Java FX and Java SE 6 Update N

Raghavan “Rags” N. Srinivas
CTO, Technology Evangelism
Sun Microsystems Inc.

Agenda

- State of Java
- Java FX
- Java SE 6 update N
- Future

How Much Java Technology Is Out There?

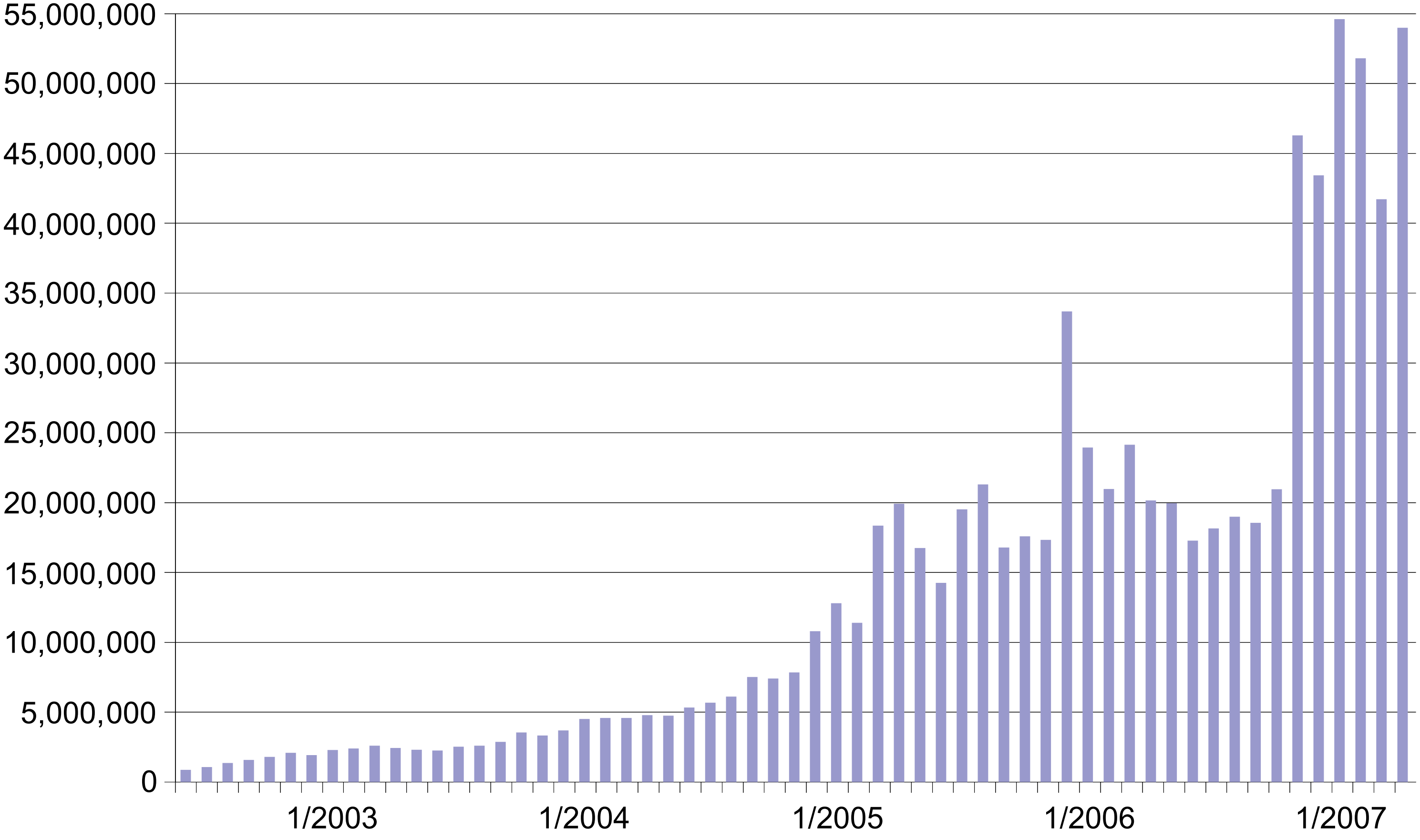
- >91% of all PCs run Java platform*
- ~77% of all Java technology-enabled PCs run Sun’s Java Platform, Standard Edition (Java SE platform)**
- Distribution through PC OEMs
  - > Nine of the top ten PC OEMs ship the Sun JRE software
  - > Representing >60% of all shipped PCs
  - > 58 white box vendors have signed JRE software redistribution agreements
- Download/installs
  - > ~44m installations / month for the last six months on Windows
  - > >50M in Jan, Feb, April, 2007

