1 Sebon Koo
JavaScript language
JavaScript

SEBON KOO
JavaScript

- Developed by Sun Microsystems and Netscape
- Client-side scripting language
- Widely used for enhanced user interface in websites
- Implemented by web browsers
- Multi-paradigm, weakly-typed language
Good Aspects

- Flexibility and Efficiency of Use
  - Multi-paradigm, weakly-typed
  - Supports imperative, object-oriented and functional programming styles -> many implementation options
Bad Aspects

- Consistency and Standards
  - Web browsers have made their own JavaScript engine
  - No official reference
  - Cross-browser compatibility issue (especially IE)
  - Solution -> use JavaScript framework (jQuery)

```javascript
// IE
document.getElementById("header").style.styleFloat = "left";
// Firefox
document.getElementById("header").style.cssFloat = "left";
```
Bad Aspects

- **Error Prevention**
  - Function parameter
  - Wrong data type
  - Parameters can be omitted

```javascript
window.onload = function () {
  foo(10, function() { return 'Hello World' }); // NaN
  foo(10, 10); // 100
}

function foo(a,b) {
  console.log(a*b);
}
```
Bad Aspects

- Error Prevention
  - No compiler
  - No way to catch an error in advance
  - Have to run the code to find errors
  - Solution -> Use JavaScript console
2 Steven Gardiner
App Inventor
(Google/MIT) App Inventor

Steve Gardiner

January 22, 2013
1. Goal: empower us to “take creative control of our everyday gadgets, like cell phones.”
1. Goal: empower us to “take creative control of our everyday gadgets, like cell phones.”
1. Goal: empower us to “take creative control of our everyday gadgets, like cell phones.”
1. Goal: empower us to “take creative control of our everyday gadgets, like cell phones.”
2. App Inventor Designer (web app, integrated with google)
1. Goal: empower us to “take creative control of our everyday gadgets, like cell phones.”
2. App Inventor Designer (web app, integrated with google)
3. App Inventor Block Editor (java applet, on local machine)
<table>
<thead>
<tr>
<th>Introduction to App Inventor</th>
<th>Kudos</th>
<th>Flaws</th>
<th>Closing Thought</th>
</tr>
</thead>
</table>

1. Integration with google account
1. Integration with google account
2. Immediate propagation of changes to phone
1. Integration with google account
2. Immediate propagation of changes to phone
3. Puzzle-piece metaphor
1. disjoint Designer and Block Editor

![Layout](image1)

![Behavior](image2)
1. disjoint Designer and Block Editor

Layout
Server-side Web application

Behavior
Client-side Java applet

Steve Gardiner on (Google/MIT) App Inventor
1. disjoint Designer and Block Editor
2. Lots of clicking
<table>
<thead>
<tr>
<th>Introduction to App Inventor</th>
<th>Kudos</th>
<th>Flaws</th>
<th>Closing Thought</th>
</tr>
</thead>
</table>

1. disjoint Designer and Block Editor
2. Lots of clicking
3. Types
I was able to assemble an app in about an hour without studying Android development.
I was able to assemble an app in about an hour without studying Android development

- Why don’t we all assemble pieces visually?
I was able to assemble an app in about an hour without studying Android development

- Why don’t we all assemble pieces visually?
- Is it because we like to manually write (probably buggy) code?
I was able to assemble an app in about an hour without studying Android development

▶ Why don’t we all assemble pieces visually?
▶ Is it because we like to manually write (probably buggy) code?
▶ Or is typing code faster?
I was able to assemble an app in about an hour without studying Android development

- Why don’t we all assemble pieces visually?
- Is it because we like to manually write (probably buggy) code?
- Or is typing code faster?
- Or because we need all O(1000) instructions rather than the O(100) exposed in visual programming?
3 Lia Qu
Balsamiq
Features

- Have **75 built-in user interface components** and **187 icons**, plus a whole log of community-generated components. User can simply drag and drop UI components from UI library to Mockup Canvas.

- Have linking that let UX developers generate **click-through** prototype. Because of the **sketching-like drawing**, the wireframe elicits honest feedbacks, which lead to better user interfaces.

- Export mockups as PNG or PDF, or as code using 3 party tools. Easy to share and present among stakeholders.
Positive feedbacks

- Allow user leave in the middle and resume immediately (Visibility of system status/Progressive)

- Help user **expedite work** (recognize data format, type ahead, undo/redo, easy to make changes - group, lock)

- Easy to use and **Easy to learn**

- Aesthetic and simple

- Help documents are always available
Negative feedbacks

- No **rowspan or colspan** feature on Data Grid/Table
- Data Grid/Table cannot be resized manually
- No **pencil tool** to draw basic lines or shapes
- No place to add **notes or annotations** on mockups
THANK YOU & QUESTION?
4 Yang Piao
backbone JS
Backbone.js
A Cognitive Dimensions Evaluation

Yang Piao, yp@cmu.edu, INI
Backbone.js

- A JavaScript library for creating web apps
- One of JavaScript MV* frameworks
  - Model- View- Whatever (Controller, Router, ViewModel)
  - Maintainable code
  - Spine, Ember.js, AngularJS, Knockout, etc.
- Built on jQuery/Zepto, Underscore.js/Lo-Dash, and json2.js (and template libraries that you might choose)
- Used by companies like Airbnb, Foursquare, Pandora, LinkedIn, Groupon, Walmart, ...
- backbonejs.org

Yang Piao (yp@cmu.edu)
var TodoRouter = Backbone.Router.extend({
    routes: { 'show/:id', 'show' },
    show: function(id) { ... },
    initialize: function() { ... },
    start: function() { ... },
    ...
});

var app = new TodoRouter();
app.start();
Abstraction

```javascript
var Note = Backbone.Model.extend({
    initialize: function() { ... },
    ...
});
var PrivateNote = Note.extend({ ... });
var NoteView = Backbone.View.extend({});
var NoteApp = Backbone.Router.extend({
    routes: {'': 'index'},
    index: function() {}
});
```
Consistency

Backbone.Model.extend({ ... });
Backbone.View.extend({ ... });
Backbone.Router.extend({ ... });
Backbone.Collection.extend({ ... });

var object = new SomeView({});
// event binding for all objects in Backbone
object.on('event', function(e) {});
var TodoView = Backbone.View.extend({
    ...
    template:
        _.template('<h3><%= description %></h3>'),
    render: function() {
        var attributes = this.model.toJSON();
        this.$el.html(this.template(attributes));
    }
});

var view = new TodoView({model: todoItem})
Meaning of Code

// model class
var TodoItem = Backbone.Model.extend({});

// model instance
var todoItem = new TodoItem({
    description: 'Pick up milk',
    status: 'incomplete'
});
todoItem.get('description');
todoItem.set({status: 'complete'});

Yang Piao (yp@cmu.edu)
Things to Remember

• How to define/derive a class
• Names of some special methods and attributes
• How events work
• How router works
• How views and models interact
• How to write template in views
• The best practices
• ... ...
Other Thoughts

• Progressive evaluation
• Relatively low viscosity
• Lack of good development environments
• Design of usable APIs is hard
• Trade-offs: usability, performance, constraints of platforms, complexity of implementations
Thanks for your attention
5 Ramya Balaraman
Microsoft Expressions Blend
HEURISTIC EVALUATION OF MICROSOFT EXPRESSION BLEND

- By Ramya Balaraman (rbalaram@andrew.cmu.edu)
About Microsoft Expression Blend

- Visual tool used for designing and prototyping desktop and web applications.

