ORACLE®

JavaOne™

Using CSS to Style JavaFX UI Controls
The Visual Designer's Experience

{ **Eileen Bugée and Tim Dunn**
Principal Visual Designers | Oracle Corporation }

Hi. Welcome to our BOF...
I'm Eileen ....(and I'm Tim)
We're Visual Designers in the Middleware User Experience group at Oracle.
Over the past year, we've worked closely with Engineering on the Look of the JavaFX UI Components, and we learned so much that we thought it would be a great idea to share it with you.

Outline

. **The Caspian Theme**

. **Styling Overview**

. **Desktop Re-theming**

. **TV: Current and Future**

. **Q&A**

```
.text-box {
    -fx-font: 12pt "Amble-Condensed";
    -fx-padding: 0;
    -fx-background-color: -fx-shadow-highli
fx-mid-border-color, -fx-control-inner-back
    -fx-background-radius: 6, 5 4;
    -fx-background-insets: 0, .5, 1.5;
    -fx-text-fill: -fx-text-inner-color;
    -fx-prompt-text-fill: derive(-fx-control-in
    -fx-accent: #114fd9;
    -fx-base: #3f3f3f;
    -fx-highlight-fill: -fx-accent;
    -fx-background: #040404;
    -fx-highlight-text-fill: white;
    -fx-control-inner-background: white;
    -fx-pannable: true;
    -fx-focus-color: #31BAED;
    -fx-select-on-focus: true;
    -fx-dark-text-color: black;
    -fx-mid-text-color: #292929;
    -fx-light-text-color: white;
    -fx-color: -fx-base;
    -fx-hover-base: ladder -fx-base stops (20%,derive(-fx-base,20%)) (35%,derive(-f
(50%,derive(-fx-base,40%));
    -fx-pressed-base: derive(-fx-base,-40%);
    -fx-box-border: #000000;
    -fx-text-box-border: ladder -fx-background stops (0.1,black) (0.3,derive(-fx-bac
    -fx-shadow-highlight-color:#3B3B3B;
    -fx-outer-border: #000000;
    -fx-inner-border: linear (0%, 0%) to (0%, 100%) stops (0%, #7A7A7A) (100%, #52
    -fx-body-color: linear (0%,0%) to (0%,100%) stops (0,derive(-fx-color,34%)) (1,d
color,-18%));
```

So, what are we going to talk about tonight....

First, I'll give you a bit of context around the Caspian Theme.
Then I'll share how the UI Controls are created and styled using CSS.
Next I'll do some Desktop re-theming...
and Tim will show you CSS theming for TV, with some VERY cool modifications.
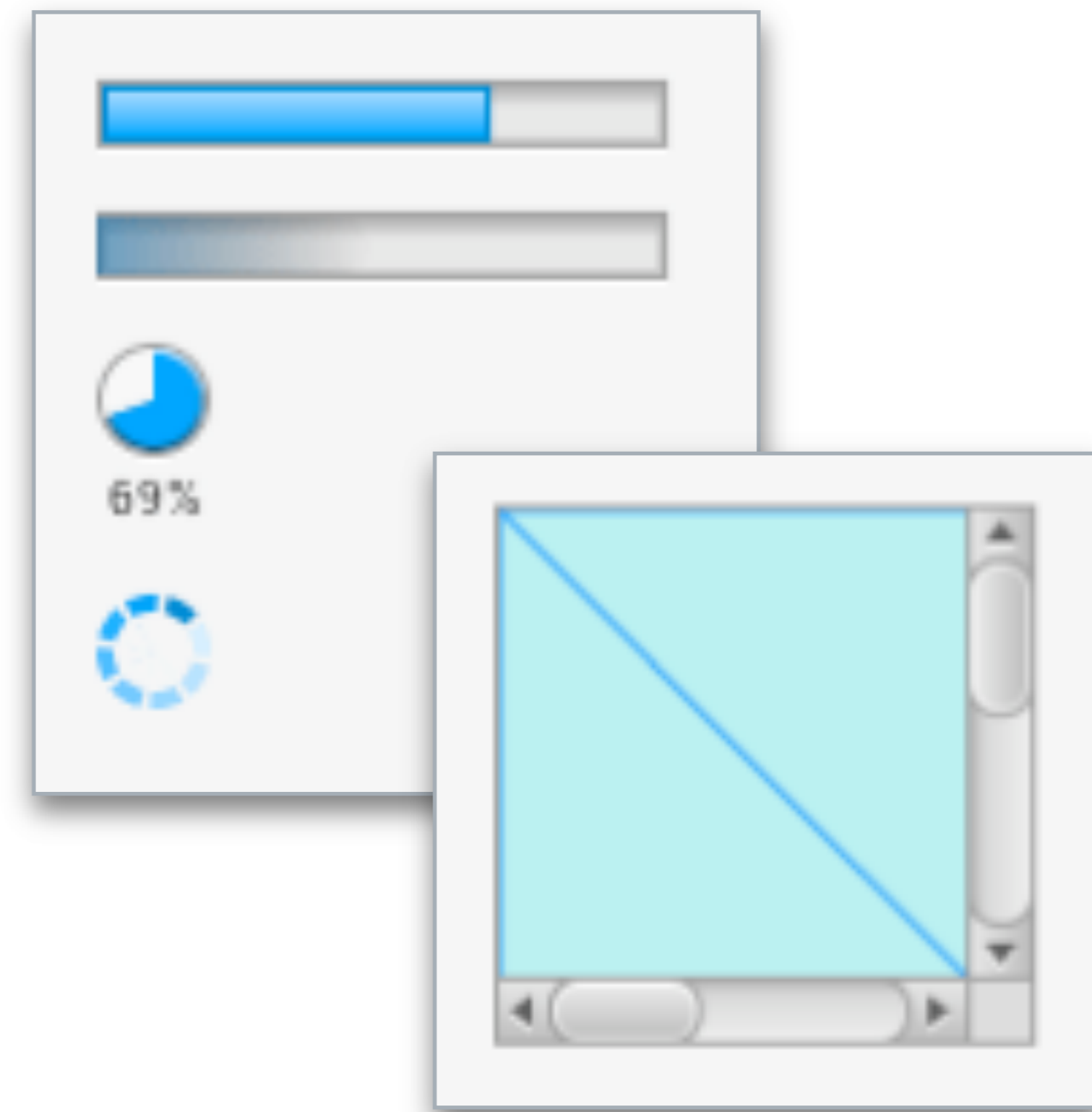We'll take questions at the end.

Outline

. **The Caspian Theme**

. Styling Overview

. Desktop Re-theming

. TV: Current and Future

. Q&A

```
.text-box {
    -fx-font: 12pt "Amble-Condensed";
    -fx-padding: 0;
    -fx-background-color: -fx-shadow-highli
fx-mid-border-color, -fx-control-inner-back
    -fx-background-radius: 6, 5 4;
    -fx-background-insets: 0, .5, 1.5;
    -fx-text-fill: -fx-text-inner-color;
    -fx-prompt-text-fill: derive(-fx-control-in
background, 30%);
    -fx-highlight-fill: -fx-accent;
    -fx-highlight-text-fill: white;
    -fx-pannable: true;
    -fx-select-on-focus: true;
}
-fx-accent: #114fd9;
-fx-base: #3f3f3f;
-fx-background: #040404;
-fx-control-inner-background: white;
-fx-focus-color: #31BAED;
-fx-dark-text-color: black;
-fx-mid-text-color: #292929;
-fx-light-text-color: white;
-fx-color: -fx-base;
-fx-hover-base: ladder -fx-base stops (20%,derive(-fx-base,20%)) (35%,derive(-
(50%,derive(-fx-base,40%));
-fx-pressed-base: derive(-fx-base,-40%);
-fx-box-border: #000000;
-fx-text-box-border: ladder -fx-background stops (0.1,black) (0.3,derive(-fx-bac
-fx-shadow-highlight-color:#3B3B3B;
-fx-outer-border: #000000;
-fx-inner-border: linear (0%, 0%) to (0%, 100%) stops (0%, #7A7A7A) (100%, #52
-fx-body-color: linear (0%,0%) to (0%,100%) stops (0,derive(-fx-color,34%)) (1,d
color,-18%));
```

Let's start with some background on the Caspian theme.

JavaFX "Caspian" Theme

✳ Default "look" for Desktop JavaFX UI Controls

The "Caspian" theme is the "out-of-the box" look for the JavaFX UI Controls.   The visual style is matte, not glossy.
Gradients are soft, and the lighting is "top-down."
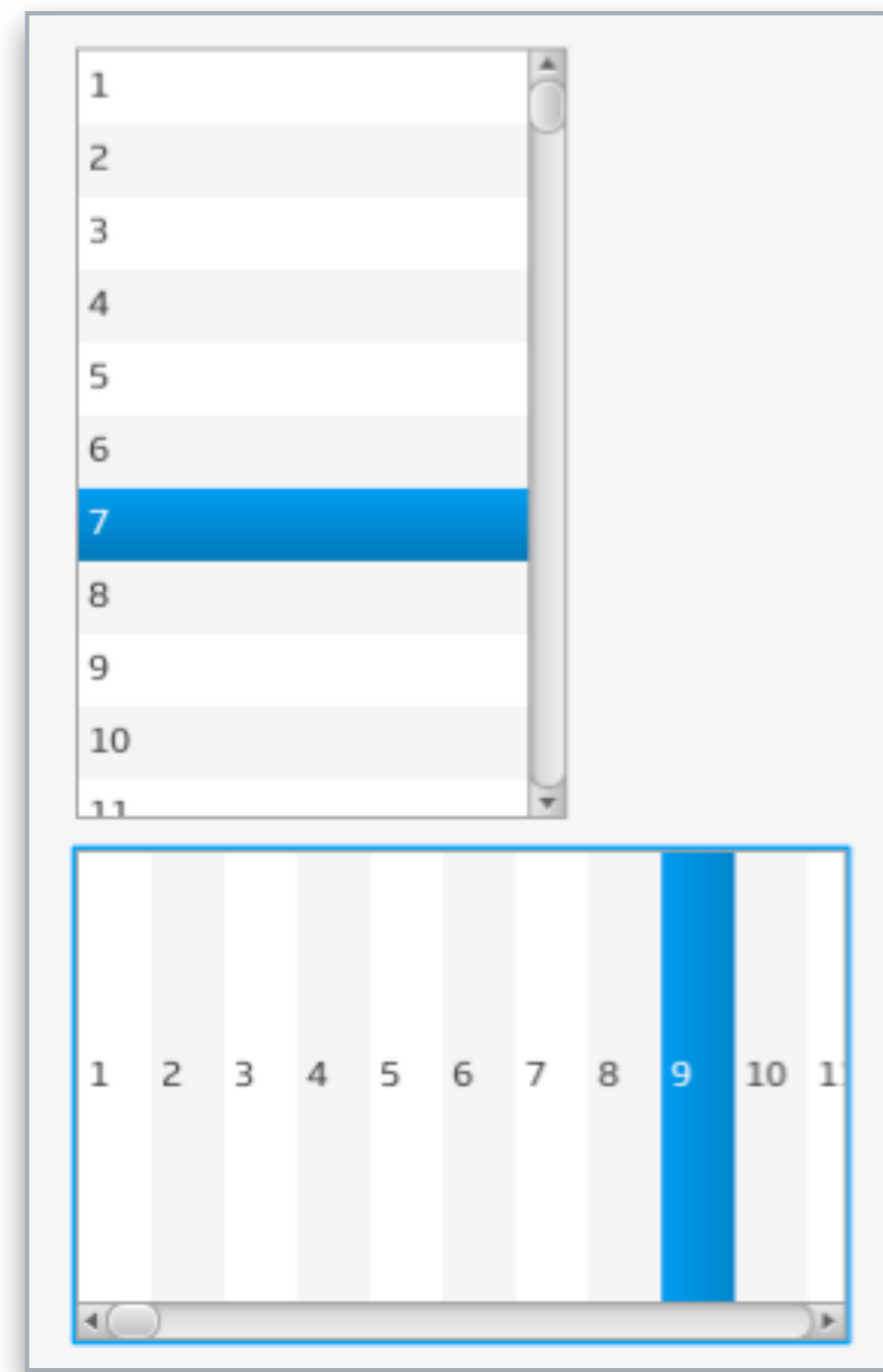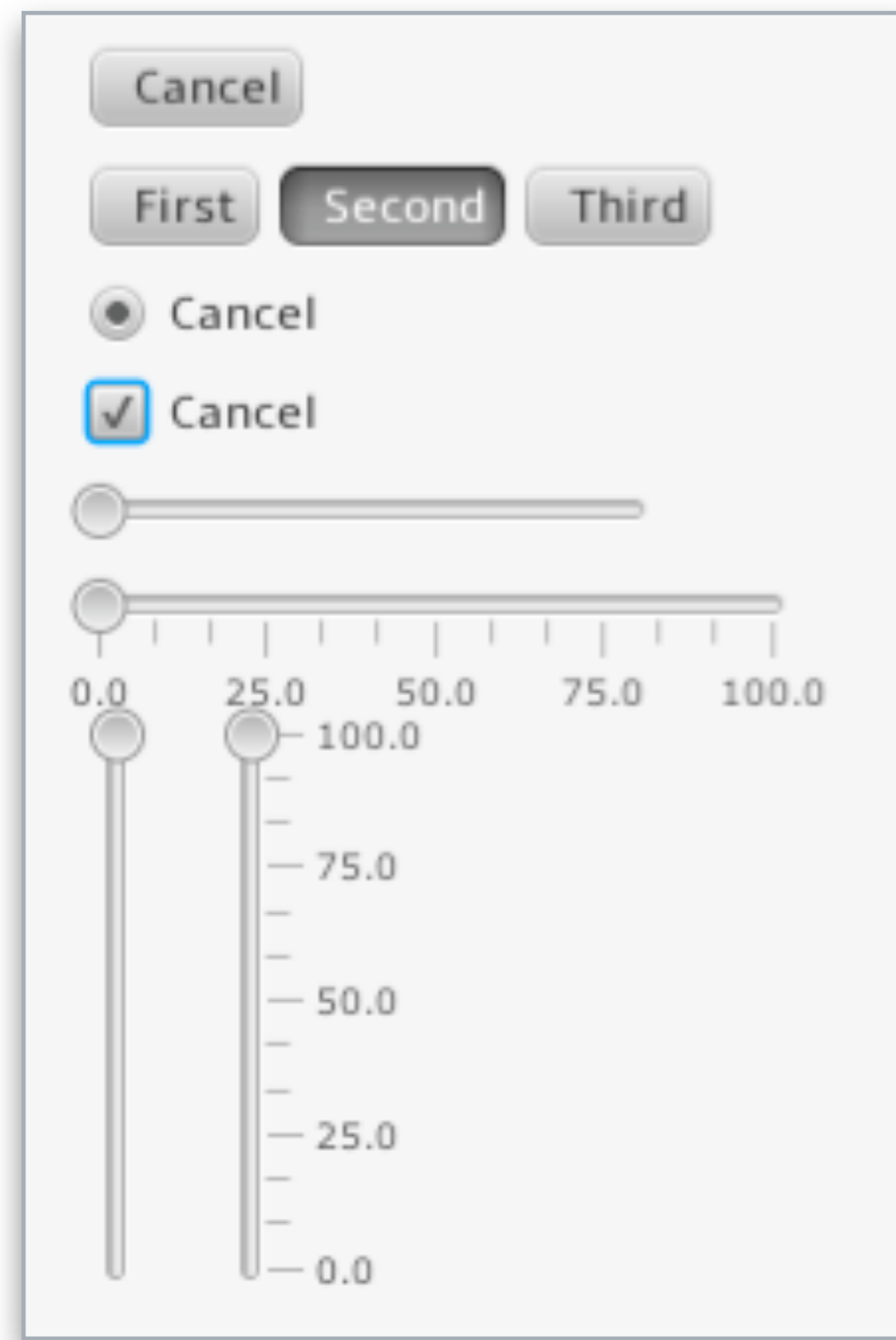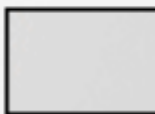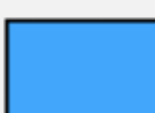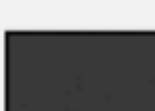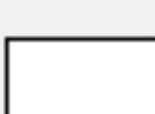
JavaFX "Caspian" Theme

✳ Color Palette: A Selection of Greys

"Base" Colors

Solid Fills	
▮	-fx-accent
▮	-fx-base
▯	-fx-background
▯	-fx-control-inner-background
▮	-fx-focus-color
▮	-fx-dark-text-color
▮	-fx-mid-text-color
▯	-fx-light-text-color

#000000   #0093FF

#292929   #D0D0D3

#FFFFFF   #F4F4F4

ORACLE®

The "Caspian" color palette is mostly grey with some blue accents. A Caspian color can be a solid fill or a gradient fill.
Here you see the six "base" colors of the Caspian color palette.

You'll find the entire set of Caspian colors, including gradients, in the ".scene {"   portion of the caspian.css file.

JavaFX "Caspian" Theme

✳ ## Design Considerations

- Enterprise-ready
- Easy to customize
- Win/Mac/Linux --- complementary yet unique
- Performant

ORACLE®

Caspian was designed to work in enterprise UIs, and to be easily customized for cross-platform use.
So, the controls needed to fit comfortably in Windows, Mac & Linux environments, but still look unique.

