CS/ENGRD 2110 FALL 2014

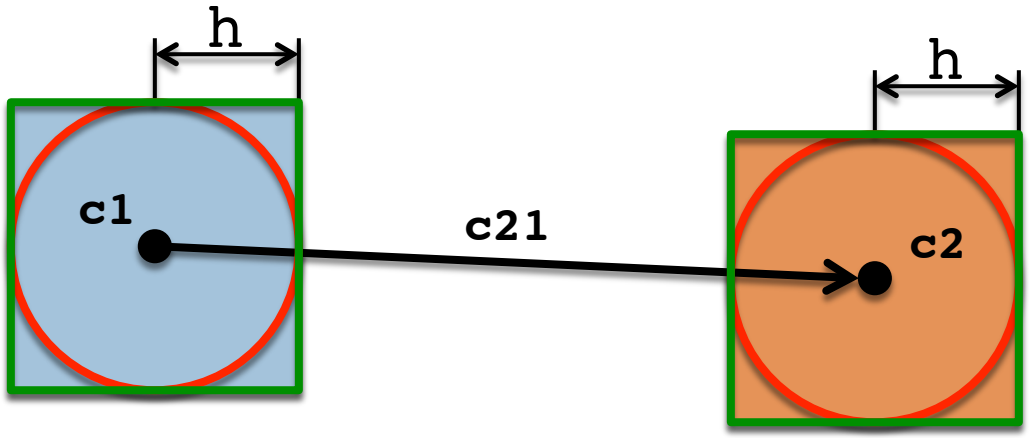
Lecture 14: Graphical User Interfaces (GUIs): Laying out components

http://courses.cs.cornell.edu/cs2110

Announcement: A4 Collision Detection

- Note: Block.overlap() is approximate:

```
public static boolean overlaps(Block a, Vector2D u, Block b, Vector2D v) {
        Vector2D c1 = Vector2D.add(new Vector2D(a.position), u);
        Vector2D c2 = Vector2D.add(new Vector2D(b.position), v);
        return Vector2D.dist(c1, c2) < a.halfwidth + b.halfwidth;
}
```



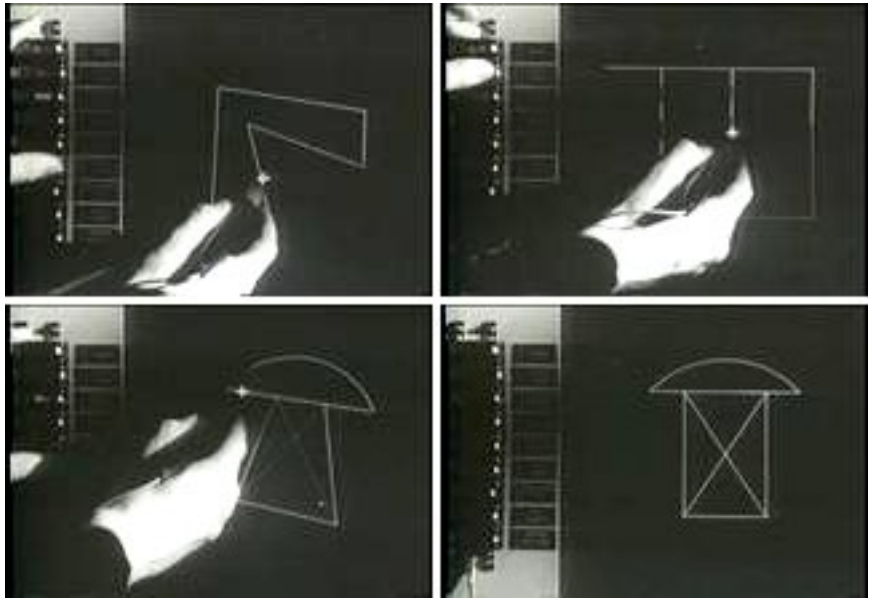
$$\|\vec{c}_{21}\|_2 \leq 2h$$

$$\|\vec{c}_{21}\|_\infty \leq 2h$$

$$\|\vec{c}\|_\infty \equiv \max_i |c_i|$$

Graphical User Interface: a type of user interface that allows users to interact with programs through manipulation of graphical elements as opposed to text-based interfaces.

Some GUI history

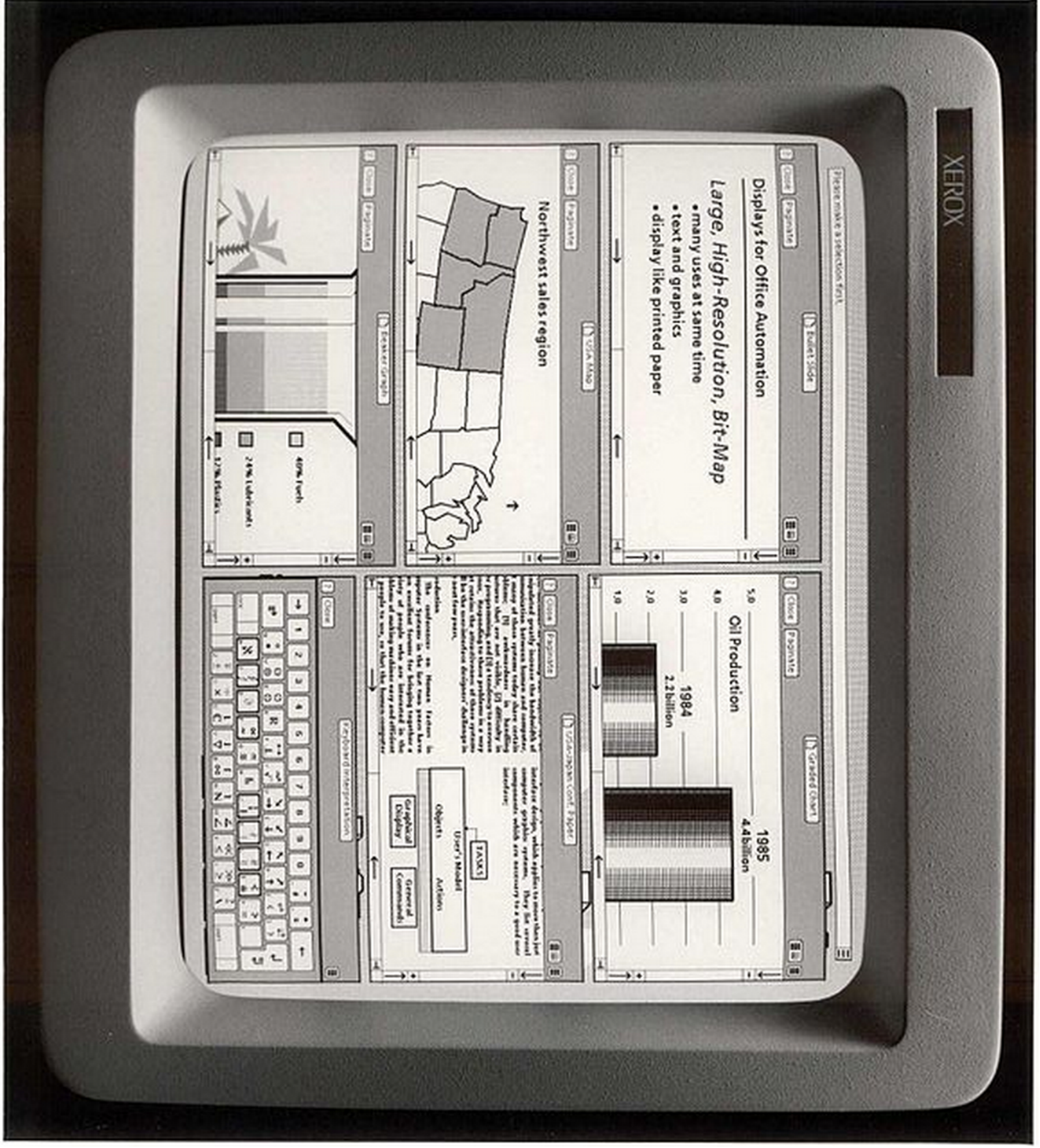


`60s: Sutherland's "SketchPad"

The first prototype of a computer mouse, as designed by Bill English from Engelbart's sketches[1]

http://en.wikipedia.org/wiki/History_of_the_graphical_user_interface

http://toastytech.com/guis/guitimeline.html



`81: Xerox's "Star" workstation

Mouse tales…

Their mouse had a mean time between failure of … a week … it would jam up irreparably, or ... jam up on the table-- ... It had a flimsy cord whose wires would break. Steve Jobs: "... Xerox says it can't be built for < $400, I want a $10 mouse that will never fail and can be mass produced, because it's going to be the primary interface of the computer ..."

... Dean Hovey ... came back, "I've got some good and some bad news. Good news: we've got a new project with Apple. Bad news: I told Steve we'd design a mouse for 10 bucks."

... year later ... we … filed … and were granted a patent, on the electro-mechanical-optical mouse of today; ... we ended up ... [making] the mouse as invisible to people as it is today.

Steve Sachs interview on first computer with GUI: Apple Lisa (~$10K in 1982).
http://library.stanford.edu/mac/primary/interviews/sachs/trans.html

(see also https://alumni.stanford.edu/get/page/magazine/article/?article_id=37694)

Magic Mouse

Buy Now

Suddenly, everything clicks. And swipes. And scrolls.

Introducing Magic Mouse. The world's first Multi-Touch mouse.
Now included with every new iMac. And available on its own for just $69.

View the Magic Mouse gallery  Watch the Magic Mouse video 

We've built a better mouse.