- Visual design represented by XAML, the markup language for WPF.

- Can be used to create user interfaces for –
  1. Microsoft Windows application built on Windows Presentation Foundation (WPF).
  2. Web applications built on Microsoft Silverlight.
  3. Interactive prototypes using SketchFlow.
  4. Windows phone applications.
Sample Project – Zune3D

Showing images and objects rotated in a 3D projection transformation
Things Done Well

- Visibility of System Status
- Flexibility and efficiency of use
- Help users recognize, diagnose and recover from error.
- Help and Documentation/ Recognition rather than Recall
Visibility of System Status
Visibility of System Status
Visibility of System Status

Gradient Eyedropper tool for picking color
Flexibility & Efficiency of Use

[Image showing a screenshot of a software interface with settings for layout, including Width, Height, Row, Column, ZIndex, HorizontalAlignment, and Margin values.]
Flexibility & Efficiency of Use
Help users recognize, diagnose & recover from error

Invalid XAML

Check the Results panel for detailed information.

```
19 <GradientStop Color="#FD59F22" Offset="0.069"/>
20 </LinearGradientBrush>
21 </Ellipse.Fill>
22 <Button>
23 <Button>
24 </Button>
25 </Button>
</Grid>
</Ellipse>

The closing XML tag "Button" is mismatched.

```
```
<LinearGradientBrush EndPoint="0.5,1" StartPoint="0.5,0">
  <GradientStop Color="Black" Offset="0.871"/>
  <GradientStop Color="#FF45340E" Offset="0.431"/>
</LinearGradientBrush>

```
```
<Ellipse Margin="253,167,243,185" Fill=""/>

```
```
```
The property "Fill" is set multiple times.
```
Help and Documentation/ Recognition rather than Recall
Violations

- Standards
- User Control and Freedom
- Recognition rather than Recall
User Control & Freedom

```xml
<Ellipse.Fill>
  <LinearGradientBrush EndPoint="0.5,1" StartPoint="0.5,0">
    <GradientStop Color="Black" Offset="0.871"/>
    <GradientStop Color="#FF45340E" Offset="0.401"/>
    <GradientStop Color="#FFD59F22" Offset="0.069"/>
    <GradientStop Color="#FF926D18" Offset="0.237"/>
    <GradientStop Color="#FF271D07" Offset="0.621"/>
  </LinearGradientBrush>
</Ellipse.Fill>
```
Recognition rather than Recall

Project > Add Reference >
Microsoft.Expressions.Effects.dll from
Program Files/ Microsoft SDK’s/ Expression/
Blend/ Silverlight/ v4.0/ Libraries
Thank You!!!
6 Kerry Chang
Microsoft touchdevelop
Overview

• A programming environment designed for mobile touch devices.

• Lets people write and execute code directly on the mobile devices.

• Current runs as a Web application at https://www.touchdevelop.com/app/
Overview

• Challenge: *Small screen estate*
  – Output
  – Input

• Solution: Code is inserted (mostly) by tapping on icons instead of directly typing on keyboard
Example 1: Hello World
```
action main ()
```
```java
public class Main {
    public static void main() {
        System.out.println("Hello world");
        // we have a String here; it doesn't do anything by itself; insert 'post to wall' if you want to display it or just delete it
    }
}
```
```java
public class Main {  
  public static void main(String[] args) {  
    System.out.println("Hello world");  
  }  
}
```

<table>
<thead>
<tr>
<th>post to wall -- Displays string on the wall</th>
</tr>
</thead>
<tbody>
<tr>
<td>new line</td>
</tr>
<tr>
<td>123</td>
</tr>
</tbody>
</table>
action main()

"Hello world" → post to wall

"Today is so cold" → post to wall
Today is so cold
Hello world
Example 2: Bouncing ball
action main ()
time→ create
(year, month, day, hour, minute, second) : DateTime :: Creates a new date instance

media→ create board
(height) : Board :: Creates a new game board

social→ create contact
(nickname) : Contact :: Creates a new contact

maps→ create full map
 : Map :: Creates a full screen Bing map. Use 'post to wall' to display it.

media→ create landscape board
(width, height) : Board :: Creates a new game board in landscape mode. On rotatable devices it will take the...
```javascript
function main ()
{
    media-> create board(640)
    // 'create board' returns a 'Board'; insert 'post to wall' if you want to display it
}
```

create board(height : Number) : Board -- Creates a new game board

store in var
- new variable
- Adds numbers
- Subtracts numbers
- Multiplies numbers
- Divides numbers
- Compares numbers for equality
- Compares numbers for more
- there's more (1/3)

paste  copy  cut  mark  backspace  undo
main ()

board := |media| -> create board(640)

create board(heigh : Number) : Board --> Creates a new game board
```plaintext
action main()