The controls also need to render quickly, even on devices with limited capabilities.
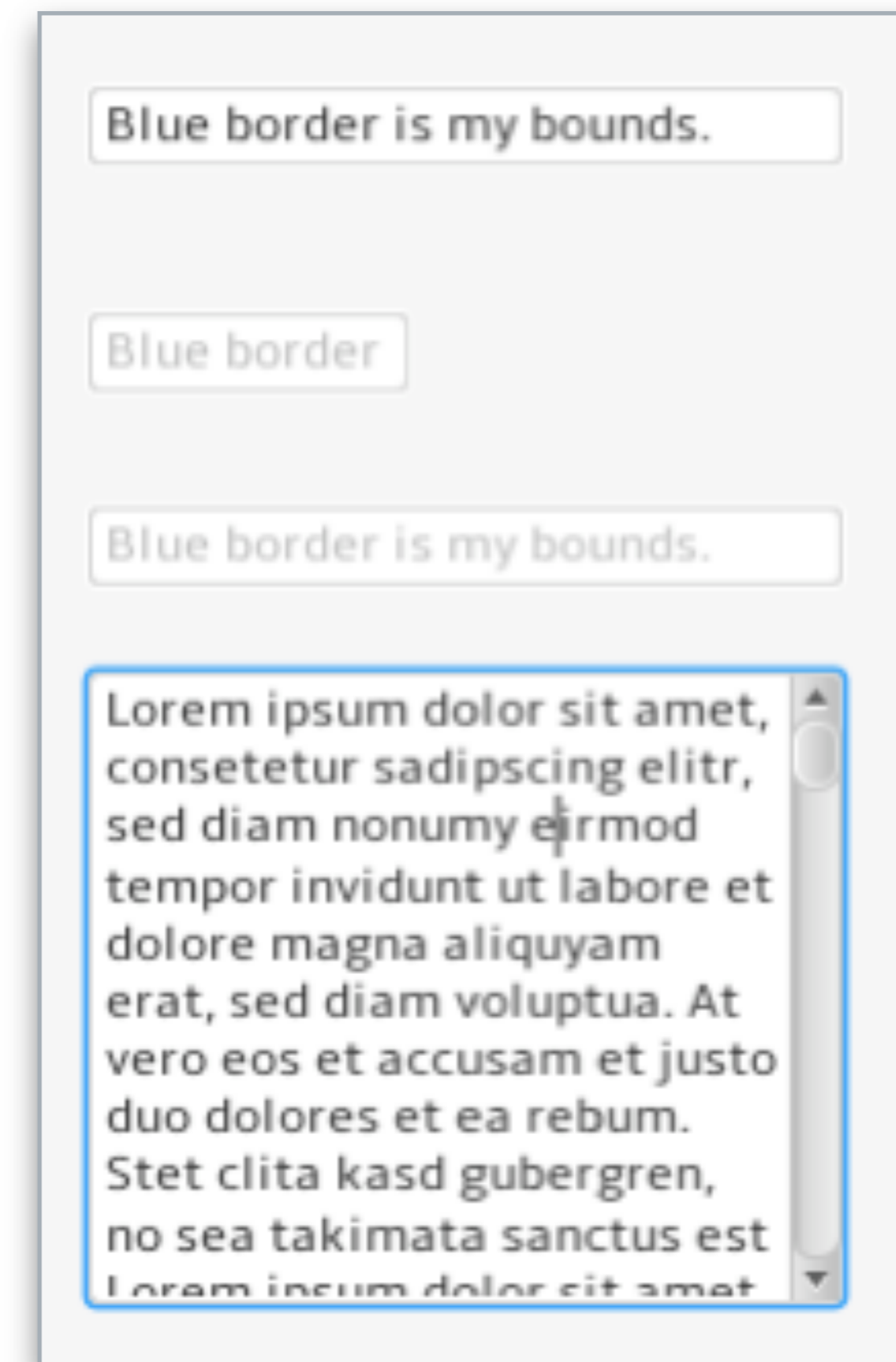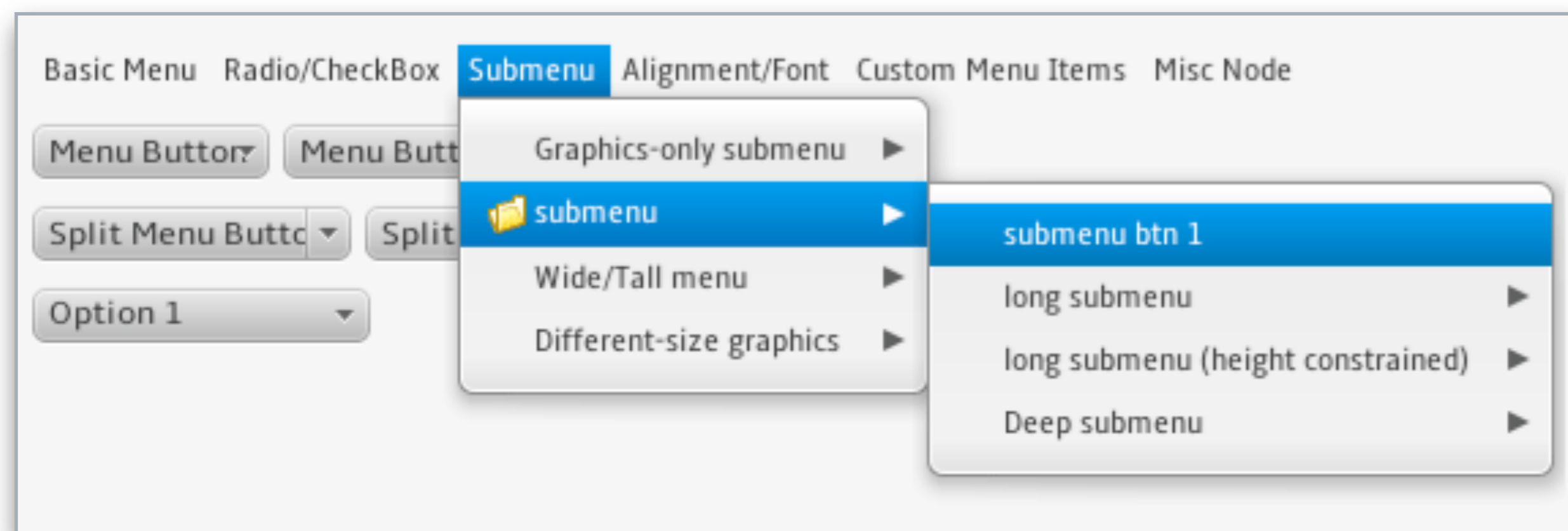
JavaFX "Caspian" Theme

* **2 Caspian Releases**
  - 2009: Hard-coded visuals
  - 2010: CSS-coded visuals and Amble font

the Amble font family

ORACLE®

In our first Caspian release, the look was hard-coded, much of it in private APIs.
That's no problem for all you developers out there, but for visual designers like Tim and me, who don't write Java code --- it was difficult for us to implement our UI designs.

Our second Caspian release, in summer 2010, introduced CSS theming for all the UI Controls.
This is a really powerful addition, and we've had lots of fun exploring ways to use CSS to create and theme the controls.

Another cool feature in this summer's release was a default set of fonts. (the "Amble" font family)
Now you have a font you can bundle with your app and use consistently across all the platforms on which your application will run.

Outline

. The Caspian Theme

. **Styling Overview**

. Desktop Re-theming

. TV: Current and Future

. Q&A

```
.text-box {
    -fx-font: 12pt "Amble-Condensed";
    -fx-padding: 0;
    -fx-background-color: -fx-shadow-highli
fx-mid-border-color, -fx-control-inner-back
    -fx-background-radius: 6, 5 4;
    -fx-background-insets: 0, .5, 1.5;
    -fx-text-fill: -fx-text-inner-color;
    -fx-prompt-text-fill: derive(-fx-control-in
background, 30%);
    -fx-highlight-fill: -fx-accent;
    -fx-highlight-text-fill: white;
    -fx-pannable: true;
    -fx-select-on-focus: true;
}

-fx-accent: #114fd9;
-fx-base: #3f3f3f;
-fx-background: #040404;
-fx-control-inner-background: white;
-fx-focus-color: #31BAED;
-fx-dark-text-color: black;
-fx-mid-text-color: #292929;
-fx-light-text-color: white;
-fx-color: -fx-base;
-fx-hover-base: ladder -fx-base stops (20%,derive(-fx-base,20%)) (35%,derive(-
(50%,derive(-fx-base,40%));
-fx-pressed-base: derive(-fx-base,-40%);
-fx-box-border: #000000;
-fx-text-box-border: ladder -fx-background stops (0.1,black) (0.3,derive(-fx-bac
-fx-shadow-highlight-color:#3B3B3B;
-fx-outer-border: #000000;
-fx-inner-border: linear (0%, 0%) to (0%, 100%) stops (0%, #7A7A7A) (100%, #52
-fx-body-color: linear (0%,0%) to (0%,100%) stops (0,derive(-fx-color,34%)) (1,d
color,-18%));
```

So, how do we build JavaFX UI Controls, and how do we style them?

Rendered Stacked Shapes

Button = 4 stacked shapes...

●-fx-shadow-highlight-color
●-fx-outer-border

●-fx-inner-border

●-fx-body-color

The shapes are gradient vectors

ORACLE®

Let's look at a simple Button Control.
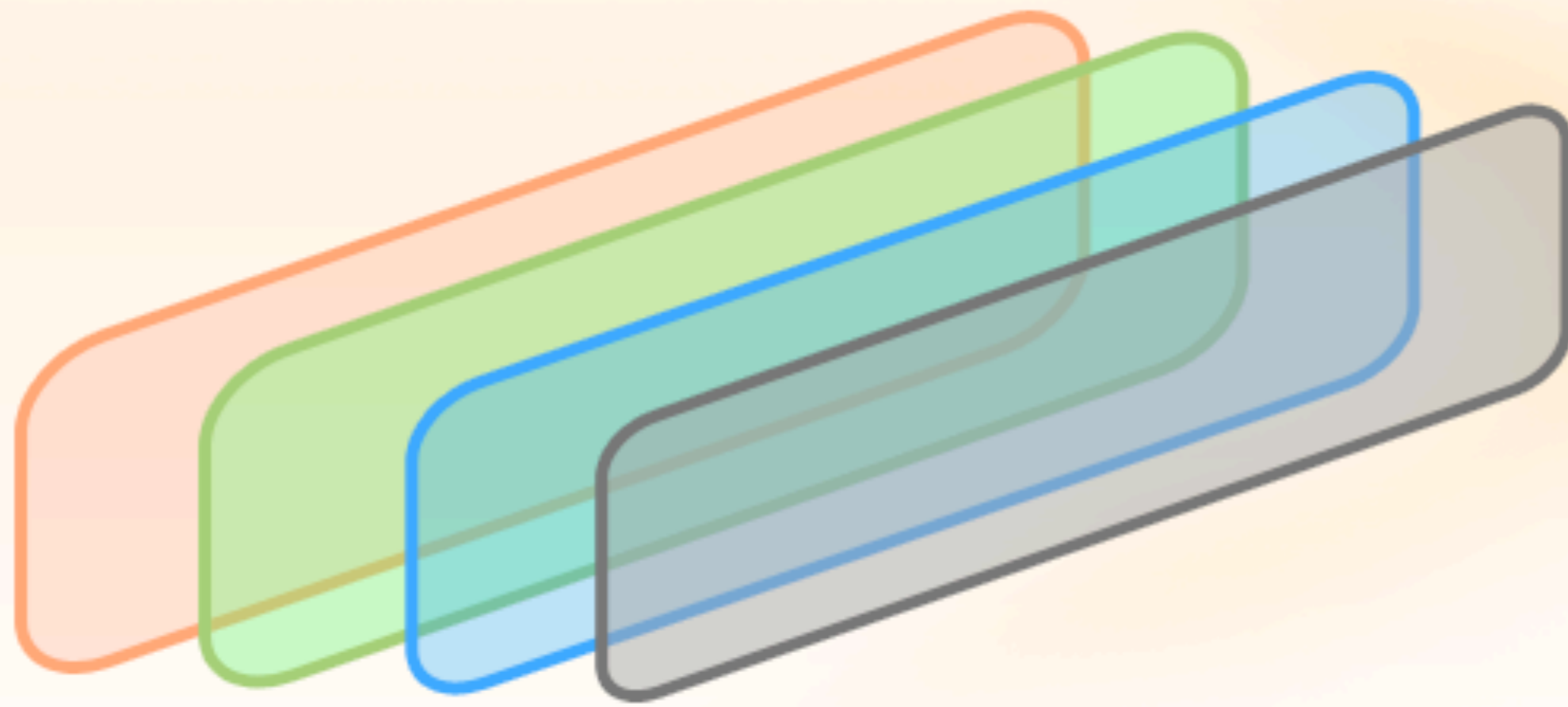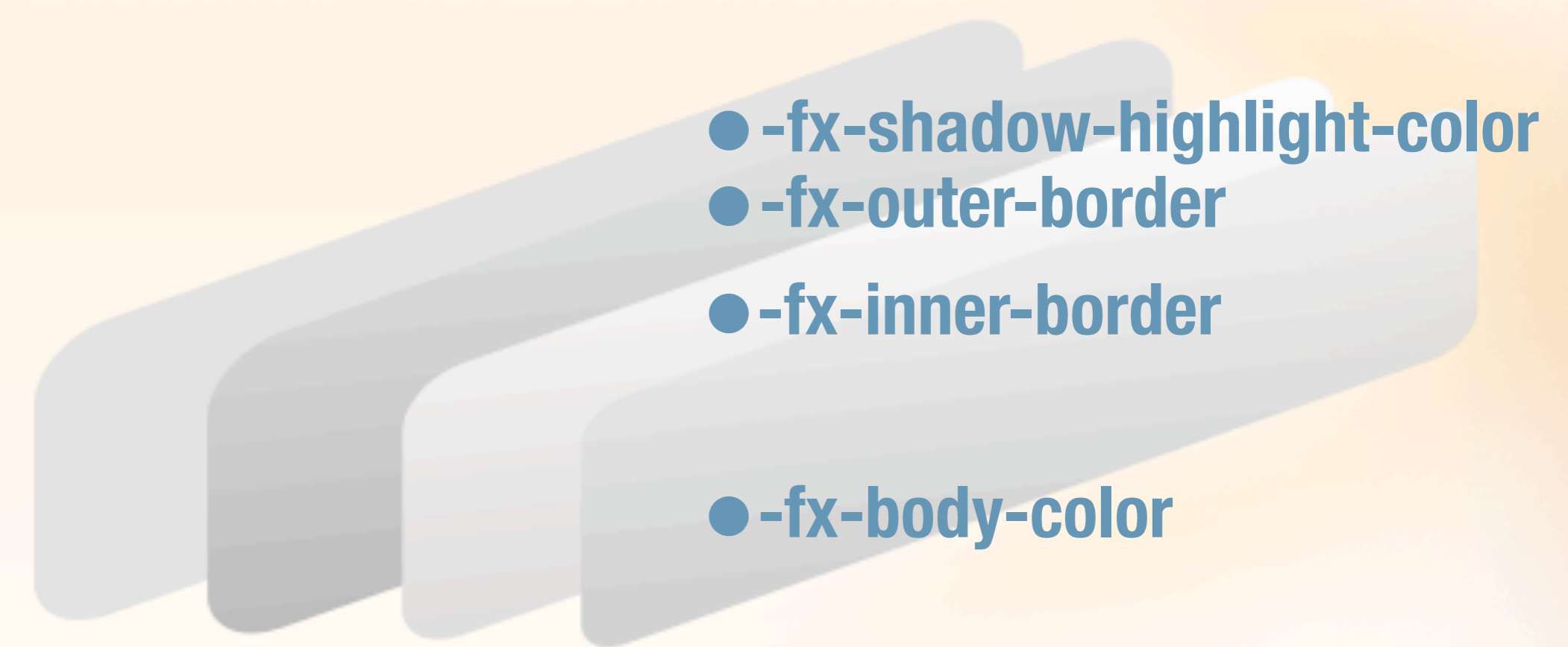
To create it, we stack 4 shapes on top of each other.
These shapes are gradient vectors, and are drawn with code...
They're rendered using CSS StyleClasses, IDs, Properties, and Values.

Centered, Nested Shapes

specification example: background-radius: 5, 5, 4, 3;

A|5 B|5 C|4 D|3

A | 5 pixel radius on all four corners
B | 5 pixel radius on all four corners
C | 4 pixel radius on all four corners
D | 3 pixel radius on all four corners

Visualisierung
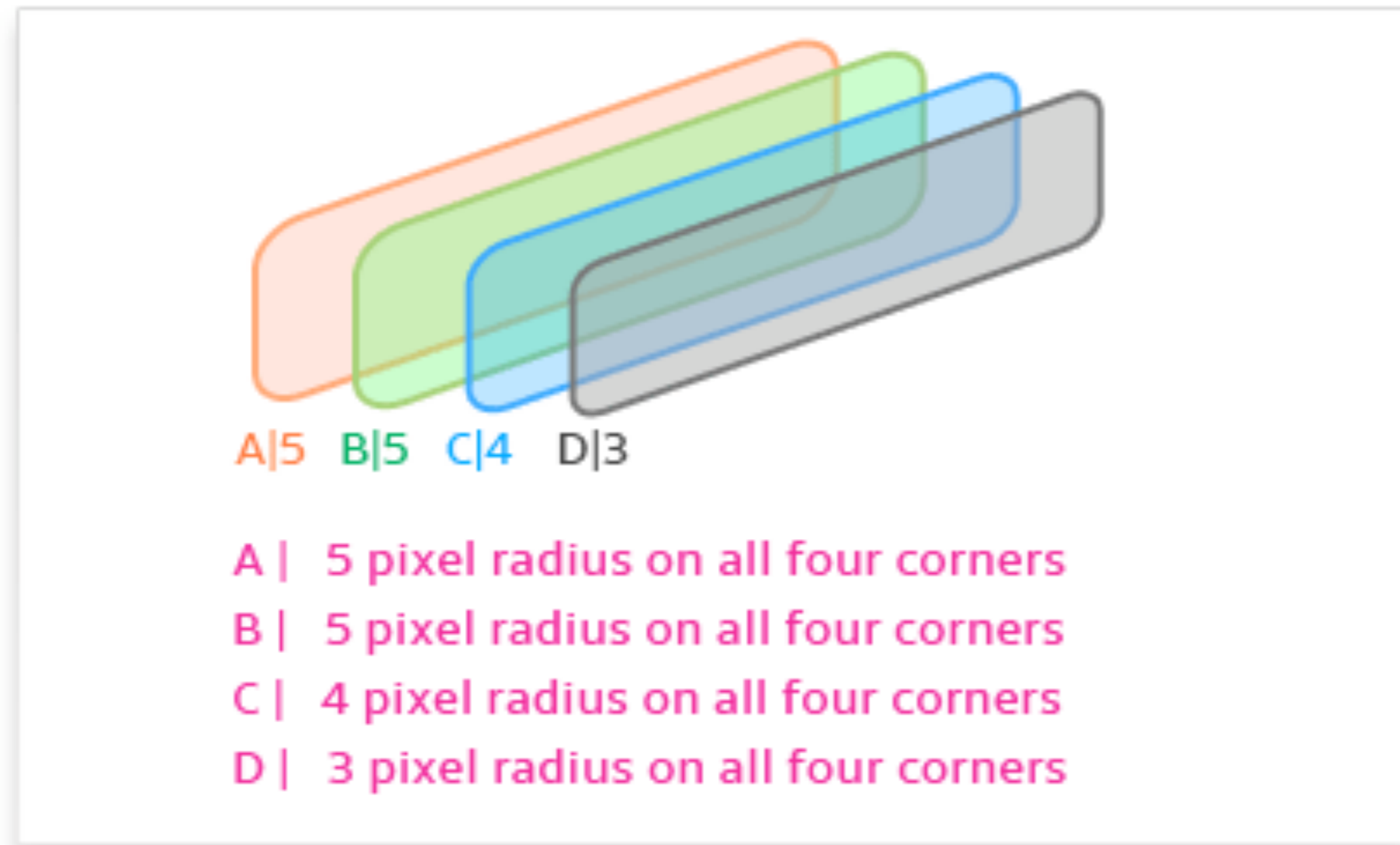
Progressively smaller shapes & radii

Offset bottom shape = shadow/highlight effect

ORACLE®

Each shape is smaller than the one underneath it, and has a smaller corner radius.
The shapes are centered and they "nest" inside each other using a CSS "inset" specification.
The bottom shape is offset below the outer-border shape, and creates our "shadow-highlight" effect.

You might be thinking "Why didn't they just use strokes?"
Well, it turns out that drawing a lot of stroked objects is much slower than drawing the same number of shapes.

"Styling" a UI Control

```
86   *
87   *
88   * CSS Styles for the Button control
89   *
90   *******************************************************************************
91
92   .button {
93       -fx-padding: 3 10 2 10;
94       -fx-background-color: -fx-shadow-highlight-color, -fx-outer-border,
95           -fx-inner-border, -fx-body-color;
96       -fx-background-radius: 5, 5, 4, 3;
97       -fx-background-insets: 0 0 -1 0, 0, 1, 2;
98       -fx-text-fill: -fx-text-base-color;
99       -fx-vpos: CENTER;
100      -fx-graphic-vpos: CENTER;
101  }
102
103  .button:hover {
104      -fx-color: -fx-hover-base;
105  }
106
107  .button:armed {
108      -fx-color: -fx-pressed-base;
109  }
110
111  .button:focused {
112      -fx-background-color: -fx-focus-color, -fx-outer-border,
113          -fx-inner-border, -fx-body-color;
114      -fx-background-radius: 6.4, 5, 4, 3;
115      -fx-background-insets: -1.4, 0, 1, 2;
116  }
117
118  .button:strong {
119      -fx-base: -fx-accent
120  }
121
122  .button:disabled {
123      -fx-opacity: .4;
124  }
```

"Button"
section
from the
caspian.css
file

ORACLE®

Now let's look at some CSS.
We use a single style sheet named caspian.css to style our JavaFX UI Controls.

Here, I'm showing the CSS for the default Button control.
... if you've written with CSS before, you'll recognize lots of the syntax.

"Styling" a UI Control

```
 87     *
 88     * CSS Styles for the Button control
 89     *
 90     ***********************************************************
 91
 92    .button {
 93        -fx-padding: 3 10 2 10;
 94        -fx-background-color: -fx-shadow-highlight-color, -fx-outer-border,
 95            -fx-inner-border, -fx-body-color;
 96        -fx-background-radius: 5, 5, 4, 3;
 97        -fx-background-insets: 0 0 -1 0, 0, 1, 2;
 98        -fx-text-fill: -fx-text-base-color;
 99        -fx-vpos: CENTER;
100        -fx-graphic-vpos: CENTER;
101    }
102
103    .button:hover {
104        -fx-color: -fx-hover-base;
105    }
106
107    .button:armed
108        -f           x-pressed-base:
109
110
11     on:focused {
       -fx-background-color: -fx-focus-color, -fx-ou  er
           -fx-inner-border, -fx-body-color;
       -fx-background-radius: 6.4, 5, 4, 3;
       -fx-background-insets: -1.4, 0, 1
118
119        -fx-
120    }
121
122    .button:disabled {
123        -fx-opacity: .4;
124    }
```

"Button" section from the caspian.css file

ORACLE

Our JavaFX syntax is based on CSS3, but we've customized some of the properties for our needs.
One of the changes you can see here is our addition of the prefix "dash-fx-dash" to all the property names.