* Omniture, April 2007

**Mapsolute/Map24.com, April 2007

Completed Java SE Platform Downloads

Windows numbers only

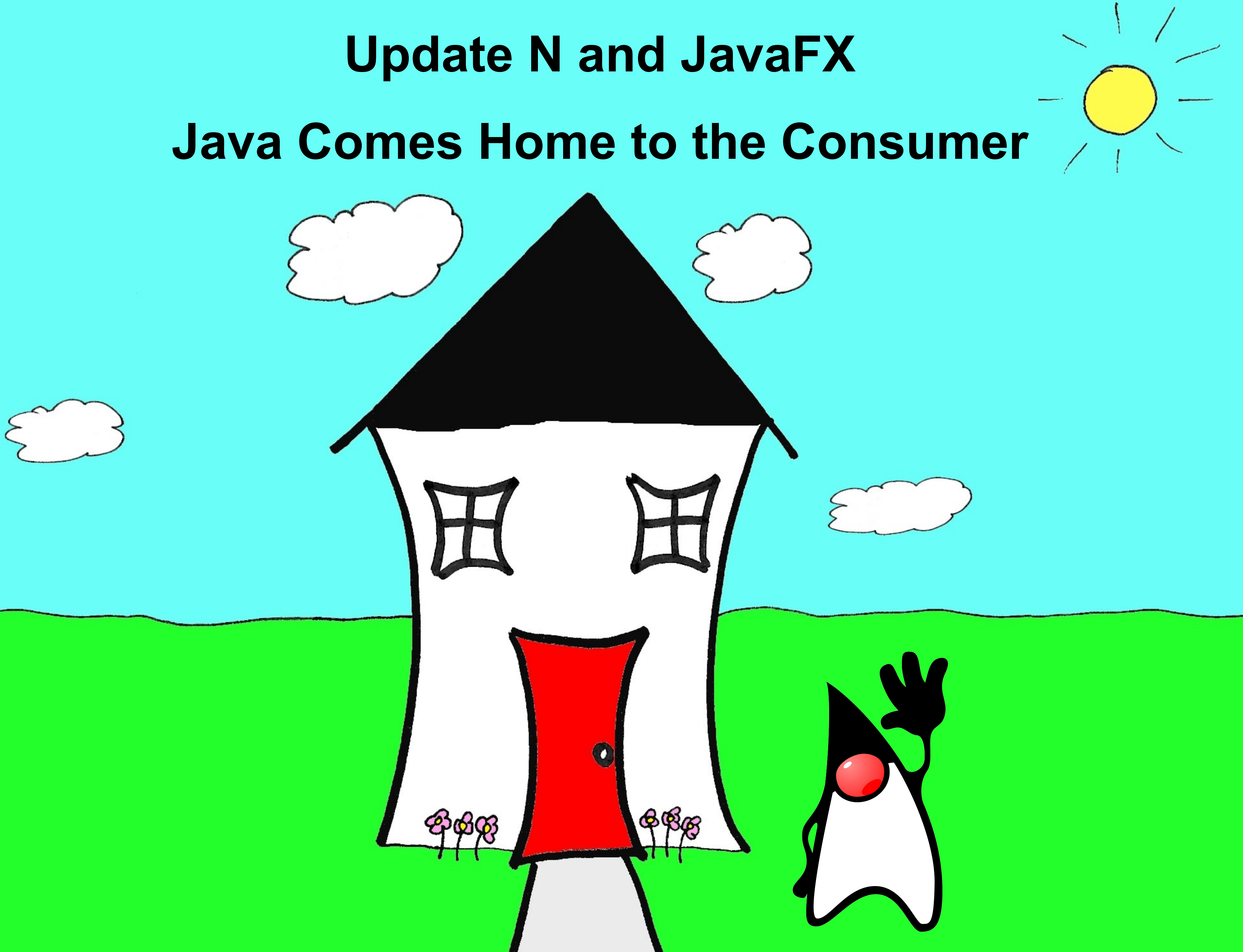


Agenda

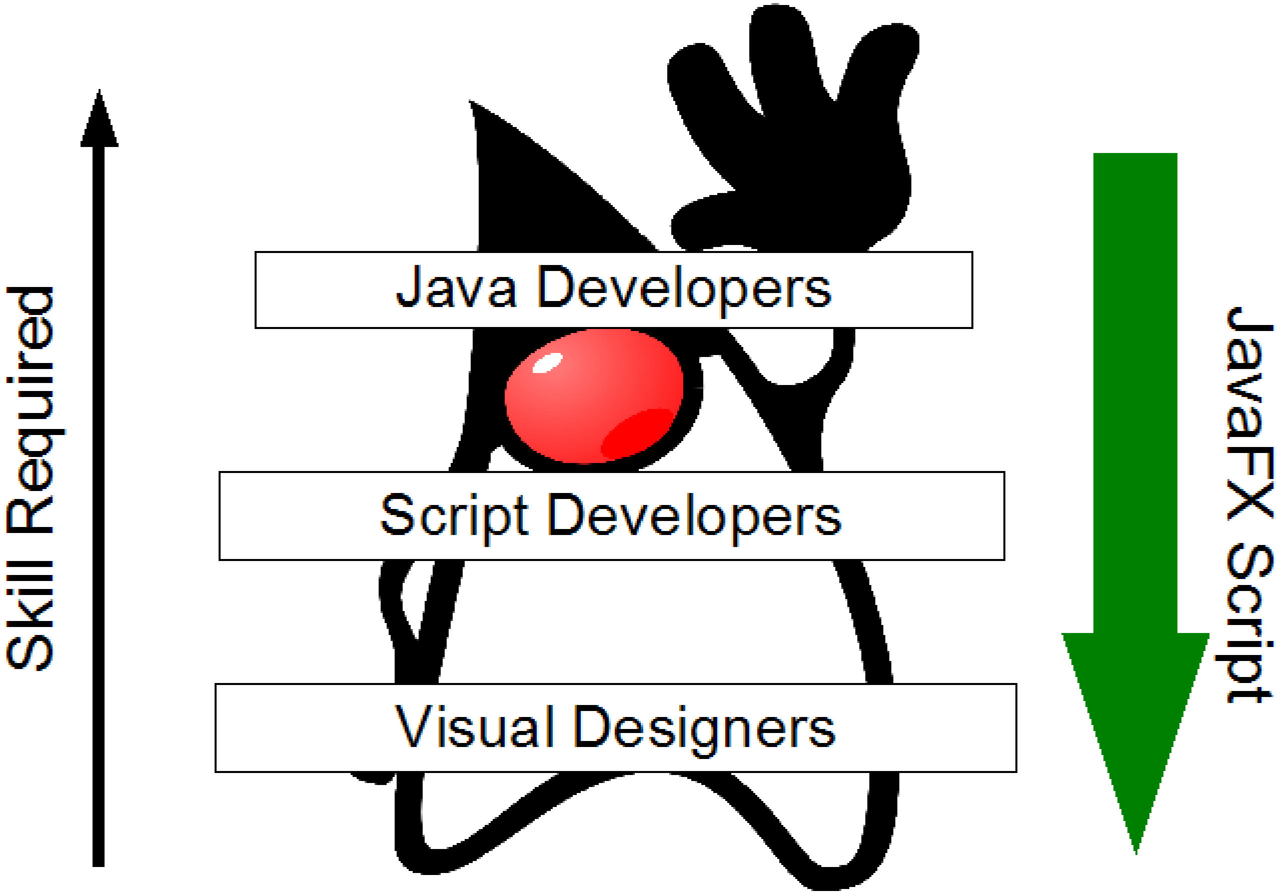
- State of Java
- Java FX
- Java SE 6 update N
- Future