var board := media→ create board(640)
```
`board := media → create board(640)`

**board**

- We have a Board here; it doesn't do anything by itself; insert 'post to wall' if you want to display it or just delete it.

<table>
<thead>
<tr>
<th>board : Board -- a local variable</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>:=</th>
<th>rename</th>
<th>promote to data</th>
<th>set background</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignment</td>
<td>set local variable name</td>
<td>to global var</td>
<td>Sets the background color</td>
</tr>
<tr>
<td>create ellipse</td>
<td>update on wall</td>
<td>post to wall</td>
<td>backspace</td>
</tr>
<tr>
<td>Create a new ellipse sprite.</td>
<td>Make updates visible.</td>
<td>Shows the board on the wall.</td>
<td>undo</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>123</th>
<th>&quot;(),&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>number entry</td>
<td>&quot;&quot;, not true</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>move cursor</th>
<th>move cursor</th>
</tr>
</thead>
<tbody>
<tr>
<td>left</td>
<td>right</td>
</tr>
</tbody>
</table>
def main():
    board := media
    create board(640)

    board.set background(colors random)

random: Color -- Picks a random color

- simplify: current expression
- darken: Makes a darker color by a delta
- make transparent: Creates a new color by changing
- lighten: Makes a lighter color by a delta
- equals: Checks if the color is equal to the current expression
- concatenate: Concatenates two pieces of text
- R: Gets the red value (0.0-1.0)
- there's more (1/3): More options
- 123: Number entry
- "(,)": "...", not true
- move cursor
- move cursor
- backspace
- undo
main()

board := media
create board(640)

board -> set background(colors)

' set background' expects Color here, got Colors

set background(color : Color) -- Sets the background color
action main ()

var board := media→ create board(640)

board→ set background(colors→ blue)

board→ create ellipse(20, 20)

'create ellipse' returns a 'Sprite'; insert 'post to wall' if you want to display it

board→ post to wall
code

new
add new action

events

new
add new event

data
gameloop()  an event handler
shake()  an event handler
phone face up()  an event handler
phone face down()  an event handler
phone portrait()  an event handler
phone landscape left()  an event handler
phone landscape right()  an event handler
event gameloop ()

- board ➔ evolve
- board ➔ update on wall
Evaluation

1. Error prevention
2. Language design
3. Recognition rather than recall
4. Help and documentation
5. Consistency
6. Aesthetic
& learn

Kerry Chang

showcase

tags

1. Getting started
2. Advanced editing
3. Create a game!
4. Export to Windows Store
5. Learn More

Watch Video
Watch Video
Live Tutorial

TouchInvaders
David Renton

RPN Calculator
Stefan Falk

See More

turtle triangle spiral
TouchDevelop Sample

Wabbit Wars
Wabbit Apps

snowman in snowfall
white FOX

© 2012 Microsoft

privacy legal

feedback

entertainment
tools
games
Evaluation

7. Efficiency
8. No redo and very limited undo
9. No other debugging tools besides error message
10. Running in the browser
In main()

board := media -> create board()

_not enough parameters supplied to create board_

sprite := board -> create ellipse(40, 40)

rite := set color(colors, blue)

create board(height : Number) : Board -- Creates a new game board

create ellipse
Create a new ellipse sprite.

post to wall
Shows the board on the wall.

set background
Sets the background color

create text
Create a new text sprite.

update on wall
Make updates visible.

width
Gets the width in pixels

height
Gets the height in pixels

there's more (1/5)

123

"\(,\)"

"...", not, true

move cursor

move cursor
7 Vishal Dwivedi
Yahoo Pipes
Evaluating Yahoo Pipes
Based on Neilsen's Usability Heuristics

Vishal Dwivedi
Institute for Software Research
Carnegie Mellon University

05-830: Advanced User Interface Software

Home Work 1
Yahoo Pipes

Find an Apartment in Pittsburgh

Ever wanted to find an apartment close to a park, a school or Whole Foods? Now you can...

Pipe Web Address: http://pipes.yahoo.com/pipes/pipes/pipe.info?id=48c4809d7b3fe91a90656c55799dde4f (edit)

Configure this Pipe

What is your minimum price? 100
What is your maximum price? 3000
Find an apartment near what cmu
Minimum distance (miles) 3

Run Pipe

Use this Pipe

Get as a Badge + My Yahoo! Google Get as RSS Get as JSON More options

Map List

7 items
## Heuristic Evaluation

<table>
<thead>
<tr>
<th>Supported (+)</th>
<th>Inhibited (-)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visibility of system status (+)</td>
<td>User control and freedom (-)</td>
</tr>
<tr>
<td>Consistency and standards (+)</td>
<td>Error prevention (-)</td>
</tr>
<tr>
<td>Recognition rather than recall (+)</td>
<td>Help users recognize, diagnose, and recover from errors (-)</td>
</tr>
<tr>
<td>Aesthetic and minimalist design (+)</td>
<td></td>
</tr>
<tr>
<td>Help and Documentation (+)</td>
<td></td>
</tr>
<tr>
<td><strong>Match between system and the real world (+/-)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Flexibility and efficiency of use (+/-)</strong></td>
<td></td>
</tr>
</tbody>
</table>
Error Prevention (-)

~81% of Yahoo Pipes have problems (Stolee et al, ICSE 2011)

- Yahoo pipes does a real bad job with error prevention.
  - Dumps XML warnings for erroneous pipes
  - Error checks limited to only restricting port types
  - Accidental module deletion → Start from scratch
    - UNDO – REDO “extremely difficult” to implement because of the architecture

Suggestions:
- Support an analyses windowpane that allows automated analyses
- Save intermediate pipes
Recognition rather than recall (+)

- Minimize the user's memory load by making objects, actions, and options visible.
- Yahoo Pipes does a good job here
  - Allow iterative addition of rules
  - Simplified constructs for designing rules
Conclusion

- Overall,
  - Yahoo Pipes does a good job with aesthetics and simplicity.
  - Needs significant improvement over error prevention and diagnosis
  - The choice of the UI library restricts usability mechanisms.
8  Juan Gonzalez Restrepo
   Pencil Project
Pencil Project

usability evaluation
Advanced User Interface Software
Spring 2013
Juan Gonzalez
Pencil Project

- Pencil is an open source GUI prototyping tool that allows creating mockups for different platforms: desktop, android, iOS.
Good usability aspects

- Minimalistic design
  - The system follows the principles of minimalistic design. Dialogs are designed to show what the user needs to see.
Good usability aspects

- **Flexibility and efficiency of use**
  - Save created GUI Objects
Good usability aspects

- Recognition rather than recall and Flexibility and efficiency of use

  - GUI Objects with description and search feature
Bad usability aspects

- Consistency and standards & efficiency of use
  - Delete key doesn’t work as in most graphic interface builders
Bad usability aspects

- *Error prevention*
  - Dragging objects out of the canvas
Bad usability aspects

- *Match between system and the real world*
  - Code used as names
Bad usability aspects

• Recognition rather than recall
  ➢ No button to close dialogs, have to use “esc” key
9  Chen Feng
Flex
REPORT FOR EVALUATING THE USABILITY OF FLASH CS6 PROFESSIONAL

Chen Feng
The Tool
Good Feature
Good Feature
Good Feature
Good Feature
Good Feature
Good Feature
Bad Feature
10 Truc Nguyen
LabVIEW
LABVIEW EVALUATION
Overview

- Data-flow programming language paradigm
- Drag-and-drop graphical language “G”
- Front Panel
  - UI elements
- Block Diagram
  - Wires, function/variable blocks, and boxes
Overview

Front Panel

Block Diagram
Heuristics Violated

- Simple arithmetic, logical or array operations must be completed using wires
  - Flexibility and efficiency of use

- Block icons not visually distinct
  - Recognition rather than recall

- Debugging tools are limited in LabVIEW
  - Help users recognize, diagnose, and recover from errors
Simple Operations Tedious to Wire
Simple Operations Tedious to Wire
Block Icons Not Visually Distinct
Limitations of Debugging

- Difficult to feed in test/"dummy" data
- Sometimes not possible to inspect variables/run-time stack when stepping through code
Things Done Well

- Language design allows for modularity (Sub-Vis)
  - Flexibility and efficiency of use
- UI elements (controls and indicators) mimic physical lab equipment
  - Match between system and real world
- Contextual help option on right-click
  - Help and documentation
Someone Who Didn’t Use Sub-VIs
Sub-VIs
Mapping to Real Instruments
Mapping to Real Instruments
Contextual Help

Delete From Array Function

- **Owning Palette:** Array Functions
- **Requires:** Base Package

This function deletes an element or subarray from an n-dimensional array and returns the edited array. It operates on an array of length `length`, with a subset deleted from the position specified by `index`. The default index is the last element in the array.

**Details**
- `n-dimensional array`: The array from which you want to delete elements. This can be an array of any type, including elements, rows, columns, and pages.
- `length`: Determines how many elements, rows, columns, or pages to delete.
- `index`: Specifies what you want to delete from the array. In an n-dimensional array, row is the first major index, column is the last minor index. In a 2D array, row is followed by page, then column.

**Example**