It began with iPhone. Then came iPod touch. Then MacBook Pro. Intuitive, smart, dynamic. Multi-Touch technology introduced a remarkably better way to interact with your portable devices — all using gestures. Now

Shop for **apple mouse** on Google

Sponsored

Apple® - Magic Wireless Las...
$69.99
Best Buy
★★★★½ (1k+)

Apple Magic **Mouse** with M...
$69.00
Apple Store
★★★★½ (1k+)

Apple® - Optical **Mouse**
$49.99
Best Buy
★★★★☆ (102)

Free shipping Bluetooth 2.4...
$20.00
AliExpress.com
★★★★½ (1k+)

Apple Magic Wireless Laser
$34.49
eBay
★★★★½ (1k+)

Shop for **microsoft mouse** on Google

Sponsored

Microsoft Arc Touch **Mouse**...
$69.99
Abt Electr...
★★★★½ (155)

Microsoft Sculpt Comfo...
$39.95
MicrosoftStore
★★★★☆ (169)

Microsoft Touch **Mouse**
$79.95
MicrosoftStore
★★★☆☆ (127)

Microsoft Wireless Mob...
$29.95
MicrosoftStore

Microsoft Arc **Mouse** - Black
$39.99
Newegg.c...
★★★★☆ (613)

Shop for **monoprice mouse** on Google

Sponsored

6-Key Ergonomic G...
$10.16
Monoprice...

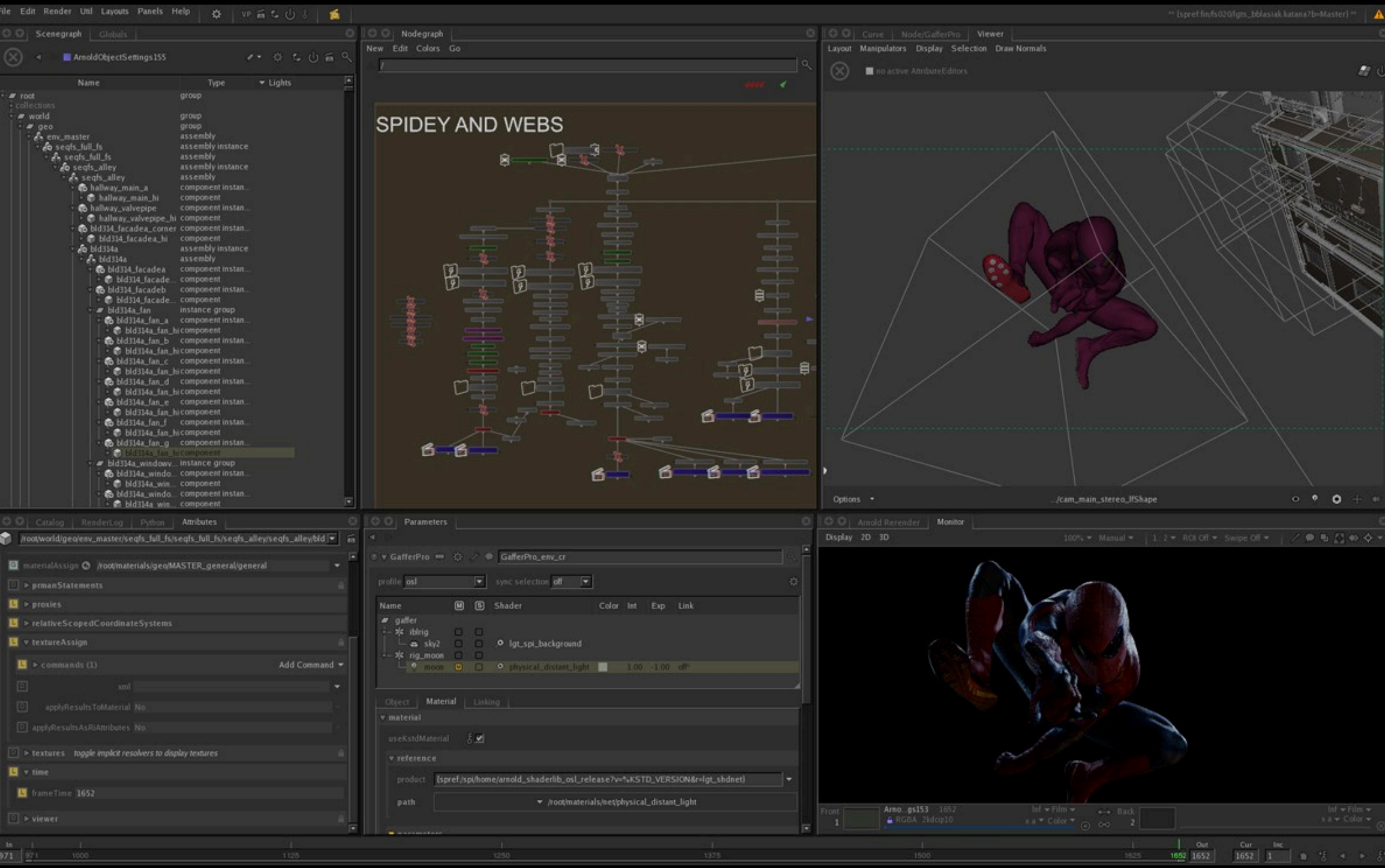
6-Key Gaming **Mouse** with A...
$10.16
Monoprice...

5-Button Optical Laser...
$11.43
Monoprice...

Mini/Travel Optical Mous...
$4.59
Monoprice...

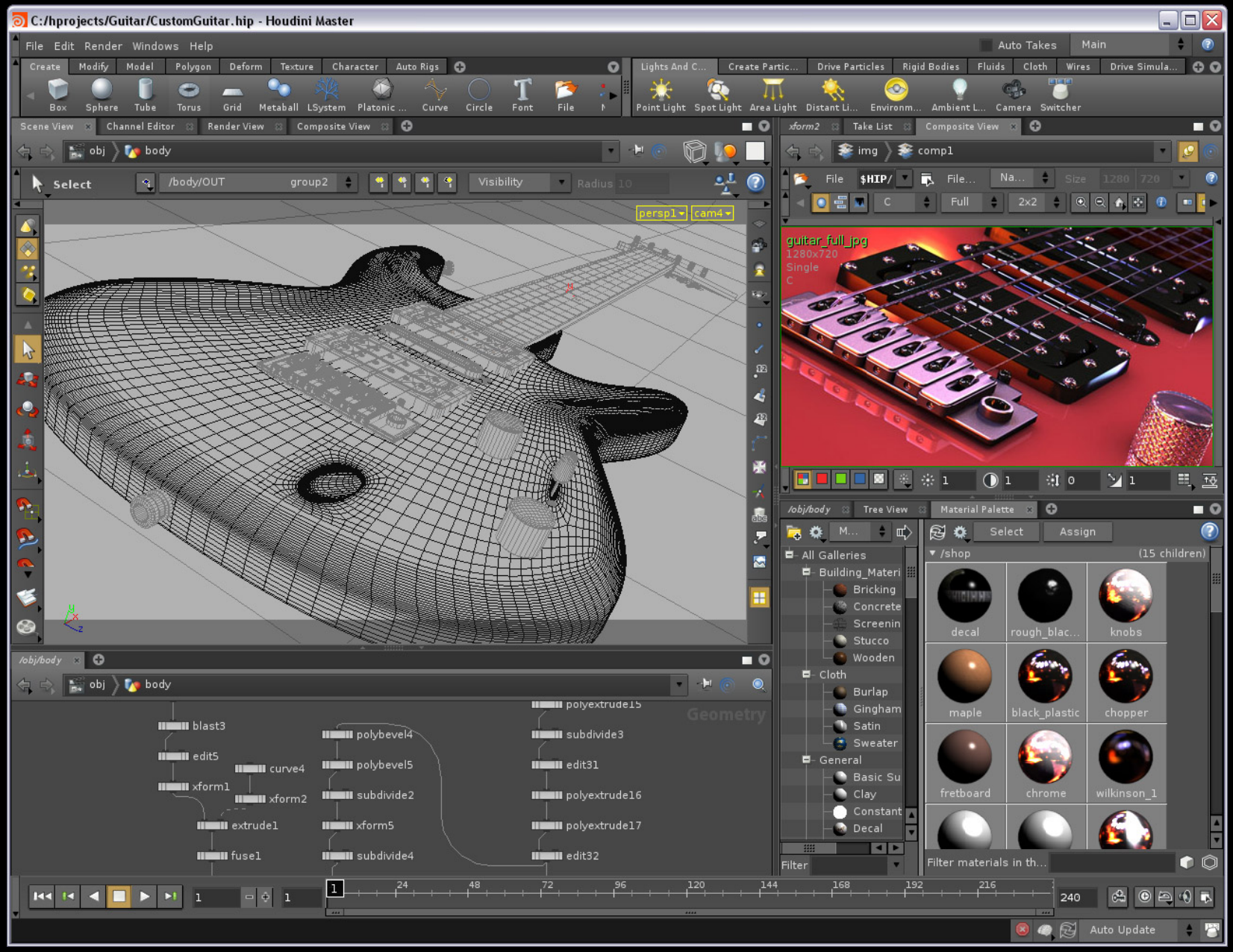
Super Slim Optical USB...
$5.19
Monoprice...

