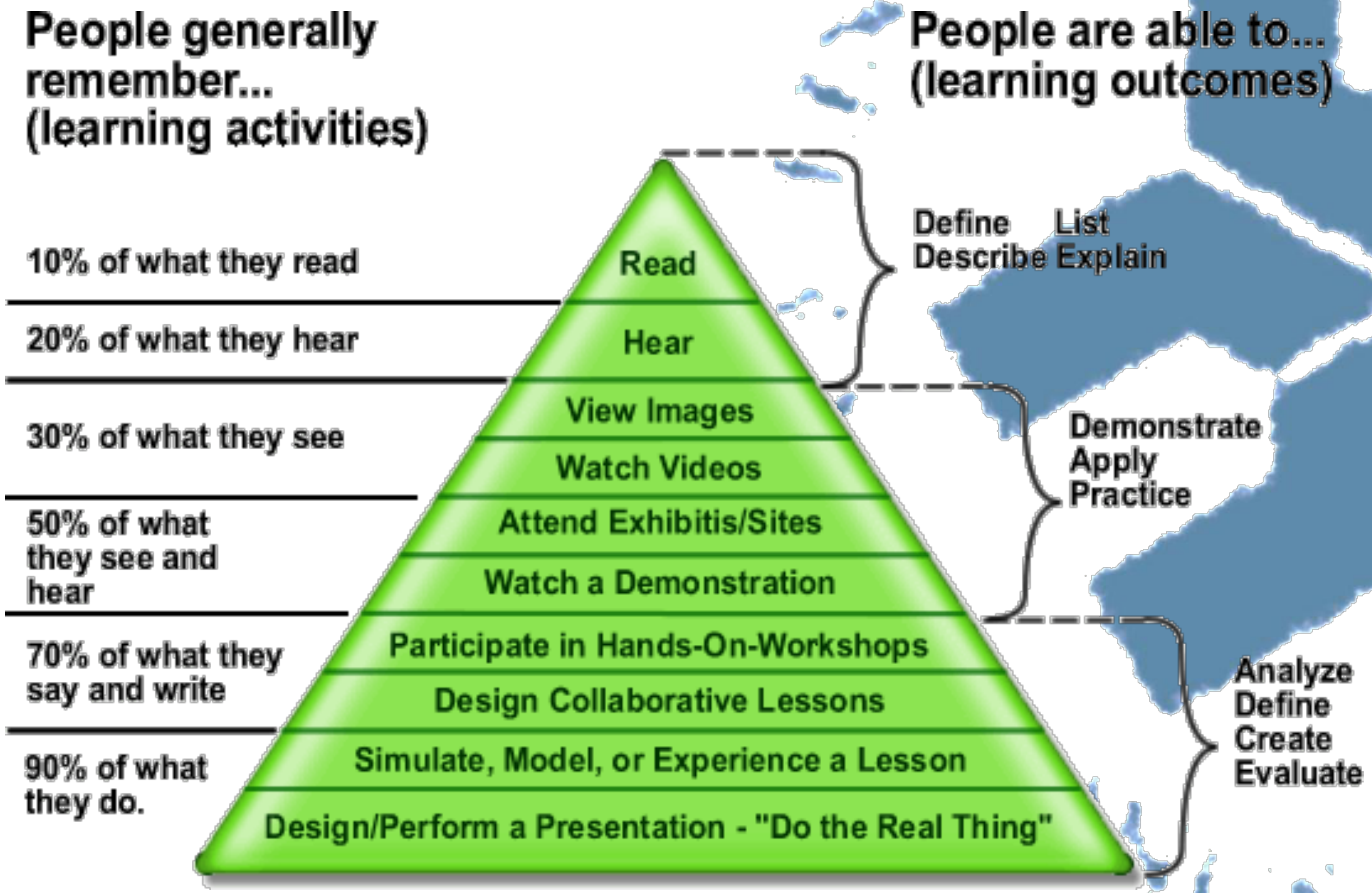
Operations

Name	Parameters	Return type	
exit	int,	void	Invoke
exit	void	void	Invoke
halt	int,	void	Invoke
traceMethodCalls	java.lang.Boolean,	void	Invoke

Interaction Patterns
for user requirements:
What to build

Now how do **you** get there

- Welcome to this course
- More interaction programming
- This is a programming course (G2 level)
  - Focus on writing code
  - Focus on writing for others
- You will learn from working not listening


People generally remember... (learning activities)
People are able to... (learning outcomes)
10% of what they read
20% of what they hear
30% of what they see
50% of what they see and hear
70% of what they say and write
90% of what they do.
Read
Hear
View Images
Watch Videos
Attend Exhibitis/Sites
Watch a Demonstration
Participate in Hands-On-Workshops
Design Collaborative Lessons
Simulate, Model, or Experience a Lesson
Design/Perform a Presentation - "Do the Real Thing"
Define List Describe Explain
Demonstrate Apply Practice
Analyze Define Create Evaluate