Update N and JavaFX

Java Comes Home to the Consumer



JavaFX



Skill Required

Java Developers

Script Developers

Visual Designers

JavaFX Script

JavaFX Features

- Java FX Script
- Scene graph
- Media
- HTML
- Multiple Device

JavaFX Script

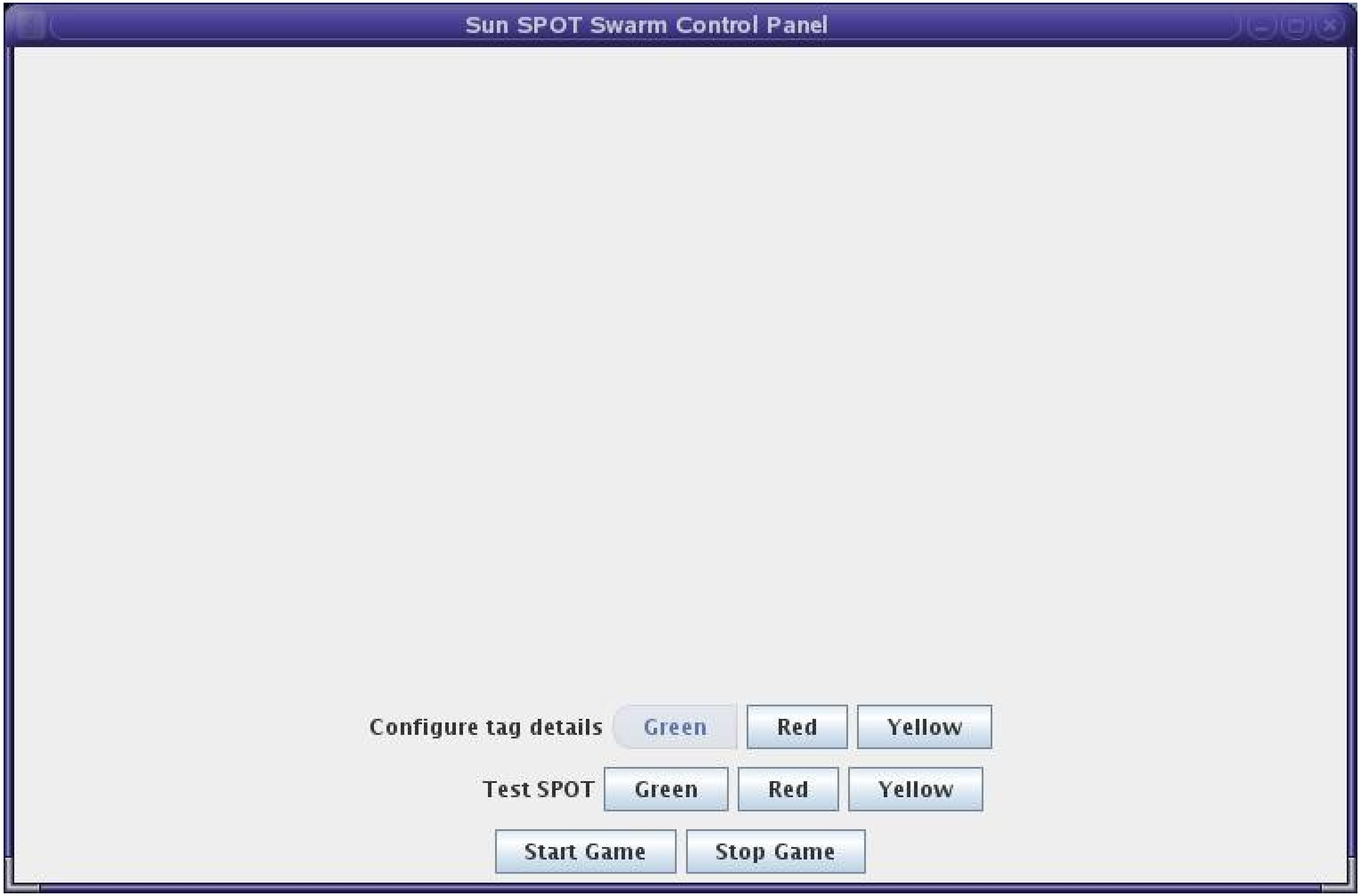
JavaFX Script

- Language
  - Simple data binding
  - Fast prototyping
  - Declarative GUIs and graphics
  - Easily programmed animations
- Compiler
  - Interpreter (now): okay for demos
  - Compiler (soon): necessary for real applications

The Java GUI Quagmire

- Java Swing and 2D APIs are very powerful and yet
  - Why does it take a long time to write GUI programs?
  - How can we avoid the “Ugly Java technology GUI” stereotype?
  - Why do Flash programs look different than Java programs?
  - Why does it seem easier to write web apps than Swing programs?
  - How can I avoid having an enormous mass of listener patterns?
    - Fundamental problem with event-oriented programming model

A Basic Java GUI: Not Very Pretty



Java FX and Java 2D API

- Java 2D API doesn't have compositional behavior
  - The barrier to entry for many Java code programmers is too high
- Declarative GUI composition is the main goal for JavaFX
  - In addition to Swing Components, a language that includes first-class elements which can be composed together into higher-level components.
  - A language to allow declarative expression of this composition

What is JavaFX Script?

- Formerly known as F3 (Form Follows Function)
- http://blogs.sun.com/chrisoliver/category/JavaFX
- Java Language that is “scripting friendly”
  - Object-oriented
  - Static typing + type inference
  - Declarative Syntax
  - Automatic Data Binding
  - Mixed functional/procedural evaluation model
- Extensive widget library encompassing Swing components and Java2D objects
- Development Tools including IDE Plugins

JavaFX: Not Yet Another Scripting Language

- JavaFX is for GUIs and not a general scripting langauge (yet)
  - > Only the Java platform can marry client- and browser-based technologies with RIAs to run across virtually any device
  - > Designed to optimize the creative process of building rich and compelling UIs leveraging Java Swing, Java 2D and Java 3D for developers and content authors
  - > Structure of JavaFX code closely matches the actual layout of the GUI, making it more readable and maintainable

Declarative and Composition Syntax

```
frame = new Jframe();
mb = new JMenuBar();
JMenuItem gi,mi;

fm = new JMenu("Foo");
fm.add(gi = new JMenuItem("Goo"));
gi.setMnemonic('G');
fm.addSeparator();
fm.add(mi = new JMenuItem("Moo"));
mi.setMnemonic('M');
frame.add(fm);
```

```
Frame {
  menubar: MenuBar {
    menus: Menu {
        text: "Foo" mnemonic: F
        items: [
          MenuItem { text: "Moo" mnemonic: M },
          MenuSeparator,
          MenuItem { text: "Goo" mnemonic: G }
        ]
    }
  }
  visible: true
}
```

JavaFX – More Comparisons

In JavaFX Script

```
picks.opacity = [0, .01 .. 1 ] dur 1000 linear
```

In Java (SwingLabs Timing Framework)

```
public void class Guitar {
   private GuitarPick pick = ...;
   public Guitar() {
      pick.setOpacity(.5f);
      Animator a =
           PropertySetter.createAnimator(
               300, pick, "opacity", .5f, 1.0f);
      MouseTrigger.addTrigger(pick, a,
               MouseTriggerEvent.ENTER, true);
   }
}
```