"Styling" a UI Control

* Padding

* Color, including gradients

* Radius

* Insets

* Text-fill color

* Opacity

* Background Image

* Tick-mark-stroke

* Scale-shape

* Pannable

* Icon-gutter

* Cursor

* Echo-char

* ...etc.

ORACLE®

You can style things like padding, color, radius, cell-border-style, and more.
And you can specify these styles for pseudo-states... like Hover, Focused, Pressed, and Disabled.

"Styling" Complex Shapes

✳ Use SVG Code

- Photoshop ---> Illustrator ---> SVG code ---> CSS
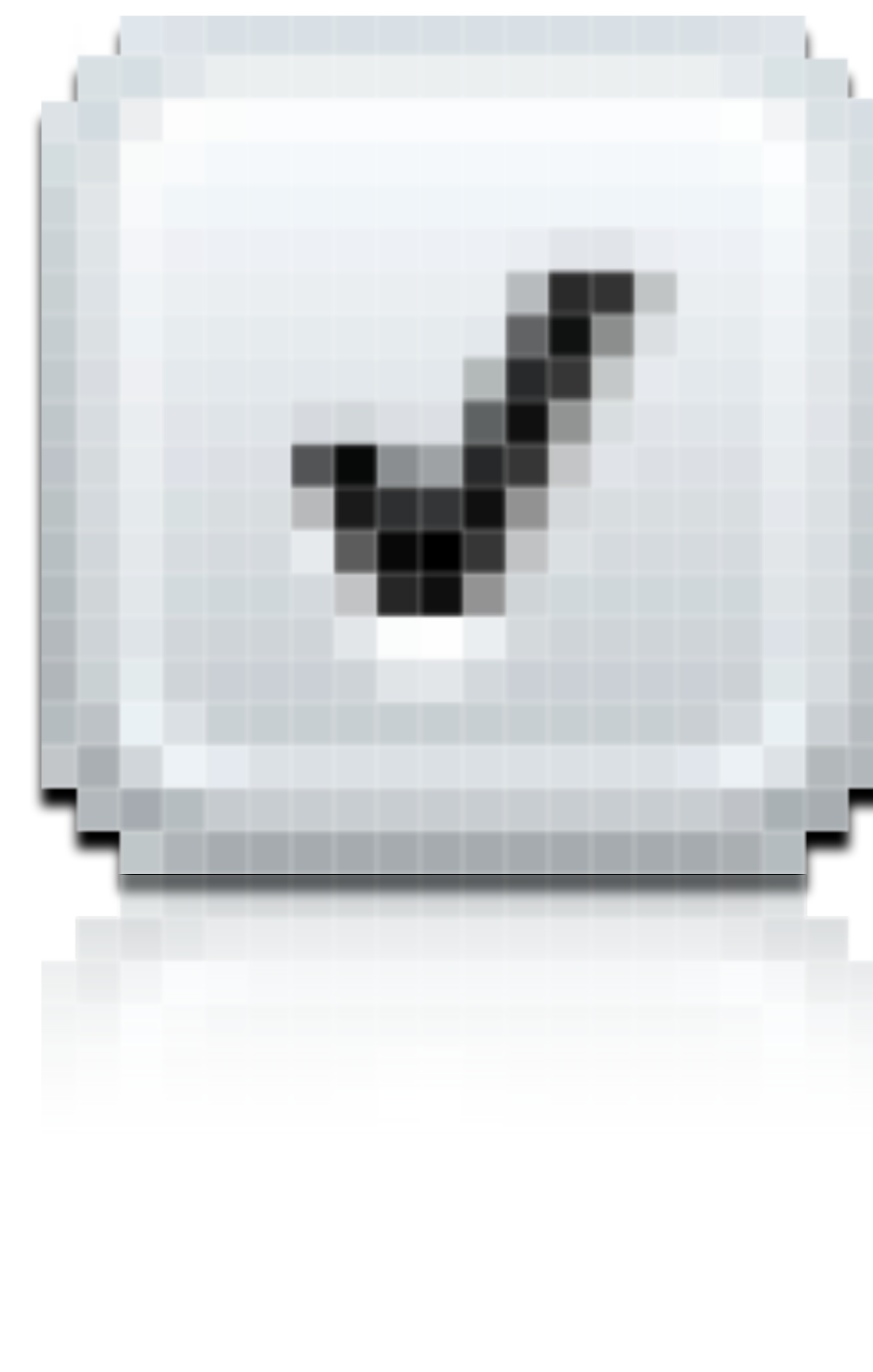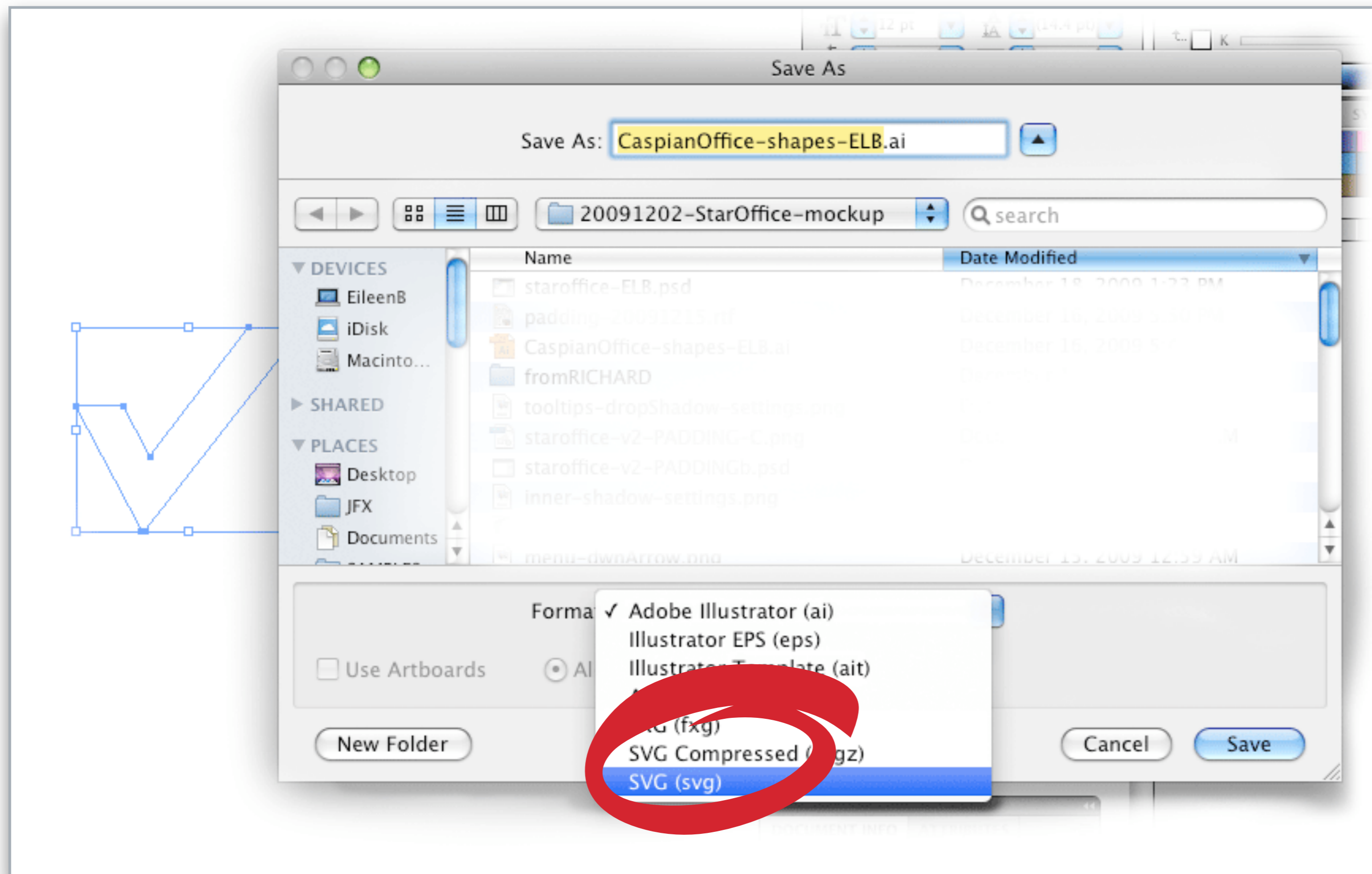
ORACLE®

What about 'a complex' shape, like a checkmark?
Can you style that with CSS?
Well, Yes you can!... and here's one way to do it, using SVG code.

First I create a vector shape in Adobe Photoshop.
(You can also create a shape directly in Adobe Illustrator, or in any other vector drawing program)

I copy the shape and paste it into Illustrator as a "compound shape," and then I save this new file in SVG format.

"Styling" Complex Shapes

✳ ## Use SVG Code

- Photoshop ---> Illustrator ---> SVG code ---> CSS

A new dialog box is displayed, titled "SVG Options".
At the bottom of this dialog box, I select the button "Show SVG Code".
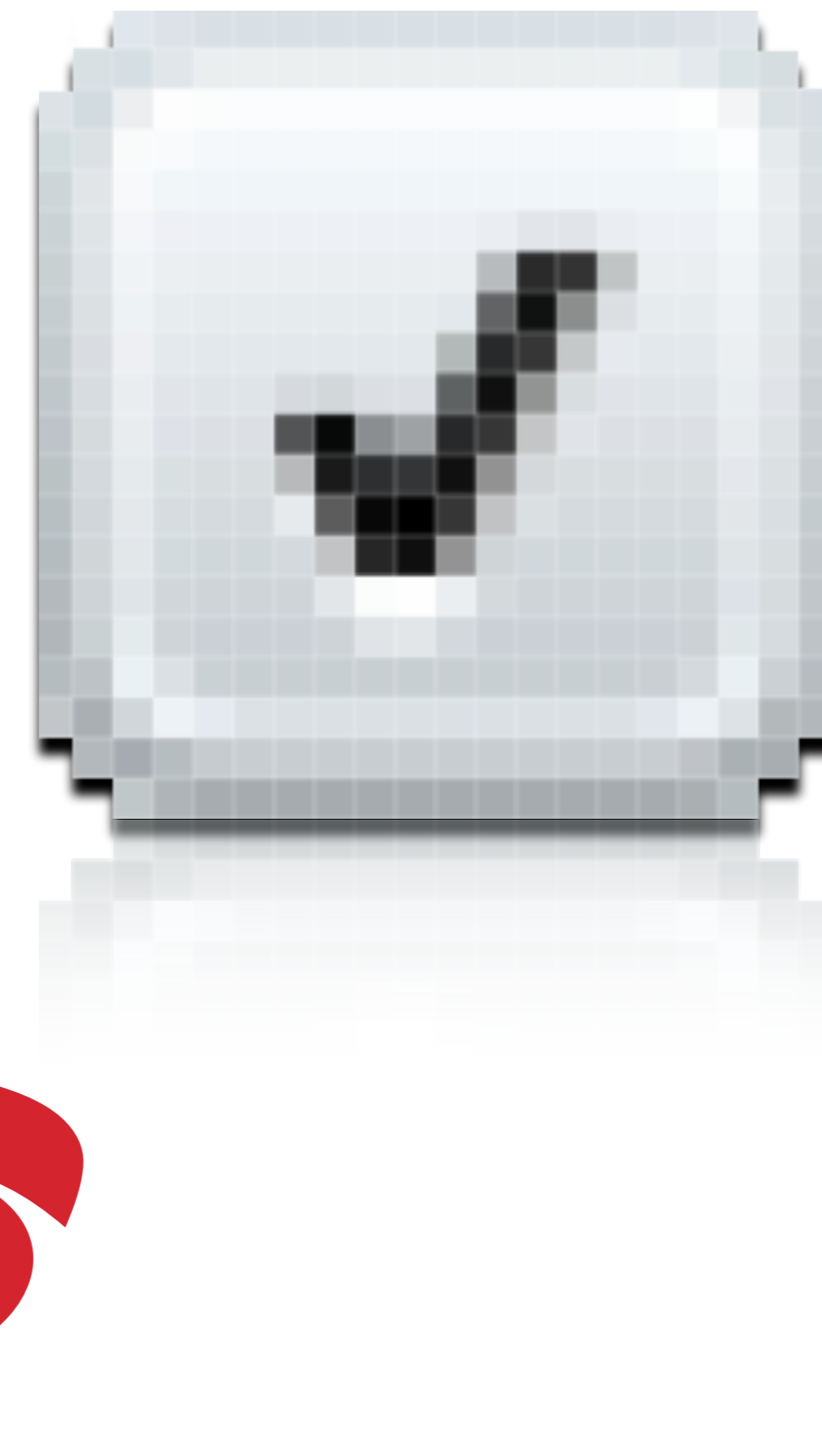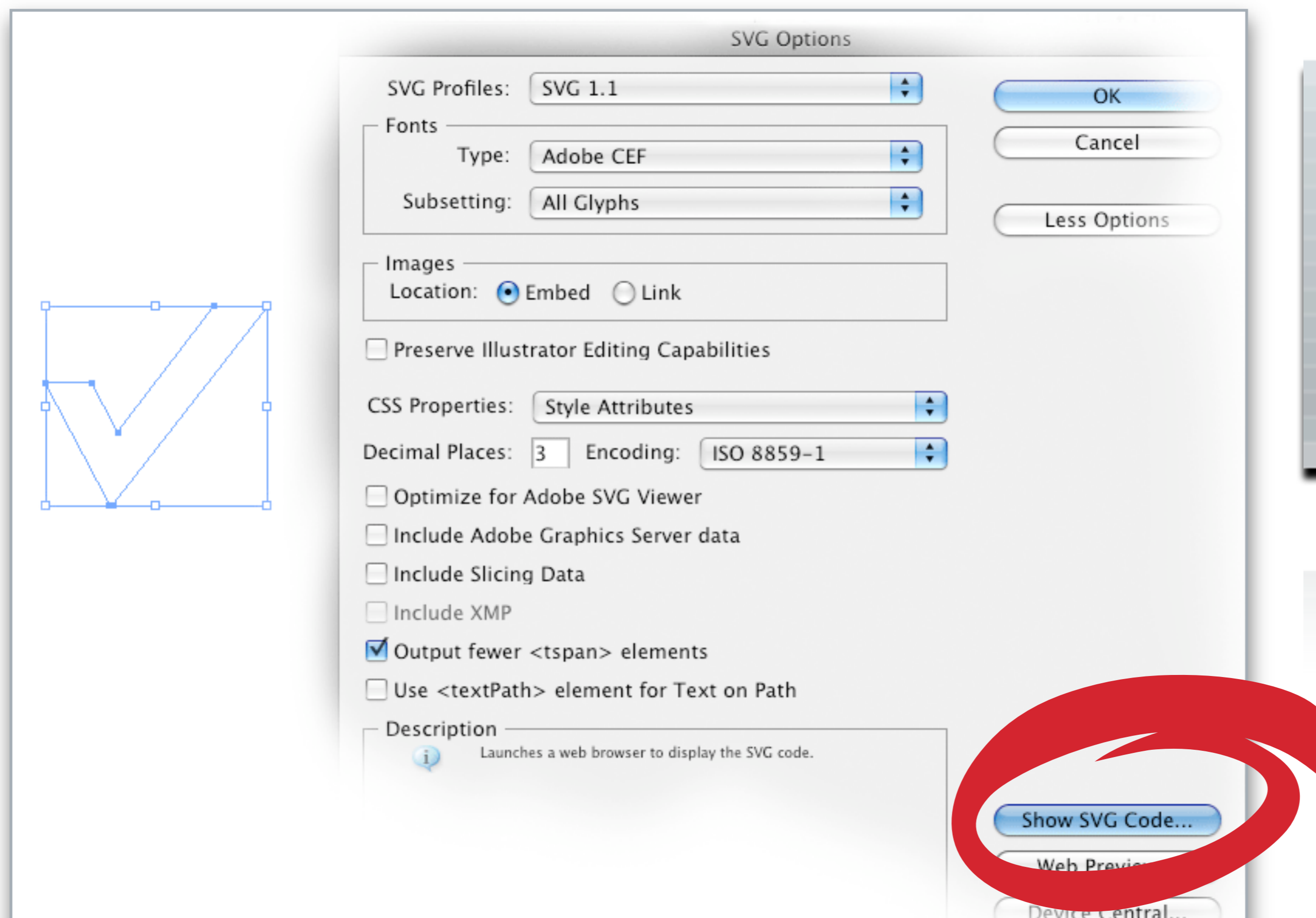
"Styling" Complex Shapes

✳ ## Use SVG Code

- Photoshop ---> Illustrator ---> SVG code ---> CSS

```
⬤ ⬤ ⬤                    📄 ai12857174131.txt

<?xml version="1.0" encoding="iso-8859-1"?>
<!-- Generator: Adobe Illustrator 14.0.0, SVG Export Plug-In . SVG Version: 6.00 Build 43363)  -->
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN" "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
<svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px"
         width="612px" height="792px" viewBox="0 0 612 792" style="enable-background:new 0 0 612 792;" xml:space="preserve">
<path style="fill-rule:evenodd;clip-rule:evenodd;fill:none;" d="M0,4H2L3,6L6,0H8L4,8H2Z"/>
</svg>
```

SVG path for checkmark shape

```
284
285     .check-box.selected *.mark {
286         -fx-shape: "M0,4H2L3,6L6,0H8L4,8H2Z";
287         -fx-background-color:
288             -fx-mark-highlight-color,
289             -fx-mark-color;
290         -fx-background-insets: 1 0 -1 0, 0;
291     }
292
```

ORACLE®

A text-filled window is displayed.
From within this Adobe Illustrator "Show SVG Code" text window, I copy the SVG code string which begins with the letter "M",
 and which is located between the quotation marks near the end of the text output.

Then I paste that SVG code string into my CSS file, as the value for my checkmark's "-fx-shape".

"Styling" Complex Shapes

✳ Use SVG Code

- Photoshop ---> Illustrator ---> SVG code ---> CSS

✳ Examples:
- Check Mark
- Menu Arrows
- Scrollbar Arrows
- Tooltip Page Curl
- Tree Turner
- ... etc.

ORACLE®

Other 'complex' shapes we've styled using SVG code include Scrollbar and ChoiceBox Arrows, the page-curl on our Tooltip, and so on.

Outline

```
.text-box {
    -fx-font: 12pt "Amble-Condensed";
    -fx-padding: 0;
    -fx-background-color: -fx-shadow-highli
fx-mid-border-color, -fx-control-inner-back
    -fx-background-radius: 6, 5 4;
    -fx-background-insets: 0, .5, 1.5;
    -fx-text-fill: -fx-text-inner-color;
    -fx-prompt-text-fill: derive(-fx-control-in
background, 30%);
    -fx-highlight-fill: -fx-accent;
    -fx-highlight-text-fill: white;
    -fx-panrable: true;
    -fx-select-on-focus: true;
}
```

```
-fx-accent: #114fd9;
-fx-base: #3f3f3f;
-fx-background: #040404;
-fx-control-inner-background: white;
-fx-focus-color: #31BAED;
-fx-dark-text-color: black;
-fx-mid-text-color: #292929;
-fx-light-text-color: white;
-fx-color: -fx-base;
-fx-hover-base: ladder -fx-base stops (20%,derive(-fx-base,20%)) (35%,derive(-
(50%,derive(-fx-base,40%));
-fx-pressed-base: derive(-fx-base,-40%);
-fx-box-border: #000000;
-fx-text-box-border: ladder -fx-background stops (0.1,black) (0.3,derive(-fx-bac
-fx-shadow-highlight-color:#3B3B3B;
-fx-outer-border: #000000;
-fx-inner-border: linear (0%, 0%) to (0%, 100%) stops (0%, #7A7A7A) (100%, #52
-fx-body-color: linear (0%,0%) to (0%,100%) stops (0,derive(-fx-color,34%)) (1,d
color,-18%));
```

Although the Caspian look is great, suppose you want to tweak it?

Maybe you need to incorporate some of your company's branding into your application....
Or maybe you're porting your app to a different form factor than the Desktop...