```
// Example of using the Delete From Array Function

// Create an array
Array arr = [1, 2, 3, 4, 5];

// Delete element at index 2
arr = Delete From Array(arr, 2);

// Display the result
print(arr); // Output: [1, 2, 4, 5]
```
GTK+

- GIMP Toolkit
- Originally created to support the GIMP image editing software on Gnome/Linux
- Now cross platform
  - Microsoft Windows
  - Apple OSX
  - Some mobile devices
  - OLPC
GTK+ API Hierarchy
Examples
Good Features

• Easily Themed for native look and feel
• Accessibility is baked in
• Expressive syntax
  – gtk_new_window (...)  
  – gtk_drawing_area_new (...)  
  – gtk_container_add (...)

Concerns

• Underscore character disruptive and error prone
• Diffusive
  – Avoidable when used with languages like Python
12 Julia Schwarz
Qt framework
if (m_PixelData[y * m_numRows + x] > 0) {
    painter.setBrush(QBrush(Qt::black, Qt::SolidPattern));
} else {
    painter.setBrush(QBrush(Qt::white, Qt::SolidPattern));
}
painter.drawRect(QRect(x * stepX, y * stepY, (x+1) * stepX, (y+1) * stepY));
Good: Signals + Slots

private slots:
    void timer_timeout();

// initialize timer
m_timer = new QTimer(this);
connect(m_timer, SIGNAL(timeout()),
       this, SLOT(timer_timeout()));

... emit timeout();

---

VS.

//Step 1. Class that defines data for the event
//
public class AlarmEventArgs : EventArgs
{
    private readonly bool snoozePressed = false;
    private readonly int nrings = 0;
    // Constructor.
    public AlarmEventArgs(bool snoozePressed, int nrings) {...}
    // Properties.
    public int NumRings{ get { return nrings;}}
    public bool SnoozePressed { get { return snoozePressed;}}
    public string AlarmText { get {...}}
}

//Delegate declaration.
public delegate void AlarmEventHandler(object sender, AlarmEventArgs e);

// Class definition.
//
public class AlarmClock
{
    //Step 3. The Alarm event is defined using the event keyword.
    //The type of Alarm is AlarmEventHandler.
    public event AlarmEventHandler Alarm;
    //
    //Step 4. The protected OnAlarm method raises the event by invoking
    //the delegates. The sender is always this, the current instance of
    //the class.
    //
    protected virtual void OnAlarm(AlarmEventArgs e)
    {
        if (Alarm != null)
        {
            //Invokes the delegates.
            Alarm(this, e);
        }
    }
}
Good: Describe Interface 3 Ways

```xml
<xml version="1.0" encoding="UTF-8">
<root version="4.0">
  <class>QWidget</class>
  <widget class="QMainWindow" name="MainWindow">
    <property name="geometry">
      <rect>
        <x>0</x>
        <y>0</y>
        <width>409</width>
        <height>425</height>
      </rect>
    </property>
    <property name="windowTitle">
      <string>life</string>
    </property>
  </widget>
  <widget class="QWidget" name="centralWidget">
    <widget class="QWidget" name="layoutWidget">
      <property name="geometry">
        <rect>
          <x>10</x>
          <y>10</y>
          <width>391</width>
          <height>141</height>
        </rect>
      </property>
    </layout>
  </widget>
</root>
```

```java
QMainWindow life = new QMainWindow();
centralWidget = new QWidget(life);
centralWidget->setObjectName(QStringLiteral("centralWidget"));
layoutWidget = new QWidget(centralWidget);
layoutWidget->setObjectName(QStringLiteral("layoutWidget"));
verticalLayout = new QVBoxLayout(layoutWidget);
verticalLayout->setSpacing(6);
verticalLayout->setContentsMargins(11, 11, 11, 11);
startButton = new QPushButton(layoutWidget);
startButton->setObjectName(QStringLiteral("startButton"));
```
Bad: Error Visibility in IDE

VS.

Bad: Error Visibility in IDE

VS.
13 YoungSeok Yoon
Java SWT
EVALUATION OF JAVA SWT USING THE COGNITIVE DIMENSIONS

05-830 Advanced User Interface Software, Spring 2013
Homework#1

YoungSeok Yoon
(youngseok@cs.cmu.edu)
The Standard Widget Toolkit (SWT)

- A graphical widget toolkit for Java, mainly developed for Eclipse GUI
  - Standalone applications can also be built using SWT
  - Provides widgets such as buttons, and various layout managers

- Java GUI toolkits: AWT $\rightarrow$ Swing $\rightarrow$ SWT
  - Carefully designed to have the advantages of both AWT and Swing
    - Ease of development, portability, native look-and-feel and performance

- SWT is often used in conjunction with JFace, which provides useful helper classes built on top of SWT
  (This presentation focuses only on SWT and its documentation)
Quality documentation (Learning style +)

- Meets the needs of both the opportunistic/pragmatic and systematic developers by providing different types of documentation.

- Widget catalog

- Javadoc

```
**Overview** | **Package** | **Class** | **Use** | **Tree** | **Deprecated** | **Index** | **Help**

Package org.eclipse.swt.widgets

SWT widget public API classes.

See: Description

**Interface Summary**

- **Listener**
  Implementers of **Listener** provide a simple **handleEvent**() method that is used internally by SWT to dispatch events.

**Class Summary**

- **Button**
  Instances of this class represent a selectable user interface object that issues notification when pressed and released.

- **Canvas**
  Instances of this class provide a surface for drawing arbitrary graphics.

- **Caret**
  Instances of this class provide an i-beam that is typically used as the insertion point for text.

- **ColorDialog**
  Instances of this class allow the user to select a color from a predefined set of available colors.

