Extreme GUI Makeover 2007

Chris Campbell, Shannon Hickey, Romain Guy

Extreme GUI Makeover Artists

TS-3548

Why Are You Here?

To learn how to take advantage of Swing and Java 2D™ API to create visually stunning applications…

Why Are You Here?

To learn how to take advantage of Swing and Java 2D™ API to create visually stunning applications…

And Have Fun!

Who Are We?

Chris Campbell
Sun Microsystems, Java 2D Master

Shannon Hickey
Sun Microsystems, Swing Sorcerer

Romain Guy
Google, GUI Pimp

Agenda

Introduction

Last year's application, this year's application

Love at First Sight

Look and feel, splash screen, validation

Unlike Any Other

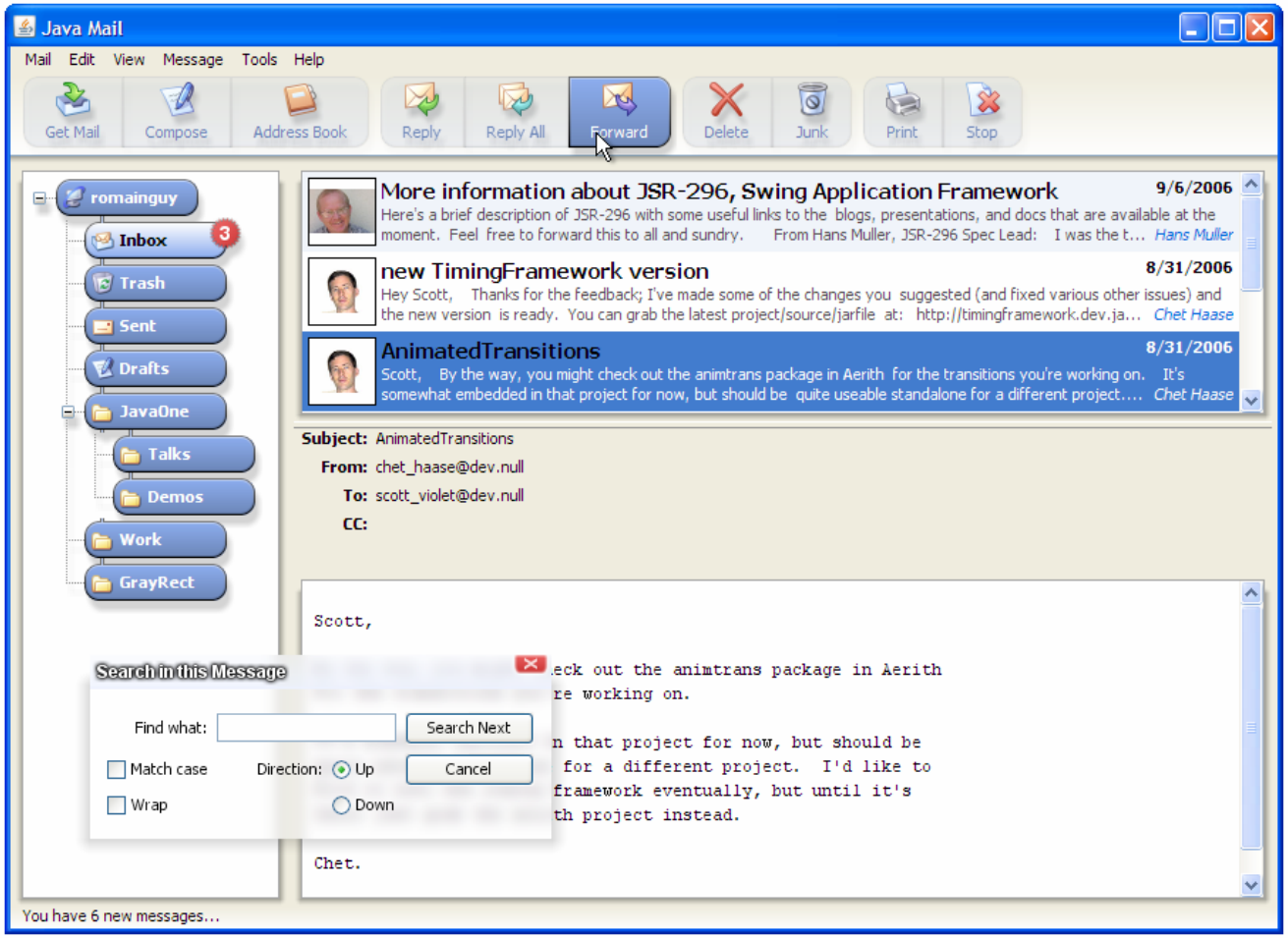
Custom components, fling scrolling

It's All About Tables

Spanning, highlighting, filtering, animating

Surprises

Last Year


Java Mail
Mail Edit View Message Tools Help
Get Mail
Compose
Address Book
Reply
Reply All
Forward
Delete
Junk
Print
Stop
romainguy
Inbox 3
Trash
Sent
Drafts
JavaOne
Talks
Demos
Work
GrayRect
More information about JSR-296, Swing Application Framework 9/6/2006
Here's a brief description of JSR-296 with some useful links to the blogs, presentations, and docs that are available at the moment. Feel free to forward this to all and sundry. From Hans Muller, JSR-296 Spec Lead: I was the t... Hans Muller
new TimingFramework version 8/31/2006
Hey Scott, Thanks for the feedback; I've made some of the changes you suggested (and fixed various other issues) and the new version is ready. You can grab the latest project/source/jarfile at: http://timingframework.dev.ja... Chet Haase
AnimatedTransitions 8/31/2006
Scott, By the way, you might check out the animtrans package in Aerith for the transitions you're working on. It's somewhat embedded in that project for now, but should be quite useable standalone for a different project.... Chet Haase
Subject: AnimatedTransitions
From: chet_haase@dev.null
To: scott_violet@dev.null
CC:
Scott,
eck out the animtrans package in Aerith
re working on.
n that project for now, but should be
for a different project. I'd like to
framework eventually, but until it's
th project instead.
Chet.
Search in this Message
Find what:
Search Next
Match case
Direction: Up
Cancel
Wrap
Down
You have 6 new messages...

We Listened to You

- No chat client
- No email client
- No media player
- No pictures manager
- But a corporate application!