Our JavaFX CSS makes it easy to customize your interface to suit your needs.

Changing the Default Look

✳ ListView: Re-themed for a Touch device

Desktop ListView   Touch Device ListView

List Item 1
List Item 2
List Item 3
List Item 4
List Item 5
List Item 6
List Item 7
List Item 8

List Item 1
List Item 2
List Item 3
List Item 4
List Item 5

ORACLE®

As I mentioned, the Caspian default theme is designed & sized for Desktop use.

But what if you have a touch-screen device?
You'll need to make changes to the UI to accommodate things like finger touch-target sizes.

Here's an example of a ListView...   on the left is the default Caspian appearance for Desktop.
On the right is the same ListView control, re-themed for touch by making changes to the caspian.css file.

Changes include:
-- removing the ListCell even/odd coloring
-- increasing the ListCell font size and the cell-height
-- specifying a color and thickness for the ListCell bottom border, to act as a separator between list items
-- changing the color of the focused 'selection bar'
-- giving a rounded corner effect to the LIstView container, and specifying a border color & thickness for it
-- changing the vertical scrollbar to a scroll "indicator" by removing the increment/decrement arrows, corner piece and borders, and removing the horizontal scrollbar
-- restyling the scrollbar thumb and track

Ways to Implement JavaFX Theming

✳ Change the "base set" of Caspian theme colors

✳ Change specs for individual controls inside the "caspian.css" file

✳ Add new styleClasses to the "caspian.css" and ".fx" files

✳ Create an external ".css" file in which to make customizations

✳ etc.

ORACLE®

There are several ways you can implement theming in JavaFX...

The easiest way to get a new theme is to change the "base set" of Caspian colors.
You can also change individual controls...
You can add new styleClasses...
You can create an external style sheet for your theming...

And that's what we did for the demo I'll show you next.

Changing the Default Look ~ a Demo

Original application screen
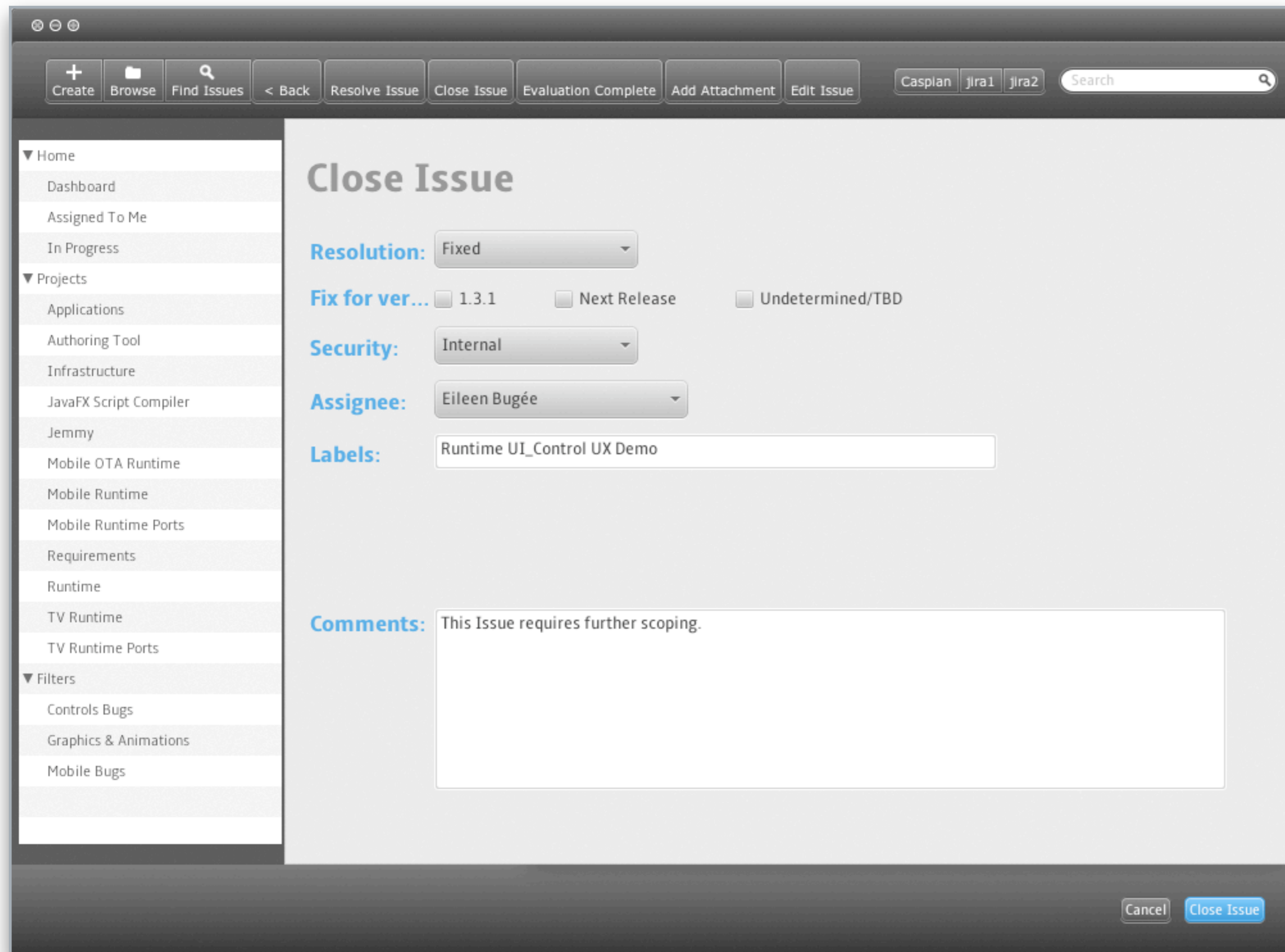
Photoshop mock-up
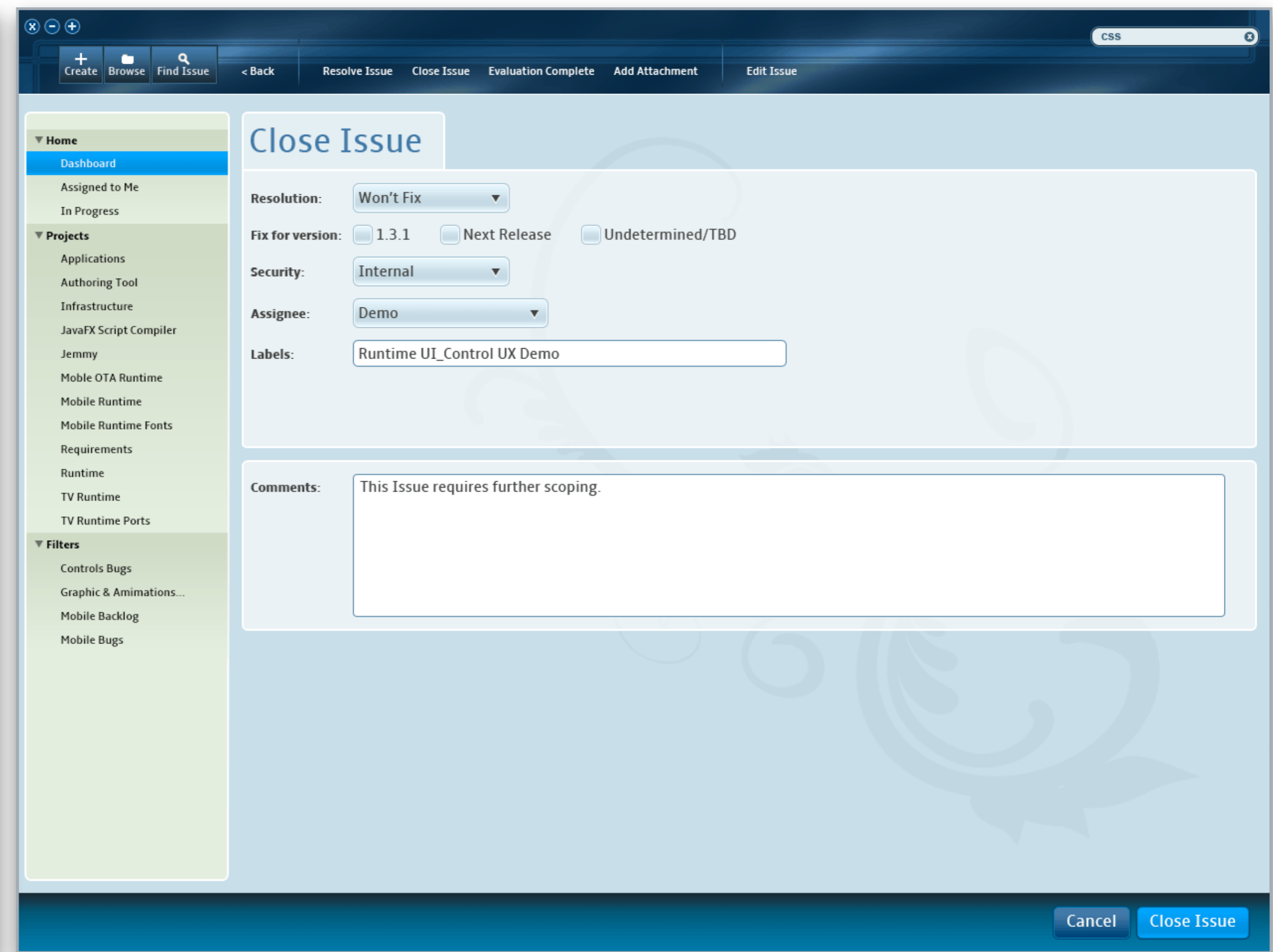
I'm going to show you how I used CSS to re-theme this application screen.
It's from a working version of the JIRA Dashboard app, written in JavaFX by one of Oracle's developers.

On the left you can see the Caspian-themed screen ...
And on the right is my Adobe Photoshop mock-up, with my new theme.

Let's see how close we can get to that, using CSS...

<--------- The next section of slides represents a live demo which was given at JavaOne 2010. --------->

Toolbar

Default Caspian CSS

```
/**********************************
 * CSS Styles for the Toolbar control
 **********************************/


.tool-bar:vertical {
    -fx-padding: 2;
    -fx-spacing: 2;
}

.tool-bar:horizontal {
    -fx-padding: 2;
    -fx-spacing: 2;
}
```

CSS in External Style Sheet

```
/*************************
  Section 1 - Toolbar & Main Background
 *************************/

/* ====== MAIN WINDOW background ============ */
#main-window {
    -fx-background-color: #3e3e3e;
    -fx-background-radius: 6 6 0 0;
}

/* ====== used for Min/Max/Close ====== */
frame-button {
    -fx-background-color: #616161, #464646, #505050;
    -fx-background-insets: 0,1,2;
    -fx-background-radius: 6,5,4;
    -fx-padding: 2;
}
.frame-button Rectangle {
    -fx-fill: #949494;
}

/* ====== SEGMENT BUTTONS - Create/Browse/Find ======= */
#left-segement {
    -fx-background-radius: 5 0 0 5 , 5 0 0 5, 4 0 0 4, 3 0 0 3;
    -fx-background-insets: 0 0 -1 0, 0, 1 0 1 1, 2 1 2 2;
}
#center-segement {
    -fx-background-radius: 0, 0, 0, 0;
    -fx-background-insets: 0 0 -1 0, 0, 1 0 1 1, 2 1 2 2;
}
#right-segement {
    -fx-background-radius: 0 5 5 0 , 0 5 5 0, 0 4 4 0, 0 3 3 0;
}

/* ====== TOOLBAR ====== */
#toolbar {
    -fx-padding: 2 10 2 10;
    -fx-base: #434343;
    -fx-background: #434343;
    -fx-background-color:  linear (0%,0%) to (0%,100%) stops (0%,
#6f6f6f) (100%, #4b4b4b),
                linear (0%,0%) to (0%,100%) stops (0%, #595959)
(100%, #282828);
    -fx-background-radius: 5 5 0 0, 4 4 0 0;
    -fx-background-insets: 0, 1 1 0 1;
}
#toolbar *.button {
    -fx-font: 10pt "Verdana";
    -fx-padding: 5 6 3 6;
    -fx-graphic-text-gap: 2;
    -fx-graphic-hpos: center;
    -fx-graphic-vpos: top;
}
```

Caspian-themed Toolbar

ORACLE®

For my demo... I'm using NetBeans 6.9 and JavaFX 1.3.1 to run the JIRA Dashboard app. I'll make changes to our external style sheet by typing into it within the NetBeans editing pane.
Then I'll rebuild the app and run it in NetBeans to see the result of my changes.

The first area I want to change is the Toolbar, at the top of the application screen.
As I mentioned earlier, we've added an external style sheet to override, and add to, some of the existing caspian.css styleClasses.

In the center of this slide is the CSS for the default Caspian Toolbar.
On the right is the CSS of our external style sheet. It extends the default Toolbar styling, and it overrides values set for same-named properties in the caspian.css file.
The external style sheet 'code' on this slide adds specs for the Main Window, Min/Max/Close buttons, and all the buttons aligned in a row on the toolbar.
This style sheet also specifies the color & appearance of the Toolbar itself.

Toolbar

CSS in External Style Sheet

```
/**************************
    Section 1 - Toolbar & Main Background
   **************************/

/* ===== MAIN WINDOW background ============= */
#main-window {
    -fx-background-color: #3e3e3e;
    -fx-background-radius: 6 6 0 0;
}

/* ===== used for Min/Max/Close ====== */
.frame-button {
    -fx-background-color: #616161, #464646, #505050;
    -fx-background-insets: 0,1,2;
    -fx-background-radius: 6,5,4;
    -fx-padding: 2;
}
.frame-button Rectangle {
    -fx-fill: #949494;
}

/* ===== SEGMENT BUTTONS - Create/Browse/Find ====== */
#left-segement {
    -fx-background-radius: 5 0 0 5 , 5 0 0 5, 4 0 0 4, 3 0 0 3;
    -fx-background-insets: 0 0 -1 0, 0, 1 0 1 1, 2 1 2 2;
}
#center-segement {
    -fx-background-radius: 0, 0, 0, 0;
    -fx-background-insets: 0 0 -1 0, 0, 1 0 1 1, 2 1 2 2;
}
#right-segement {
    -fx-background-radius: 0 5 5 0 , 0 5 5 0, 0 4 4 0, 0 3 3 0;
}

/* ===== TOOLBAR ====== */
#toolbar {
    -fx-padding: 2 10 2 10;
    -fx-base: #434343;
    -fx-background: #434343;
    -fx-background-color:  linear (0%,0%) to (0%,100%) stops (0%, #6f6f6f) (100%, #4b4b4b),
                           linear (0%,0%) to (0%,100%) stops (0%, #595959) (100%, #282828);
    -fx-background-radius: 5 5 0 0, 4 4 0 0;
    -fx-background-insets: 0, 1 1 0 1;
}
```

Changes to CSS

```
/**************************
    Section 1 - Toolbar & Main Background
   **************************/

/* ===== MAIN WINDOW background ============= */
#main-window {
    -fx-background-color: #bdd6e2;
    -fx-background-radius: 6 6 0 0;
}

/* ===== used for Min/Max/Close ====== */
.frame-button {
    -fx-background-color: transparent;
    -fx-background-insets: 0,1,2;
    -fx-background-radius: 6,5,4;
    -fx-padding: 2;
}
.frame-button Rectangle {
    -fx-fill: white;
}

/* ===== SEGMENT BUTTONS - Create/Browse/Find ====== */
#left-segement {
    -fx-background-radius: 5 0 0 5 , 5 0 0 5, 4 0 0 4, 3 0 0 3;
    -fx-background-insets: 0 0 -1 0, 0, 1 0 1 1, 2 1 2 2;
    -fx-background-color: linear (0%,0%) to (0%,100%) stops (0%, #336699) (100%, #15232d);
    -fx-border-color: #2b5974;
    -fx-border-width: 1;
}
#center-segement {
    -fx-background-radius: 0, 0, 0, 0;
    -fx-background-insets: 0 0 -1 0, 0, 1 0 1 1, 2 1 2 2;
    -fx-background-color: linear (0%,0%) to (0%,100%) stops (0%, #336699) (100%, #15232d);
    -fx-border-color: #2b5974;
    -fx-border-width: 1;
}
#right-segement {
    -fx-background-radius: 0 5 5 0 , 0 5 5 0, 0 4 4 0, 0 3 3 0;
    -fx-background-color: linear (0%,0%) to (0%,100%) stops (0%, #336699) (100%, #15232d);
    -fx-border-color: #2b5974;
    -fx-border-width: 1;
}

/* ===== SEGMENT BUTTONS - Back/Resolve/Close/etc. ======= */
#action-left-segement {
    -fx-background-color: transparent;
    -fx-background-color: linear (0%,0%) to (0%,100%) stops (0%, #142129) (50%, #326496) (100%, #142129);
    -fx-border-width: 0 2 0 0;
}
#action-center-segement {
    -fx-background-color: transparent;
    -fx-border-width: 0;
}
#action-right-segement {
    -fx-background-color: transparent;
    -fx-background-color: linear (0%,0%) to (0%,100%) stops (0%, #142129) (50%, #326496) (100%, #142129);
    -fx-border-width: 0 0 0 2;
}

/* ===== TOOLBAR ====== */
#toolbar_top {
    -fx-padding: 2 10 2 10;
    -fx-base: #434343;
    -fx-background: #434343;
    -fx-background-image: "Header_1261x26-8bt.png";
    -fx-background-radius: 5 5 0 0, 4 4 0 0;
    -fx-background-insets: 0, 1 1 0 1;
}
#toolbar {
    -fx-padding: 2 10 2 10;
    -fx-base: #434343;
    -fx-background: #434343;
    -fx-background-image: "Header_1261x60-8bt.png";
    -fx-background-radius: 0 0;
    -fx-background-insets: 0, 1 1 0 1;
}
#toolbar *.button {
    -fx-font: 10pt "Verdana";
    -fx-padding: 5 6 3 6;
    -fx-graphic-text-gap: 2;
    -fx-graphic-hpos: center;
    -fx-graphic-vpos: top;
}
/* ===== SEARCH BOX =================================== */
#search-box-textbox {
    -fx-background-color: #396789, #142129, #DAE8EE;
}
```

Caspian Theme
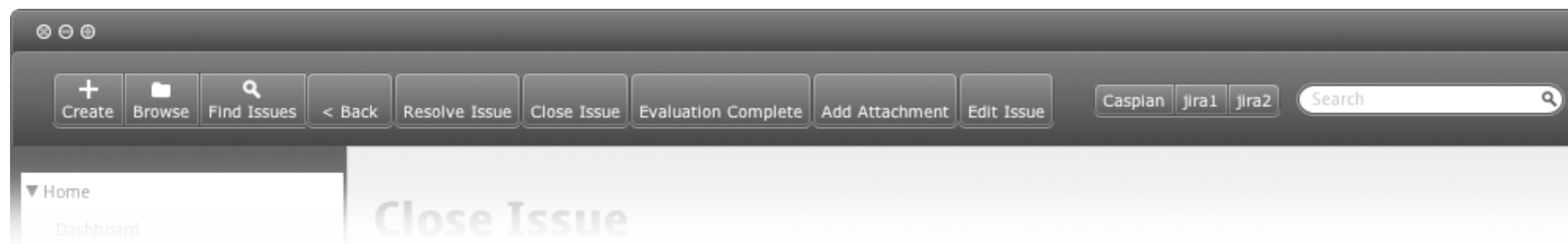
My New Theme
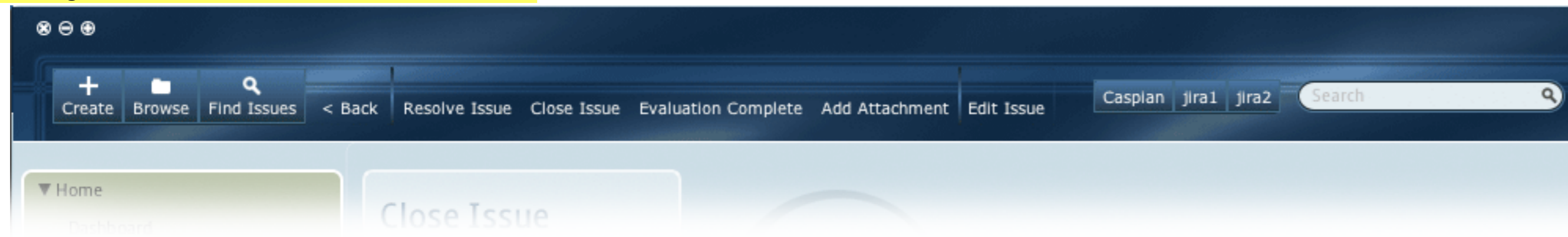
ORACLE®

In order to get the Toolbar area closer to my Photoshop mock-up, I need to change some of the CSS specifications in our external style sheet.