Course outline - Your work

- 3 Labs
  - Component placement (Four frameworks)
  - User interaction and component coupling
  - Communication with the rest of the world
- Mini project
  - Your own mini Library
    - Implement 2 (at lest) high-level interaction patterns in a Library.
    - You may need to develop supporting components

Course outline - our work

- Introduction to Course (right now)
- Two ui lectures
  - Closely related to the labs
- One lecture on adjacent topics
  - Testing
  - Framework/Library design

Labs

- Cover basic GUI-programming
  - Component placement
  - Component Interaction
  - Component-System/network interaction

Mini library of high level components

- A Library with of interaction patters (two of them)
- A example app using your Library

Grading for Course

- 3 Labs and mini project
- 4
  - grade 3 + UI Testing
- 5
  - grade 4 + UI toolkit (low level)

Examination

- To pass the course you need to
  - Pass the labs (assistants)
  - Pass the mini project (assistants)
  - Pass the oral examination (Anders)

Feedback from last year

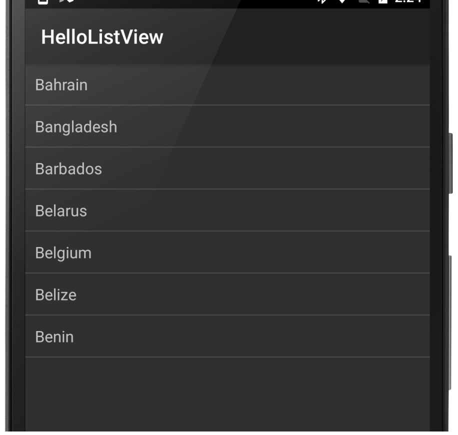
- Fun
  - But time consuming
- Lab 3 was a bit to open
  - What is “trending” on GitHub
- The project was/is easier than the labs

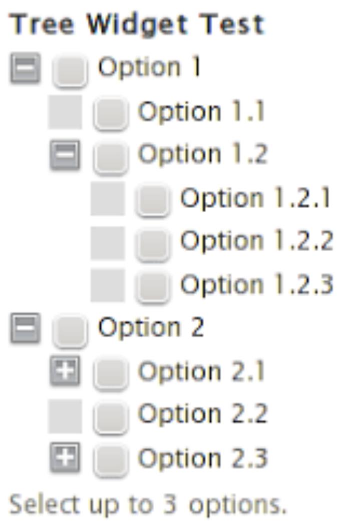
GUI SDKs

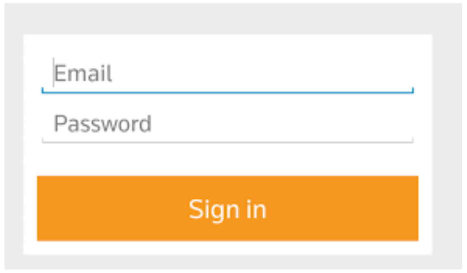
- GUI SDK contains
  - Runtime -a mapping between Abstract UI and native OS/SDK components/primitives
  - Widgets - Buttons, Menus , Text Field etc
  - Widget Messages - interoperability among widgets
  - Messaging structures for in-/output - Messages
  - Rule sets/widget constraints - for controlling layout

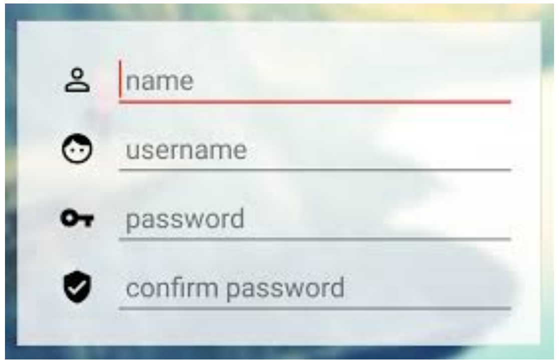
UI -Widgets

- Components
  - A set of visual reusable objects
  - Buttons, Tree, Table, Text-field
    - Android : View,Button , List, EditText, TextView
- Containers
  - Placeholders for Components and Containers
    - Usually allows for nested Containers
    - Windows, Panels, Menus, Toolbars
    - Android: ViewGroup, Layouts