DEMO

Meet REALiTY

Agenda

Introduction

Last year's application, this year's application

Love at First Sight

Look and feel, splash screen, validation

Unlike Any Other

Custom components, fling scrolling

It's All About Tables

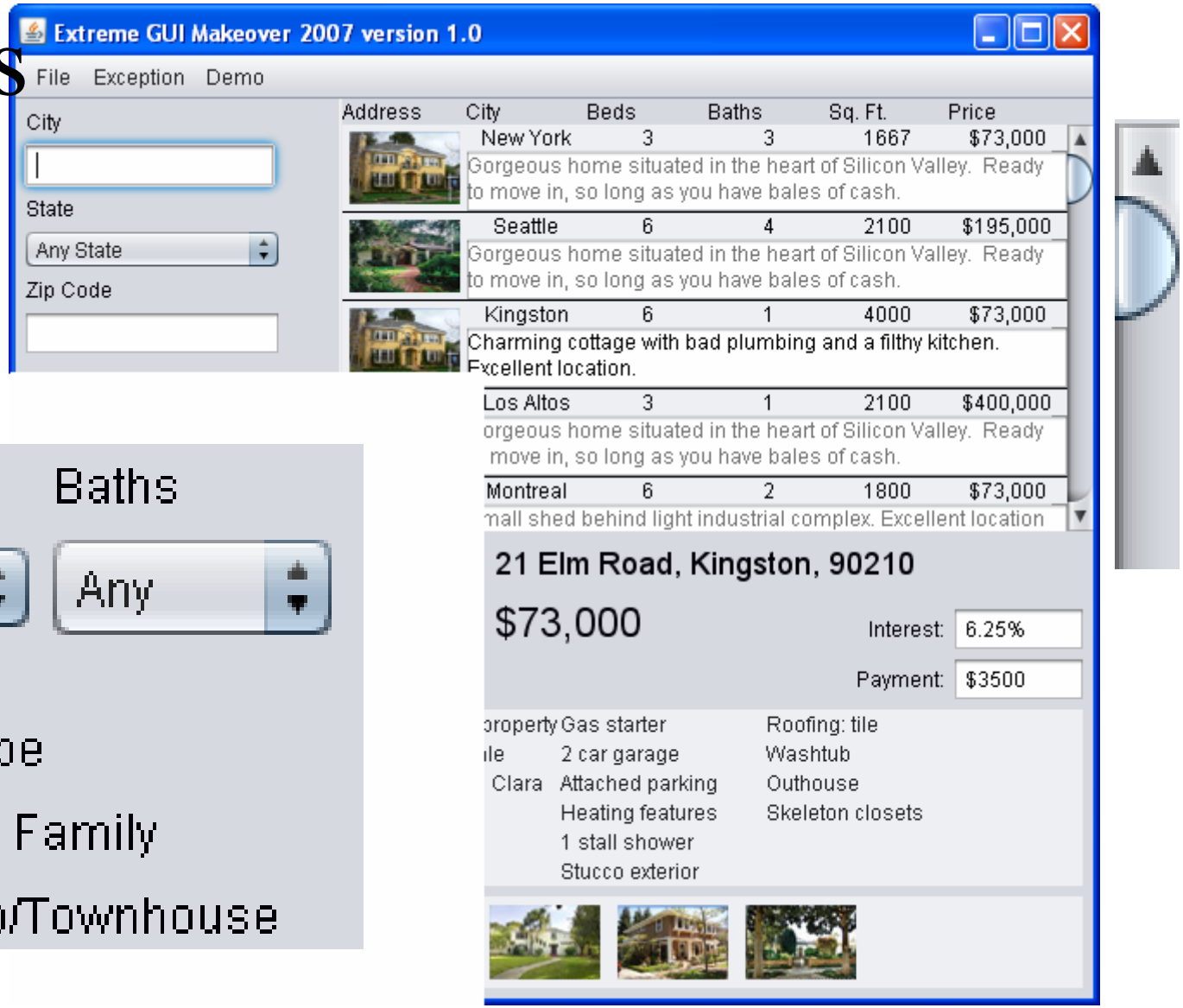
Spanning, highlighting, filtering, animating

Surprises

Love At First Sight

- Use a modern Look and Feel
  - Nimbus, nimbus.dev.java.net
- Add a splash screen
  - New Java™ Platform v.6, Standard Edition (Java™ SE Platform v.6) VM flag-splash
  - Non-rectangular
  - Drop shadow
  - Show progress

Nimbus


Extreme GUI Makeover 2007 version 1.0
File
Exception
Demo
City
State
Any State
Zip Code
Address
City
Beds
Baths
Sq. Ft.
Price
New York 3 3 1667 $73,000
Gorgeous home situated in the heart of Silicon Valley. Ready to move in, so long as you have bales of cash.
Seattle 6 4 2100 $195,000
Gorgeous home situated in the heart of Silicon Valley. Ready to move in, so long as you have bales of cash.
Kingston 6 1 4000 $73,000
Charming cottage with bad plumbing and a filthy kitchen. Excellent location.
Los Altos 3 1 2100 $400,000
orgeous home situated in the heart of Silicon Valley. Ready move in, so long as you have bales of cash.
Montreal 6 2 1800 $73,000
mall shed behind light industrial complex. Excellent location
21 Elm Road, Kingston, 90210
$73,000
Interest: 6.25%
Payment: $3500
roperty Gas starter
ıle 2 car garage
Clara Attached parking
Heating features
1 stall shower
Stucco exterior
Roofing: tile
Washtub
Outhouse
Skeleton closets
Beds
Baths
Any
Any
Property Type
Single Family
Condo/Townhouse

DEMO

Splash Screen

Color Fade Splash Screen

```
// java -splash:sepia.png ...
SplashScreen splash = SplashScreen.getSplashScreen();
if (splash != null) {
  Graphics2D g2 = splash.createGraphics();
  // clear the splash screen
  g2.setComposite(AlphaComposite.Clear);
  g2.fillRect(0, 0,
      splash.getSize().width, splash.getSize().height);
  g2.setPaintMode();

  // draw colors on top of sepia
  g2.drawImage(sepiaSplashImage, 0, 0, null);
  g2.setComposite(AlphaComposite.SrcOver.derive(progress));
  g2.drawImage(colorSplashImage, 0, 0, null);

  g2.dispose();
  splash.update();
}
```

Validating Data

- Forms = input errors
- Validate data in the UI
  - Immediate feedback
  - Responsive application
  - Ease the learning curve
- Visual feedback
  - JGoodies-like
  - Shaking

DEMO

Validation

JGoodies Way

```
JLayeredPane pane = getRootPane().getLayeredPane();
// where is the text field?
Point p = field.getLocationOnScreen();
SwingUtilities.convertPointFromScreen(p, pane);

JLabel icon = new JLabel(warningIcon);
pane.add(icon, JLayeredPane.POPUP_LAYER);

// move the icon over the text field's corner
Dimension size = icon.getPreferredSize();
icon.setBounds(p.x - size.width / 3,
    p.y + field.getHeight() - size.height * 2 / 3,
    size.width, size.height);
icon.setVisible(true);
```

Shaking

```
final Point p = text.getLocation();
// moves the field to west->north->east->south
KeyValues<Point> v = KeyValues.create(
    translate(p, -3,  0), translate(p,  0, -3),
    translate(p, +3,  0), translate(p,  0, +3));
KeyFrames f = new KeyFrames(v);
// animates the field's location
Animator shakeAnimator =
    PropertySetter.createAnimator(
        70, text, "location", f);
// the animation is repeated 15 times
shakeAnimator.setRepeatCount(15);
shakeAnimator.start();
```

Agenda

Introduction

Last year's application, this year's application

Love at First Sight

Look and feel, splash screen, validation

Unlike Any Other

Custom components, fling scrolling

It's All About Tables

Spanning, highlighting, filtering, animating

Surprises

DEMO

Constraints and Details Panels

Basic Master/Detail View

Boring…Cluttered…and Boring!

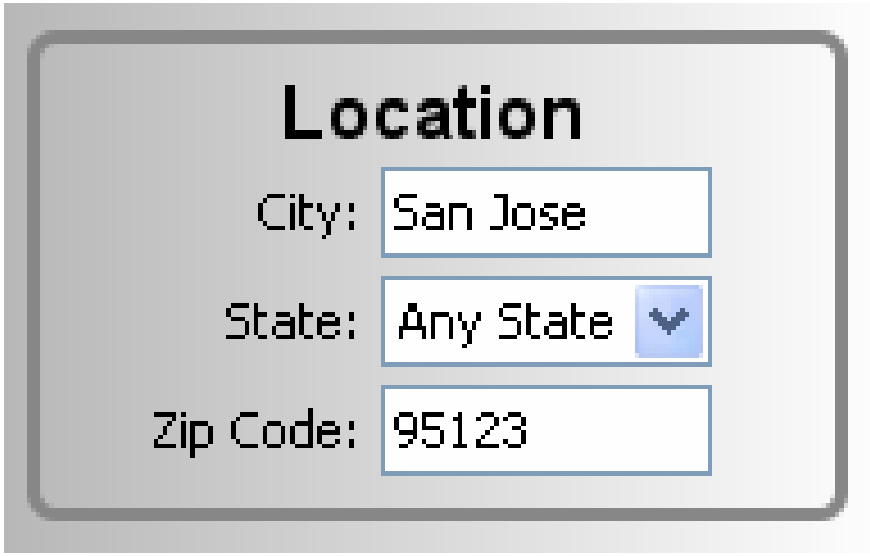
- Constraints panel
  - No logical grouping of controls
  - Range widgets are awkward to use
- Detail panel
  - Takes up valuable screen real estate
  - No visual separation from master table
  - List items are crammed together
- Master table
  - Don’t get me started!