Modern GUI Examples: "Katana"



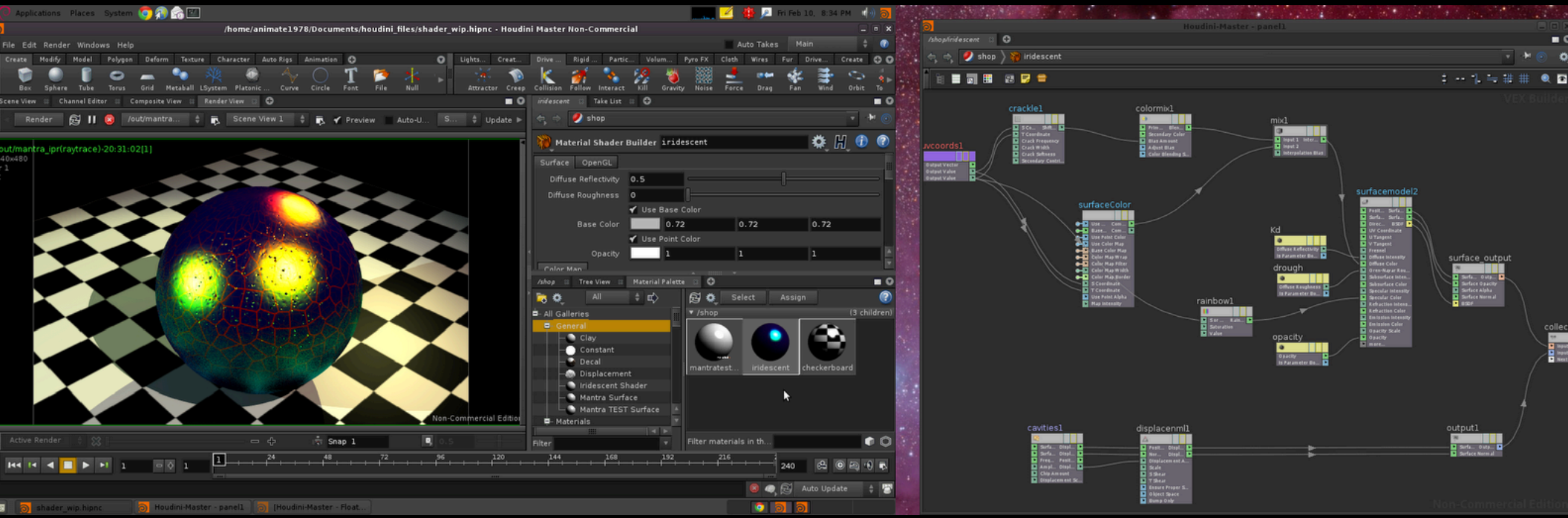
http://www.fxguide.com/featured/katana-in-production-changing-the-structure-of-the-workflow/

Modern GUI Examples: "Houdini"



http://www.creativeobserver.com/img/276_houdini_9.1_screenshot_lg.jpg

Modern GUI Examples: "Houdini"



http://animate1978rendering.blogspot.com/2012/02/iridescent-houdini-mantra-shader.html

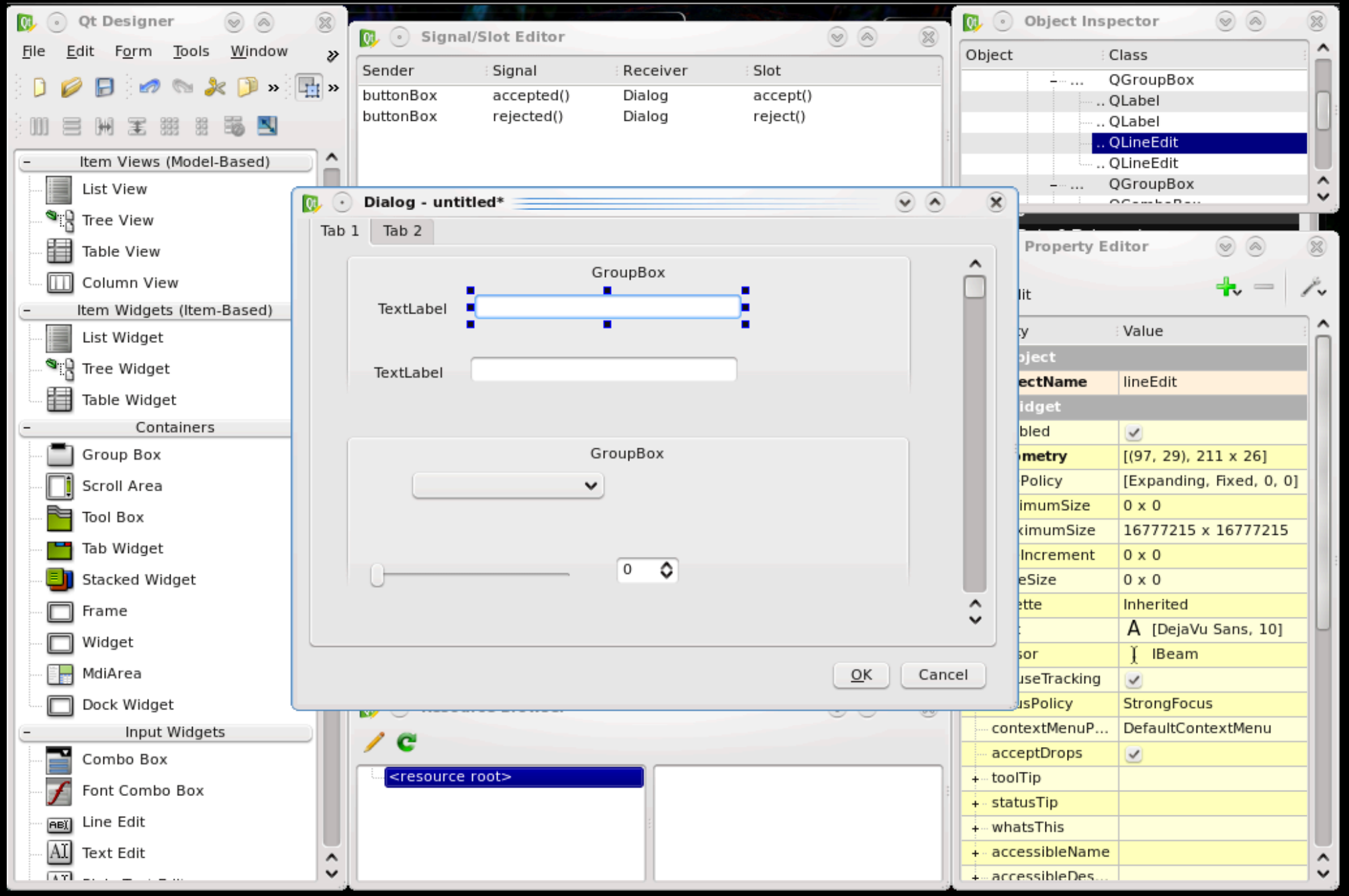
Modern GUI Examples: Drum machine

http://plus.pointblanklondon.com/wp-content/uploads/2013/07/nithonat-gui-big.jpg

Many GUI tools & APIs

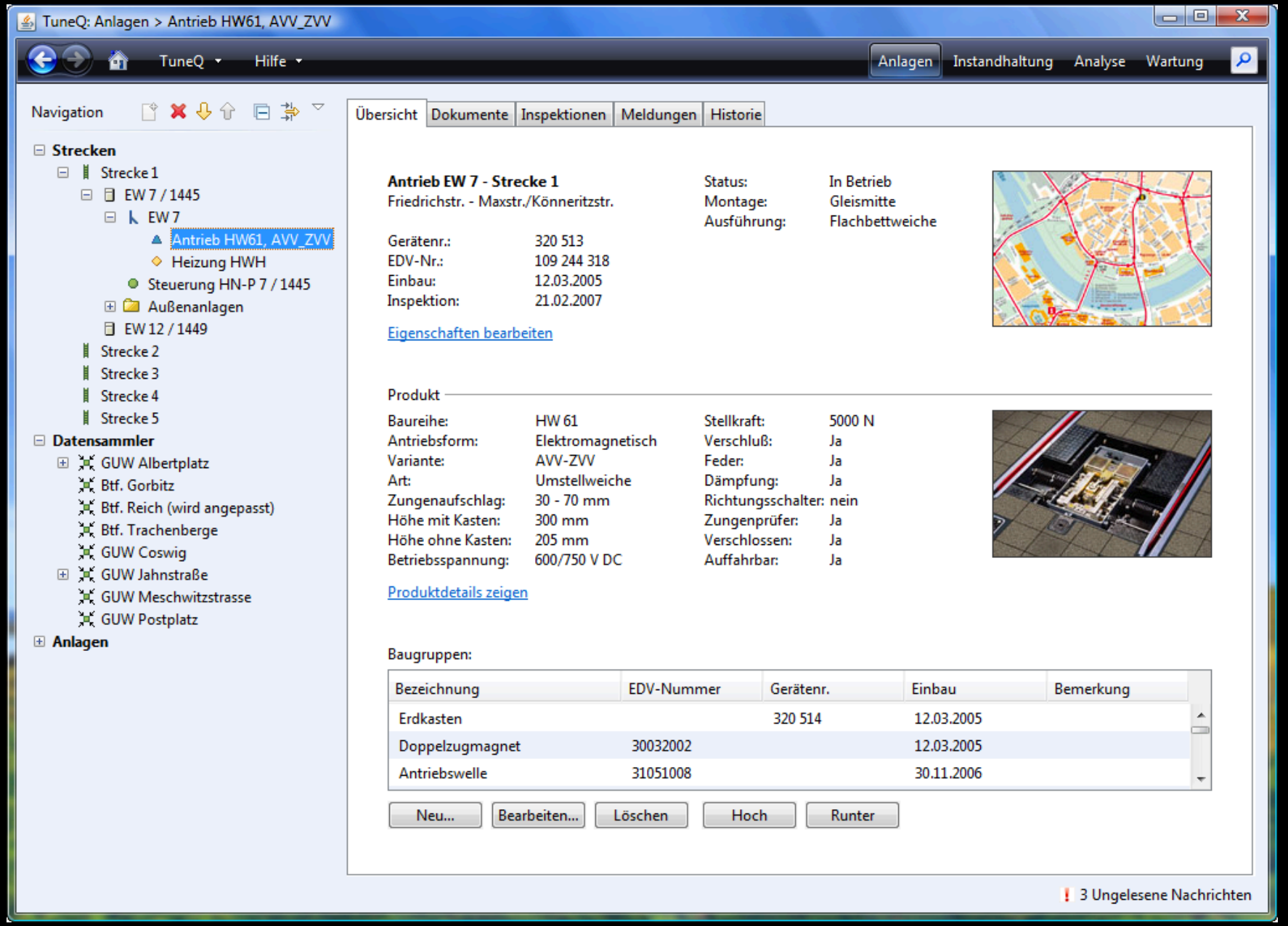
http://www.webdesignshock.com/wp-content/uploads/2012/05/Gui_Apollo_WDS_901.jpg

Many GUI tools & APIs: QT (cross platform)