In Java

Main guitar class

```
guitarAnimationThread = new StringOpThread();
......
......
if (guitarAnimationThread != null) {
    guitarAnimationThread.run();
}
.........
```

StringOpThread class

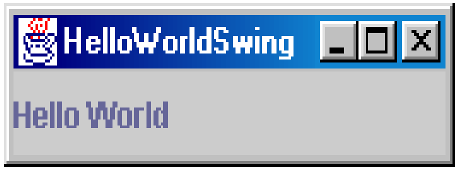
```
        ...........
    public void run() {
             opacityBegin = 0.01;
        opacityEnd = 1.0;
             opacityIncreStep = 0.02;
             opacitySleep = 2;

         for(currOpacity = opacityBegin; \
                 currOpacity < opacityEnd;    \
                    currOpacity+=opacityIncreStep) {
            setPickOpacity();
            repaint();
            try {
               thread.sleep(opacitySleep);
            } catch (InterruptedException e) { }
```

Benefits of Static Typing

- High-quality, compile-time error reporting
- High-quality IDE support
  - Code-completion
  - Searching
  - Refactoring
- Efficient compilation

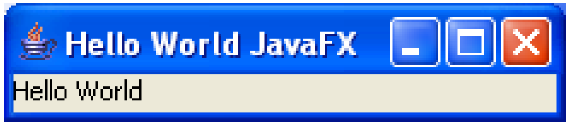
Hello World with Swing

```
import javax.swing.*;
public class HelloWorldSwing {
    public static void main(String[] args) {
        JFrame frame = new JFrame("HelloWorldSwing");
        final JLabel label = new JLabel("Hello World");
        frame.getContentPane().add(label);


  frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.pack();
        frame.setVisible(true);
    }
}
```



Hello World with JavaFX

```
import javafx.ui.*;
    Frame {
        title: "Hello World JavaFX"
        width: 200
        height: 50
        content: Label {
            text: "Hello World"
        }
        visible: true
    }
```



Hello World

Declarative Syntax

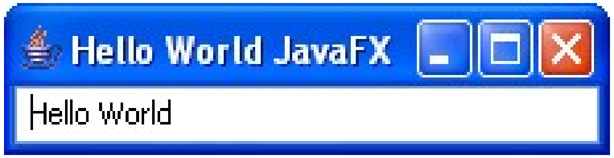
- Tell me what you want. Not How.
- Common in Web applications
- For example, ease of composing styled text
  - HTML vs. JTextPane
- HTML Table using JSTL versus JTable
- JavaFX brings that same ease of use to Swing and Java 2D API programming

Data Binding in JavaFX

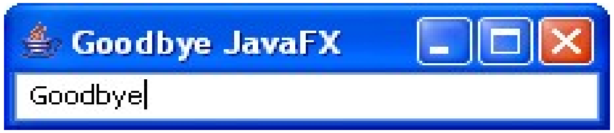
- Cause and Effect—Responding to change
- The JavaFX bind operator—Allows dynamic content to be expressed declaratively
- Dependency-based evaluation of any expression
- Automated by the system—Rather than manually wired by the programmer
- You just declare dependencies and the JavaFX runtime takes care of performing updates when things change
- Eliminates listener patterns

Example: Dynamic Behavior

```
class HelloWorldModel {
     attribute saying: String;
}
var model = HelloWorldModel {
     saying: "Hello World"
};
var win = Frame {
      title: bind "{model.saying} JavaFX"
      width: 200
      content: TextField {
           value: bind model.saying
      }
      visible: true
};
```



Hello World



Goodbye

Understanding filters

```
import javafx.ui.filter.*;
Canvas {
    content: Circle {
          cx: 80
          cy: 80
          radius: bind [50..0] dur 1000
          fill: red
          stroke: purple
          strokeWidth: 2
          filter: [Noise, ShapeBurst]
    }
}
```

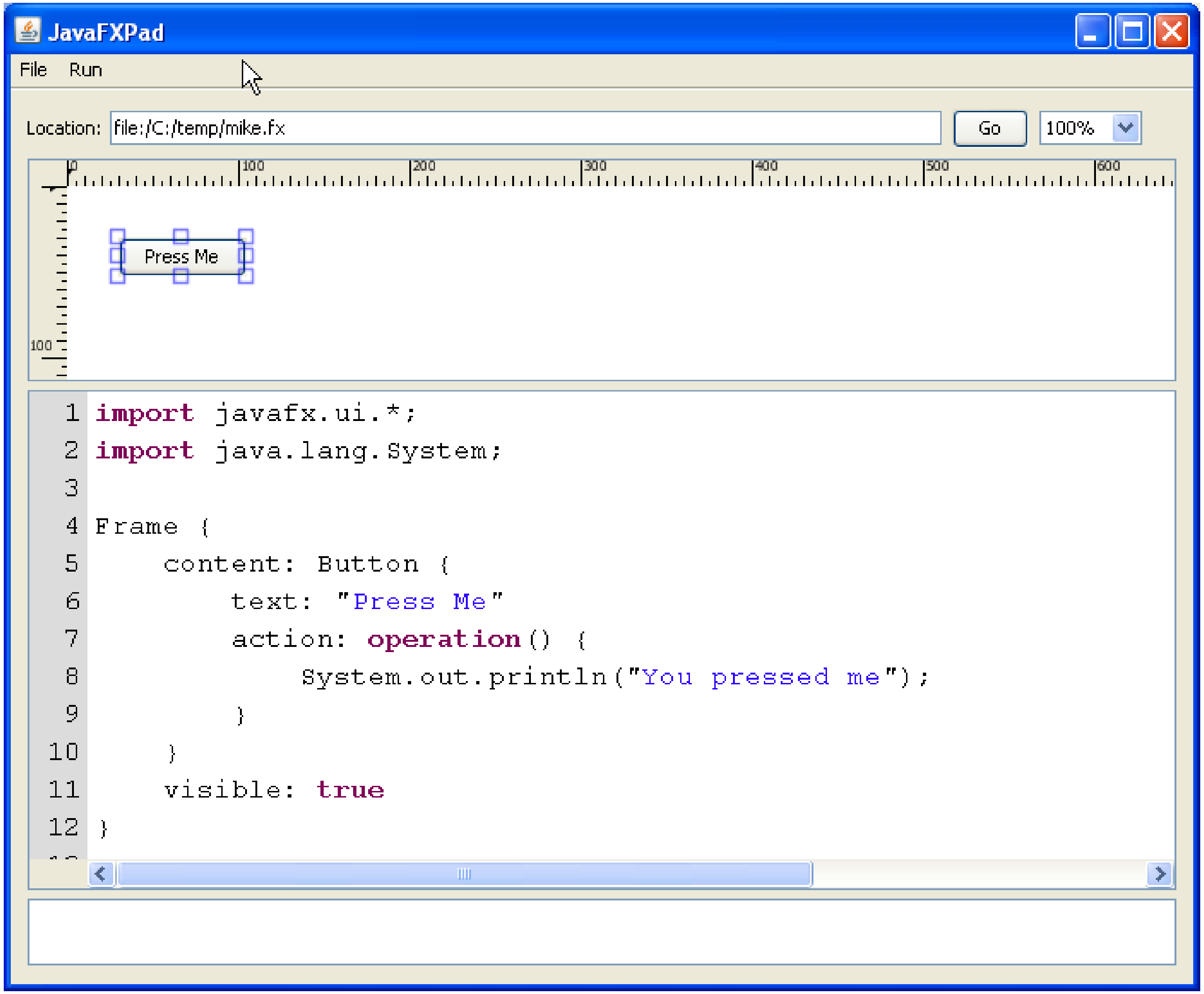
Example : Hello world

```
$ cat HelloWorld.fx
import java.lang.System;
System.out.println("Hello, world");

$ javafx HelloWorld.fx
compile thread: Thread[AWT-EventQueue-0,6,main]
compile 0.017
Hello, world

$ javafxc HelloWorld.fx
$ java -cp javafxc.jar;javafxrt.jar;. HelloWorld
```