Constraints Panel

The problems

- Widgets are unorganized
  - User may be overwhelmed with options
- Price range box is not very flexible
  - Can’t select more than a single range at once
- More awkward combo boxes
  - Can’t change bed/bath constraints quickly
- Not very attractive in general!

Border and Gradient



Simple Rounded Border

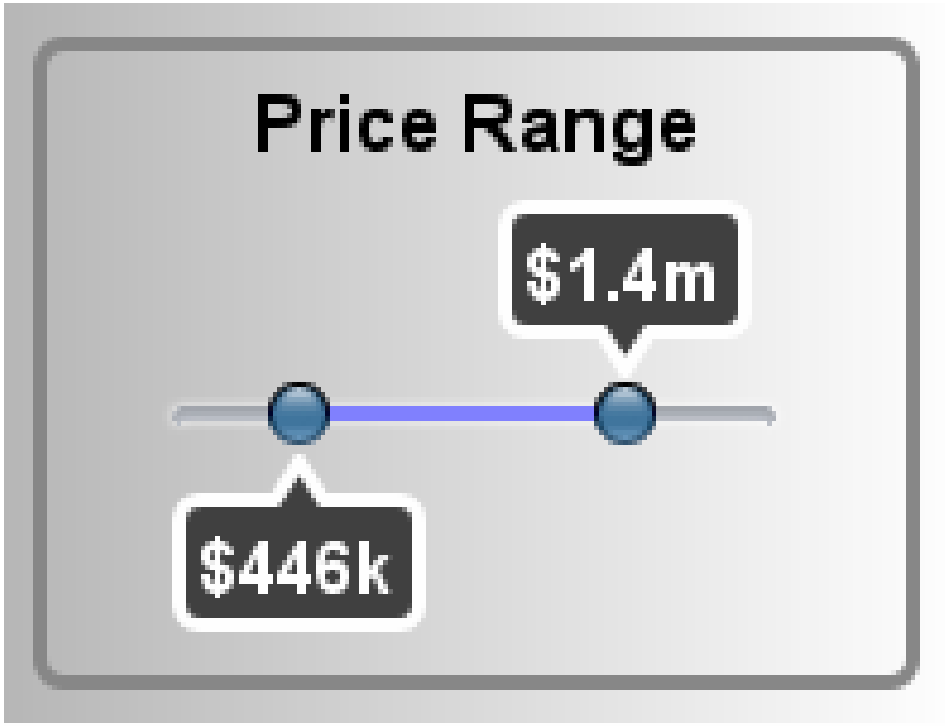
```
private class RoundedBorder extends AbstractBorder {
    private final Color FILL = new Color(135, 135, 135);
    private final Stroke STROKE = new BasicStroke(3f);

    @Override public void paintBorder(...) {
        Graphics2D g2 = (Graphics2D)g.create();
        g2.setRenderingHint(
            RenderingHints.KEY_ANTIALIASING,
            RenderingHints.VALUE_ANTIALIAS_ON);
        g2.setColor(FILL);
        g2.setStroke(STROKE);
        g2.drawRoundRect(10, 10, w-20, h-20, 10, 10);
        g2.dispose();
    }
}
```

Gradients Everywhere!

```
public class ConstraintsPanel extends JPanel {
    public void paintComponent(Graphics g) {
        Graphics2D g2 = (Graphics2D)g;
        // use cyclic gradient for improved performance
        Paint bg =
            new GradientPaint(0, 0, COLOR1,
                              0, getWidth(), COLOR2,
                              true);
        g2.setPaint(bg);
        g2.fillRect(0, 0, getWidth(), getHeight());
    }
}
```

A Better Range Widget



A Better Range Widget

```
public class MultiSlider extends JPanel {
    public void paintComponent(Graphics g) {
        calculateThumbBounds();
        Graphics2D g2 = (Graphics2D)g.create();
        // draw track
        g2.drawImage(imgTrackL, XPAD, y, null);
        g2.drawImage(imgTrackR, xc2, y, null);
        // use scaling drawImage() to stretch center img
        g2.drawImage(imgTrackC,
            xc1, y, xc2-xc1, imgTrackC.getHeight(), null);
        // draw track highlight
        g2.setColor(HILITE);
        g2.fill(trackHilite);
        // draw thumbs
        g2.drawImage(imgThumb, thumbLo.x, thumbLo.y,null);
        g2.drawImage(imgThumb, thumbHi.x, thumbHi.y,null);
    }
}
```

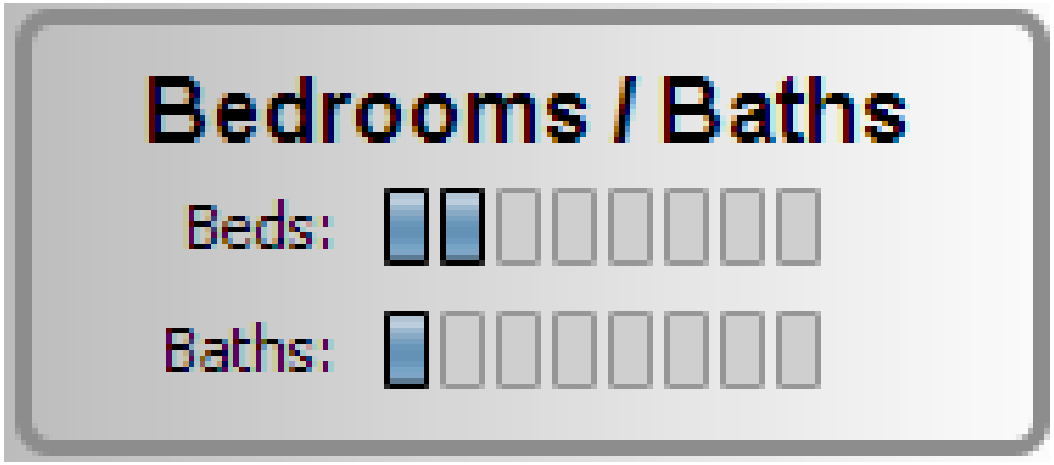
Centering Bubble Text

```
public class MultiSlider extends JPanel {
    public void paintComponent(Graphics g) {
        Graphics2D g2 = (Graphics2D)g.create();
        g2.setRenderingHint(...);
        g2.setColor(Color.WHITE);
        g2.setFont(FONT);
        FontMetrics metrics = g2.getFontMetrics();
        int textW = metrics.stringWidth(textLo);
        int textH = metrics.getAscent();
        g2.drawString(textLo,
            pointX - (textW/2),
            pointY + pointH + pad + textH);
        ...
        g2.dispose();
    }
}
```

VUMeter Custom Component

- Simple JComponent subclass
  - on/off segment Icons
  - 0 < value < segmentCount
  - Input handlers update value property
  - Paint method draws segment icons

VUMeter: Bonus Optical Illusion!