In the center of this slide is the CSS for the 'original/Caspian-themed' Toolbar.
On the right you see the same base CSS, but with my additions and revisions highlighted.

Here's what I changed:
-- replaced the main window's grey background color with light blue (-fx-background-color: #bdd6e2;)
-- changed the Min/Max/Close buttons' background color to transparent, and their icons to white (frame-button) (frame-button Rectangle)
-- replaced the inherited grey colors of the "Create/Browse/Find" Toolbar buttons with blue gradients and a blue border (SEGMENT BUTTONS - Create/Browse/Find)
-- removed the background color completely from the "Back/Resolve/Close/etc" Toolbar buttons (-fx-background-color: transparent;)
-- styled one vertical border of these buttons with a gradient to act as a separator between button groups (#action-left-segement) (#action-right-segement)
     (I am misspelling "segment" as "segement" in the CSS to follow the original engineer's misspelling, so as to not break the build)
-- replaced the grey gradient backgrounds on #toolbar_top and #toolbar with background-images. These 2 images "butt" up against each other, and combine to give the look of a single background,
-- specified new colors for the SearchBox

Toolbar ~ Re-themed

Here's the result of my NetBeans build, with my Toolbar revisions added to our external style sheet...
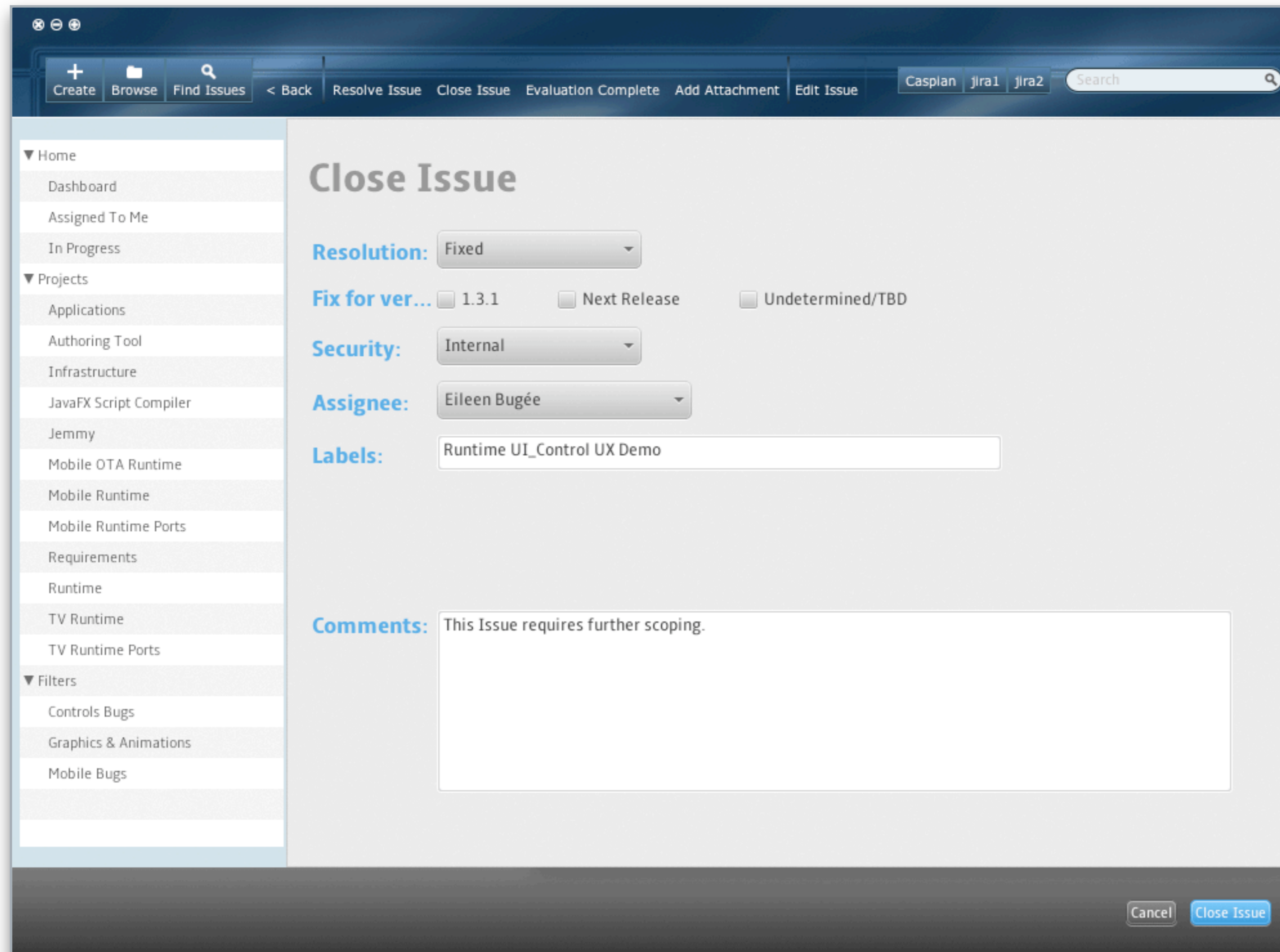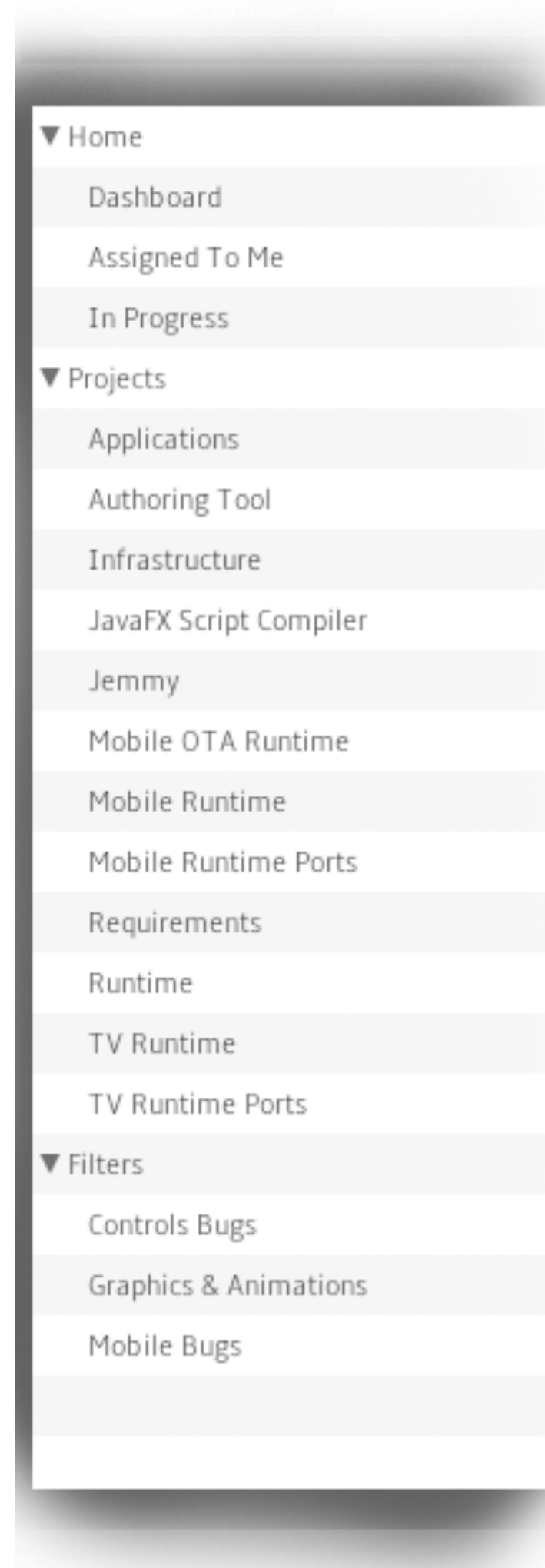
TreeView

Caspian-themed TreeView

```
/***************************************
 * CSS Styles for the TreeView control including the TreeCell
 ***************************************/
.tree-view {
    -fx-padding: 1;
    -fx-background-color: pink, yellow;
    xxx-fx-background-color: -fx-box-border,
-fx-control-inner-background;
    -fx-background-insets: 0,1;

}
.tree-view:focused {
    -fx-padding: 1;
    -fx-background-color: pink, yellow;
    xxx-fx-background-color: -fx-focus-color,-fx-box-border,
-fx-control-inner-background;
    -fx-background-insets: -1.4, 0, 1;
    -fx-background-radius: -1.4, 0, 0;
}
.tree-view > * > .scroll-bar:vertical{
    -fx-background-insets: 0, 0 0 0 1;
    -fx-padding: -1 -1 -1 0;
}
.tree-view > * > .scroll-bar:horizontal{
    -fx-background-insets: 0, 1 0 0 0;
    -fx-padding: 0 0 -1 -1;
}
.tree-cell {
    -fx-cell-size: 24;
    -fx-indent: 6;
    -fx-padding: 3;
    -fx-background-color: pink, yellow;
    xxx-fx-background-color: -fx-control-inner-background;
}
.tree-cell .label {
    -fx-padding: 0 0 0 3;
}
.tree-cell:odd {
    -fx-background-color: pink, yellow;
    xxx-fx-background-color:
derive(-fx-control-inner-background,-5%);
}
.tree-cell:filled:selected {
    -fx-background: -fx-accent;
    -fx-background-color: pink, yellow;
    xxx-fx-background-color: -fx-selection-bar;
    -fx-text-fill: -fx-selection-bar-text;
}
.tree-disclosureNode {
    -fx-background-color: pink, yellow;
    xxx-fx-background-color: -fx-mark-color;
    -fx-padding: 4;
    -fx-shape: "M 0 -4 L 8 0 L 0 4 z";
}
.tree-cell:filled:selected .tree-disclosureNode {
    -fx-background-color: pink, yellow;
    xxx-fx-background-color: -fx-selection-bar-text;
}
```

```
/***************************************
 * CSS Styles for the TreeView control including the TreeCell
 ***************************************/
/* nothing added to default */
```

ORACLE®

The next area I want to change is the TreeView, on the left side of the application screen.

In the center of this slide is the CSS for the default Caspian TreeView.
Next to it is the CSS of our external style sheet for the TreeView. You can see that the engineer used the default TreeView for his JIRA Dashboard application.

However, I'm going to <u>change</u> the TreeView's appearance to match my Photoshop mock-up. To do this, I'll need to add some new CSS 'code' to our external style sheet...
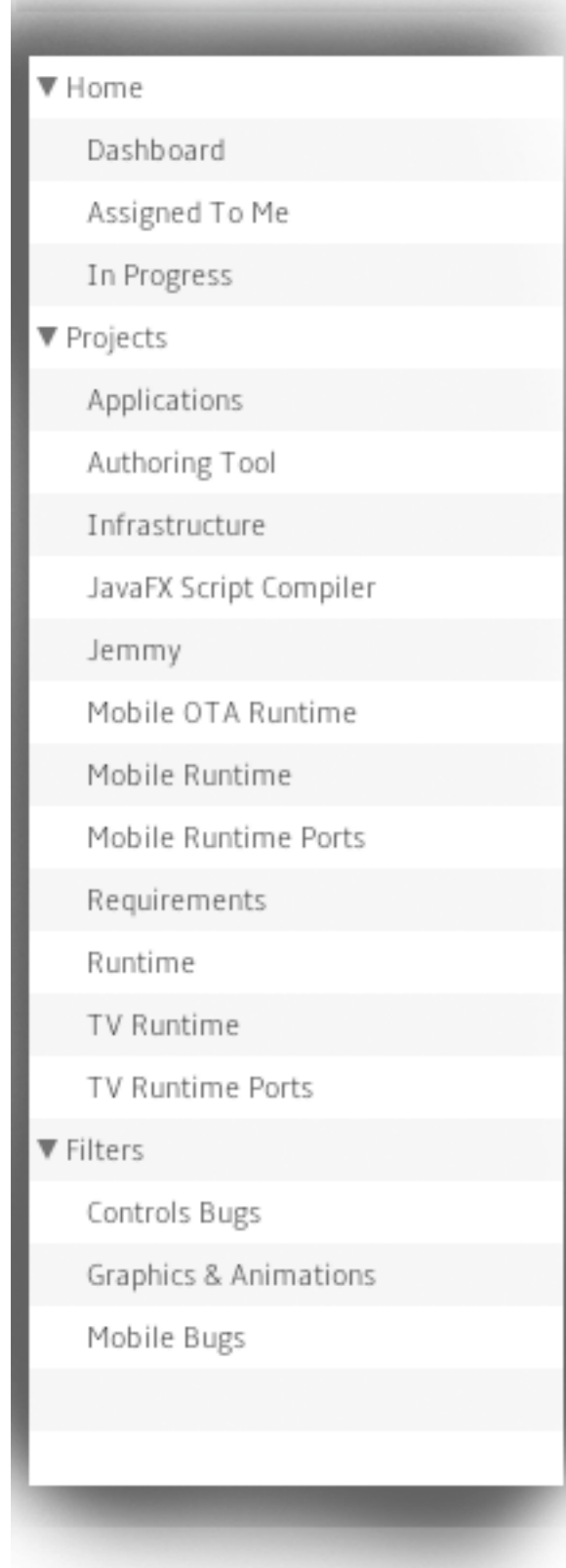
TreeView
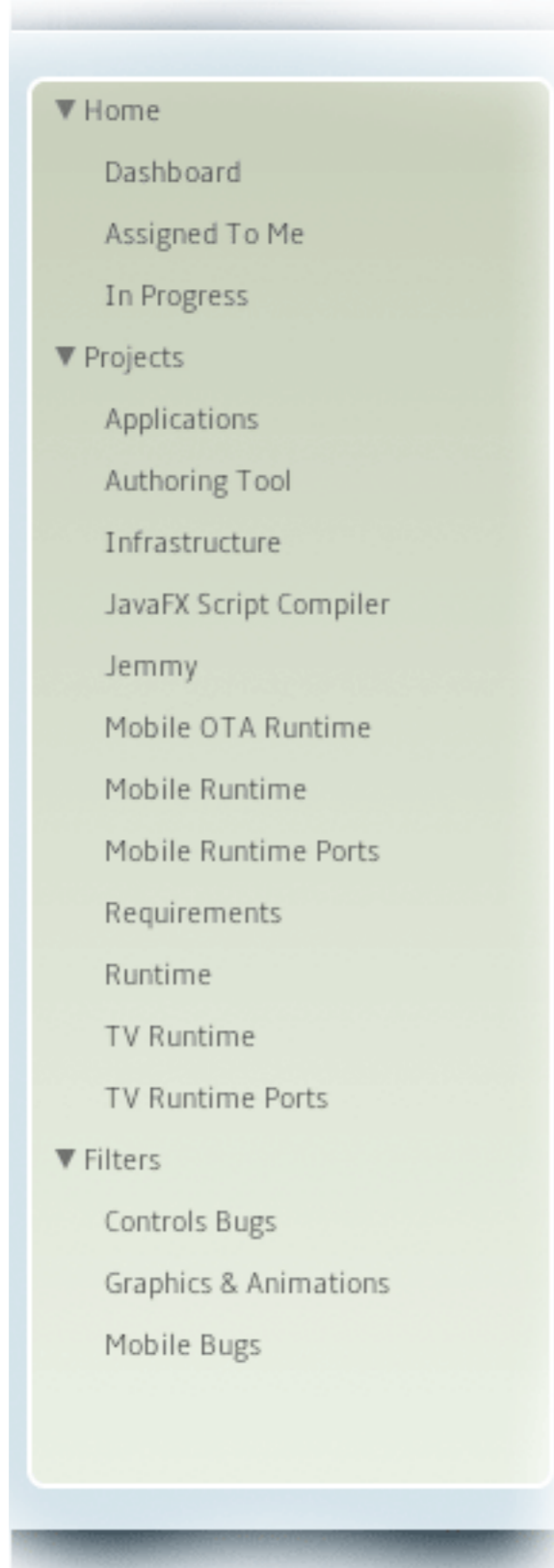
Caspian Theme

My New Theme

Default Caspian CSS

```
/***********************************
 * CSS Styles for the TreeView control including the TreeCell
 ***********************************/
.tree-view {
    -fx-padding: 1;
    -fx-background-color: pink, yellow;
    xxx-fx-background-color: -fx-box-border,
-fx-control-inner-background;
    -fx-background-insets: 0,1;

}
.tree-view:focused {
    -fx-padding: 1;
    -fx-background-color: pink, yellow;
    xxx-fx-background-color: -fx-focus-color,-fx-box-border,
-fx-control-inner-background;
    -fx-background-insets: -1.4, 0, 1;
    -fx-background-radius: -1.4, 0, 0;
}
.tree-view > * > .scroll-bar:vertical{
    -fx-background-insets: 0, 0 0 0 1;
    -fx-padding: -1 -1 -1 0;
}
.tree-view > * > .scroll-bar:horizontal{
    -fx-background-insets: 0, 1 0 0 0;
    -fx-padding: 0 0 -1 -1;
}
.tree-cell {
    -fx-cell-size: 24;
    -fx-indent: 6;
    -fx-padding: 3;
    -fx-background-color: pink, yellow;
    xxx-fx-background-color: -fx-control-inner-background;
}
.tree-cell .label {
    -fx-padding: 0 0 0 3;
}
.tree-cell:odd {
    -fx-background-color: pink, yellow;
    xxx-fx-background-color:
derive(-fx-control-inner-background,-5%);
}
.tree-cell:filled:selected {
    -fx-background: -fx-accent;
    -fx-background-color: pink, yellow;
    xxx-fx-background-color: -fx-selection-bar;
    -fx-text-fill: -fx-selection-bar-text;
}
.tree-disclosureNode {
    -fx-background-color: pink, yellow;
    xxx-fx-background-color: -fx-mark-color;
    -fx-padding: 4;
    -fx-shape: "M 0 -4 L 8 0 L 0 4 z";
}

.tree-cell:filled:selected .tree-disclosureNode {
    -fx-background-color: pink, yellow;
    xxx-fx-background-color: -fx-selection-bar-text;
}
```

New CSS in External Style Sheet

```
/*************************
 ** Section 2 - Tree
 *************************/

.mystyle  {
    -fx-padding: 1;
    -fx-background-color:
linear (0%, 0%) to (0%, 100%)
stops (0%, #adbb9b) (100%, #deebd7);
    -fx-background-insets: 2,2,2,2;
    -fx-border-width: 2 2 2 2;
    -fx-border-color: white;
    -fx-border-radius: 6;
}

.mystyle:focused  {
    -fx-border-color: white;
}


/* ====== TreeItem/TreeCell ===== */
.tree-cell {
    -fx-cell-size: 24;
    -fx-indent: 6;
    -fx-padding: 10;
    -fx-background-color: transparent;
}

.tree-cell .label {
    -fx-padding: 0 0 0 3;
}

.tree-cell:odd {
    -fx-background-color: transparent;
}
```

ORACLE®

In order to get the TreeView to look like my Photoshop mock-up, I need to add some specifications & revise other specs in our external style sheet.

In the center of this slide is the CSS for the 'original/Caspian-themed' TreeView.
On the right you see my additions to our external style sheet, and a couple of revisions to the default caspian.css.