Introduction and Getting Started

- Invoke JavaFXPad
- JavaFXPad
  - Running FX Programs manually
  - Loading and Saving FX Programs
  - Developing FX Programs
- Simple JavaFX editor

JavaFXPad



Location: file:/C:/temp/mike.fx

100%

```
import javafx.ui.*;
import java.lang.System;

Frame {
    content: Button {
        text: "Press Me"
        action: operation() {
            System.out.println("You pressed me");
        }
    }
    visible: true
}
```

NetBeans Plugins

- Understand the Edit-Compile-Debug lifecycle for JavaFX programs on NetBeans
- Run JavaFX programs in NetBeans easily with the JavaFX shell
- NetBeans plugins installation instructions:
  - https://openjfx.dev.java.net/javafx-nb55-plugin-install.html
  - https://openjfx.dev.java.net/javafx-nb60-plugin-install.html
- Getting started with JavaFX:
  - https://openjfx.dev.java.net/Getting_Started_With_JavaFX.html

Java and JavaFX

```
ScriptEngineManager manager = new
  ScriptEngineManager();

ScriptEngine engine =
  manager.getEngineByExtension("fx");


engine.put("now:java.util.Date", new Date());


InputStreamReader reader = new
  InputStreamReader(FxScriptLauncher2.class

    .getResourceAsStream("HelloWorld.fx"));

engine.eval(reader);
```

Java and JavaFX

```
import javafx.ui.*;
import javafx.ui.canvas.*;

Frame {
    title: "Hello World!"
    content: Label {
        text: "Hello World with Java
  date: {now:<<java.util.Date>>}"
    }
    visible: true

}
```

Mobile

JavaFX Mobile

- Java FX Script isn't just for desktop
  - Java FX Mobile platform
- Development skills that carry between platforms
  - Not *mobile developers*
  - Developers with skills that are *mobile*
- Easier authoring of applications for multiple devices

Agenda

- State of Java
- Java FX
- Java SE 6 update N
- Future

Consumer JRE

~~Consumer JRE~~

Project Hamburg

~~Consumer JRE~~

~~Project Hamburg~~

Java SE 6, Update X

~~Consumer JRE~~

~~Project Hamburg~~

~~Java SE 6, Update X~~

Java SE 6, Update N

~~Consumer JRE~~

~~Project Hamburg~~

~~Java SE 6, Update X~~

~~Java SE 6, Update N~~

Java SE 6, Update 10

Problems to fix

- Startup Time
- Install Time
- JRE Detection
- Applet support

Quickstarter

- “Coldstart” vs. “Warmstart”
- Root problem:
  - Large files + Disk access speed
- Solution: QuickStarter
  - Pre-warm the disk cache
- Note: QuickStarter != running VM
  - Smaller footprint, more targeted disk pages