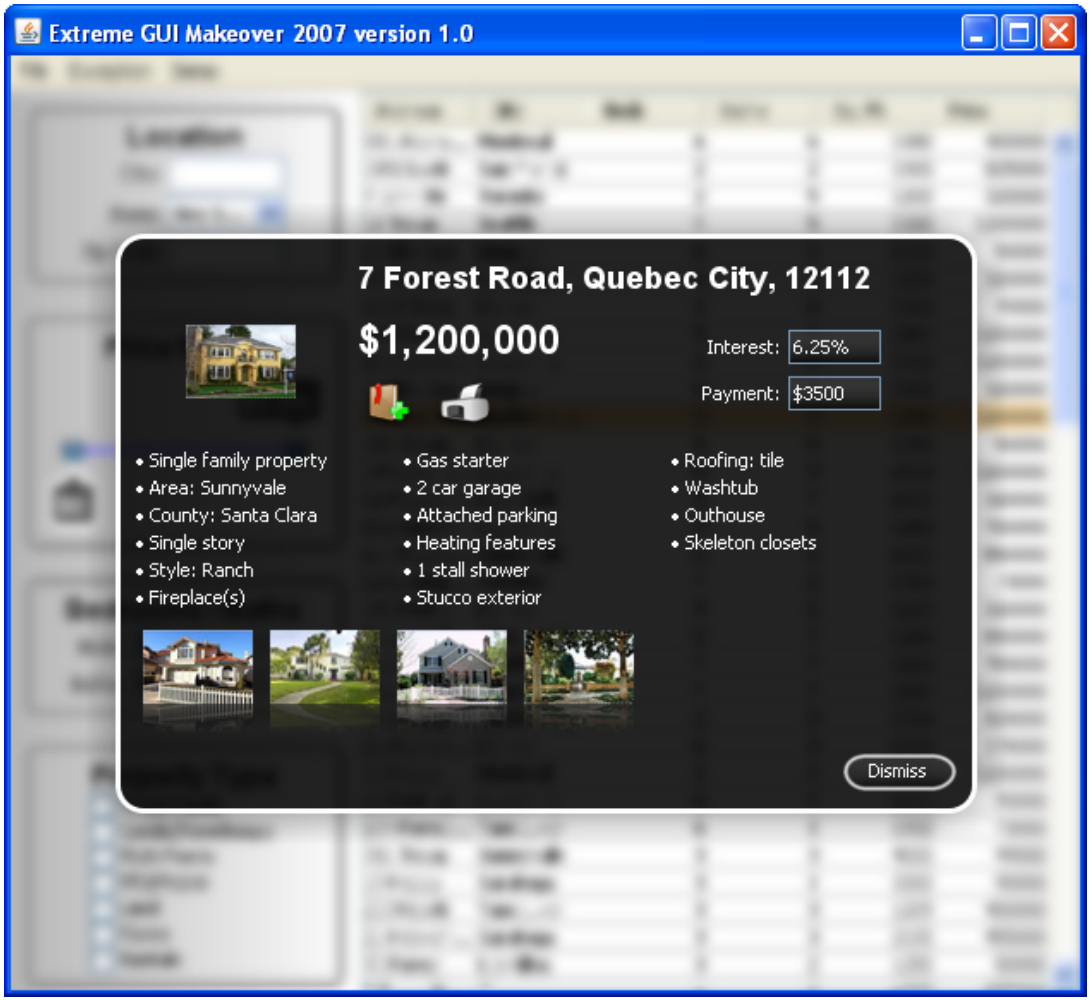


Detail Panel

The problems

- Takes space away from master table
- Bullet items should, umm, have bullets
- Everything is crammed together
  - Don’t be afraid to use “whitespace”
- Again, not very attractive!

Fancy Detail Panel



Using JList.VERTICAL_WRAP

```
public class DetailPanel extends JPanel {

    private void initComponents() {
        bulletList = new JList();
        bulletList.setOpaque(false);
        bulletList.setVisibleRowCount(6);
        bulletList.setLayoutOrientation(
            JList.VERTICAL_WRAP);
        bulletList.setCellRenderer(
            new BulletCellRenderer());
    }
}
```

Bullets and Padded Columns

```
private class BulletCellRenderer
    extends DefaultListCellRenderer {
    public Component getListCellRendererComponent(...) {
        // disable cell selection and focus
        JLabel c = (JLabel)
            super.getListCellRendererComponent(
                list, value, index, false, false);
        // prepend quick-and-dirty bullet
        c.setText("\u2022 " + c.getText());
        // adjust to get 3 evenly spaced columns
        c.setPreferredSize(new Dim(list.getWidth()/3,
            c.getPreferredSize().height);
        c.setOpaque(false);
        c.setForeground(Color.WHITE);
        return c;
    }
}
```

Thumbnail Reflections

```
import org.jdesktop.swingx.graphics.ReflectionRenderer;

private class PhotoListModel extends AbstractListModel {
    public void setPhotos(List<ImageIcon> photos) {
        this.photoList.clear();
        ReflectionRenderer renderer =
            new ReflectionRenderer();
        for (ImageIcon orig : photos) {
            BufferedImage origImg = (BufferedImage)
                orig.getImage();
            BufferedImage newImg =
                renderer.appendReflection(origImg);
            this.photoList.add(new ImageIcon(newImg));
        }
        fireContentsChanged(this, 0, photoList.size());
    }
}
```

Agenda

Introduction

Last year's application, this year's application

Love at First Sight

Look and feel, splash screen, validation

Unlike Any Other

Custom components, fling scrolling

It's All About Tables

Spanning, highlighting, filtering, animating

Surprises

Table Makeover

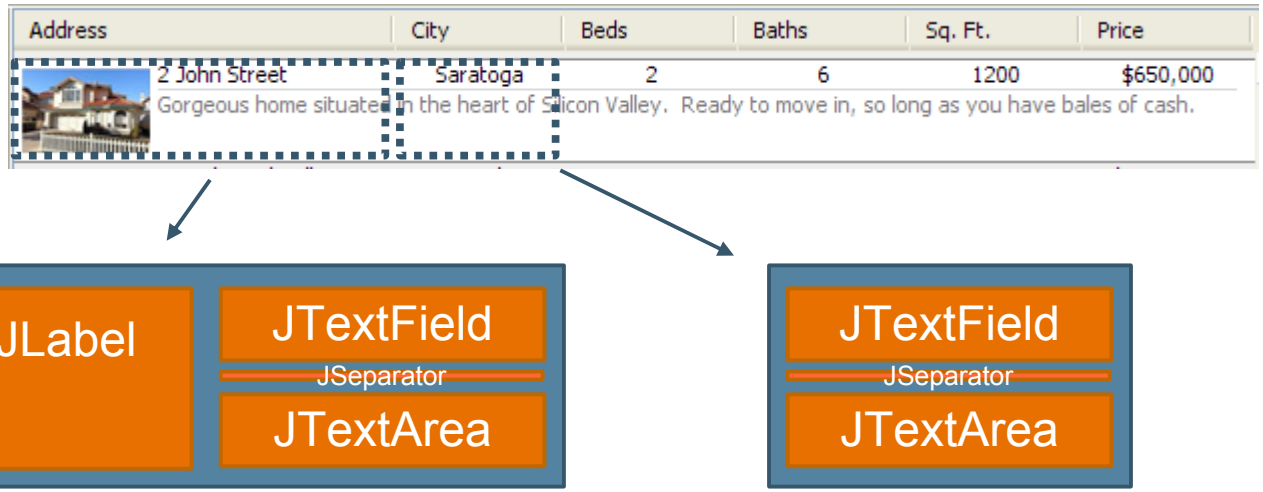
- Fancy cell rendering
- Light-up filtering
- Animated sorting and filtering

DEMO

Fancy Cell Rendering and Light-up Filtering

Complex Cell Renderer

- Default cell render is a JLabel: string/icon
- Ours are defined with a JPanel
  - GridBagLayout (totally)
  - One for the first column, another for all the rest