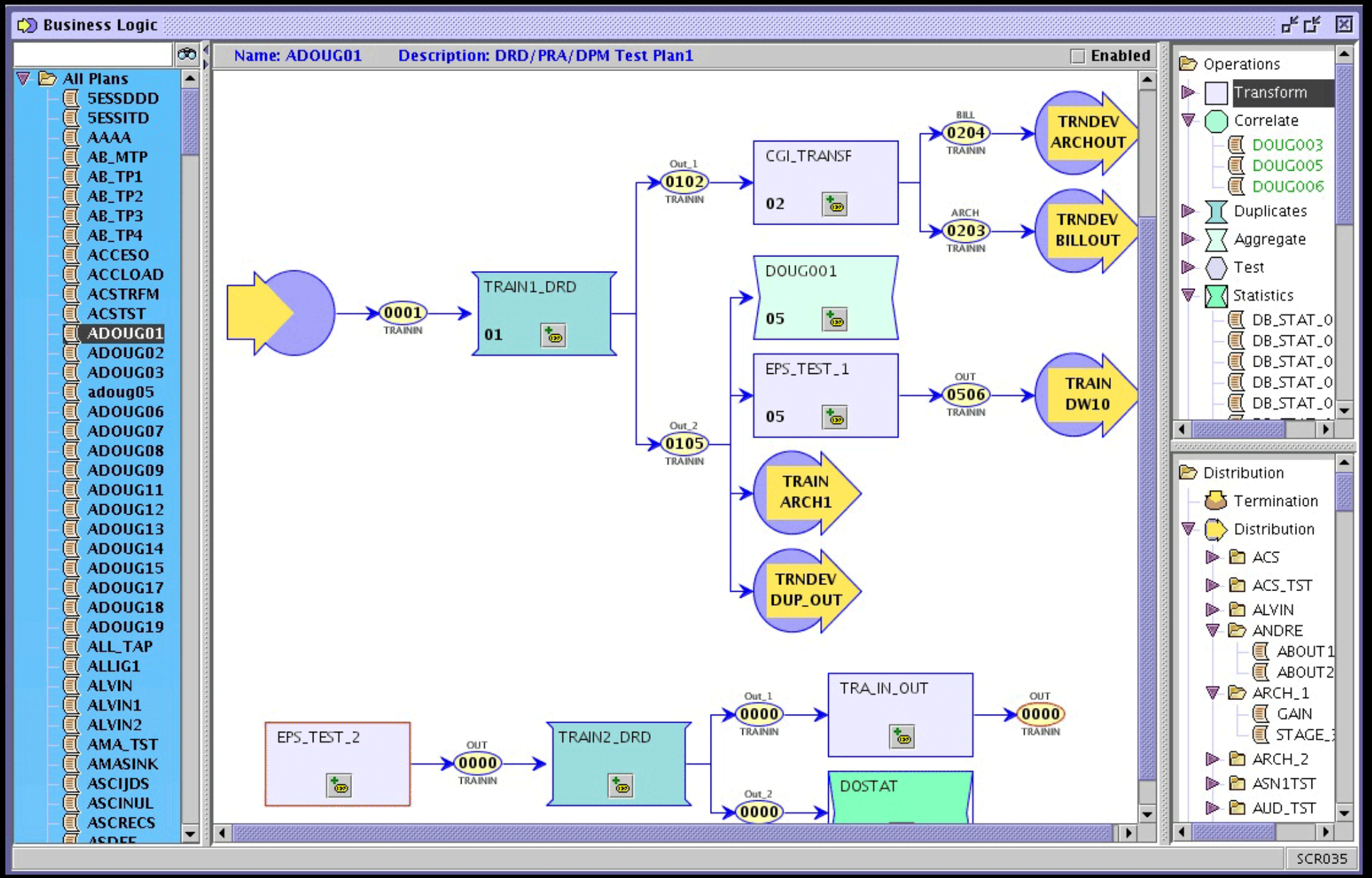
http://upload.wikimedia.org/wikipedia/commons/f/f3/Qt_Designer_4_4_3.png

Many GUI tools & APIs: Swing (Java)



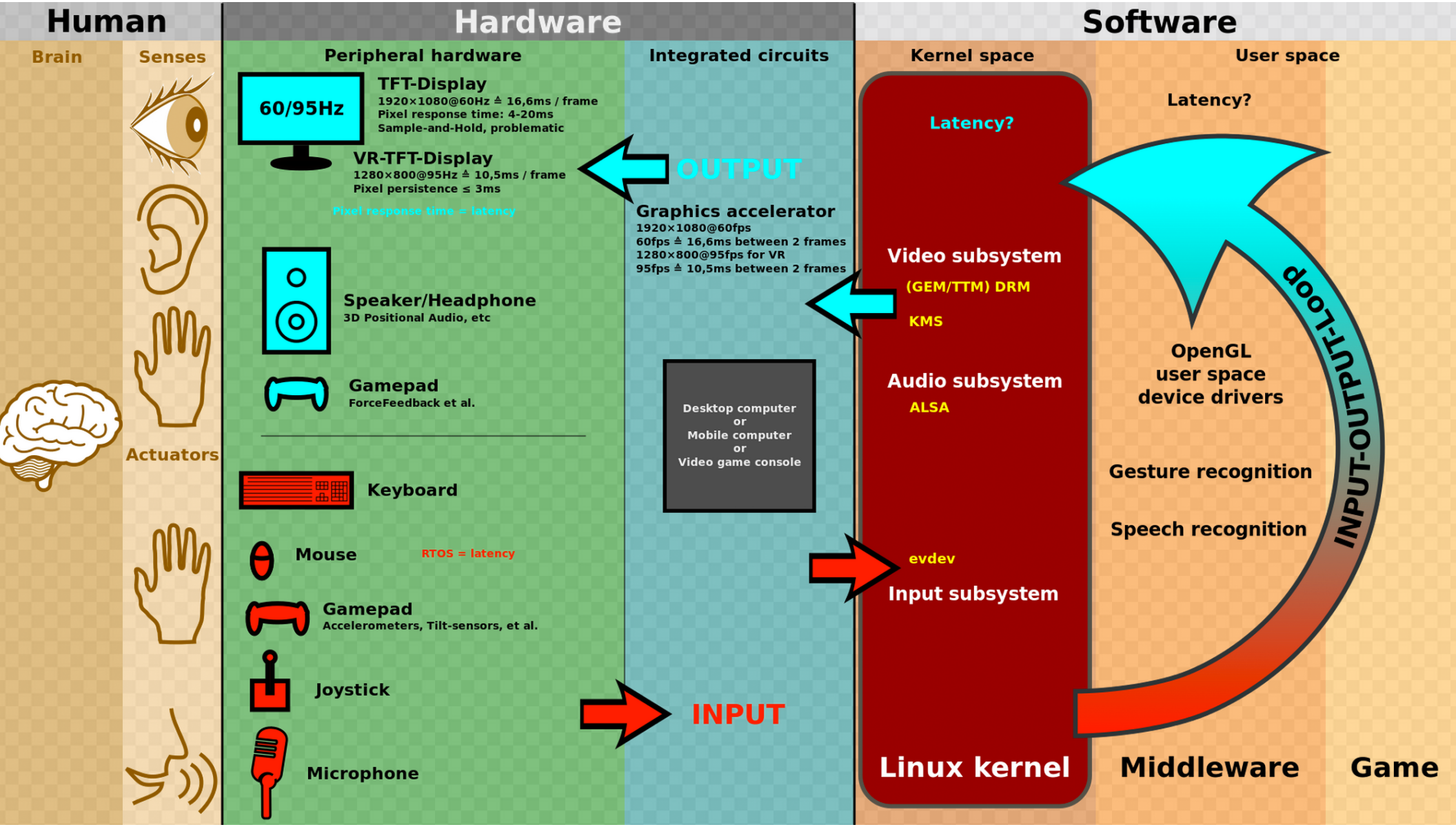
http://www.jgoodies.com/wp-content/uploads/2012/04/01-Antrieb.png

Example: Swing (Java)



http://stackoverflow.com/questions/11692095/how-to-create-movable-nodes-and-edges-in-java-swing-gui

GUI (Graphical User Interface)



http://en.wikipedia.org/wiki/Graphical_user_interface

GUI (Graphical User Interface)

- Provides a friendly interface between user and program
- Allows event-driven or reactive programming: The program reacts to events such as button clicks, mouse movement, keyboard input
- Often is multi-threaded: Different threads of execution can be going on simultaneously

We use Java's two packages for doing GUIs:

- **AWT** (Abstract Window Toolkit) —first one; very simple
- Swing —a newer one, which builds on AWT as much as possible

Two aspects to making a GUI:

1. Laying out components (buttons, text, etc.) in it. TODAY
2. Listening/responding to events Next Lecture

Class JFrame

JFrame object: associated with a window on your monitor.

Generally, a GUI is a JFrame object with various components placed in it

Some methods in a JFrame object

hide() show() setVisible(boolean)

getX() getY() (coordinates of top-left point)

getWidth() getHeight() setLocation(int, int)

getTitle() setTitle(String)

getLocation() setLocation(int, int)

Over 100 methods in a JFrame object!

Class JFrame is in package javax.swing

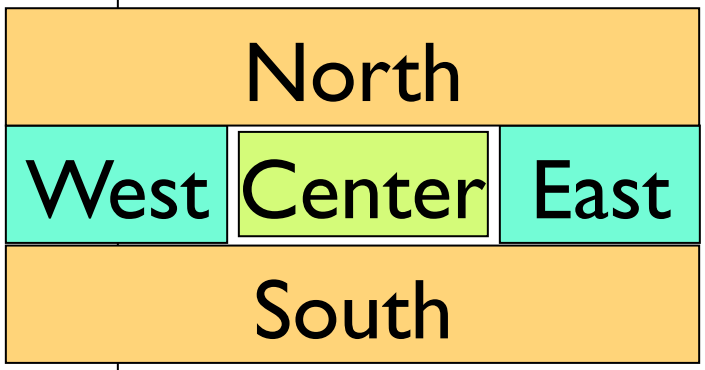
Placing components in a JFrame

Layout manager: Instance controls placement of components.

JFrame layout manager default: BorderLayout.