Here's what I changed:
-- created a styleClass for a new background shape (.mystyle). The styleClass "mystyle" was also added to the MainWindow.fx file.
-- styled this background with rounded corners, a white border and a green background gradient. (.mystyle)
-- changed the TreeView's even/odd striped background to transparent, so that the green background gradient shows through. (.tree-cell) (.tree-cell:odd)
-- unpinned the TreeView from the screen edge and "floated" it within the left pane area. (in the MainWindow.fx file)

TreeView ~ Re-themed
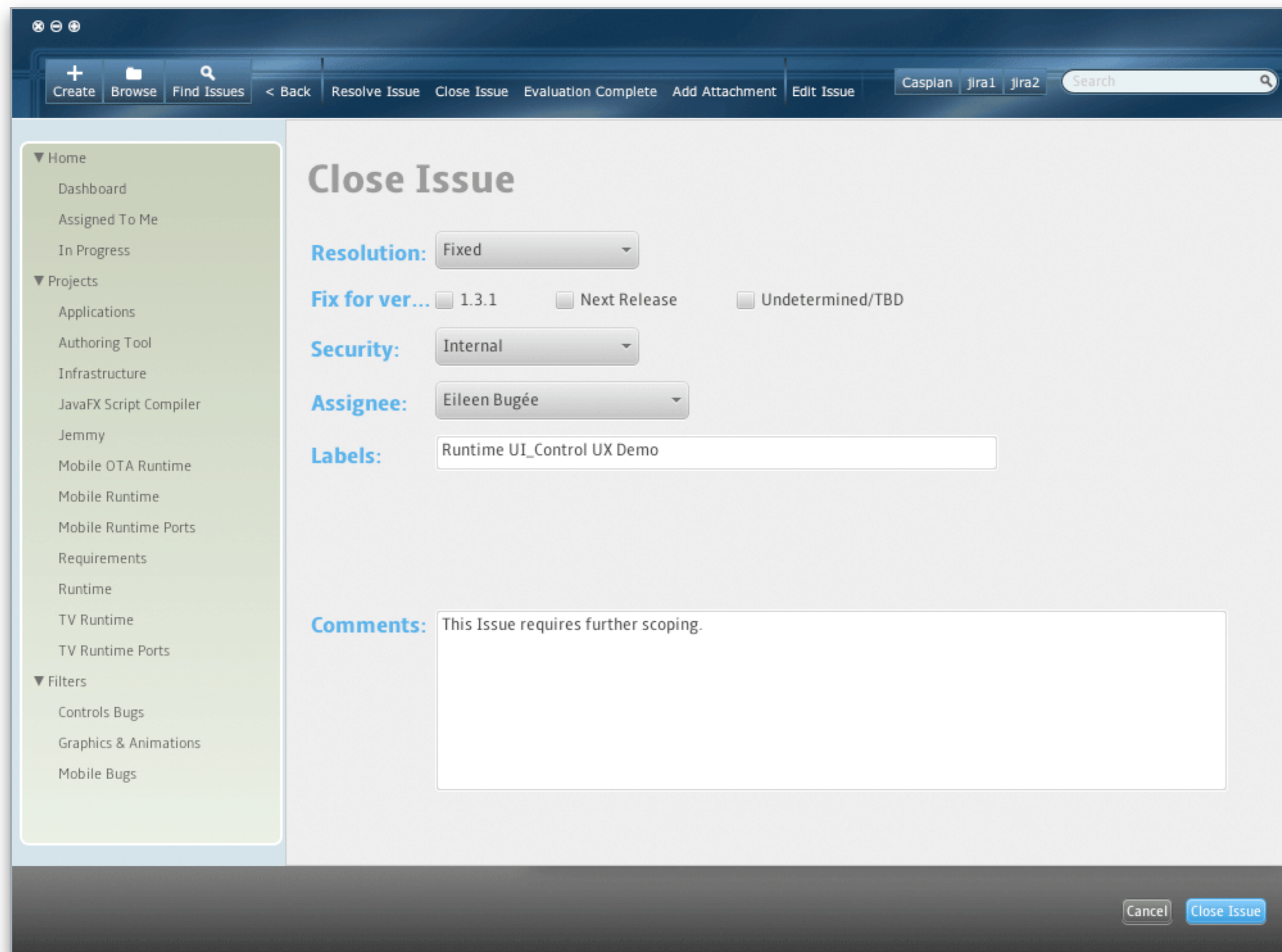
Here's the result of my NetBeans build, with my TreeView revisions added to our external style sheet.

It's getting close to looking like my Photoshop mock-up, but now the "body" of the screen looks out of place. Let's fix that next...

Body

New CSS in External Style Sheet

```
/***************************************
 ** Section 3 - Right Panel & Status bar
 ***************************************/
/* ====== Close Action Region ====== */
#issue-action-region {
    -fx-font-size: 16pt;
    -fx-background: #e5e5e5;
    -fx-padding: 17;
    -fx-background-color: #b6b6b6, #e5e5e5;
    -fx-background-insets: 0,1;
    -fx-effect: dropshadow( three-pass-box ,
rgba(0,0,0,0.4) , 25, 0.0 , 0 , 0 );
    -fx-border-color: white;
    -fx-border-radius: 6;
    -fx-background-color: #bdd6e2;
    -fx-text-fill: #336699;
    -fx-background-image: "watermark80pct.png";
}
#issue-action-region-title {
    -fx-font-size: 50;
    -fx-text-fill: #003366;
    -fx-effect: dropshadow( one-pass-box , white , 0,
0.0 , 0 , 1 );
    -fx-border-width: 2 2 0 2;
    -fx-border-color: white;
    -fx-border-radius: 6 6 0 0;
    -fx-padding: 4 16 4 6;
    -fx-background-color: #e5eff3;
    -fx-background-radius: 6 6 0 0;
    -fx-opacity: .6;
}
.issue-action-region-label {
    -fx-font-size: 14pt;
    -fx-font-weight: bold;
    -fx-text-fill: #333333;
    -fx-effect: dropshadow( one-pass-box , white , 0,
0.0 , 0 , 1 );
}
#issue-action-region-button {
    -fx-font-size: 18pt;
    -fx-padding: 6 15 6 15;
    -fx-background-radius: 6, 6, 5, 4;
    -fx-background-color: linear (0%,0%) to (0%,100%)
stops (0%, #336699) (100%, #033669);
    -fx-background-color: #003366, #006699,
linear (0%,0%) to (0%,100%) stops (0%, #336699)
(100%, #003366);
    -fx-text-fill: white;
}
#issue-action-region-button:strong {
    -fx-background-color: #003366, #006699,
linear (0%,0%) to (0%,100%) stops (0%, #4bb1fc)
(75%, #0072ce) (95%, #0093ff);
}
```

```
#issue-action-region-choice-box {
    -fx-padding: 5 18 6 6;
    -fx-background-color: #92b6c7, white,
linear (0%,0%) to (0%,100%) stops (0%, #e1f4fd)
(70%, #aaccdd) (100%, #e1f4fd);
    -fx-background-radius: 6, 5, 4;
    -fx-background-insets: 0, 1, 2;
    -fx-vpos: CENTER;
    -fx-graphic-vpos: CENTER;
    -fx-text-fill: #333333;
    -fx-color: -fx-pressed-base;
    -fx-font-size: 16pt;
}
#issue-check-box .box {
    -fx-background-color: #92b6c7, white,
linear (0%,0%) to (0%,100%) stops (0%, #e1f4fd)
(70%, #aaccdd) (100%, #e1f4fd);
    -fx-background-radius: 6, 5, 4;
    -fx-background-insets: 0, 1, 2;
    -fx-padding: 7 7 7 7;
    -fx-font-size: 16pt;
}
#issue-action-region-roundedbox {
    -fx-background-color: #deebf1;
    -fx-background-radius: 6px;
    -fx-padding: 10px;
    -fx-border-width: 2;
    -fx-border-color: white;
    -fx-border-radius: 6;
    -fx-opacity: .6;
}
#issue-action-region-mostly-roundedbox {
    -fx-background-color: #deebf1;
    -fx-background-radius: 6px;
    -fx-padding: 10px;
    -fx-border-width: 2;
    -fx-border-color: white;
    -fx-border-radius: 0 6 6 6;
    -fx-opacity: .6;
}

/* ====== STATUSBAR ====== */
#statusbar {
    -fx-padding: 2 20 2 10;
    -fx-base: #434343;
    -fx-background: #434343;
    -fx-background-color: linear (0%,0%) to (0%,100%)
stops (0%, #142129) (100%, #006699);
}
#statusbar *.button {
    -fx-padding: 7 13 9 13;
}
```

Default Caspian CSS

```
/***************************************
 ** Section 3 - Right Panel & Status bar
 ***************************************/
/* Close Action Region */
#issue-action-region {
    -fx-font-size: 14pt;
    -fx-background: #e5e5e5;
    -fx-padding: 20;
    -fx-background-color: #b6b6b6, #e5e5e5;
    -fx-background-insets: 0,1;
    -fx-effect: dropshadow( three-pass-box ,
rgba(0,0,0,0.4) , 25, 0.0 , 0 , 0 );
}
#issue-action-region-title {
    -fx-font-size: 50;
    -fx-font-weight: bold;
    -fx-text-fill: #767676;
    -fx-effect: dropshadow( one-pass-box , white ,
0, 0.0 , 0 , 1 );
}
.issue-action-region-label {
    -fx-font-size: 18pt;
    -fx-font-weight: bold;
    -fx-text-fill: #2092e4;
    -fx-effect: dropshadow( one-pass-box , white ,
0, 0.0 , 0 , 1 );
}
#issue-action-region-button {
    -fx-font-size: 12pt;
    -fx-padding: 6 15 6 15;
    -fx-background-radius: 5, 5, 4, 3;
}
#issue-action-region-button:strong {
    -fx-text-fill: white;
}
```

```
.check-box .box {
    -fx-padding: 3 3 3 3;
    -fx-background-color: -fx-shadow-highlight-color,
-fx-outer-border, -fx-inner-border, -fx-body-color;
    -fx-background-radius: 2, 2, 1, 1;
    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
}
.choice-box {
    -fx-padding: 2 6 3 6;
    -fx-background-color: -fx-shadow-highlight-color,
-fx-outer-border, -fx-inner-border, -fx-body-color;
    -fx-background-radius: 5, 5, 4, 3;
    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
    -fx-vpos: CENTER;
    -fx-graphic-vpos: CENTER;
}
/* ====== STATUSBAR ====== */
#statusbar {
    -fx-padding: 2 20 2 10;
    -fx-base: #434343;
    -fx-background: #434343;
    -fx-background-color: linear (0%,0%) to (0%,100%)
stops (0%, #595959) (100%, #212121);
}
#statusbar *.button {
    -fx-padding: 3 4 2 4;
}
```

As I mentioned, the final area that I want to change is the body/content area of the application screen.

At the left of this slide are sections from the default caspian.css.
At the right is CSS of our external style sheet, showing all of my changes to these sections, as well as some new specifications.

Here's what I changed:
-- increased the font size of the Controls' text (#issue-action-region { -fx-font-size: 16pt;)
-- added a background image (-fx-background-image: "watermark80pct.png";)
-- increased the size and changed the color of the screen's Title text (#issue-action-region-title { -fx-font-size: 50;) (-fx-text-fill: #003366;)
-- decreased the size and changed the weight and color of the Label text (.issue-action-region-label { -fx-font-size: 14pt; -fx-font-weight: bold; -fx-text-fill: #333333;)
-- increased the dimensions and font size, and changed the colors of the status bar buttons (#issue-action-region-button) (#issue-action-region-button:strong)
-- specified new colors, radii, padding and font size, and removed the shadow-highlight shape for the ChoiceBox and CheckBox (#issue-action-region-choice-box) (#issue-check-box .box)
-- added rounded, translucent white shapes as backgrounds behind the controls (#issue-action-region-title) (#issue-action-region-roundedbox) (#issue-action-region-mostly-roundedbox)
-- specified different button padding and gradient colors for the StatusBar background (-fx-background-color: linear (0%,0%) to (0%,100%) stops (0%, #142129) (100%, #006699);)

Body ~ Entire Screen Re-themed
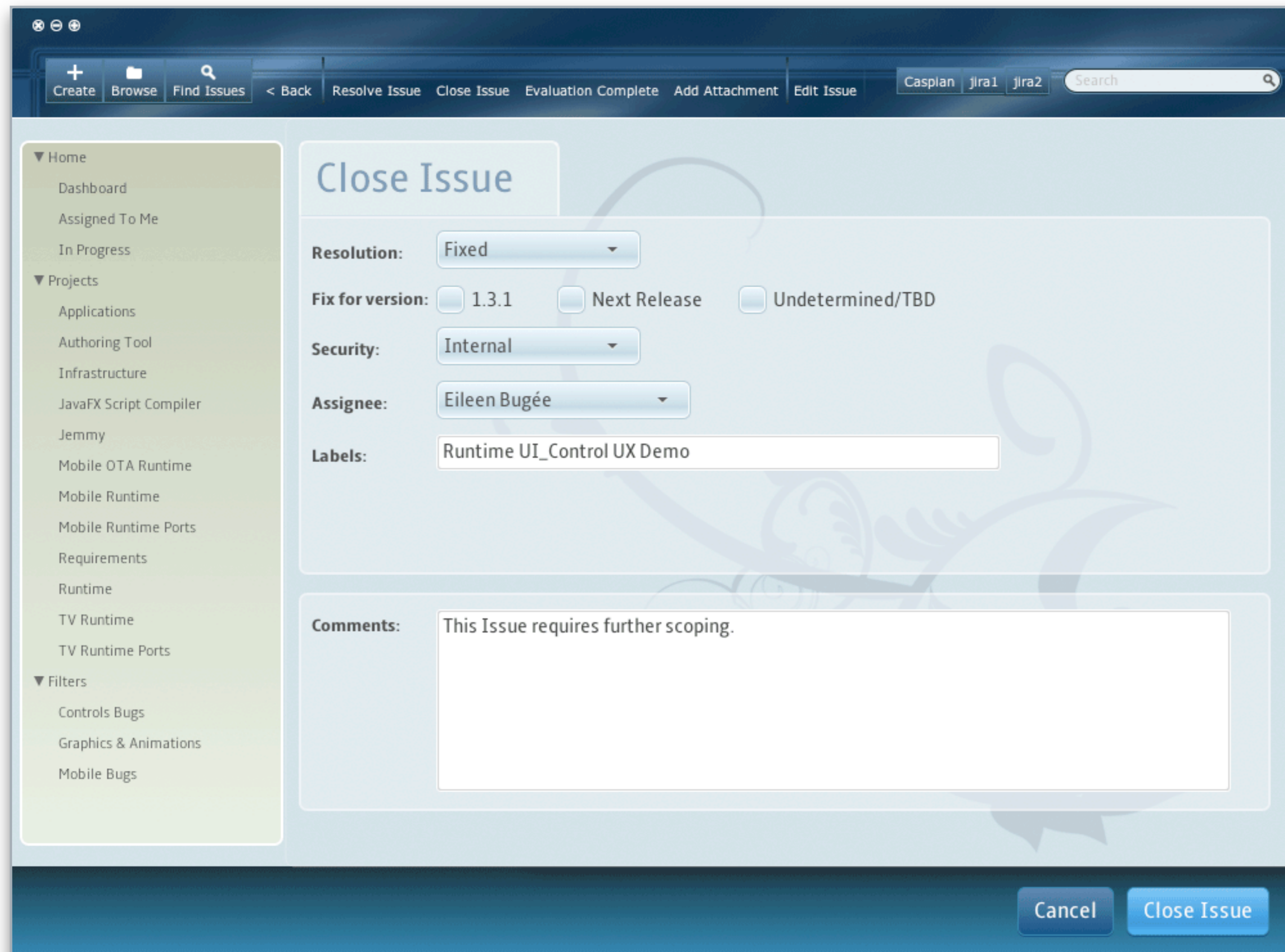
And here's the result of my NetBeans build, with my 'body' revisions added to our external style sheet...
It's pretty close to my Photoshop mock-up!

Before and After... Summary
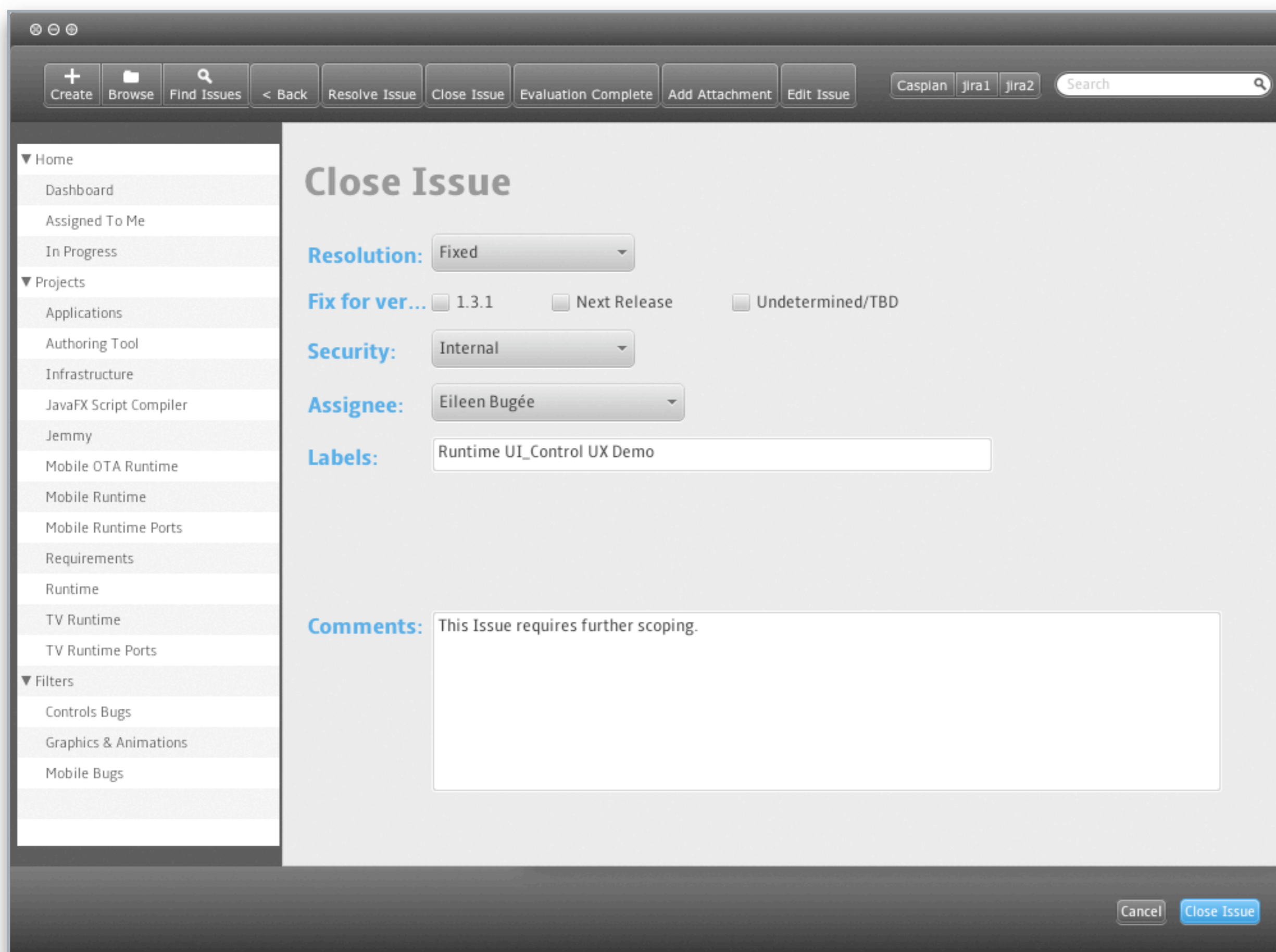
Original Application Screen
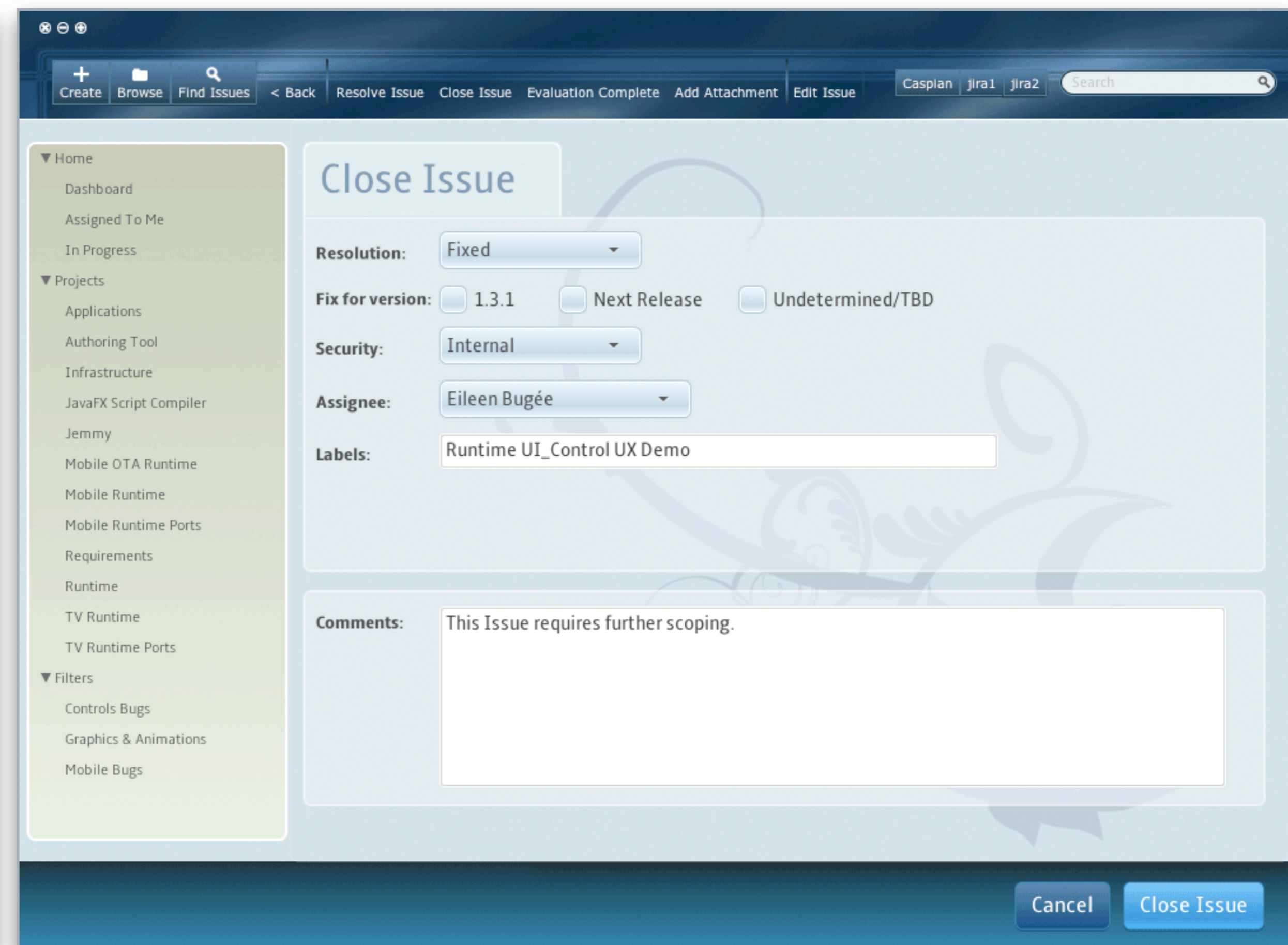
Re-themed Application Screen

Here's a recap of what I did....