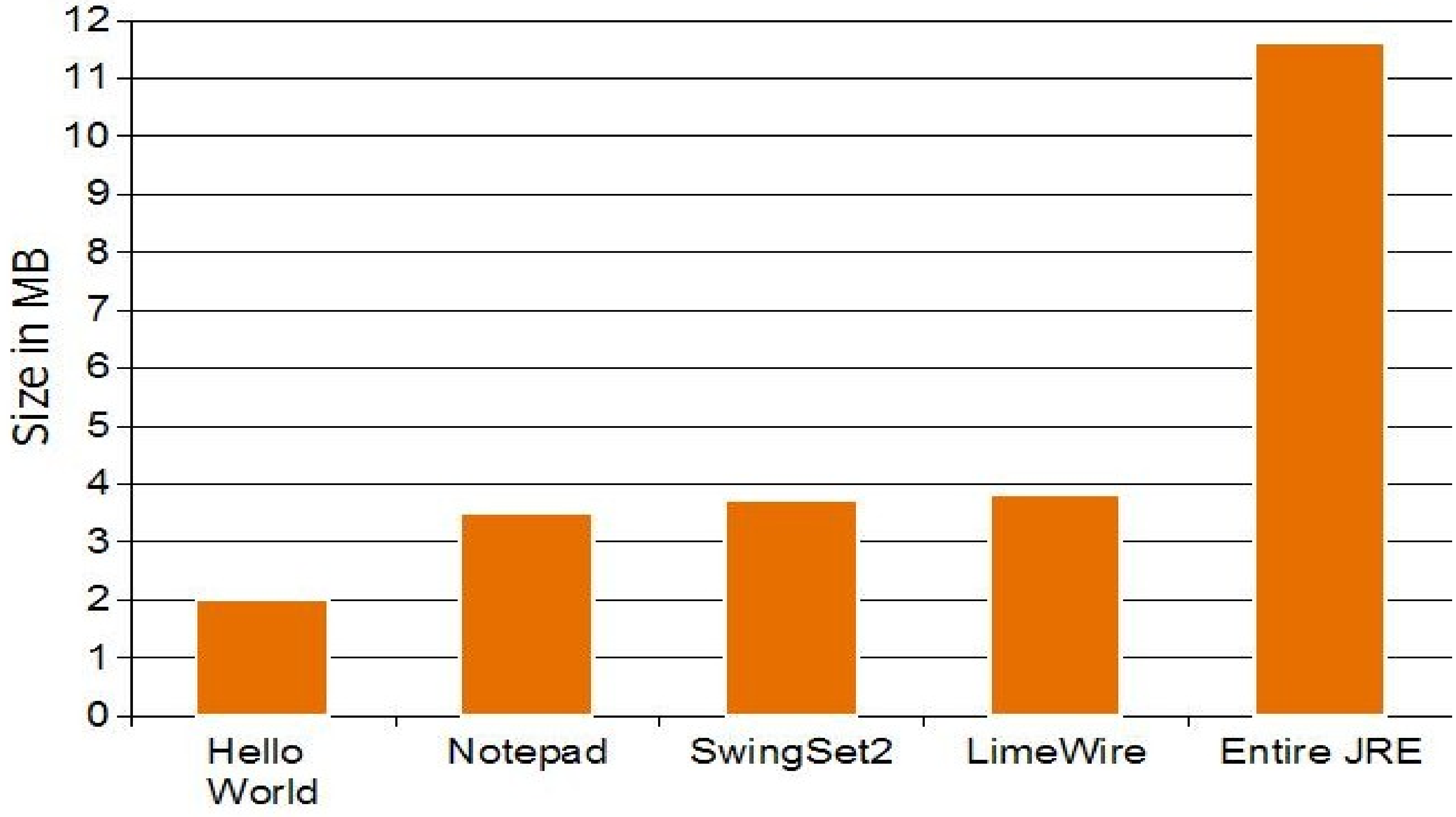
Problems to fix

- Startup Time
- Install Time
- JRE Detection
- Applet support

Install Time

- Java's not small
  - J2SE 5.0: 7.1 MB
  - Java SE 6: 10+ MB
  - rt.jar: 40+ MB on disk
- Lots of bits being moved around
  - Download, Unzip, Unpack200, Copying
- Solution: Java Kernel
  - Download only core dependencies first
  - Launch application
  - Download and install in the background

Look Mom! We Shrunk the JRE



Problems to fix

- Startup Time
- Install Time
- JRE Detection
- Applet support

Deployment Tookit

- Detecting Java from a web page is hard
  - Does the user have Java?
  - What version?
  - How to get user to install and return?
- Result: Most applets use old (1.1) APIs
  - Allows them to run compatibly on old versions
    - e.g., MS VM
- Solution: Deployment Toolkit
  - JavaScript hosted at Sun for general detection
  - Plugins (ActiveX and Mozilla) for more specific detection
  - Detect, start installation if necessary, launch

Problems to fix

- Startup Time
- Install Time
- JRE Detection
- Applet support

New Plugin Architecture

- Out-of-process
  - Small core process for overall plugin
  - Separate process(es) for applets
  - Faster startup
  - Better signed applet behavior on Vista
- Increased robustness
- Better JavaScript/Java communication support
- Different JVM options per-applet
- Different applets can run different JRE versions
- Built for IE 6&7 and Firefox 3

New Plugin Architecture

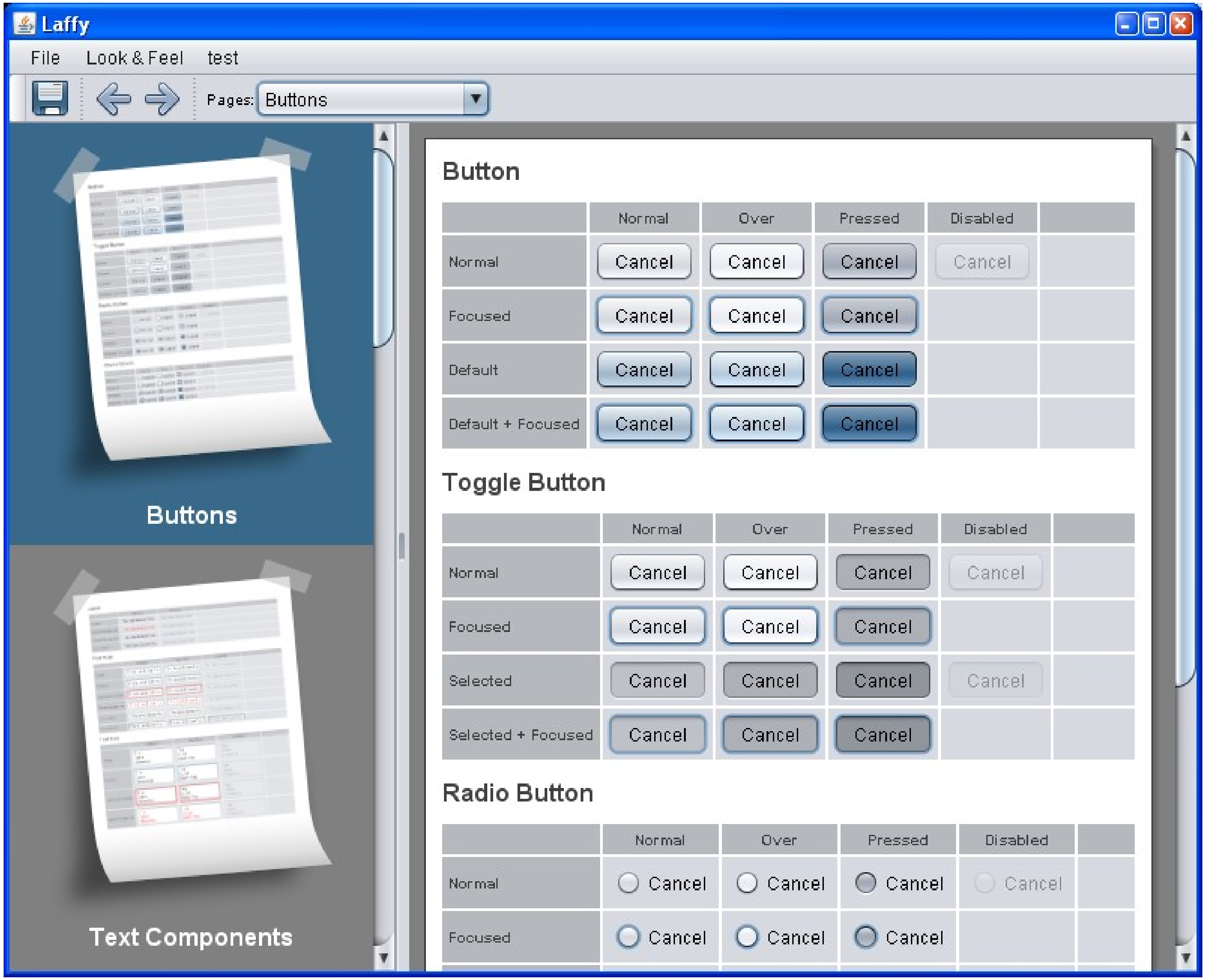
- Please test it and let us know!
  - https://jdk6.dev.java.net/6uNea.html
- Windows:
  - Java Control Panel, “Advanced” tab
  - “Java Plug-In” node
  - Select the checkbox:
    - “Enable the next-generation Java Plug-in”
- Linux/Solaris
  - Symlink to lib/{i386,sparc}/libnpjp2.so from firefox/plugins or ~/.mozilla/plugins directory
  - Remove any old symlinks to libjavaplugin_oji.so