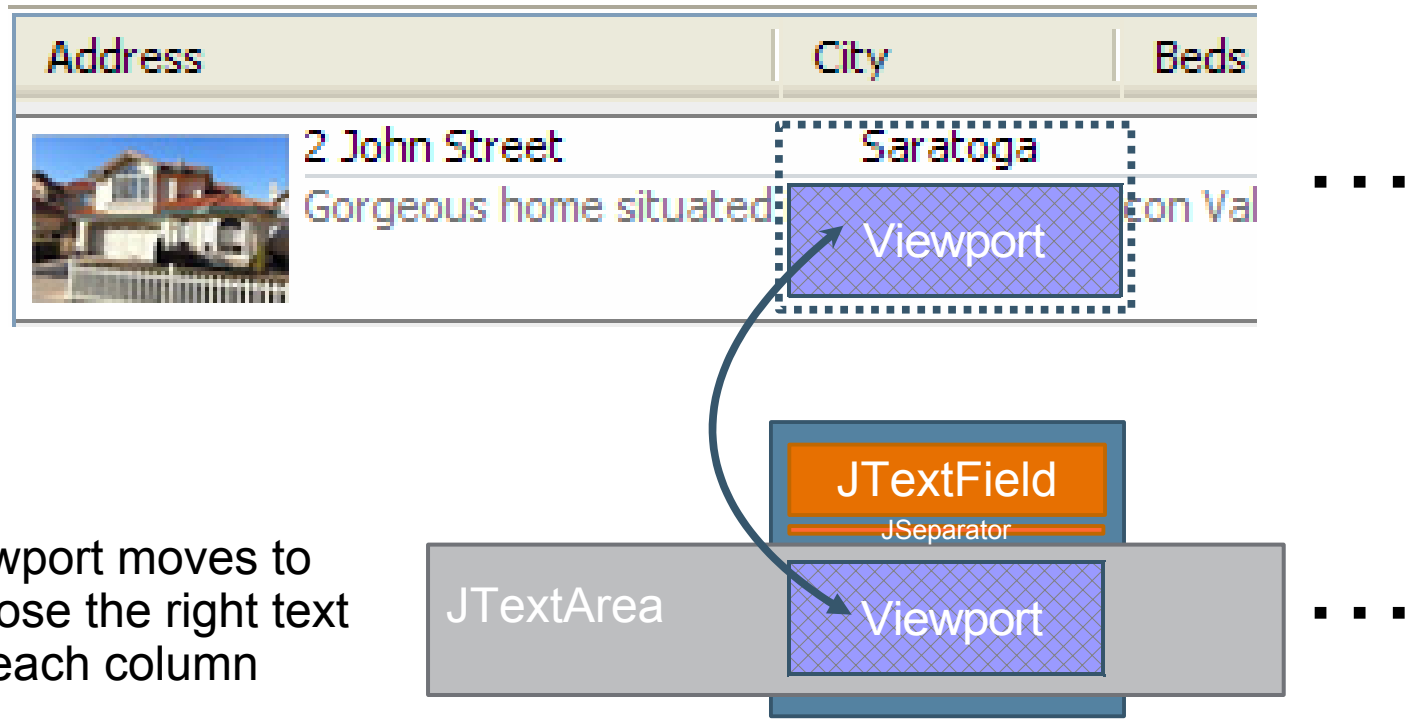


1st Column Cell Renderer Panel

Rest Column Cell Renderer Panel

Cell Renderer: Column Spanning

Viewport clips column contents



Viewport moves to expose the right text for each column

Light-Up Filtering

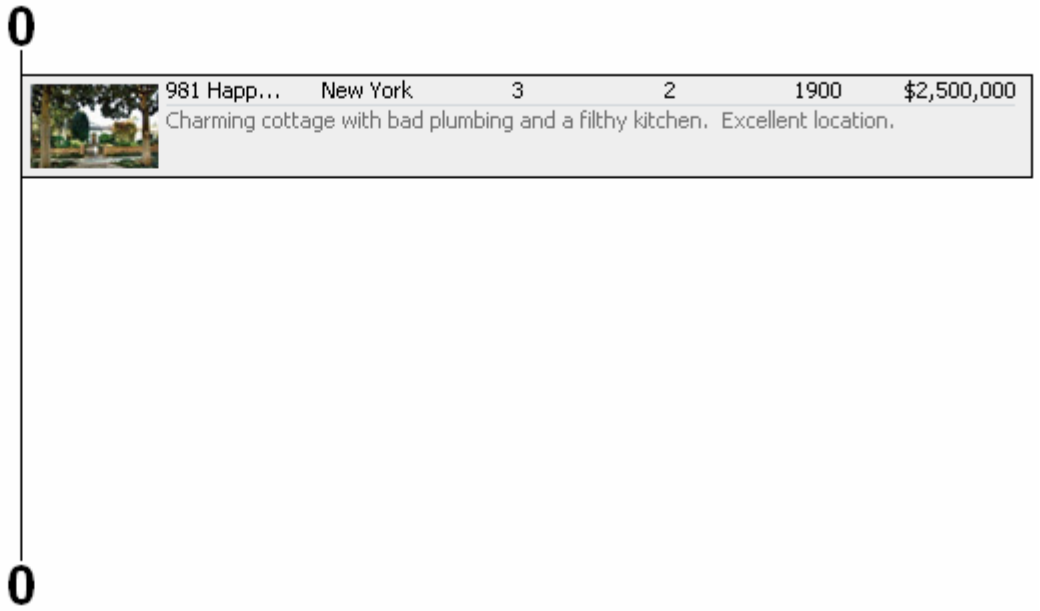
The setup

```
ChangeListener luListener = new ChangeListener() {
    public void stateChanged(ChangeEvent ce) {
        char type = 'Z';
        if (singleFamilyCB.getModel().isRollover()) {
            type = 'S';
        } else if (condoCB.getModel().isRollover()) {
            type = 'C';
        } // etc...

        setLightupType(type);
        table.repaint(table.getVisibleRect());
    }
};
singleFamilyCB.getModel().addChangeListener(luListener);
condoCB.getModel().addChangeListener(luListener);
// etc...
```

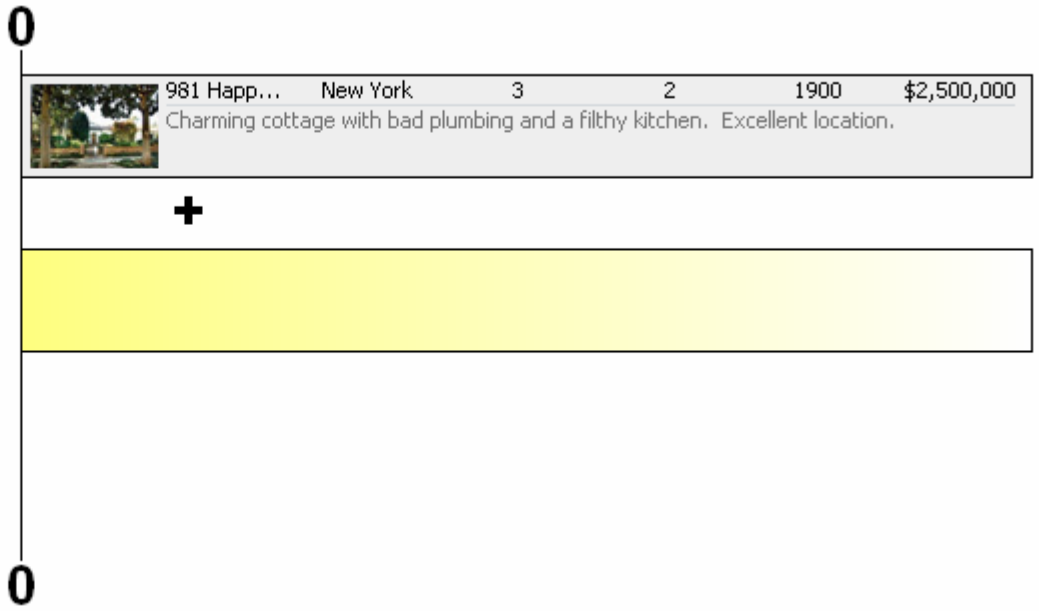
Light-Up Filtering

The Light-Up (in images)