- **Combo**
  Instances of this class are controls that allow the user to choose an item from a list of items, or optionally enter a new value by typing it into an editable text field.
```
Consistency within the API (Consistency +)

• Widget constructors
  - Browser (Composite parent, int style)
  - Button (Composite parent, int style)
  - Canvas (Composite parent, int style)
  - Combo (Composite parent, int style)
  - Composite (Composite parent, int style)
  - DateTime (Composite parent, int style)
  - Group (Composite parent, int style)
  - Label (Composite parent, int style)

• Layout class names
  - RowLayout – RowData
  - GridLayout – GridData
  - FormLayout – FormData

Once the user gets used to some part of the toolkit, the usage of the rest can be easily inferred.
How to obtain a **BLACK** color object?
(*Work-step unit – / Penetrability –*)

```java
<SWT>
Color black = 
    Display.getDefault().getSystemColor(SWT.COLOR_BLACK);

<AWT/Swing>
Color black = Color.BLACK;

<.NET Windows Forms>
Color black = Color.Black;
```

- **Problems**
  - The color ID is in SWT class, whose name has nothing to do with colors.
  - A method call is required for getting a simple color object.
  - `getSystemColor` method is in the “Display” class, which is not very intuitive.
  - In order to get the Display object, a static factory method should be called.
    (which is known to have usability problems)

- **cf. other toolkits**
Not intended to be subclassed?
(API elaboration – / Penetrability –)

• Most widgets are not intended to be subclassed.
• Difficult to add new functionalities to a widget.
  - e.g., Button → Roll-over Image Button?
• Users might not be aware of this.
  (the class is not marked as final)
• Hard to find the alternatives.
• In fact, users CAN subclass, though not recommended, but then they also must override “checkSubclass” method and make it do nothing.
  (very hard to get to know this)

```java
public class Button extends Control

Instances of this class represent a selectable user interface object that issues notification when pressed and released.

Styles:
  ARROW, CHECK, PUSH, RADIO, TOGGLE, FLAT, WRAP UP, DOWN, LEFT, RIGHT, CENTER

Events:
  Selection

Note: Only one of the styles ARROW, CHECK, PUSH, RADIO, and TOGGLE may be specified.

Note: Only one of the styles LEFT, RIGHT, and CENTER may be specified.

Note: Only one of the styles UP, DOWN, LEFT, and RIGHT may be specified when the ARROW style is specified.

IMPORTANT: This class is not intended to be subclassed.

See Also:
  Button snippets, SWT Example, ControlExample, Sample code and further information

Restriction:
  This class is not intended to be subclassed by clients.
```
Conclusion

• In general, SWT is very well-documented and easy to use
  • But still, Cognitive Dimensions (CDs) could identify several usability problems that could have been fixed fairly easily.

• Often a single problem relates to two or more cognitive dimensions.

• It is difficult to evaluate an API in isolation.
The followings should also be taken into account:
  • The programming language in which the API is written (e.g. Java, Javadoc)
  • The IDE used by the users (e.g. Eclipse and code completion feature)
  • Other APIs that are often used together with the target API (e.g. JFace with SWT)
  • Any interactive tools that are related to the API (e.g. WindowBuilder)
14 Erik Harpstead
Unity GUI
Unity

- Game engine architecture
- Programming is done using an Entity-Component model
- GUI code can be implemented within any component in a scene
using UnityEngine;
using System.Collections;

public class GUIExample : MonoBehaviour {
    void OnGUI() {
        // Make a background box
        GUI.Box(new Rect(10, 10, 100, 90), "Loader Menu");

        // Make the first button. If it is pressed,
        // Application.LoadLevel (1) will be executed
        if (GUI.Button(new Rect(20, 40, 80, 20), "Level 1")) {
            Application.LoadLevel(1);
        }

        // Make the second button.
        if (GUI.Button(new Rect(20, 70, 80, 20), "Level 2")) {
            Application.LoadLevel(2);
        }
    }
}
Flexibility and Efficiency of Use

+ Simplified single line implementation puts positioning, drawing, and handling in a single function call

```csharp
void OnGUI() {
    if (GUI.Button(new Rect(10, 10, 40, 25), "My Button")) {
        Debug.Log("Click");
    }
}
```
Consistency and Standards

Windowing and Layout systems are conceptually similar but have completely different functional APIs.
Visibility of System Status

+ Unity’s active editing system allows for editing of properties while the game is running

- Active editing is not as integrated into the GUI as other systems
Error Diagnosis and Recovery

- `OnGUI()` can be implemented anywhere within a scene making it difficult to diagnose where errors occur
Match Between System and the Real World

+ The system is designed from the perspective of a game engine.

– The system does not resemble a normal GUI system and so it can be hard for new developers.
Documentation and Help

+ Unity’s documentation is consistent for all three supported languages

+ Unity provides a concise set of manual pages on the use of all of the GUI Controls and features
15 Chalalai Chaihirunkarn
Microsoft Visual Studio
Homework #1: Evaluate the Usability of a User Interface Tool or Toolkit

Microsoft Visual Studio

By: Chalalai Chaihirunkam

05-830, Advanced User Interface Software, Spring, 2013
Microsoft Visual Studio

- An integrated development environment (IDE)
- Provide a source code editor, built automation, a debugger, a designer, and other software development tools
- Used to develop desktop applications, mobile applications, websites, etc.
- Supports many programming languages
- Available in many editions: Express, Professional, Premium, Ultimate, and Test Professional
Microsoft Visual Studio 2012 Ultimate Edition
Neilsen's Ten Usability Heuristics

- Visibility of system status
- Match between system and the real world
- User control and freedom
- Consistency and standards
- Error prevention
- Recognition rather than recall
- Flexibility and efficiency of use
- Aesthetic and minimalist design
- Help users recognize, diagnose, and recover from errors
- Help and documentation
Visibility of system status

“The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.”

☑️ Good
Match between system and the real world

“The system should speak the users’ language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.”
"Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo."

☑️ Good
Consistency and standards

“Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.”

Bad
Help users recognize, diagnose, and recover from errors

“Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.”

☑️ Good
16 Yanan Qi
Java Swing
05-830, AU IS
Homework #1:
Java Swing Evaluation

Yanan Qi
yananq@andrew.cmu.edu
School of Computer Science
MSIT eBusiness Technology
Java Swing

- Part of the Java Foundation Classes (JFC)
- Provides a sophisticated set of GUI components
- Used to create a Java program with a graphical user interface (GUI)
Extensible
Customizable
Configurable
Lightweight UI
Loosely Coupled and MVC
What’s good?

- Visibility of System Status
- User Control and Freedom
- Consistency and Standards
- Error Prevention
- Recognition Rather Than Recall
- Flexibility and Efficiency of Use
- Help and Documentation
writecheck.addActionListener(this);

makedeposit.

dateField =
descriptionField =
amountField =
errormessage =
textArea =
textArea.setFont(fixedFont);
```java
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.JButton;

textArea = new JTextArea(30, 100);
textArea.setFont(fixedFont);
textArea.setEditable(false);