But Wait, There’s More!

Graphics Goodies

- Better Windows graphics performance
  - > GPU-accelerated, using Direct3D
  - > On by default
- Nimbus:
  - > Cross-platform look & feel
  - > More modern than current Metal look & feel
  - > Opt-in: Not the default look & feel
    - > Preserves backwards compatibility
  - > Design tool for creating new skins

Nimbus



Button

	Normal	Over	Pressed	Disabled	
Normal	Cancel	Cancel	Cancel	Cancel	
Focused	Cancel	Cancel	Cancel		
Default	Cancel	Cancel	Cancel		
Default + Focused	Cancel	Cancel	Cancel		

Toggle Button

	Normal	Over	Pressed	Disabled	
Normal	Cancel	Cancel	Cancel	Cancel	
Focused	Cancel	Cancel	Cancel		
Selected	Cancel	Cancel	Cancel	Cancel	
Selected + Focused	Cancel	Cancel	Cancel		

Radio Button

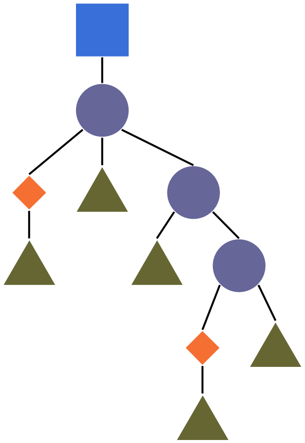
	Normal	Over	Pressed	Disabled	
Normal	○ Cancel	○ Cancel	● Cancel	○ Cancel	
Focused	○ Cancel	○ Cancel	● Cancel		

Update whe[N]?

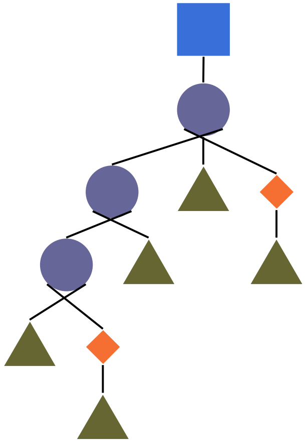
- Early Access available now
  - https://jdk6.dev.java.net/6uNea.html
  - Everything seen here except:
    - Java Kernel will be available by Beta
- Beta planned for February
- GA planned for early second half of 2008

Agenda

- State of Java
- Java FX
- Java 6 update N
- Future

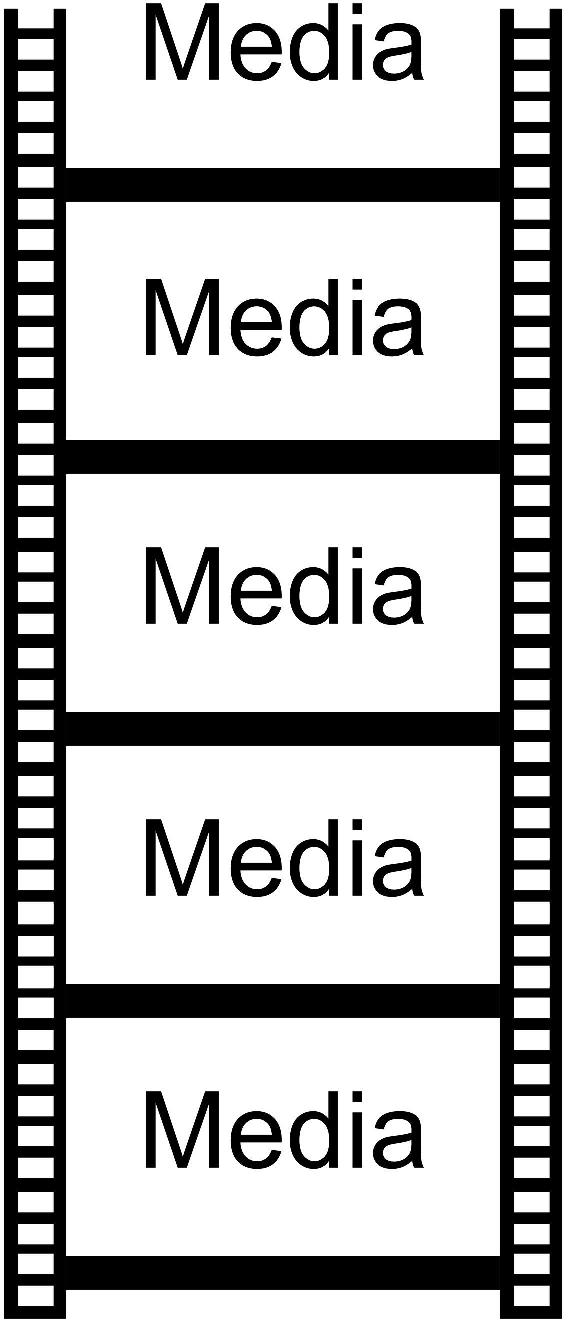


Scene Graph



Scene Graph

- “Retained mode” graphics
  - > Versus immediate-mode 2D API
- More declarative model for:
  - > Graphics
  - > GUI
  - > Media
  - > Animation
- Used by FX Script
  - > Also usable from Java



Media
Media
Media
Media
Media

Media

- Java Technology needs a standard media solution
- Media components
  - Swing components for video/audio playback
  - Support for native formats through native players
    - Windows Media Player
    - Flash
    - DirectShow
    - QuickTime
  - Cross-platform codec for lightweight integration