BorderLayout layout manager: Can place 5 components:

```
public class C extends JFrame {
   public C() {
      Container cp= getContentPane();
      JButton jb= new JButton("Click here");
      JLabel jl= new JLabel( "label 2");
      cp.add(jb, BorderLayout.EAST);
      cp.add(jl, BorderLayout.WEST);
      pack();
      setVisible(true);
 }
```



JFrameDemo.java

Putting components in a JFrame

```
import java.awt.*;    import javax.swing.*;
/** Demonstrate placement of components in a JFrame.
    Places five components in 5 possible areas:
      (1) a JButton in the east,          (2) a JLabel in the west,
      (3) a JLabel in the south,          (4) a JTextField in the north
      (5) a JTextArea in the center.   */

public class ComponentExample extends JFrame {
   /** Constructor: a window with title t and 5 components */
   public ComponentExample(String t) {
      super(t);
      Container cp=  getContentPane();
      cp.add(new JButton("click me"), BorderLayout.EAST);
      cp.add(new JTextField("type here", 22), BorderLayout.NORTH);
      cp.add(new JCheckBox("I got up today"), BorderLayout.SOUTH);
      cp.add(new JLabel("label 2"), BorderLayout.WEST);
      cp.add(new JTextArea("type\nhere", 4, 10), BorderLayout.CENTER);
      pack();
   }
```

Add components to its contentPane

ComponentExample.java

Packages --Components

Packages that contain classes that deal with GUIs:
java.awt: Old package. **javax.swing: New package.**

javax.swing has a better way of listening to buttons, text fields, etc. Components are more flexible.

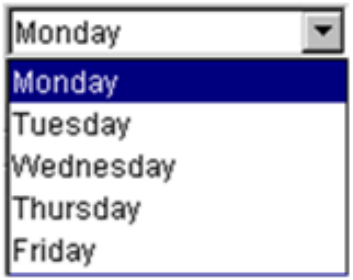
xxxx in awt
Jxxxx in Swing

Component: Something that can be placed in a GUI window. They are instances of certain classes, e.g.

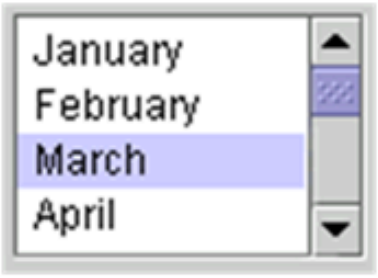
JButton, Button: Clickable button
JLabel, Label: Line of text
JTextField, TextField: Field into which the user can type
JTextArea, TextArea: Many-row field into which user can type
JPanel, Panel: Used for graphics; to contain other components
JCheckBox: Checkable box with a title
JComboBox: Menu of items, one of which can be checked
JRadioButton: Same functionality as JCheckBox
Container: Can contain other components
Box: Can contain other components

Basic Components

Buttons

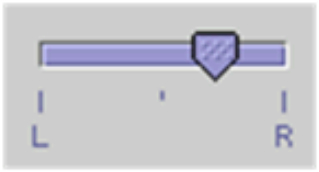


Combo Box



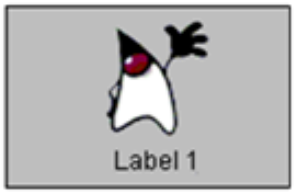
List

TextField

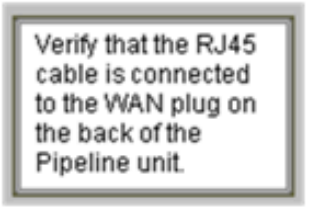


Slider

Menu



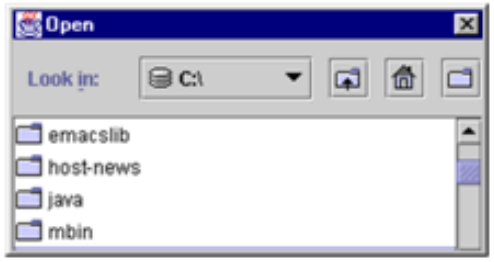
Label



Text Area

Tool Tip

Progress Bar

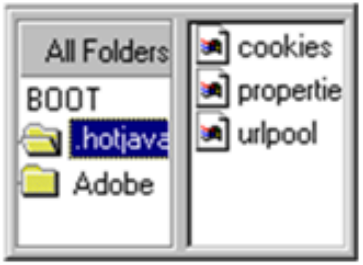


File Chooser

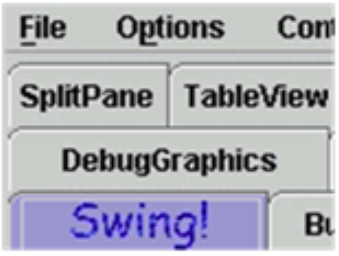
Color Chooser

Table

Tree



Split Pane



Tabbed Pane

http://www.ntu.edu.sg/home/ehchua/programming/java/images/Swing_Components.png

Basic Components

Component
- Button, Canvas
- Checkbox, Choice
- Label, List, Scrollbar
- TextComponent
  - TextField, TextArea
- Container
  - JComponent
  - AbstractButton
    - JButton
    - JToggleButton
      - JCheckBox
      - RadioButton
  - JLabel, JList
  - JOptionPane, JPanel
  - JPopupMenu, JScrollBar, JSlider
  - JTextComponent
  - JTextField, JTextArea

Component: Something that can be placed in a GUI window. These are the basic ones used in GUIs

Note the use of subclasses to provide structure and efficiency. For example, there are two kinds of JToggleButtons, so that class has two subclasses.

Components that can contain other components

- Component
  - Box
  - Container
    - JComponent
    - JPanel
    - Panel
      - Applet
  - Window
    - Frame
      - JFrame
    - JWindow

java.awt is the old GUI package.

javax.swing is the new GUI package. When they wanted to use an old name, they put J in front of it.

(e.g. Frame and JFrame)

When constructing javax.swing, the attempt was made to rely on the old package as much as possible.

So, JFrame is a subclass of Frame.

But they couldn't do this with JPanel.

```
import java.awt.*;    import javax.swing.*;
/** Instance has labels in east /west, JPanel with four buttons in center.  */
public class PanelDemo extends JFrame {
    JPanel p= new JPanel();
    /** Constructor: a frame with title "Panel demo", labels in east/west,
        blank label in south, JPanel of 4 buttons in the center */
    public PanelDemo() {
        super("Panel demo");
        p.add(new JButton("0"));   p.add(new JButton("1"));
        p.add(new JButton("2"));   p.add(new JButton("3"));
        Container cp= getContentPane();
        cp.add(new JLabel("east"), BorderLayout.EAST);
        cp.add(new JLabel("west"), BorderLayout.WEST);
        cp.add(new JLabel("    "), BorderLayout.SOUTH);
        cp.add(p, BorderLayout.CENTER);
        pack();
    }
}
```

JPanel as a container

JPanel layout manager default: FlowLayout.

FlowLayout layout manager: Place any number of components. They appear in the order added, taking as many rows as necessary.

```
import javax.swing.*;  import java.awt.*;
/** Demo class Box. Comment on constructor says how frame is laid out. */
public class BoxDemo extends JFrame {
   /** Constructor: frame with title "Box demo", labels in the east/west,
       blank label in south, horizontal Box with 4 buttons in center. */
   public BoxDemo() {
      super("Box demo");
      Box b= new Box(BoxLayout.X_AXIS);
      b.add(new JButton("0"));     b.add(new JButton("1"));
      b.add(new JButton("2"));     b.add(new JButton("3"));
      Container cp= getContentPane();
      cp.add(new JLabel("east"),  BorderLayout.EAST);
      cp.add(new JLabel("west"), BorderLayout.WEST);
      cp.add(new JLabel(" "),       BorderLayout.SOUTH);
      cp.add(b,                          BorderLayout.CENTER);
      pack();
   }
}
```

Class Box: a container

Box layout manager default: BoxLayout.

BoxLayout layout manager: Place any number of components. They appear in the order added, taking only one row.

```
public class BoxDemo2 extends JFrame {
 /** Constructor: frame with title t and 3 columns with n, n+1, and n+2 buttons. */
 public BoxDemo2(String t, int n) {
      super(t);
      // Create Box b1 with n buttons.
           Box b1= new Box(BoxLayout.Y_AXIS);
           for (int i= 0;  i != n;  i= i+1)
                b1.add(new JButton("1 " + i));
      // Create Box b2 with n+1 buttons.
           Box b2= …
      // Create Box b3 with n+2 buttons.
           Box b3= …
      // Create horizontal box b containing b1, b2, b3
           Box b= new Box(BoxLayout.X_AXIS);
           b.add(b1);
           b.add(b2);
           b.add(b3);
      Container cp= getContentPane();
      cp.add(b, BorderLayout.CENTER);
      pack();  show();
}
```

Boxes within a Box

3 vertical boxes, each a column of buttons, are placed in a horizontal box

BoxLayout layout manager: Place any number of components. They appear in the order added, taking only one row.