JScrollPane scroller = new JScrollPane(textArea);
scroller.setHorizontalScrollBarPolicy(ScrollPaneConstants.HORIZONTAL_SCROLLBAR_NEVER);
```
```java
public ButtonPanel() {
    // create buttons
    redButton = new JButton("RED");
    blueButton = new JButton("BLUE");
    // add buttons to current panel
    add(redButton);  // add button to current panel
    add(blueButton); // add button to current panel
    // register the current panel as listener for the buttons
    redButton.addActionListener(this);
    blueButton.addActionListener(this);
} // ButtonPanel constructor
```
javax.swing

Class JTextArea

javax.swing.text.JTextComponent
javax.swing.JComponent
java.awt.Container
java.awt.Component
java.lang.Object

All Implemented Interfaces:
ImageObserver, MenuContainer, Serializable, Accessible, Scrollable

public class JTextArea
extends JTextComponent

A JTextArea is a multi-line area that displays plain text. It is intended to be a lightweight component that provides source compatibility with the java.awt.TextArea class where it can reasonably do so. You can find information and examples of using all the text components in Using Text Components, a section in The Java Tutorial.

This component has capabilities not found in the java.awt.TextArea class. The
What’s bad?

- Visibility of System Status
- Error Prevention
- Flexibility and Efficiency of Use
17  Michael Helmbrecht
iOS xCode programming environment
Overview

- IDE by Apple for developing OS X and iOS
- Code checking, autocomplete, documentation, compilation, debugging, simulator
Overview

- Interface Builder
Good

- Overall good attention to design from a heuristics standpoint
- Has some quirks that some don’t like
- I’m probably biased from repeated use
Bad

- Bugs in Xcode get weird and mysterious
- Known for random passing bugs
Evaluation

- Heuristic evaluation helps understand some of the design decisions
- But only for the 98% of the time it works
Xcode
Michael Helmbrecht
18 Amber McConahy
Eclipse
Heuristic Evaluation of Eclipse

Amber Lynn McConahy
Eclipse 101

• Integrated development environment (IDE)
• Offers graphical interface to support software development activities
  • Includes editor and compiler
• Supports multiple programming languages
  • Java, C/C++, Python, Android, etc.
• Heuristic evaluation performed
Visibility of System Status

- Shows current status of running application
- Provides user with relevant feedback
Error Prevention

- Red and yellow icons in side bar with suggested fix
- Inability to run an application with syntax errors
  - This could be improved by providing a link to the syntax error and a description of the problem
Recognition Rather than Recall

• Content Assist helps developers by providing legal coding suggestions
• Toolbars and menus provide access to frequent functions
  • Toolbar uses cryptic icons that are not easily identified. This could be improved through the use of more recognizable icons
Flexibility and Efficiency of Use

- Content Assist
- Code Templates
- Code Generators
  - Source menu
  - Generate frequent functions
    - Getters and Setters
    - ToString
    - Constructors
Aesthetics and Minimalist Design

- Grouping into windows
- Use of menus and toolbars
  - Window menu icon is not obvious and may be overlooked by users.
Help Users Recognize, Diagnose, and Recover from Errors

- Provides error messages with link to code, line number, and description of error
- Many error messages rely on API exception handling, which requires API designers to implement useful messages for exceptions
- Full call stack overwhelming and could be collapsed under “Caused by”
Help and Documentation

- Searchable, easily understood documentation with step-by-step usage scenarios
- No access to API documentation
- Adding API documentation would be helpful
Questions
19 Karin Tsai
Android toolkit
Android Development Tools (ADT) for Eclipse IDE

05-830 Spring 2013 - Karin Tsai
Overview

- Officially supported IDE for Android development
- Java semantics and syntax
- Most popular mobile platform
- Used Neilsen’s Ten Usability Heuristics for evaluation
Strengths

• Visibility of System Status
  – Real-time feedback through LogCat view

• Error Prevention
  – Fully customizable Lint Error Checking

• Recognition Rather Than Recall
  – Hover-over documentation and auto-complete

• Help and Documentation
  – Thorough and professional website with documentation
LogCat