Client: old todo List

- □ Performance
- □ Data Binding
- □ Swing Framework
- □ Ease of Development
- □ More Hardware Acceleration
- □ World Peace
- □ Video
- □ Shaped windows
- □ Mixed Heavyweight/Lightweight
- □ Animation
- □ Rasterization Fixes
- □ Drag and Drop Work
- □ New Look + Feel

Client: Recent todo List

- □ PERFORMANCE
- □ DATA BINDING
- □ SWING FRAMEWORK
- □ EASE OF DEVELOPMENT
- □ MORE HARDWARE ACCELERATION
- □ WORLD PEACE

□ VIDEO

- □ SHAPED WINDOWS
- □ MIXED HEAVYWEIGHT/LIGHTWEIGHT
- □ ANIMATION
- □ RASTERIZATION FIXES
- □ DRAG AND DROP WORK
- □ NEW LOOK + FEEL

<HTML>

HTML

- Swing HTML support intentionally basic
- Need new component for “street HTML” rendering
- Will allow integrated Swing, graphics, and HTML applications

3D

- You can do 3D with Java platform now
  - Applications, applets, and web start
  - Seamless deployment of JOGL libraries through plugin
- Hardware acceleration, cross-platform, modern graphics standard…
- 3D API integrated into the JDK software in the future
  - Nothing concrete yet

Animation

- Modern desktops are getting more animated
- Swing' native look & feel supports basic control animation
  - > Possible to do much more…but very manual
- We need to make animation much easier
- Need built-in support for
  - > Better timing facilities
  - > Animations and effects

Scene Graph

- Available!
- Open source project now available
  - http://scenegraph.dev.java.net
- Early early (early) access form
  - Functional
  - APIs not final

Proposed JSRs in Java SE 7
(subject to change)

- JSR 277 Java Module System
- JSR 294 Improved Modularity Support in the Java Language
- JSR 295 Beans Binding
- JSR 303 Beans Validation
- JSR 296 Swing Application Framework
- JSR 203 NIO2
- JSR 220 Java Persistence APIs
- JSR 255 JMX 2.0
- JSR 262 Web Services Connector for JMX
- JSR 260 Javadoc Technology Update
- *JSR(s) TBD Java Language changes*

More tools Support… (NetBeans™ Software)

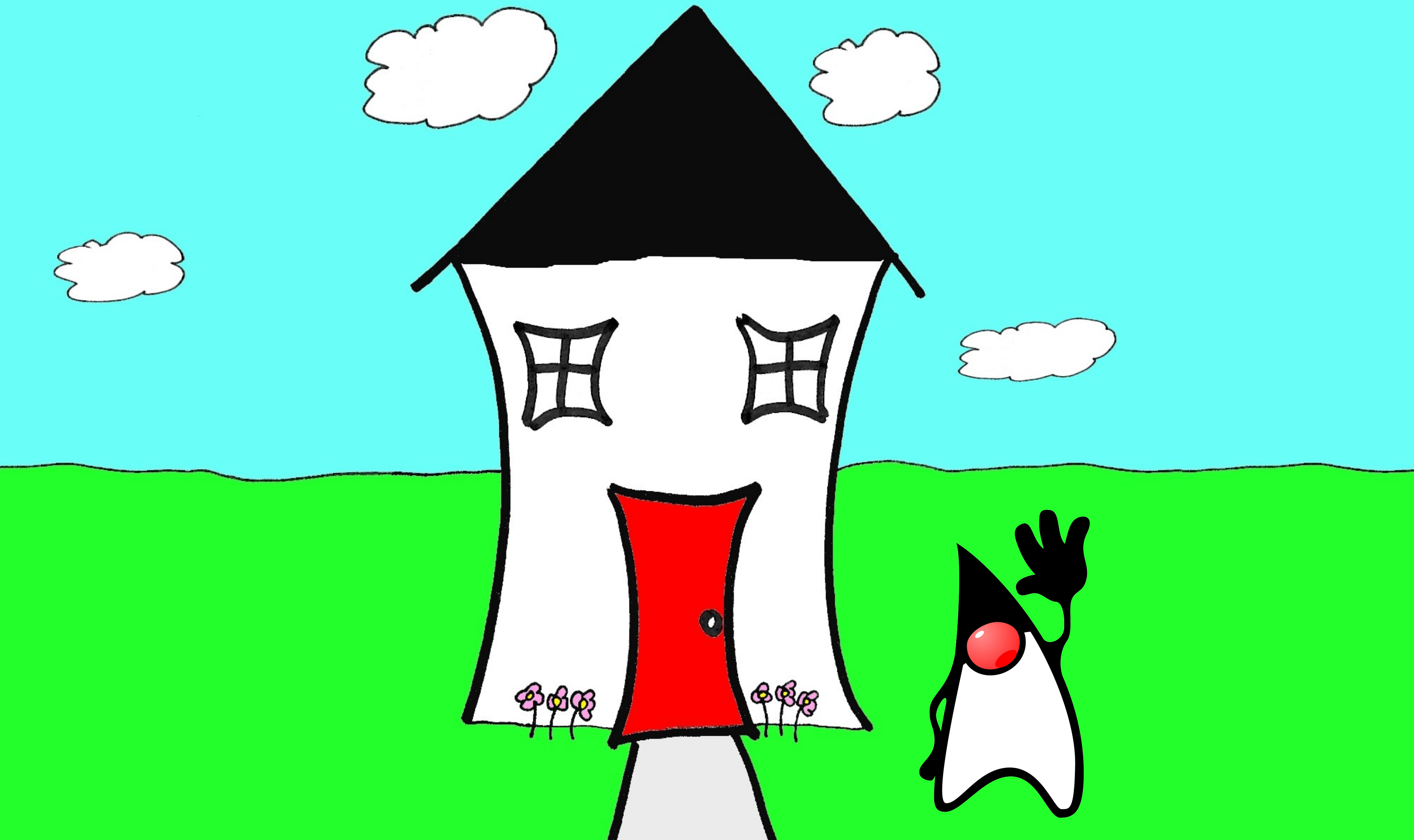
- GUI Design
  - Continued work on Project Matisse GUI Builder
- Application framework and beans binding support
- Java Web Start

Resources

- Update N
  - > https://jdk6.dev.java.net/6uNea.html
- Java FX
  - > https://openjfx.dev.java.net/

Update N and JavaFX

Java Comes Home to the Consumer



Java FX and Java SE 6 Update N

Raghavan “Rags” N. Srinivas
rags@sun.com
Sun Microsystems Inc.