Simulate BoxLayout Manager in a JFrame

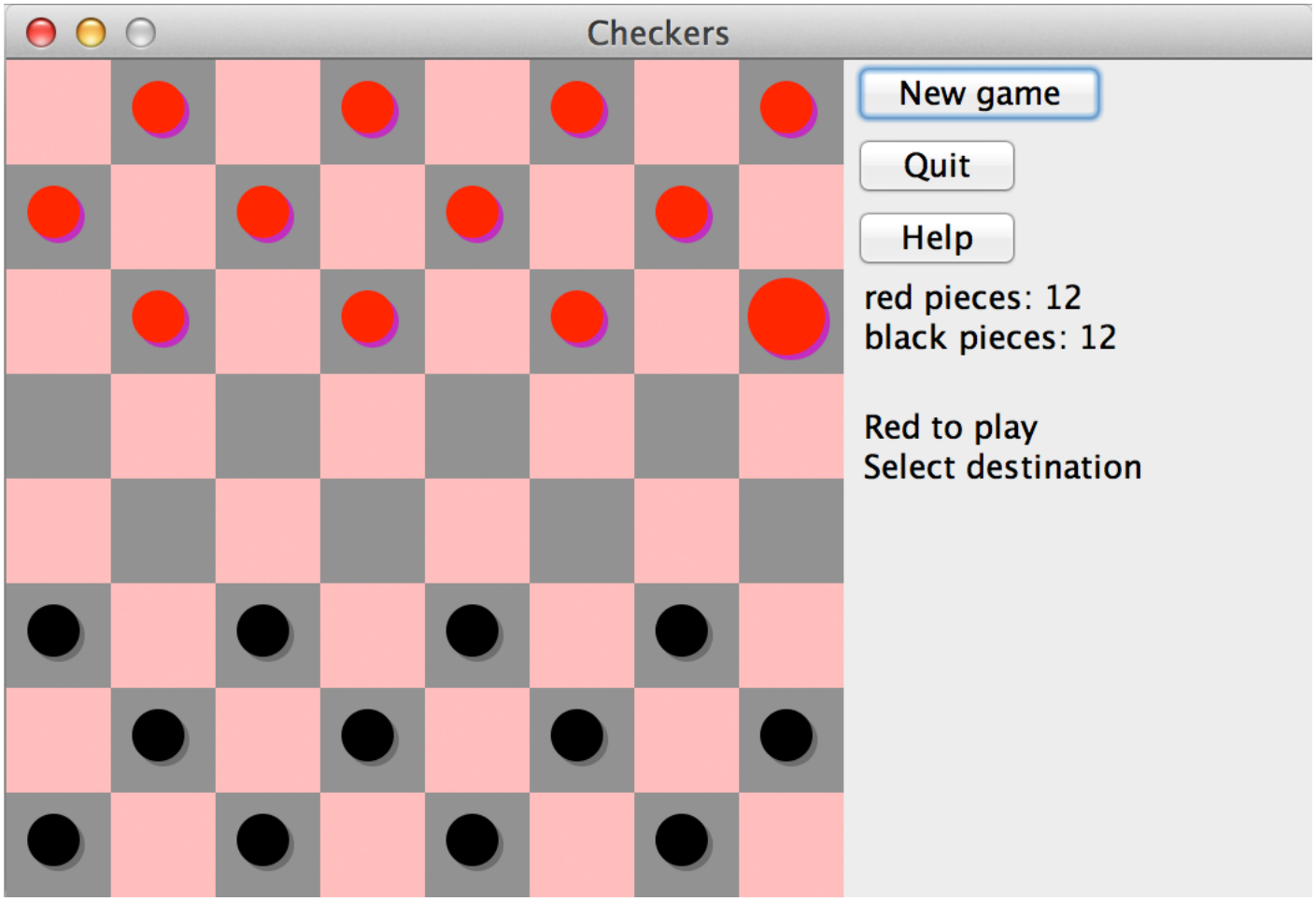
To simulate using a BoxLayout manager for a JFrame, create a Box and place it as the sole component of the JFrame:

```
JFrame jf= new JFrame("title");
Box b= new Box(BoxLayout.X_AXIS);
Add components to b;
jf.add(b,BorderLayout.CENTER);
```

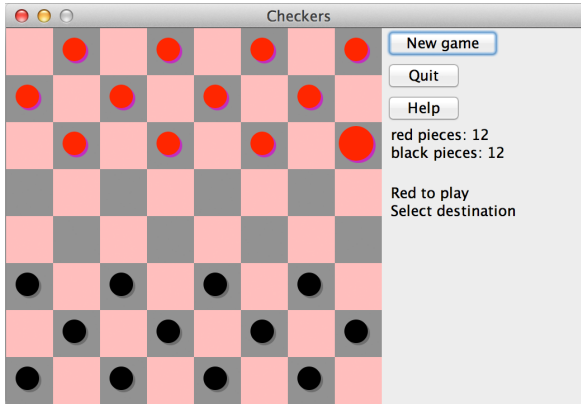
1. **Start developing a GUI by changing an already existing one.** A lot of details. Hard to get all details right when one starts from scratch and has little idea about the Java GUI package.

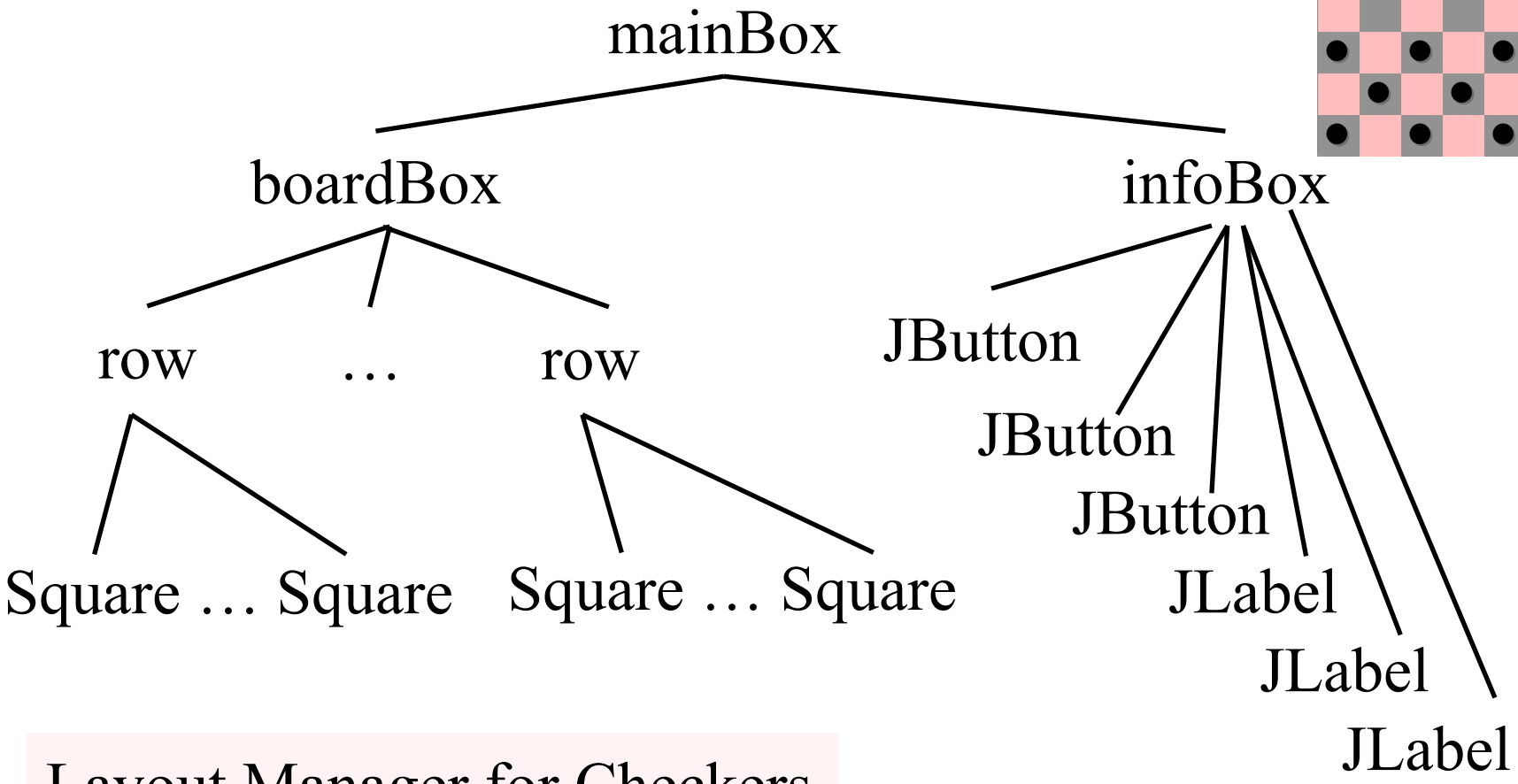
2. Showed how to place components in a GUI. Next class: how to "listen" to things like button clicks in a GUI.

Checkers Example



Checkers Example





Layout Manager for Checkers game has to process a tree

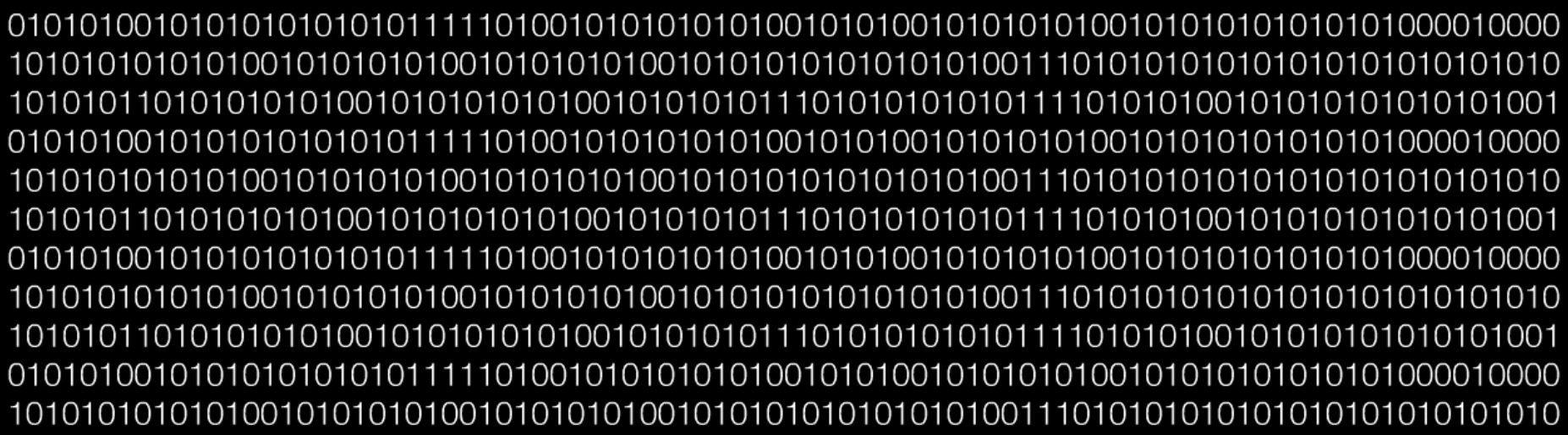
pack(): Traverse the tree, determining the space required for each component

boardBox: vertical Box
row: horizontal Box
Square: Canvas or JPanel
infoBox: vertical Box