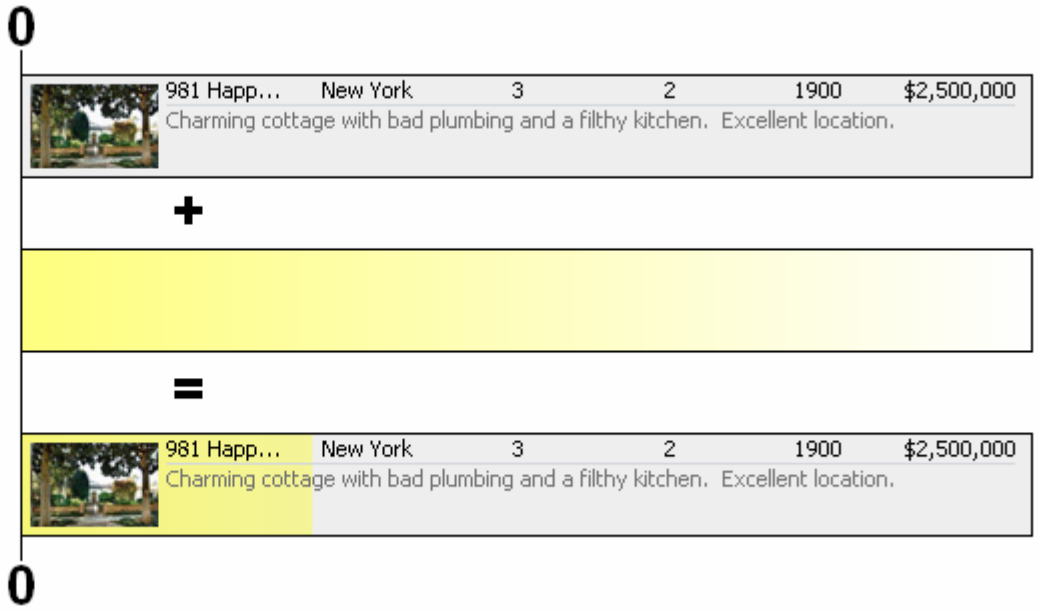
Light-Up Filtering

The Light-Up (in images)



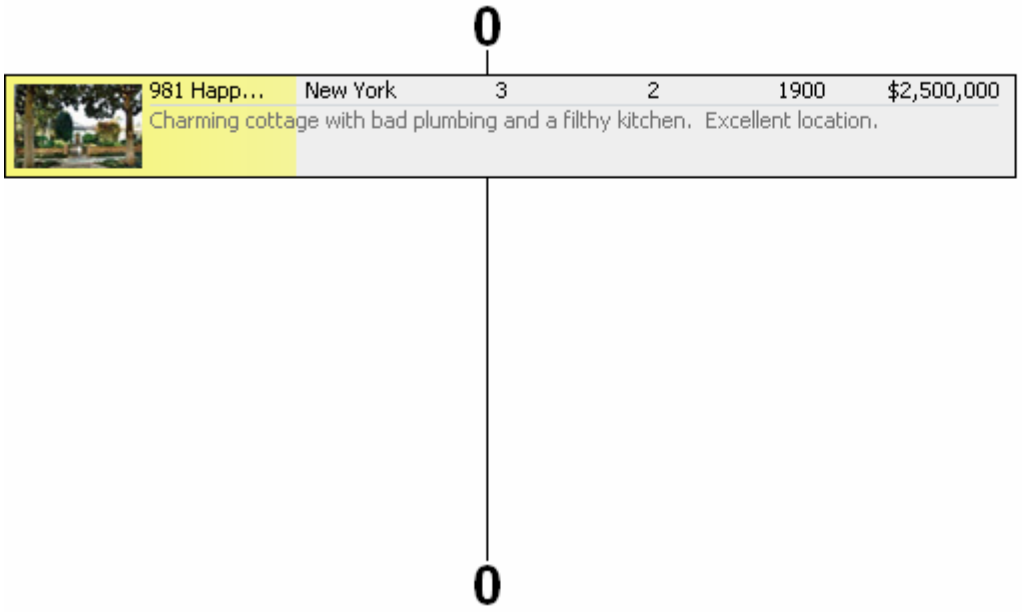
Light-Up Filtering

The Light-Up (in images)



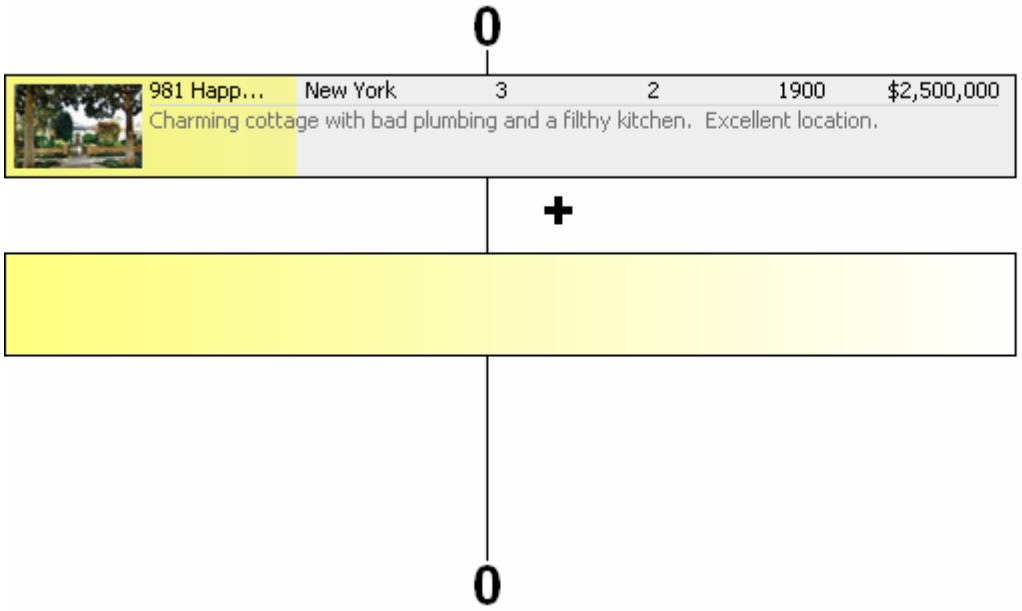
Light-Up Filtering

The Light-Up (in images)



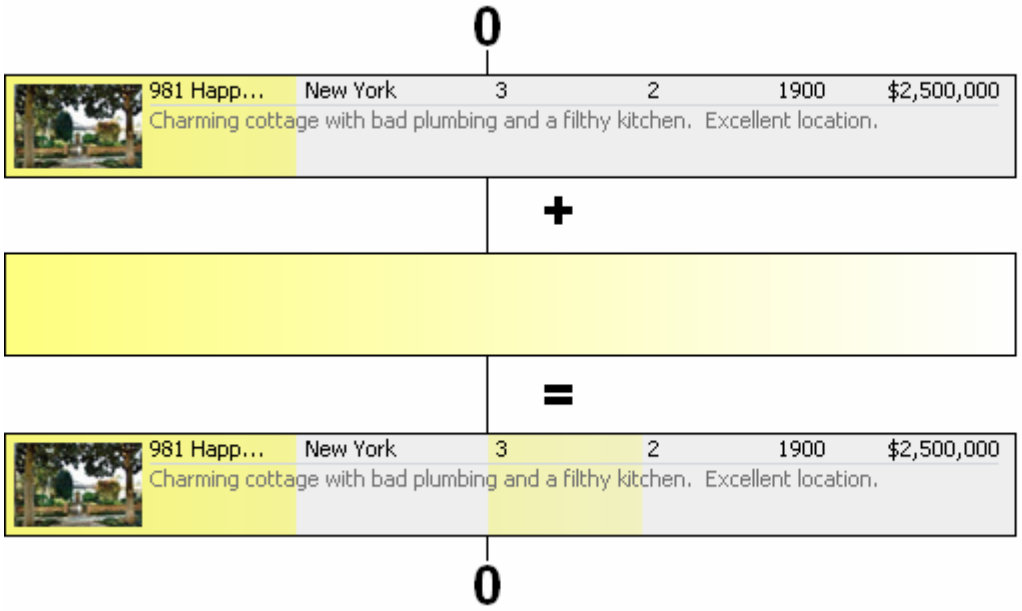
Light-Up Filtering

The Light-Up (in images)



Light-Up Filtering

The Light-Up (in images)



Light-Up Filtering

The Light-Up (in code)

```
public Component getTableCellRendererComponent(
    JTable table, ..., int vRow, int vCol) {

    int row = table.convertRowIndexToModel(vRow);
    Home home = homes.get(row);
    doHighlight = (home.getType() == getLightupType());
    gx0 = -table.getCellRect(vRow, vCol, true).x;
    gx1 = table.getWidth() - gx0;
}
```