```java
@override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_lesson_list);
    if (findViewById(R.id.lesson_detail_container) != null) {
        mTwoPane = true;
        ((LessonListFragment) getSupportFragmentManager().findFragmentById(R.id.lesson_list)).setActivateOnItemClick(true);
    }
}
System.out.println("List Created!");
```
Lint Error Checking

Lint Error Checking

- When saving files, check for errors
- Run full error check when exporting app and abort if fatal errors are found

Issues:

<table>
<thead>
<tr>
<th>ID</th>
<th>Correctness</th>
</tr>
</thead>
<tbody>
<tr>
<td>SdCardPath</td>
<td>Looks for hardcoded references to /sdcard</td>
</tr>
<tr>
<td>NewApi</td>
<td>Finds API accesses to APIs that are not supported in all targeted API versions</td>
</tr>
<tr>
<td>DuplicateIncludeIds</td>
<td>Checks for duplicate ids across layouts that are combined with include tags</td>
</tr>
<tr>
<td>DuplicateIds</td>
<td>Checks for duplicate ids within a single layout</td>
</tr>
<tr>
<td>UnknownId</td>
<td>Checks for id references in RelativeLayouts that are not defined elsewhere</td>
</tr>
<tr>
<td>UnknownIdInLayout</td>
<td>Makes sure that @+id references refer to views in the same layout</td>
</tr>
<tr>
<td>StateListReacheable</td>
<td>Looks for unreachable states in a &lt;selector&gt;</td>
</tr>
<tr>
<td>StyleCycle</td>
<td>Looks for cycles in style definitions</td>
</tr>
<tr>
<td>ScrollViewSize</td>
<td>Checks that ScrollView use wrap_content in scrolling dimension</td>
</tr>
<tr>
<td>Deprecated</td>
<td>Looks for usages of deprecated layouts, attributes, and so on.</td>
</tr>
<tr>
<td>NestedScrolling</td>
<td>Checks whether a scrolling widget has any nested scrolling widgets within</td>
</tr>
<tr>
<td>ScrollViewCount</td>
<td>Checks that ScrollView has exactly one child widget</td>
</tr>
<tr>
<td>AdapterViewChildren</td>
<td>Checks that AdapterViews do not define their children in XML</td>
</tr>
<tr>
<td>GridLayout</td>
<td>Checks for potential GridLayout errors like declaring rows and columns outside the declared grid dimensions</td>
</tr>
<tr>
<td>DalvikOveride</td>
<td>Looks for methods treated as overrides by Dalvik</td>
</tr>
<tr>
<td>OnClick</td>
<td>Ensures that onClick attribute values refer to real methods</td>
</tr>
<tr>
<td>DefaultLocale</td>
<td>Finds calls to locale-ambiguous String manipulation methods</td>
</tr>
<tr>
<td>SimpleDateFormat</td>
<td>Using SimpleDateFormat directly without an explicit locale</td>
</tr>
<tr>
<td>Registered</td>
<td>Ensures that Activities, Services and Content Providers are registered in the manifest</td>
</tr>
<tr>
<td>MissingRegistered</td>
<td>Ensures that classes referenced in the manifest are present in the project or libraries</td>
</tr>
<tr>
<td>Instantiateable</td>
<td>Ensures that classes registered in the manifest file are instantiateable</td>
</tr>
<tr>
<td>InnerclassSeparator</td>
<td>Ensures that inner classes are referenced using &quot;$&quot; instead of &quot;,&quot; in class names</td>
</tr>
<tr>
<td>MissingId</td>
<td>Ensures that XML tags like &lt;fragment&gt; specify an id or tag attribute</td>
</tr>
</tbody>
</table>

Looks for hardcoded references to /sdcard

Your code should not reference the "/sdcard" path directly; instead use "Environment.getExternalStorageDirectory(), getPath()"
Hover-Over Documentation

void android.os.Bundle.putString(String key, String value)

g
public void putString (String key, String value)

Added in API level 1

Inserts a String value into the mapping of this Bundle, replacing any existing value for the given key. Either key or value may be null.

Parameters

key a String, or null

value a String, or null

// For more information, see the Fragments API guide at:

if (savedInstanceState == null) {
    // Create the detail fragment and add it to the activity
    // using a fragment transaction.
    Bundle arguments = new Bundle();
    arguments.putString(LessonDetailFragment.ARG_ITEM_ID, getIntent()
Auto-Complete
public class ListFragment
    extends Fragment

    java.lang.Object
    + android.app.Fragment
    + android.app.ListFragment

Class Overview

A fragment that displays a list of items by binding to a data source such as an array or Cursor, and exposes event handlers when the user selects an item.

ListFragment hosts a ListView object that can be bound to different data sources, typically either an array or a Cursor holding query results. Binding, screen layout, and row layout are discussed in the following sections.

Screen Layout

ListFragment has a default layout that consists of a single list view. However, if you desire, you can customize the fragment layout by returning your own view hierarchy from `onCreateView(LayoutInflater, ViewGroup, Bundle)`. To do this, your view hierarchy must contain a ListView object with the id "@android:id/list" (or list if it's in code).

Optionally, your view hierarchy can contain another view object of any type to display when the list view is empty. This "empty list" notifier must have an id "android:empty". Note that when an empty view is present, the list view will be hidden when there is no data to display.

The following code demonstrates an (ugly) custom list layout. It has a list with a green background, and an alternate red "no data" message.
Weaknesses

• Consistency and Standards
  – Resolutions, API levels, volatility, Java inconsistencies

• Error Prevention
  – Quirks such as null static variables, destruction and re-creation of activities on orientation change

• Recognition Rather than Recall
  – User must remember to override onCreate, onStart, onResume, onPause, onStop, onDestroy, onSaveInstanceState, onSaveInstanceState…
## Fragmentation

<table>
<thead>
<tr>
<th>Version</th>
<th>Codename</th>
<th>API</th>
<th>Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.6</td>
<td>Donut</td>
<td>4</td>
<td>0.2%</td>
</tr>
<tr>
<td>2.1</td>
<td>Eclair</td>
<td>7</td>
<td>2.4%</td>
</tr>
<tr>
<td>2.2</td>
<td>Froyo</td>
<td>8</td>
<td>9.0%</td>
</tr>
<tr>
<td>2.3 - 2.3.2</td>
<td>Gingerbread</td>
<td>9</td>
<td>0.2%</td>
</tr>
<tr>
<td>2.3.3 - 2.3.7</td>
<td></td>
<td>10</td>
<td>47.4%</td>
</tr>
<tr>
<td>3.1</td>
<td>Honeycomb</td>
<td>12</td>
<td>0.4%</td>
</tr>
<tr>
<td>3.2</td>
<td></td>
<td>13</td>
<td>1.1%</td>
</tr>
<tr>
<td>4.0.3 - 4.0.4</td>
<td>Ice Cream Sandwich</td>
<td>15</td>
<td>29.1%</td>
</tr>
<tr>
<td>4.1</td>
<td>Jelly Bean</td>
<td>16</td>
<td>9.0%</td>
</tr>
<tr>
<td>4.2</td>
<td></td>
<td>17</td>
<td>1.2%</td>
</tr>
</tbody>
</table>
20 Aayush Jain
Python language
05-830 Advanced User Interface Software

EVALUATE THE USABILITY OF A USER INTERFACE TOOL OR TOOLKIT

PYTHON LANGUAGE

- AAYUSH JAIN
Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.

Application Domains –
- Web and Internet Development
- Database Access
- Desktop GUI’s
- Network programming
- Software Development
- Game and 3D Graphics
Good Aspects

- Simple syntax : Readability
- Few statements
- Lambda function
- Large standard library
- Easy problem decomposition
- Consistent user interfaces
- Code re-usability
Good Aspects

Simple Syntax

```python
print "05-830, Advanced User Interface Software"
import math
x = "Prof Brad Myers"
print x
print 2+3
```
Good Aspects

Large Libraries

```
1. print "My First Python Program" #Python 2.7
2. print ("My First Python Program") #Python 3
3. and - and keyword
4. assert - assert keyword
5. break - break keyword
6. class - Class definition (simple)
7. classes - Class definition (subclass)
8. cod - encoding comment
9. codu8 - encoding comment: utf-8
10. continue - continue keyword
11. def - Method definition (global)
12. defc - Method definition (class)
```

Console

My First Python Program
My First Python Program
My First Python Program
Good Aspects

Fewer Statements

- Python – print “Hello World”
- Java –

```java
public class HelloWorld {
    public static void main (String [] args) {
        System.out.println(“Hello World”);
    }
}
```
Bad Aspects

- Compilation: Too Slow
- Whitespaces as Tokens
- Un-riched documentation
- Variable types undeclared
- Data protection
- OOP inconsistent
- Threading issues
- Only single line lambda function
Bad Aspects

- OOP inconsistent
  Ruby code: obj.method
  Python code: len(obj)
- Data protection call by:
  ```_ method_ _```
- Variable types undeclared
  ```
  X= 42 #Error: variable type not declared
  Var X=2 #Ok: Declared.
  X=34 #ok: declared
  VarX=2#Error: Redeclaration to same variable type not allowed.
  Let y #Error: Declare value to y
  Let y = 5 #Ok: Declares value to read only y.
  ```
Thank You!
21 Salvatore Andolina
MatLab
USABILITY EVALUATION OF THE MATLAB LANGUAGE

Salvatore Andolina – Thursday, January 24, 2013
Matlab

- Matrix Laboratory
  - Highly optimized for matrix manipulations

- Allow prototyping of complex algorithms in few lines of code

- Toolboxes
  - Statistics
  - Data visualization
  - Optimization
  - Image processing
  - Speech processing
  - ...

Examples with vectors

- `>> v = 1:10`  
  \[ v = 1 2 3 ... 9 10 \]

- `>> sum(v)`  
  \[ \text{ans} = 55; \]

- `>> mean(v)`  
  \[ \text{ans} = 5.5000; \]
Examples with matrixes

- `>> A = [1 2 3; 4 5 6; 7 8 9]          A =
  1 2 3
  4 5 6
  7 8 9`

- `>> sum(A)                          ans =
  12 15 18`

- `>> mean(A)                         ans =
  4  6  5`

- `>> sum(A,2)                        ans =
  6
  15
  24`

- `>> mean(A,2)                       ans =
  2
  5
  8`

- `>> sum(A(:))                       ans =
  45`

- `>> mean(A(:))                      ans =
  5`

**Consistency**: operators work on columns by default, on rows with a 2 as second argument. With 3D matrices, we can sum along the 3rd dimension.