the GUI future…

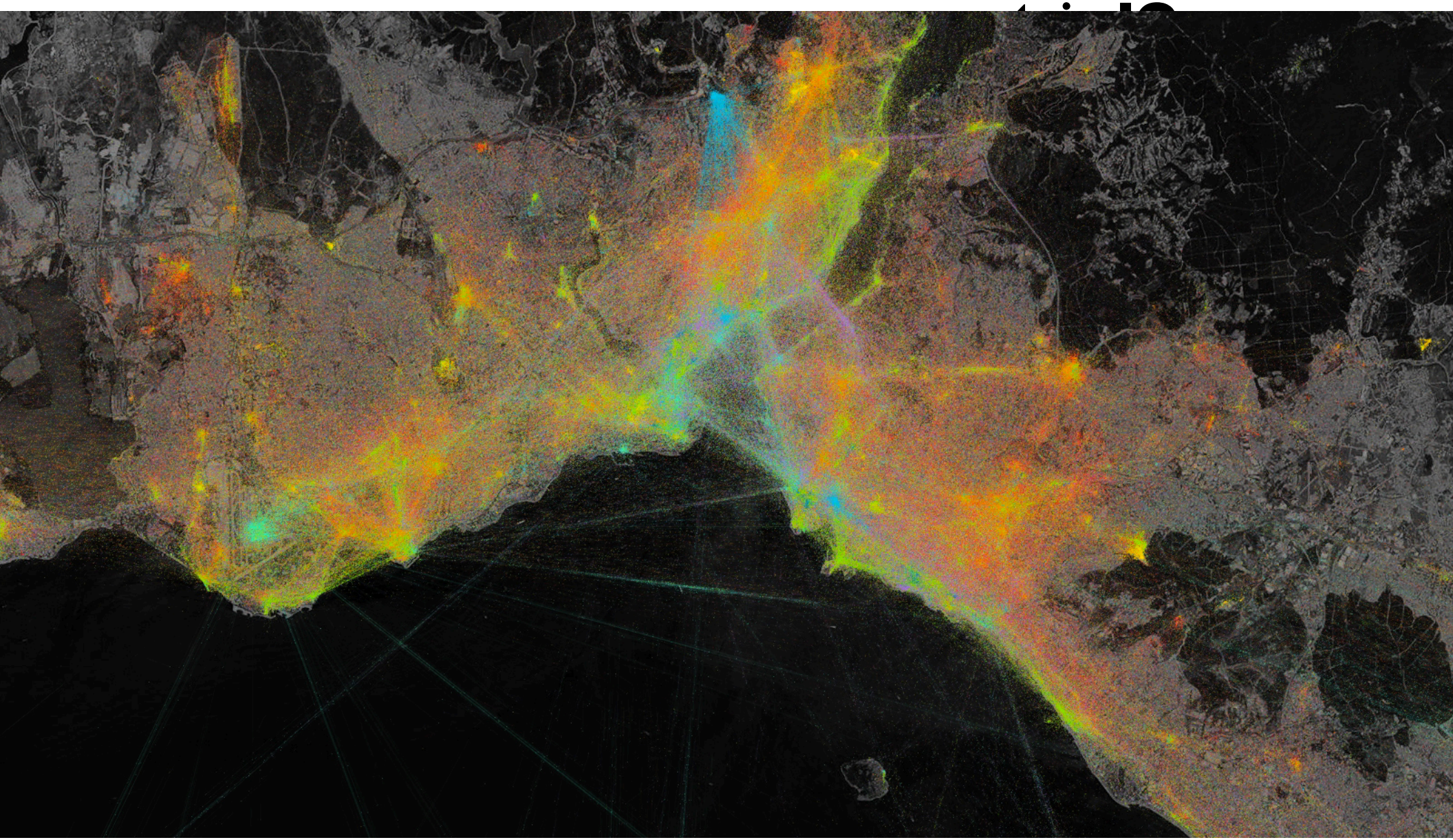
Rethinking GUIs

Big Data



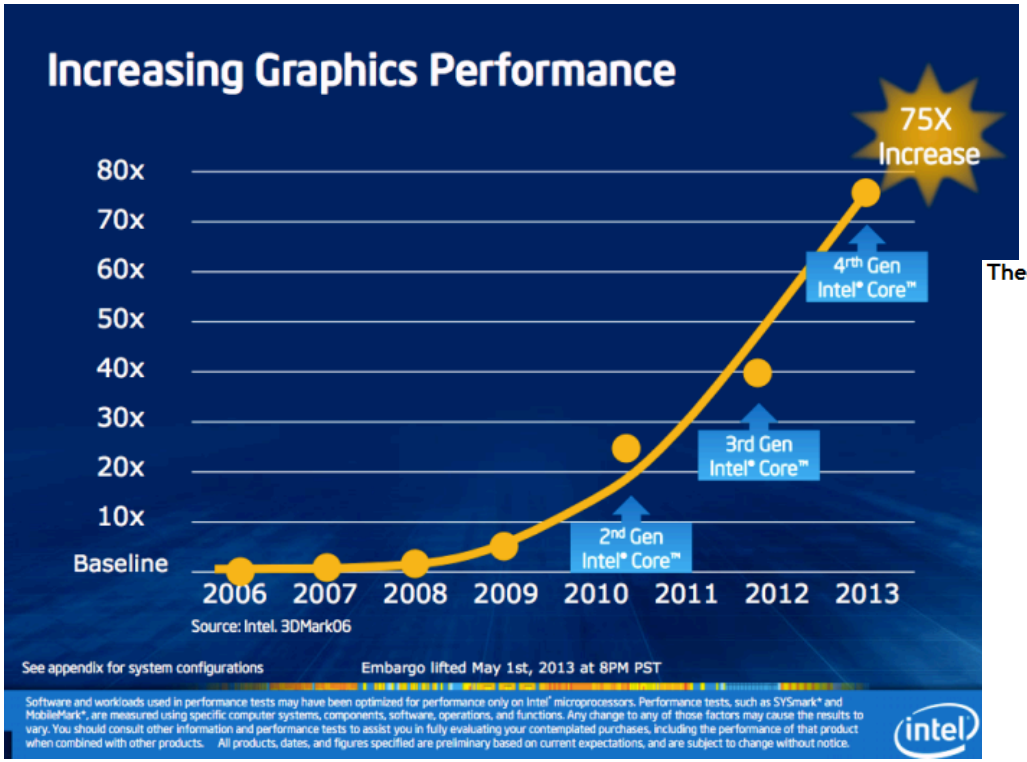
" In 2011 we will have generated more data than mankind has since the begining of history" - Peter Hirshberg (global pulse Summit)