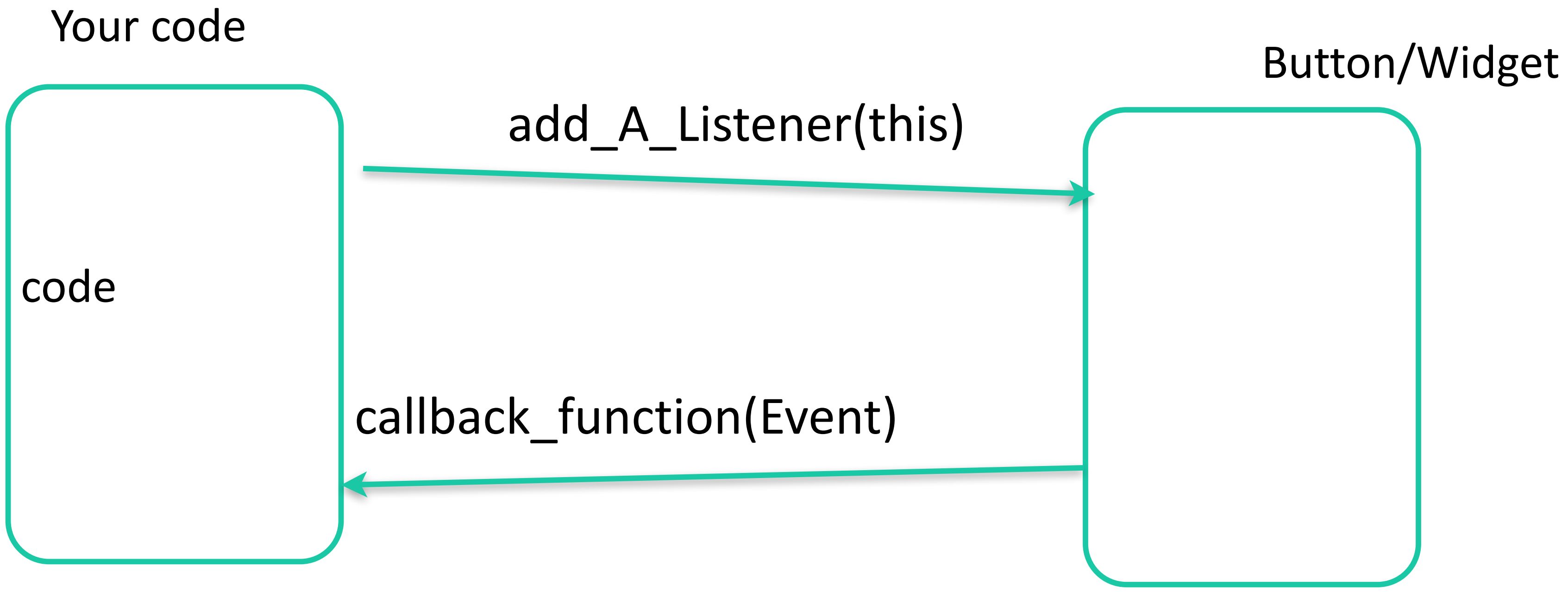
Widget Messages

- Messages from widgets to your code
  - Information about a change in state
  - Messages of user generated actions
    - Mouse movement, keyboard action
    - Touch , Sensors
- Widgets has a set of Messages (zero or more) you choose which you care about

Widget Messages

- Messages passed though callback functions
- You registers yourself with component using listeners
  - android : setOnClickListener
- Receive notification
  - onClick(View v)

Listeners


Your code
code
add_A_Listener(this)
callback_function(Event)
Button/Widget

Messages for in- and output

- GUI runs in a separate thread
  - Keep this thread clean only use for GUI related issues
- SDK provides mechanism for delivering info that can be received and processed in that thread

Threads

- Problem: UI not Thread safe
  - GUI code only in the GUI Event thread
- Solution:Place code on cue
  - Causes UI thread to run a given runnable when it can on the UI Event thread
- Use post any time you want to modify a UI object outside of a listener method

Android : {View}.post

```
mImageView.post(new Runnable() {
        public void run() {
            mImageView.setImageBitmap(bitmap);
        }
    });

// OR AsyncTask

SwingUtilities.invokeLater(
   new Runnable(){
      public void run(){
         outputArea.append(messageToDisplay);
      }
   }
);
```

Rule sets and widget constraints

- Controlling widgets
  - placement, size
  - relation to other widgets
  - reaction to external changes
    - Window size change
    - Device orientation,size,resolution

Placing components

- Layouts in Android
  - Constraint, Motion Linear- Vertical/Horizontal (Grid , Table)
  - Column,Row, Box (Composable)
- Layouts in React Native
  - Flexbox

– Layouts in Flutter
  - Layout widgets
    - Single-Child widget(s)
    - Multi-Child widget(s)

Conclusion

- What I hope you take with you
  - This is a programming course you learn by doing
  - Tools aids developers - so learn them
  - Some brief hints on how to develop todays GUI application.

- Questions