**Error-proneness**: weakly-typed

**Domain correspondence**: `A+2` sums 2 to all the elements of `A`, like in real world when adding a constant to a matrix
Scenario: Image processing

- Load an image and show it
  - >> im=imread('myimage.jpg');
  - >> imshow(im);

- Convert to grayscale
  - >> im=double(im);
  - >> im=mean(im,3);
  - >> imshow(uint8(im))
Scenario: Image processing

- Load an image and show it
  
  ```matlab
  >> im = imread('myimage.jpg);
  >> imshow(im);
  im is uint8
  ```

- Convert to grayscale
  
  ```matlab
  >> im = double(im);
  >> im = mean(im, 3);
  >> imshow(uint8(im))
  ```

Progressive evaluation

Work-step unit

Viscosity

Error-proneness: imshow behavior
- If the argument is uint8 it assumes values in the range 0...255
- If the argument is double it assumes values in the range 0...1
22 Jason Tsay
Sinatra - framework for Ruby
Web Application Framework and Domain-Specific Language for Ruby

SINATRA

Jason Tsay

*http://www.sinatrarb.com/
Sinatra

• Sinatra is a minimal web framework in Ruby
• Domain-Specific Language (DSL)
  – expresses how a web app handles HTTP actions
  – RESTful
• Used by: LinkedIn, GitHub, Apple, RedHat, BBC*

*http://www.sinatrarb.com/wild.html
Features done right

• Very easy to get up and going
• Routes
• Interoperability
Hello World!

```
# myapp.rb
require 'sinatra'

get '/' do
  'Hello world!'
end
```

*http://www.sinatrarb.com/intro*
Features done right

• Very easy to get up and going
  – Single-file web application
  – Work-Step Unit is effective

• Routes
  – Way to express how a web application handles certain types of HTTP actions

• Interoperability
Routes

```
get '/' do
  .. show something ..
end

post '/' do
  .. create something ..
end

put '/' do
  .. replace something ..
end

patch '/' do
  .. modify something ..
end

delete '/' do
  .. annihilate something ..
end

options '/' do
  .. appease something ..
end
```

*http://www.sinatrarb.com/intro*
Features done right

• Very easy to get up and going
• Routes
  – Way to express how web application handles certain types of HTTP actions
  – Good Domain Correspondence
• Interoperability
  – Applications very flexible
  – Plugins (Gems), APIs, even frameworks
    • Like Padrino
Features done less right

• Testing annoyance
  – Need to restart server to test any change

• Heavy Dependency on Rack
  – Sinatra is built on top of the Rack web server interface

• “Classic” and “Modular” styles
require 'my_sinatra_app'
require 'test/unit'
require 'rack/test'

class MyAppTest < Test::Unit::TestCase
  include Rack::Test::Methods

  def app
    Sinatra::Application
  end

  def test_my_default
    get '/'
    assert_equal 'Hello World!', last_response.body
  end

  def test_with_params
    get '/meet', :name => 'Frank'
    assert_equal 'Hello Frank!', last_response.body
  end

  def test_with_rack_env
    get '/', {}, 'HTTP_USER_AGENT' => 'Songbird'
    assert_equal "You're using Songbird!", last_response.body
  end
end

*http://www.sinatrarb.com/intro*
Features done less right

- Testing annoyance
- Heavy Dependency on Rack
  - Sinatra is built on top of the Rack web server interface
  - To be proficient at Sinatra, must also be proficient at Rack
- “Classic” and “Modular” styles
  - 2 styles of web apps supported
“Classic” vs “Modular”

“classic”

```ruby
# myapp.rb
require 'sinatra'

def hello
  put '/'
  'Hello world!
end
```

“modular”

```ruby
require 'sinatra/base'

class MyApp < Sinatra::Base
  set :sessions, true
  set :foo, 'bar'

  get '/'
  put 'Hello world!
end
```

*http://www.sinatrarb.com/intro*
Features done less right

- Testing annoyance
- Heavy Dependency on Rack
- “Classic” and “Modular” styles
  - 2 styles of web apps supported
  - Documentation assumes “classic”
    - May lead to Premature Commitment
23 Jenny Kim
VIM
VIM

How can simple be powerful?
Vim?

- A highly configurable text editor
- Distributed with most LINUX/UNIX systems
- Still loved by so many programmers
Evaluation of Vim

Co-working
Efficiency
Consistency
Working framework
Learning style
Help and documentation
Aesthetic and minimalist...
Recognition rather than...
User control and freedom
Visibility of system status
Aesthetic and minimalist design

- **So simple.** No distractions.
- Very cores of coding.
Efficiency - Speedy

- No loading time
- No need to grep mouse
- You can do it in a second
- You can go wherever. Just connect to server.
User control and freedom

• Customize whatever I want
• Variety kinds of plugins
Recognition rather than recall

• Need study - Study first to know where it is
• Too many short keys to memorize

**vi/vim graphical cheat sheet**

- **Esc**
  - normal mode
  - ^ external file
  - ! select macro
  - @ play macro
  - # prev line
  - $ col
  - % goto match
  - ^ "soft" bol
  - & repeat
  - $ next ident
  - ( begin sentence
  - ) end sentence
  - \"soft\" bol down
  - + next line
  - \- prev line

- **Q**
  - ex mode
  - W next word
  - E end word
  - R replace mode
  - T back till
  - Y yank line
  - U undo line
  - I insert at bol
  - O open above
  - P paste below

- **A**
  - append at col
  - S subst line
  - D delete to col
  - f find char
  - g go to line
  - y yank line
  - u undo
  - i insert mode
  - o open below

- **Z**
  - quit
  - X x back space
  - C change to col
  - V visual line
  - B prev word
  - N prev line
  - N next (find)
  - M screen mid
  - ? find (rev.)

**Main command line commands ('ex'):**
- :w (save), :q (quit), :q! (quit w/o saving)
- :e (open file)
- :%x/y/g (replace 'x' by 'y' filewide)
- :h (help in vim), :new (new file in vim),

**Other important commands:**
- CTRL-R: redo (vim)
- CTRL-F/E: page up/down,
- CTRL-E/-Y: scroll line up/down,
- CTRL-V: block-visual mode (vim only)

**Visual mode:**
- Move around and type operator to act on selected region (vim only)

**Notes:**
- (1) use 'x before a yank/paste/del command to use that register ('clipboard') (x=a..z,*).
  
- (2) type in a number before any action to repeat it that number of times (e.g.: 2p, d2w, 5i, 4j).

- (3) duplicate operator to act on current line (e.g.: dd = delete line, >> = indent line)
- (4) ZZ to save & quit, ZQ to quit w/o saving
- (5) zt: scroll cursor to top.
- zb: bottom, zz: center
- (6) gg: top of file (vim only).
- gf: open file under cursor (vim only)

For a graphical vi/vim tutorial & more tips, go to [www.viemu.com](http://www.viemu.com) - home of ViEmu, vi/vim emulation for Microsoft Visual Studio
Consistency

- Totally different way of control
  - Keyboard only!
- Short keys act differently
  - Ex) Paste short key is ‘P’. Not Ctrl+V.
Overall

- Simple, but powerful
  - Minimal design, Efficient, speedy