Rethinking GUIs

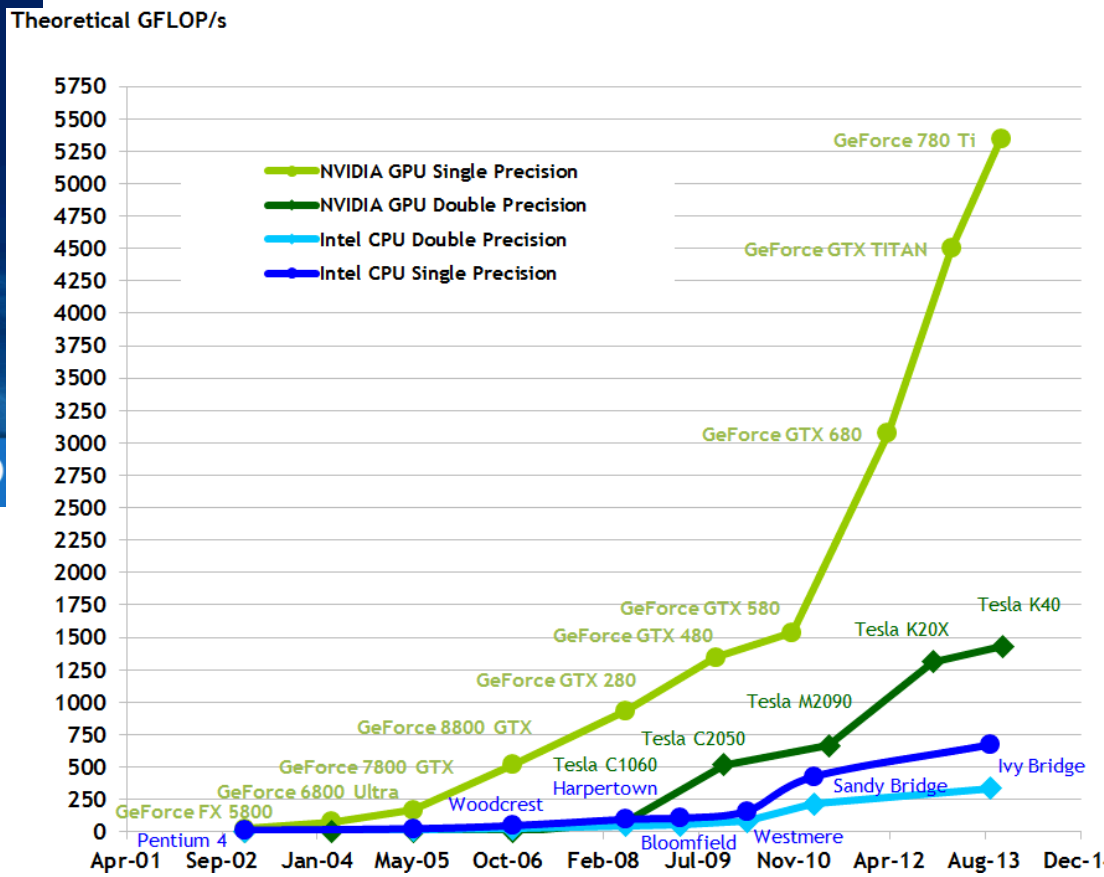


foursquare

Rethinking GUIs



Graphics Performance



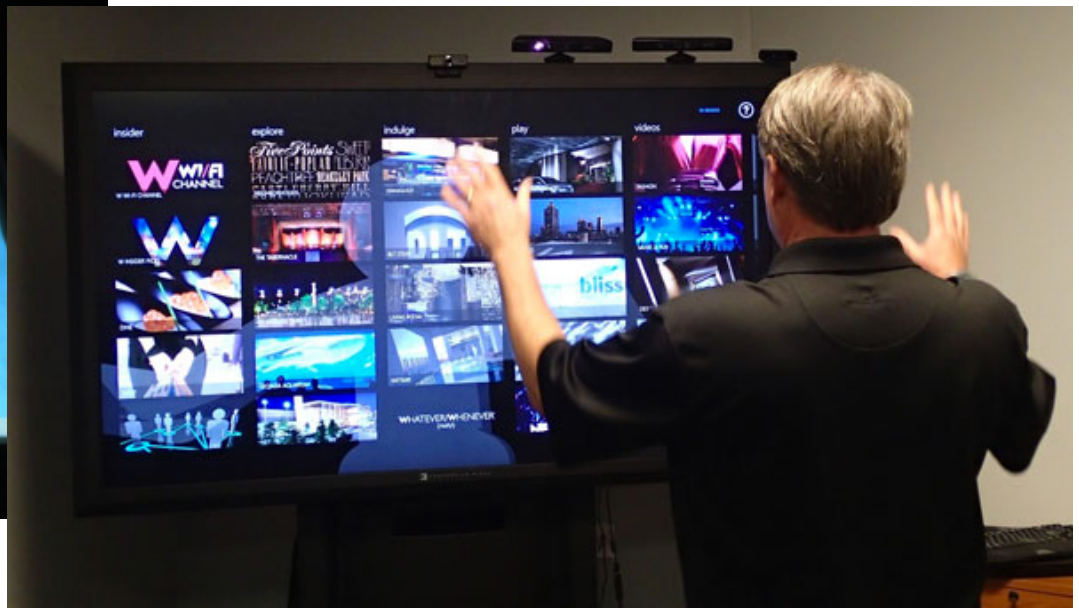
Rethinking GUIs

Human Performance

Rethinking GUIs

Natural User Interfaces

nuigroup.com



http://eecatalog.com/digital-signage/files/2013/12/131209_digitalsignage_1.jpg

Rethinking GUIs

Natural User Interfaces

http://www.inventinginteractive.com/2013/11/04/enders-game/

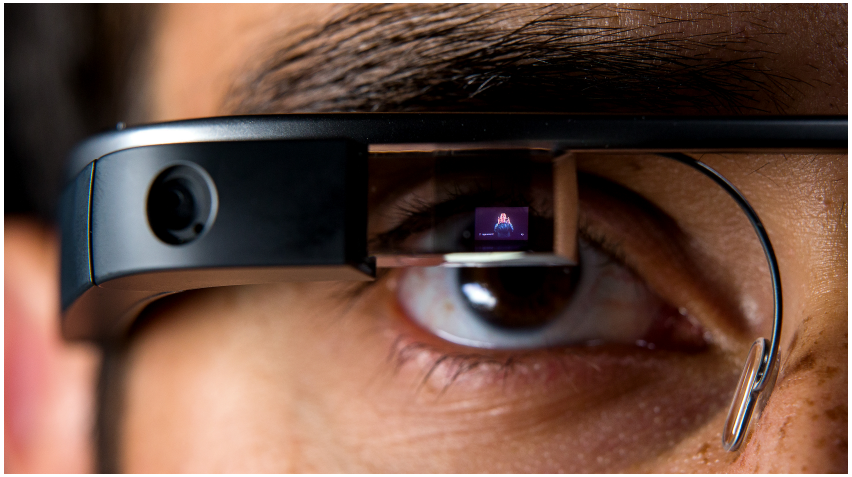
Rethinking GUIs

Natural User Interfaces

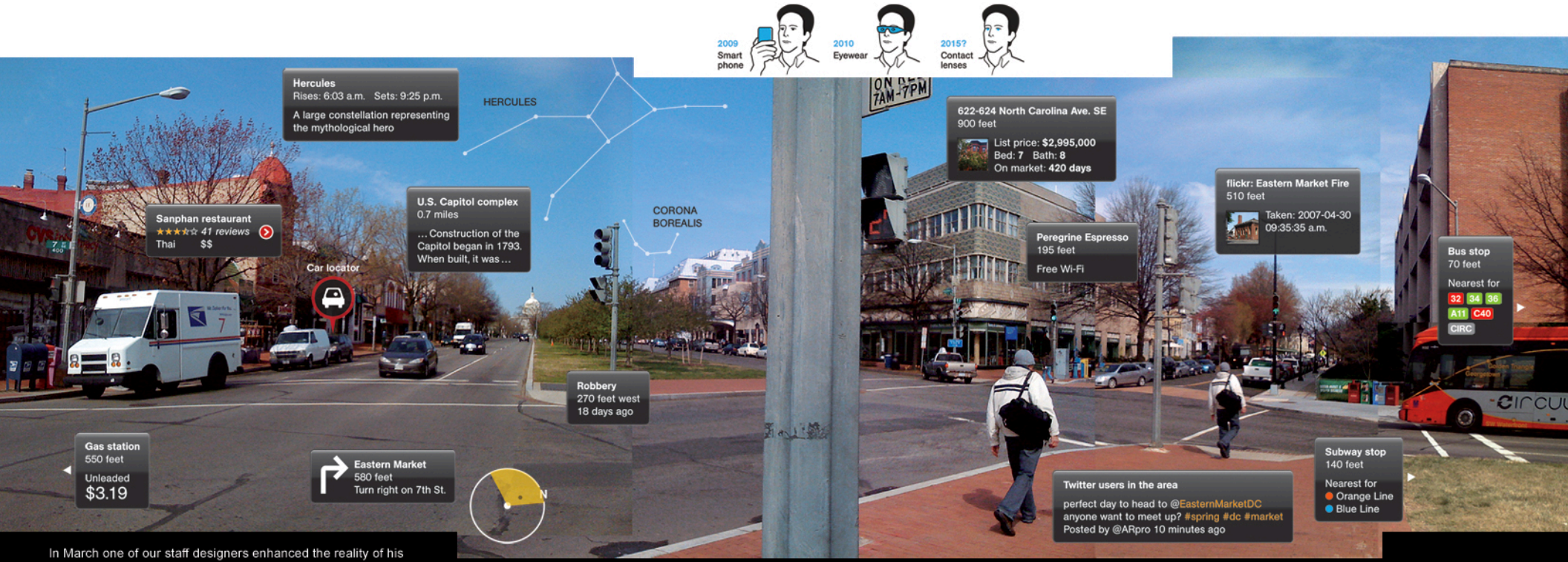
http://www.inventinginteractive.com/2013/11/04/enders-game/

Rethinking GUIs

Augmented Reality



http://www.pddnet.com/news/2014/05/google-glass-adaptation-opens-universe-deaf-students



In March one of our staff designers enhanced the reality of his Washington, D.C., neighborhood. Smart phone applications (apps) added layers of information to what he saw—called out in this composite of five photos, each taken with his phone.

UP AND AWAY Point your phone at the sky and find stars hidden by daylight. Aim at a tourist spot and see its history plus info for visitors. For an augmented-reality check, tap into crime stats.

REAL DEALS Various apps can steer you to the cheapest gas around, mass-transit options, good food, and Wi-Fi spots. You can also learn the price of that town house that's up for sale.

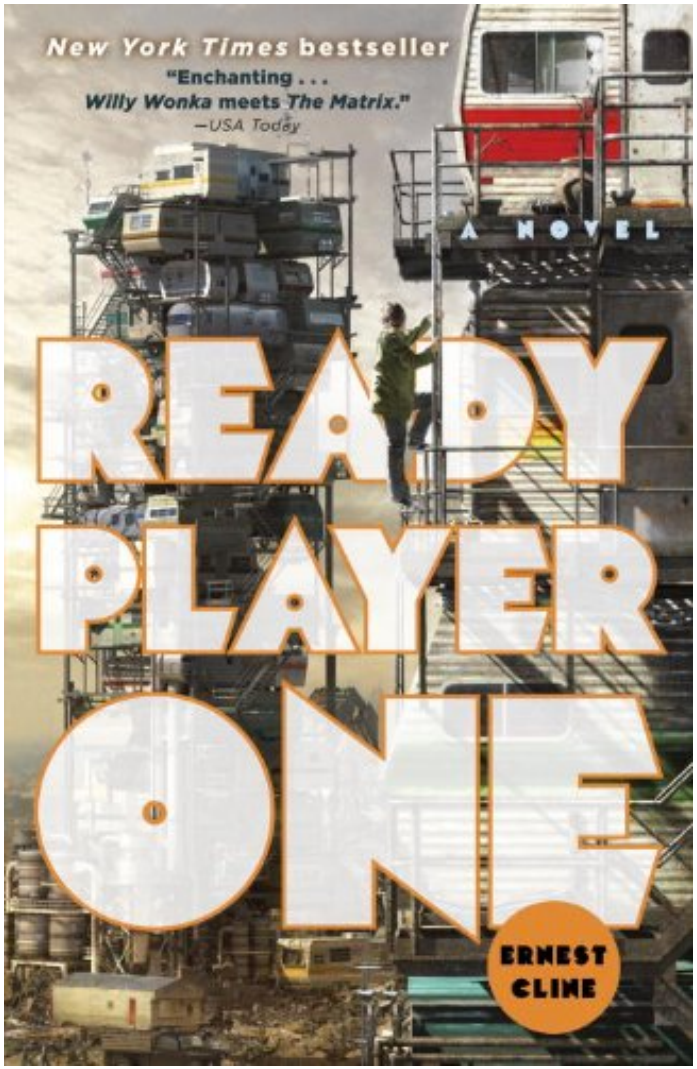
STREET PALS The Tweeps Around app tells if tweeters are near. Flickr displays area photos by members (Eastern Market, above). In the works: an app to match faces to social-network profiles.

http://ngm.nationalgeographic.com/big-idea/14/augmented-reality

Rethinking GUIs

Virtual Reality

http://upload.wikimedia.org/wikipedia/commons/7/78/AC89-0437-20_a.jpeg



http://cdn.iphonehacks.com/wp-content/uploads/2014/08/image-Oculus.jpg