Toolbar changes:
-- replaced the main window's grey background color with light blue (-fx-background-color: #bdd6e2;)
-- changed the Min/Max/Close buttons' background color to transparent, and their icons to white (frame-button) (frame-button Rectangle)
-- replaced the inherited grey colors of the "Create/Browse/Find" Toolbar buttons with blue gradients and a blue border (SEGMENT BUTTONS - Create/Browse/Find)
-- removed the background color completely from the "Back/Resolve/Close/etc" Toolbar buttons (-fx-background-color: transparent;)
-- styled one vertical border of these buttons with a gradient to act as a separator between button groups (#action-left-segement) (#action-right-segement)
        (I am misspelling "segement" as "segement" in the CSS to follow the original engineer's misspelling, so as to not break the build)
-- replaced the grey gradient backgrounds on #toolbar_top and #toolbar with background-images. These 2 images "butt" up against each other, and combine to give the look of a single background,
-- specified new colors for the SearchBox

TreeView changes:
-- created a styleClass for a new background shape (.mystyle). The styleClass "mystyle" was also added to the MainWindow.fx file.
-- styled this background with rounded corners, a white border and a green background gradient. (.mystyle)
-- changed the TreeView's even/odd striped background to transparent, so that the green background gradient shows through. (.tree-cell) (.tree-cell:odd)
-- unpinned the TreeView from the screen edge and "floated" it within the left pane area. (in the MainWindow.fx file)

Body changes:
-- increased the font size of the Controls' text (#issue-action-region { -fx-font-size: 16pt;)
-- added a background image (-fx-background-image: "watermark80pct.png";)
-- increased the size and changed the color of the screen's Title text (#issue-action-region-title { -fx-font-size: 50;) (-fx-text-fill: #003366;)
-- decreased the size and changed the weight and color of the Label text (.issue-action-region-label { -fx-font-size: 14pt; -fx-font-weight: bold; -fx-text-fill: #333333;)
-- increased the dimensions and font size, and changed the colors of the status bar buttons (#issue-action-region-button) (#issue-action-region-button:strong)
-- specified new colors, radii, padding and font size, and removed the shadow-highlight shape for the ChoiceBox and CheckBox (#issue-action-region-choice-box) (#issue-check-box .box)
-- added rounded, translucent white shapes as backgrounds behind the controls (#issue-action-region-title) (#issue-action-region-roundedbox) (#issue-action-region-mostly-roundedbox)
-- specified different button padding and gradient colors for the StatusBar background (-fx-background-color: linear (0%,0%) to (0%,100%) stops (0%, #142129) (100%, #006699);)

Summary of Re-theming Changes

Toolbar changes:
-- replaced the main window's grey background color with light blue (-fx-background-color: #bdd6e2;)
-- changed the Min/Max/Close buttons' background color to transparent, and their icons to white (frame-button) (frame-button Rectangle)
-- replaced the inherited grey colors of the "Create/Browse/Find" Toolbar buttons with blue gradients and a blue border (SEGMENT BUTTONS - Create/Browse/Find)
-- removed the background color completely from the "Back/Resolve/Close/etc" Toolbar buttons (-fx-background-color: transparent;)
-- styled one vertical border of these buttons with a gradient to act as a separator between button groups (#action-left-segement) (#action-right-segement)
    (I am misspelling "segment" as "segement" in the CSS to follow the original engineer's misspelling, so as to not break the build)
-- replaced the grey gradient backgrounds on #toolbar_top and #toolbar with background-images. These 2 images "butt" up against each other, and combine to give the look of a single background,
-- specified new colors for the SearchBox

TreeView changes:
-- created a styleClass for a new background shape (.mystyle). The styleClass "mystyle" was also added to the MainWindow.fx file.
-- styled this background with rounded corners, a white border and a green background gradient. (.mystyle)
-- changed the TreeView's even/odd striped background to transparent, so that the green background gradient shows through. (.tree-cell) (.tree-cell:odd)
-- unpinned the TreeView from the screen edge and "floated" it within the left pane area. (in the MainWindow.fx file)

Body changes:
-- increased the font size of the Controls' text (#issue-action-region { -fx-font-size: 16pt;)
-- added a background image (-fx-background-image: "watermark80pct.png";)
-- increased the size and changed the color of the screen's Title text
(#issue-action-region-title { -fx-font-size: 50;) (-fx-text-fill: #003366;)
-- decreased the size and changed the weight and color of the Label text
(.issue-action-region-label { -fx-font-size: 14pt; -fx-font-weight: bold; -fx-text-fill: #333333;)
-- increased the dimensions and font size, and changed the colors of the status bar buttons
(#issue-action-region-button) (#issue-action-region-button:strong)
-- specified new colors, radii, padding and font size, and removed the shadow-highlight shape for the ChoiceBox and CheckBox (#issue-action-region-choice-box) (#issue-check-box .box)
-- added rounded, translucent white shapes as backgrounds behind the controls
(#issue-action-region-title) (#issue-action-region-roundedbox) (#issue-action-region-mostly-roundedbox)
-- specified different button padding and gradient colors for the StatusBar background (-fx-background-color: linear (0%,0%) to (0%,100%) stops (0%, #142129) (100%, #006699);)

An alternate view of the Re-theming Summary.

Outline

. The Caspian Theme

. Styling Overview

. Desktop Re-theming

. **TV: Current and Future**

. Q&A

```
.text-box {
    -fx-font: 12pt "Amble-Condensed";
    -fx-padding: 0;
    -fx-background-color: -fx-shadow-highli
fx-mid-border-color, -fx-control-inner-back
    -fx-background-radius: 6, 5 4;
    -fx-background-insets: 0, .5, 1.5;
    -fx-text-fill: -fx-text-inner-color;
    -fx-prompt-text-fill: derive(-fx-control-in
background, 30%);
    -fx-highlight-fill: -fx-accent;
    -fx-highlight-text-fill: white;
    -fx-pannable: true;
    -fx-select-on-focus: true;
}
```

-fx-accent: #114fd9;
-fx-base: #3f3f3f;
-fx-background: #040404;
-fx-control-inner-background: white;
-fx-focus-color: #31BAED;
-fx-dark-text-color: black;
-fx-mid-text-color: #292929;
-fx-light-text-color: white;
-fx-color: -fx-base;
-fx-hover-base: ladder -fx-base stops (20%,derive(-fx-base,20%)) (35%,derive(-
(50%,derive(-fx-base,40%));
-fx-pressed-base: derive(-fx-base,-40%);
-fx-box-border: #000000;
-fx-text-box-border: ladder -fx-background stops (0.1,black) (0.3,derive(-fx-bac
-fx-shadow-highlight-color:#3B3B3B;
-fx-outer-border: #000000;
-fx-inner-border: linear (0%, 0%) to (0%, 100%) stops (0%, #7A7A7A) (100%, #52
-fx-body-color: linear (0%,0%) to (0%,100%) stops (0,derive(-fx-color,34%)) (1,d
color,-18%));

Let's take a look at the current JavaFX for TV theme and then discuss the design considerations we put into practice while developing the JavaFX UI Controls. Then I'd like to share with you how cascading style sheets can change the appearance of the UI Controls without ever touching the source code of the application.

JavaFX for TV
Design Process for UX Controls

- Desktop "look" as a basis

- Visual Designer Concepts ➡ Engineer ➡ CSS File

- Close collaborations
  - With Interaction Designer
  - With TV Engineer

ORACLE®

The JavaFX UI Controls for TV were initially designed in Adobe Photoshop and Illustrator and were based upon the Caspian desktop theme.

JavaFX for TV
Design Process for UX Controls

- Desktop "look" as a basis

- Visual Designer Concepts ➡ Engineer ➡ CSS File

- Close collaborations
  - With Interaction Designer
  - With TV Engineer

ORACLE®

The information was extracted from the controls and handed off to the TV engineer for insertion into the Caspian.css document. This was an iterative process.

JavaFX for TV
Design Process for UX Controls

- Desktop "look" as a basis

- Visual Designer Concepts ➡ Engineer ➡ CSS File

- Close collaborations
  - With Interaction Designer
  - With TV Engineer

ORACLE®

We had a close collaborative process working with the interaction designers to ensure that our visual designs aligned with their interaction specifications. We also had a close collaborative process with the TV engineers to ensure that the implemented controls aligned with our visual design specifications.

Design Considerations
For Designing UI Controls in the TV Environment

- "Content is King"

- Respect the living room context

- Design for the TV screen

- Safe area

- The text must be readable

- Make the UI intuitive

ORACLE®

First and foremost, we wanted to ensure that the visual designs of the UI controls did not compete with what our users were watching on TV.

Design Considerations
For Designing UI Controls in the TV Environment

- "Content is King"

- Respect the living room context

- Design for the TV screen

- Safe area

- The text must be readable

- Make the UI intuitive

ORACLE®

We were also thinking about what actions our users would like to perform while watching TV with their family and friends. Watching TV is a passive "sit back and relax" experience.

Design Considerations
For Designing UI Controls in the TV Environment

- "Content is King"

- Respect the living room context

- Design for the TV screen

- Safe area

- The text must be readable

- Make the UI intuitive

ORACLE®

The native resolution of the screen determines whether it is HD ready or not. If we created our images to the resolution of the screen, they would be shown as such. If we didn't, the images would be scaled to fit the screen and image distortion would occur. We targeted our images to fit the screen resolutions of 1280 x 720 and 1920 x 1080. The color displayed on a TV screen is expressed differently than it is on a computer monitor. We did a lot of testing of our colors to ensure that we had the correct color palette on the TV. We tried not to use highly saturated or very bright colors in our images as image ghosting or vibrancy could occur. We didn't use colors that had an RGB value that was greater than R-240, G-240, B-240. We didn't use large spanning gradients as color banding could occur. We tested our graphics to to ensure that they had the correct color pallet on our TV screens.

Design Considerations
For Designing UI Controls in the TV Environment

- "Content is King"

- Respect the living room context

- Design for the TV screen

- Safe area

- The text must be readable

- Make the UI intuitive

ORACLE®

Televisions have a safe central display area surrounded by a small amount of screen space that can vary in size. If we placed our UI controls outside of that safe display area, they would most likely not be visible. We used at least a 10% margin of safe area for each resolution.

Design Considerations
For Designing UI Controls in the TV Environment

- "Content is King"

- Respect the living room context

- Design for the TV screen

- Safe area

- The text must be readable

- Make the UI intuitive

ORACLE®

It's very important that the on-screen font on a TV be readable from a distance of at least 12 feet away. We targeted 21 pt. font for resolutions of 1280 x 720 and 28 pt. font for the 1920 x 1080 resolution. We increased the "leading" of the on-screen font. We chose a simple san serif font with anti- aliasing to ensure readability for the high- def screen format.

Design Considerations
For Designing UI Controls in the TV Environment

- "Content is King"

- Respect the living room context

- Design for the TV screen

- Safe area

- The text must be readable

- Make the UI intuitive

ORACLE®

We wanted to provide intuitive choices and actions that were obvious and easy to select. We wanted to provide navigation that was simple enough to be executed on a remote control. We are providing a virtual QWERTY keyboard but we realize that some of our users will conduct navigation using the directional keypads on their remote control. This limits the navigational model to up, down, right, left, and enter.

JavaFX 1.3.1 Release UI Controls

- Color palette

- The UI controls were enlarged

The color palette of the UI controls for TV has been changed to a darker set of hues so as to not compete with the content that our users are watching.

ORACLE®

JavaFX 1.3.1 Release UI Controls

- Color palette

- The UI controls were enlarged

The size of the TV theme UI controls has been enlarged to three times the size of the original Caspian desktop controls to ensure readability on the high- def screen format.

ORACLE®

JavaFX 1.3.1 Release UI Controls

- Color palette
- The UI controls were enlarged

JavaFX Desktop button control:

Normal	Hover (MouseOver)	Pressed/Armed	Focused	Disabled
Cancel	Cancel	Cancel	Cancel	Cancel

ORACLE

Here is a button control from the original Caspian desktop theme.

JavaFX 1.3.1 Release UI Controls

- Color palette
- The UI controls were enlarged

JavaFX Desktop button control:

| Normal | Hover (MouseOver) | Pressed/Armed | Focused | Disabled |
| Cancel | Cancel | Cancel | Cancel | Cancel |

JavaFX TV button control:

| Button 1 | Button 1 | Button 1 | Button 1 |
| Normal | Focused | Armed | Disabled |

ORACLE®

And here is a button control from the JavaFX for TV theme. Please notice the size differentiation between the two controls as well as the darker color palette.

JavaFX TV UI Controls ~ Summer 2010 Release

JavaFX TV Check Box Control:
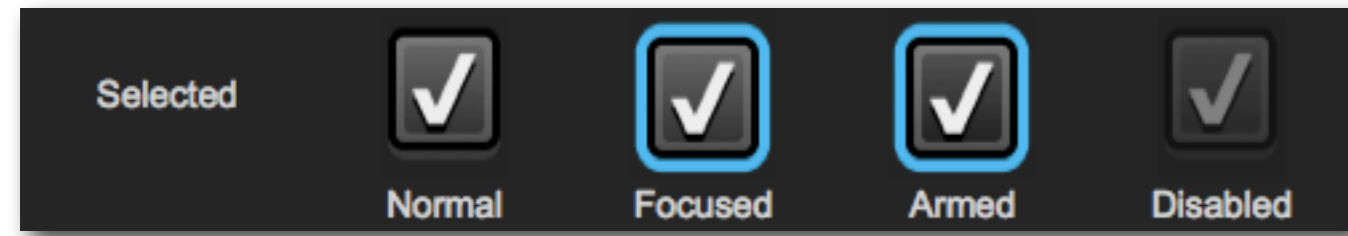
JavaFX TV Progress Bar Control:
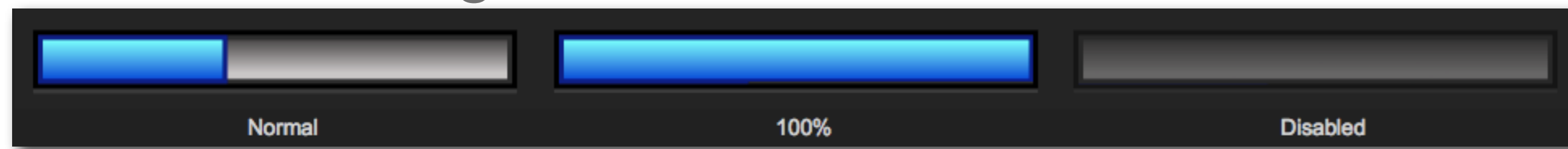
JavaFX TV List View Control:

As you can see, there is a very strong visual relationship between the Caspian desktop controls and the JavaFX for TV UI Controls.
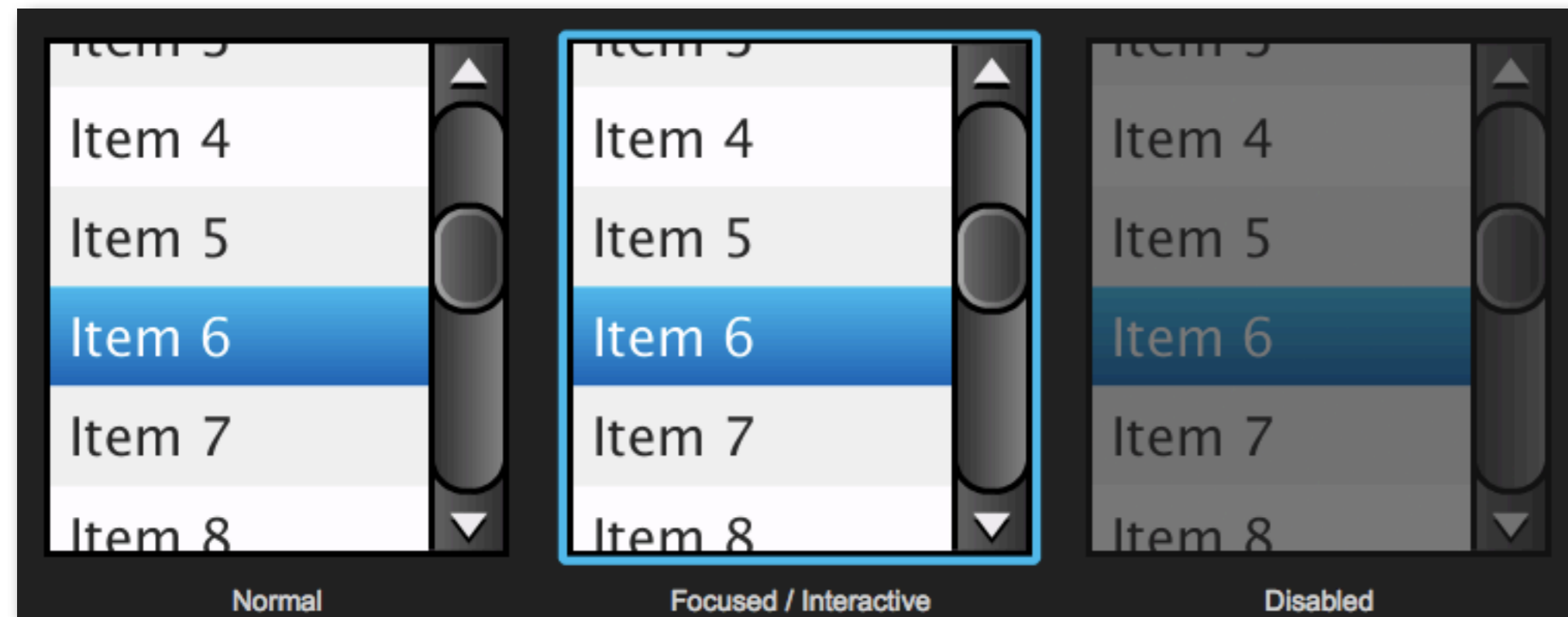
ListView UI Control
Currently Implemented in the JavaFX 1.3.1 Release