Light-Up Filtering

The Light-Up (in code)

```
public void paintComponent(Graphics g) {
    super.paintComponent(g);
    if (doHighlight) {
        Graphics2D g2d = (Graphics2D)g.create();
        GradientPaint gp =
            new GradientPaint(gx0, 0, YELLOW128A,
                              gx1, 0, YELLOW0A, true);
        g2d.setPaint(gp);
        g2d.fillRect(0, 0, getWidth(), getHeight());
        g2d.dispose();
    }
}
```

What if We Goof?

i.e. Shannon's first attempt…

	24 Aborea…	Quebec City	1	1	1980	$2,500,000
	Charming cottage with bad plumbing and a filthy kitchen. Excellent location.					
	500 Lastin…	Saratoga	3	1	1400	$320,000
	Small shed behind light industrial complex. Excellent location for observing crime. Suitable for farm animals or first-time buyers.					
	24 Forest …	Seattle	2	1	1250	$825,000
	Charming cottage with bad plumbing and a filthy kitchen. Excellent location.					
	24 Heave…	Denver	2	2	1320	$320,000

Table Animations

The setup

```
class InstrumentedSorter extends TableRowSorter {
    InstrumentedSorter(TableModel model) {super(model);}

    public void setRowFilter(RowFilter filter) {
        before();
        super.setRowFilter(filter);
        after();
    }

    public void toggleSortOrder(int column) {
        before();
        super.toggleSortOrder(column);
        after();
    }

    protected void fireRowSorterChanged(int[] lastRowIndexToModel) {
        lastViewToModel = lastRowIndexToModel;
        super.fireRowSorterChanged(lastRowIndexToModel);
    }
}
```

Table Animations

The Nitty-Gritty

```
private void before() {
    // capture the currently visible table rows into an image buffer

    // capture the states of all currently visible rows
    for (each row currently visible) {
        /*
         * store the following:
         * - current view index
         * - model index:
         *       ie. table.convertRowIndexToModel(viewIndex)
         * - y position in terms of parent:
         *       ie. SwingUtilities.convertRectangle(table, rect,
         *                                           table.getParent());
         * - height
         * - index into the image buffer
         */
    }
}
```

Table Animations

The Nitty-Gritty

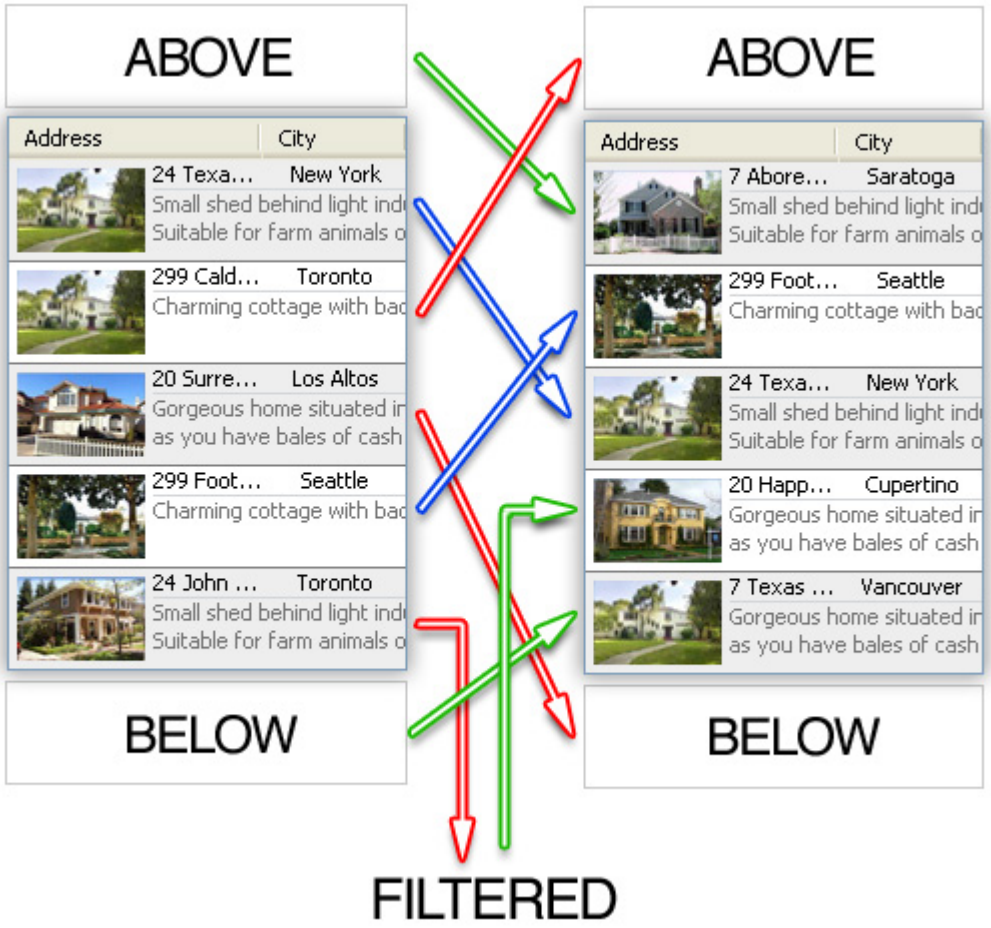
```
private void after() {
    RepaintManager.currentManager(null).validateInvalidComponents();

    // capture the newly visible table rows into an image buffer

    // capture the states of the newly visible rows

    processStates();
}
```

Animated Segue: Moving Rows



```
private void processStates() {
    // We now have a list of before-states and after-states.
    // Combine them into a single list (states) tracking where
    // things are coming from and going to:

    // first, find where the before-states are going to
    int minAfter = afterStates.get(0).viewIndex;
    for (RowState state : beforeStates) {
        int i = afterStates.indexOf(state);
        if (i != -1) {
            RowState after = afterStates.get(i);
            afterStates.remove(i);
            state.y1 = after.y;
            state.imagePos1 = after.imagePos;
        } else {
            int newView = table.convertRowIndexToView(state.row);
            if (newView == -1) {
                state.y1 = FILTERED;
            } else if (newView < minAfter) {
                state.y1 = ABOVE;
            } else {
                state.y1 = BELOW;
            }
        }
        states.add(state);
    }
```

```
    // ...continued from previous slide

    // now process the remaining after-states to see
    // where they're coming from
    for (RowState state : afterStates) {
        state.y1 = state.y;

        // getPreviousView() uses lastViewToModel that
        // we stored earlier
        int oldView = getPreviousView(state.row);

        if (oldView == -1) {
            state.y = FILTERED;
        } else if (oldView < beforeStates.get(0).view) {
            state.y = ABOVE;
        } else {
            state.y = BELOW;
        }
        states.add(state);
    }
}
```

The Rest Are Details…

- Decide on start or end positions for those items with BEFORE, AFTER, or FILTER
- Choose an algorithm that provides a nice look
- At the end of processing, use TimingFramework to drive animation of rows from their start to end positions
- Use a glasspane to paint the animation
- Special handling for things that are moving within the view and need to change background color
  - We have an image capture for both states
  - Use translucency to transition

DEMO

Fling Scrolling

Inertial (*Fling!*) Scrolling

- React to a mouse drag gesture
  - Upwards/downwards scroll has inertia
  - Like spinning a bicycle wheel
- Gesture recognizer tracks velocity acceleration
- Animation thread updates JViewport viewPosition
- Physics simulation adds inertia, friction
  - Original simulation was simple but annoying to use
  - New simulation feels good, but is a bit complicated to cover on a slide

Agenda

Introduction

Last year's application, this year's application

Love at First Sight

Look and feel, splash screen, validation

Unlike Any Other

Custom components, fling scrolling

It's All About Tables

Spanning, highlighting, filtering, animating

Surprises

We Said Extreme!

- Why stick to the window's bounds?
- There is plenty of space outside
  - Let's use it
- JNA
  - Easier Java Native Interface (JNI™), jna.dev.java.net
  - Non rectangular windows
  - Translucent windows
- Thanks to Timothy Wall for his help

DEMO

External Popup

Translucent, Shaped Windows

```
// provided by JNA! image is a translucent PNG
AlphaWindow.enableAlphaForWindow(
    window, getAlpha(), image);

// fades the window in
Animator animator = PropertySetter.createAnimator(
    400, this, "alpha", 1.0f);
animator.start();

// timing framework support
public float getAlpha() { return alpha; }
public void setAlpha(float alpha) {
    this.alpha = alpha;
    AlphaWindow.updateAlphaForWindow(window, alpha);
}
```

Following the Selection

```
// binds table selection to window's location
Binding b = new Binding(this, "${selectionBounds}",
    SwingUtilities.getWindowAncestor(detailView),
    "animatedLocation");
// converts selection location to window location
binding.setConverter(c);
b.bind();

BindingConverter c = new BindingConverter() {
  public Object sourceToTarget(Object value) {
    Rectangle b = (Rectangle) value;
    return new Point(b.x + b.width - 35,
                     b.y + b.height / 2 - 70);
  }
};
```

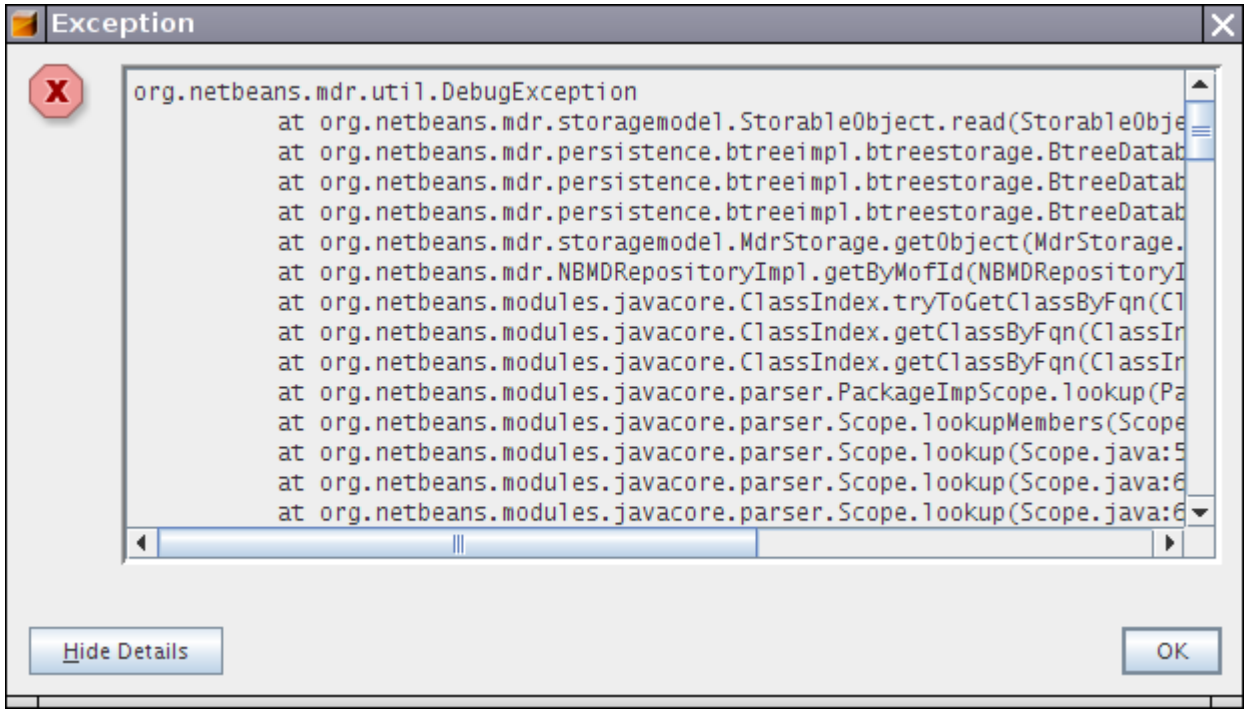
Following the Selection (Cont.)

```
// animation is easy
public Point getAnimatedLocation() {
  return window.getLocation();
}

public void setAnimatedLocation(Point p) {
  Animator animator = PropertySetter.createAnimator(
      200, window, "location", p);
  animator.setAcceleration(0.1f);
  animator.setDeceleration(0.1f);
  animator.start();
}

// do the same when the parent window moves
```

Showing the Internals



DEMO

All the Internals

Tear Effect Resources

Mask

Shadow

Tear Effect Implementation

```
public void setHoleSize(float holeSize) {
  this.holeSize = holeSize;
  int w = (int) (buffer.getWidth() * holeSize);
  int h = (int) (buffer.getHeight() * holeSize);
  int x = (int) ((holeSize / 2.0) * (-getWidth() / 3.0));
  int y = getHeight() - h;

  // "image" = copy of the frame's content pane
  g2.setComposite(AlphaComposite.Clear);
  g2.fillRect(0, 0, buffer.getWidth(), buffer.getHeight());
  g2.setComposite(AlphaComposite.SrcOver);
  g2.drawImage(image, 0, 0, null);
  g2.setComposite(AlphaComposite.DstOut);
  g2.drawImage(hole, x, y, w, h, null);

  repaint();
}
```

Tear Effect Implementation (Cont.)

```
protected void paintComponent(Graphics g) {
  Graphics2D g2 = (Graphics2D) g;
  g2.setRenderingHint(
      RenderingHints.KEY_INTERPOLATION,
      RenderingHints.VALUE_INTERPOLATION_BILINEAR);

  // scales the shadow's size to match content's
  int w = (int) (buffer.getWidth() * holeSize);
  int h = (int) (buffer.getHeight() * holeSize);
  int x = (int) ((holeSize / 2.0) * (-getWidth() / 3.0));
  int y = getHeight() - h;

  // draws the shadow first
  g2.drawImage(holeShadow, x + 2, y + 6, w, h, null);
  g2.drawImage(buffer, 0, 0, null);
}
```

Catching Uncaught Exceptions

```
Thread.setDefaultUncaughtExceptionHandler(
  // instance of Thread.UncaughtExceptionHandler
  this);

public void uncaughtException(Thread t, Throwable e) {
  // copies frame's content into a buffer
  JRootPane root = getMainFrame().getRootPane();
  BufferedImage image =
      createCompatibleImage(root.getWidth(),
                            root.getHeight());
  Graphics2D g2 = image.createGraphics();
  root.paint(g2);
  g2.dispose();
  // shows the buffer and animates the hole in it
  BurningHole c = new BurningHole(image);
}
```

Summary

- Swing and Java 2D API
  - Powerful API for stunning results
- Easier with libraries
  - Timing Framework, timingframework.dev.java.net
  - JNA, jna.dev.java.net
  - SwingX, swingx.dev.java.net
  - Nimbus, nimbus.dev.java.net
- Corporate applications can be fun!

Going Further

Sessions and BOFs

- Designing Successful UIs, Today @8:55pm
- F3, Thursday @1:30pm
- Beans Bindings, Thursday @4:10pm
- Next Generation UI Elements, Thursday @9:55pm
- Filthy Rich Clients, Friday @2:50pm

- Blogs
  - weblogs.java.net/blog/shan_man
  - weblogs.java.net/blog/campbell
  - www.curious-creature.org

We'd Like To Thank

- Hans Muller
  - Maker of Happeningness
- Jeff Dinkins
  - Designer of Graphics
- Jim Graham
  - Flinger of Table

Q&A

Chris Campbell
Shannon Hickey
Romain Guy

Extreme GUI Makeover 2007

Chris Campbell, Shannon Hickey, Romain Guy

Extreme GUI Makeover Artists

TS-3548