But, it that really appropriate for the TV format?
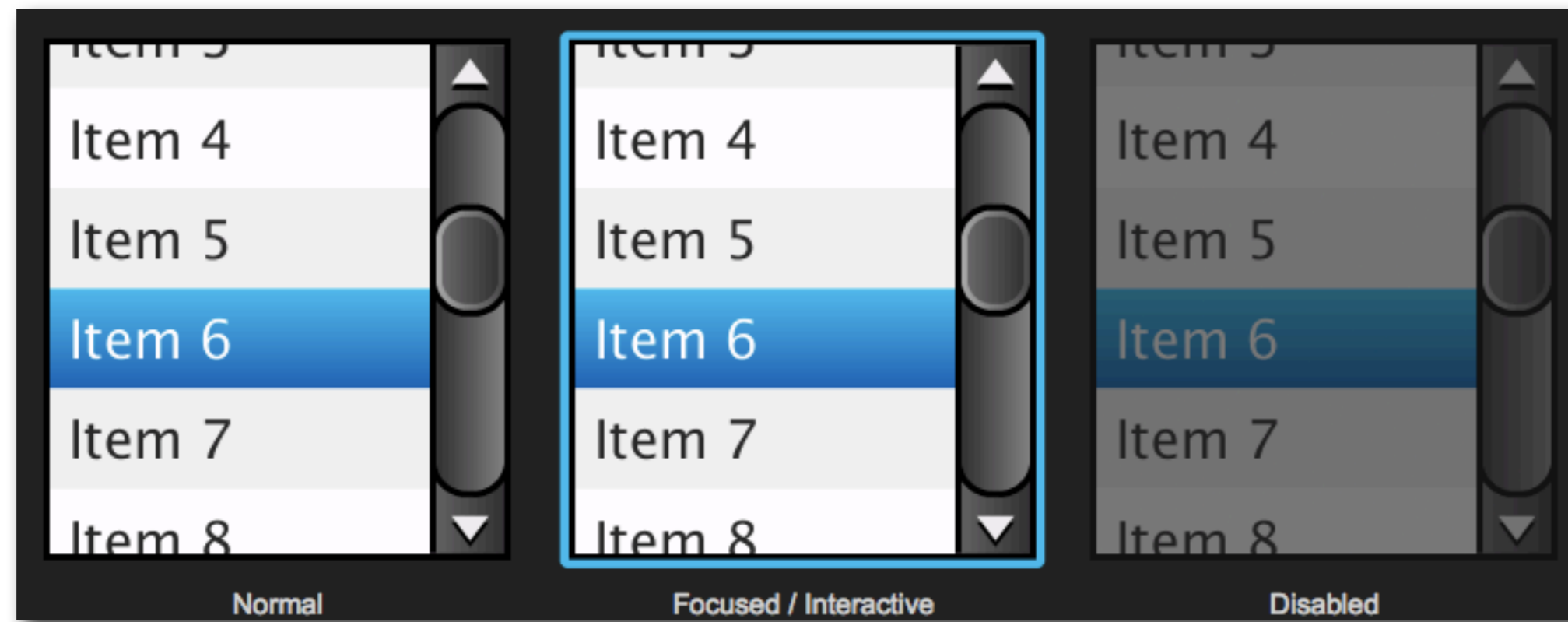
Demo

Using CSS to customize implemented

UI Controls to create a more *TV–Centric* presentation.

ORACLE®

We've been experimenting with cascading style sheets to change the appearance of our UI controls in an effort to create a more "TV-Centric" presentation. I'd like to share with you how CSS can change the appearance of your UI controls without ever touching the source code of the application. Let me show you what I mean...
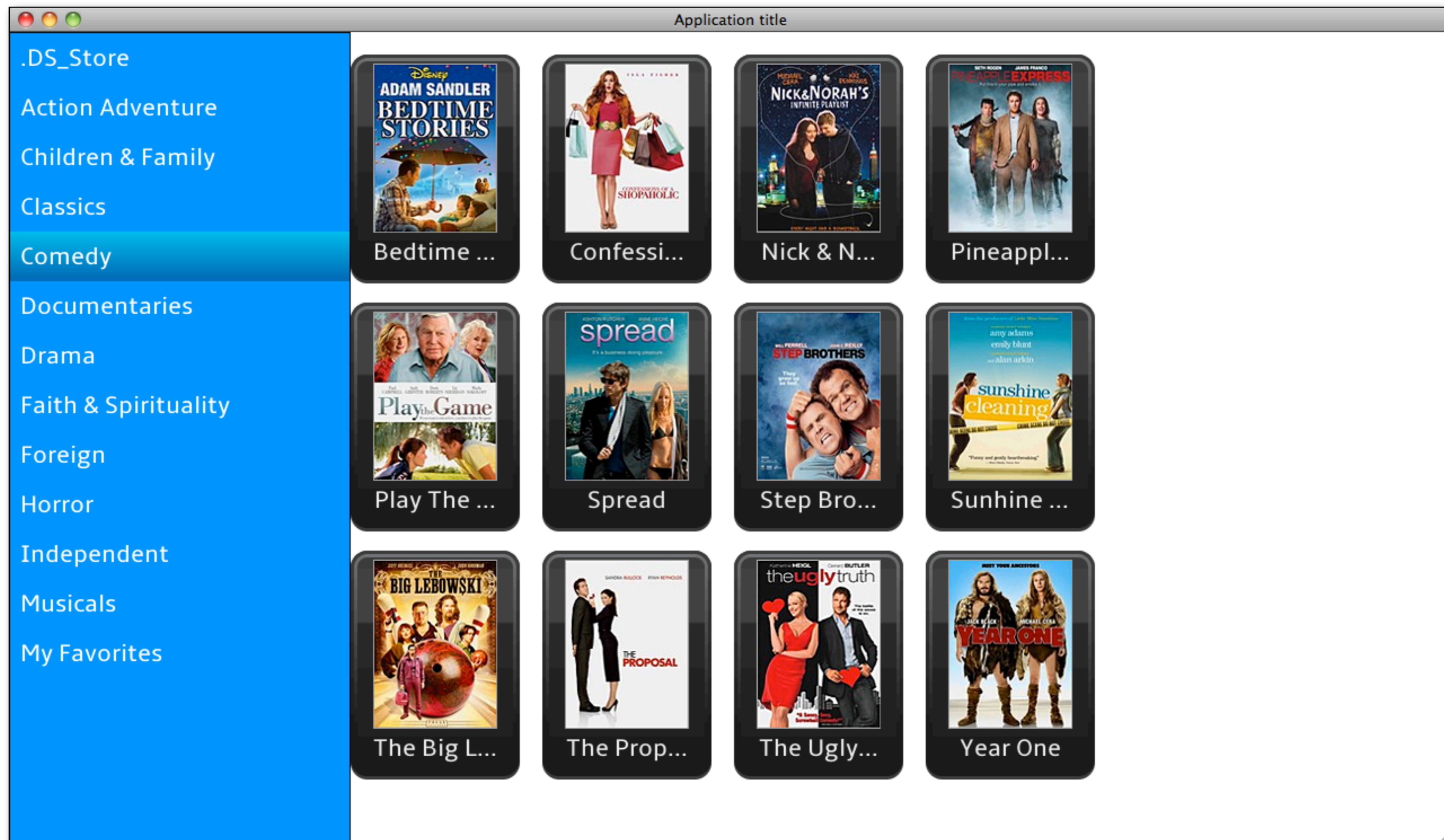
I'm running a simple TV application in the NetBeans IDE. It consists of a vertical list view control on the left side of the screen and a movie grid control in the center of the screen. As I scroll up and down through the vertical list view control, the content of the movie grid changes accordingly.

The mechanics of this application work very well, but the presentation is not very "TV Centric." I'm going to insert and modify the css code to change the look and feel of the ux controls without ever touching the source code of the application. The first element I would like to change is to take out the white background behind the movie grid and have a more "TV Centric" background appear in this application.

Left panel:

```css
@charset "UTF-8";
/* CSS Document */
/*

Application Specific Styles

These styles are specific to the application.  They use the scene properties defined by default.css.

*/


.app-background {
    -fx-fill: darkblue;
}

.app-content *.button:armed {

    -fx-color: red;
    -fx-text-fill: white;

}

.app-content *.button {
    //desktop//    padding: 2 10 3 10;
    -fx-padding: 6 30 9 30;
    -fx-background-color: -fx-shadow-highlight-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
    //desktop//    -fx-background-radius: 5, 5, 4, 3;
    -fx-background-radius: 15, 15, 12, 9;
    //desktop//    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
    -fx-background-insets: 0 0 -3 0, 0, 3, 6;
    -fx-text-fill: -fx-text-base-color;
    -fx-vpos: CENTER;
    -fx-graphic-vpos: CENTER;
}

.app-content *.button:focused {
    -fx-background-color: -fx-focus-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
    //desktop//        -fx-background-radius: 6.4, 5, 4, 3;
    -fx-background-radius: 19.2, 15, 12, 9;
    //desktop//    -fx-background-insets: -1.4, 0, 1, 2;
    -fx-background-insets: -4.2, 0, 3, 6;
}

.app-content *.list-view {
    -fx-padding: 1;
    -fx-background-color: -fx-box-border, -fx-control-inner-background;
    -fx-background-insets: 0,1;
    -fx-pannable: true
}

.app-content *.list-view:focused {
    //desktop//    -fx-padding: 1;
    -fx-padding: 3;
    -fx-background-color: -fx-focus-color,-fx-box-border,-fx-control-inner-background;
    //desktop//    -fx-background-insets: -1.4, 0, 1;
    -fx-background-insets: -4.2, 0, 3;
    //desktop//    -fx-background-radius: -1.4, 0, 0;
    -fx-background-radius: -4.2, 0, 0;
}
```

Right panel:

```css
@charset "UTF-8";
/* CSS Document */
/*

Application Specific Styles

These styles are specific to the application.  They use the scene properties defined by default.css.

*/


.app-background {
    -fx-background-image: url("background.png");
    -fx-background-repeat: no-repeat;
    -fx-background-position: left top;
    -fx-padding-top: 0px;
    -fx-margin-bottom: 0px;
}
.app-content *.button:armed {

    -fx-color: red;
    -fx-text-fill: white;
}
.app-content *.button {
    //desktop//    padding: 2 10 3 10;
    -fx-padding: 6 30 9 30;
    -fx-background-color: -fx-shadow-highlight-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
    //desktop//    -fx-background-radius: 5, 5, 4, 3;
    -fx-background-radius: 15, 15, 12, 9;
    //desktop//    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
    -fx-background-insets: 0 0 -3 0, 0, 3, 6;
    -fx-text-fill: -fx-text-base-color;
    -fx-vpos: CENTER;
    -fx-graphic-vpos: CENTER;
}
.app-content *.button:focused {
    -fx-background-color: -fx-focus-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
    //desktop//        -fx-background-radius: 6.4, 5, 4, 3;
    -fx-background-radius: 19.2, 15, 12, 9;
    //desktop//    -fx-background-insets: -1.4, 0, 1, 2;
    -fx-background-insets: -4.2, 0, 3, 6;
}
.app-content *.list-view {
    -fx-padding: 1;
    -fx-background-color: -fx-box-border, -fx-control-inner-background;
    -fx-background-insets: 0,1;
    -fx-pannable: true
}
.app-content *.list-view:focused {
    //desktop//    -fx-padding: 1;
    -fx-padding: 3;
    -fx-background-color: -fx-focus-color,-fx-box-border,-fx-control-inner-background;
    //desktop//    -fx-background-insets: -1.4, 0, 1;
    -fx-background-insets: -4.2, 0, 3;
    //desktop//    -fx-background-radius: -1.4, 0, 0;
    -fx-background-radius: -4.2, 0, 0;
}
```
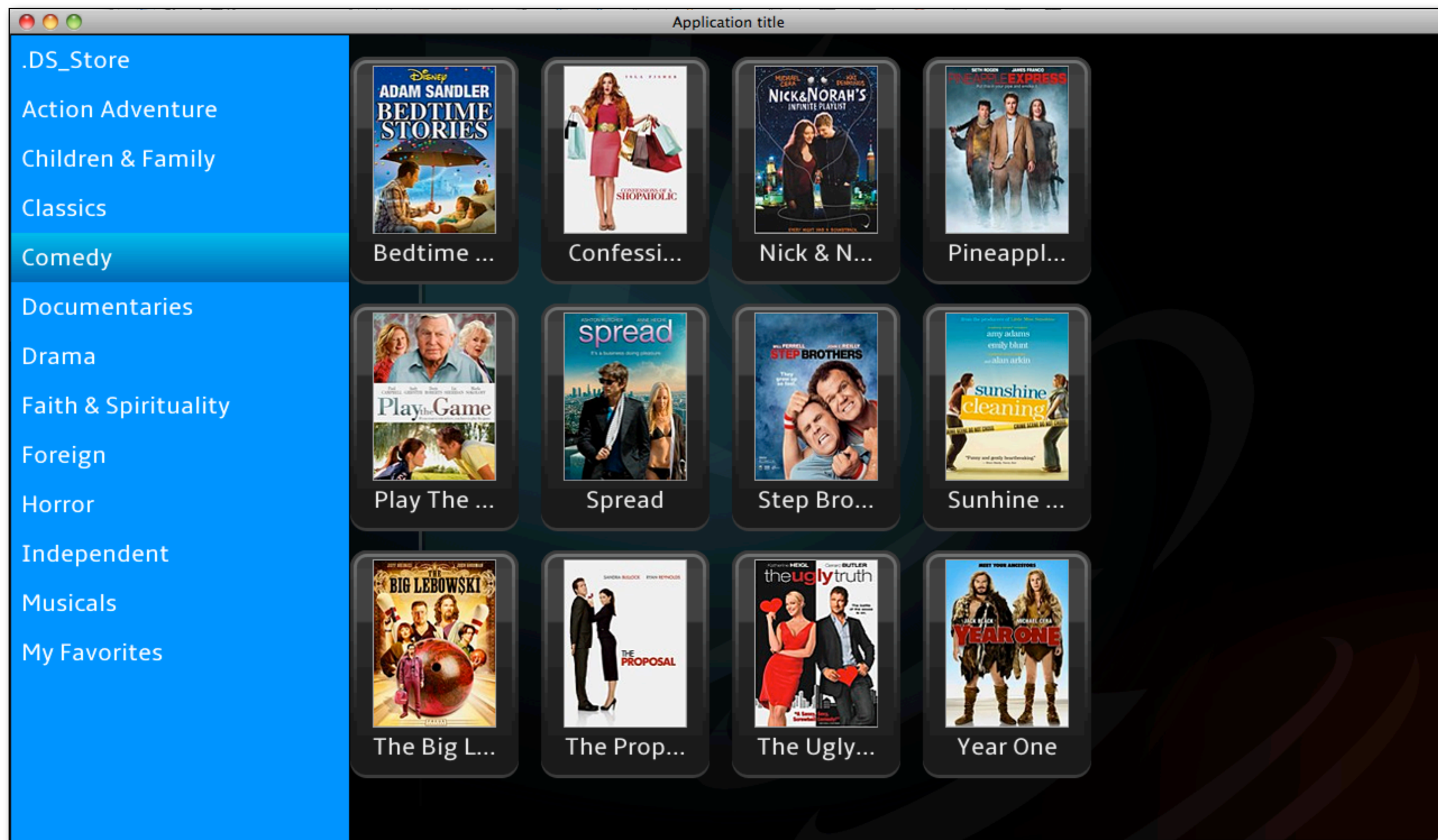
I'm going to swap the css code for the selector .app-background -fx-fill: (image upper left) to -fx-background-image: and change the value to reference the "background.png" image that I've previously created and placed in the simpletvapp resources folder. (image upper right.)

After I select the "Run Main Project" control in the NetBeans IDE, this is the result of the build. We can now see the background.png image appearing behind the movie grid control. Now I want to get rid of the medium blue hue that comprises the background of the vertical list view control. So I'll go back to the NetBeans IDE and swap out some css code to make that change.

Left panel:

```
9    */
10
11
12   .app-background {
13       -fx-background-image: url("background.png");
14       -fx-background-repeat: no-repeat;
15       -fx-background-position: left top;
16       -fx-padding-top: 0px;
17       -fx-margin-bottom: 0px;
18
19   }
20
21
22   .app-content *.button:armed {
23
24       -fx-color: red;
25       -fx-text-fill: white;
26
27   }
28
29   .app-content *.button {
     //desktop//    padding: 2 10 3 10;
31       -fx-padding: 6 30 9 30;
32       -fx-background-color: -fx-shadow-highlight-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
33       //desktop//    -fx-background-radius: 5, 5, 4, 3;
34       -fx-background-radius: 15, 15, 12, 9;
35       //desktop//    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
36       -fx-background-insets: 0 0 -3 0, 0, 3, 6;
37       -fx-text-fill: -fx-text-base-color;
38       -fx-vpos: CENTER;
39       -fx-graphic-vpos: CENTER;
40   }
41   .app-content *.button:focused {
42       -fx-background-color: -fx-focus-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
     //desktop//        -fx-background-radius: 6.4, 5, 4, 3;
44       -fx-background-radius: 19.2, 15, 12, 9;
45       //desktop//    -fx-background-insets: -1.4, 0, 1, 2;
46       -fx-background-insets: -4.2, 0, 3, 6;
47   }
48   .app-content *.list-view {
49       -fx-padding: 1;
50       -fx-background-color: -fx-box-border, -fx-control-inner-background;
51       -fx-background-insets: 0,1;
52       -fx-pannable: true
53   }
54
55   .app-content *.list-view:focused {
     //desktop//    -fx-padding: 1;
57       -fx-padding: 3;
58       -fx-background-color: -fx-focus-color,-fx-box-border,-fx-control-inner-background;
59       //desktop//    -fx-background-insets: -1.4, 0, 1;
60       -fx-background-insets: -4.2, 0, 3;
61       //desktop//    -fx-background-radius: -1.4, 0, 0;
62       -fx-background-radius: -4.2, 0, 0;
63   }
64
65
```

Right panel:

```
9    */
10
11
12   .app-background {
13       -fx-background-image: url("background.png");
14       -fx-background-repeat: no-repeat;
15       -fx-background-position: left top;
16       -fx-padding-top: 0px;
17       -fx-margin-bottom: 0px;
18
19   }
20
21
22   .app-content *.button:armed {
23
24       -fx-color: red;
25       -fx-text-fill: white;
26
27   }
28
29   .app-content *.button {
     //desktop//    padding: 2 10 3 10;
31       -fx-padding: 6 30 9 30;
32       -fx-background-color: -fx-shadow-highlight-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
33       //desktop//    -fx-background-radius: 5, 5, 4, 3;
34       -fx-background-radius: 15, 15, 12, 9;
35       //desktop//    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
36       -fx-background-insets: 0 0 -3 0, 0, 3, 6;
37       -fx-text-fill: -fx-text-base-color;
38       -fx-vpos: CENTER;
39       -fx-graphic-vpos: CENTER;
40   }
41   .app-content *.button:focused {
42       -fx-background-color: -fx-focus-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
     //desktop//        -fx-background-radius: 6.4, 5, 4, 3;
44       -fx-background-radius: 19.2, 15, 12, 9;
45       //desktop//    -fx-background-insets: -1.4, 0, 1, 2;
46       -fx-background-insets: -4.2, 0, 3, 6;
47   }
48   .app-content *.list-view {
49       -fx-padding:  50 3 50 3;
50       -fx-background-color: transparent;
51       -fx-background-insets: 0,1;
52       -fx-pannable: true
53
54   }
55
56
57   .app-content *.list-view:focused {
     //desktop//    -fx-padding: 1;
59       -fx-padding: 50 3 50 3;
60       -fx-background-color: transparent;
61       //desktop//    -fx-background-insets: -1.4, 0, 1;
62       -fx-background-insets: -4.2, 0, 3;
63       //desktop//    -fx-background-radius: -1.4, 0, 0;
64       -fx-background-radius: -4.2, 0, 0;
65   }
66
```

In the .app-content *.list-view selector, I'm going to swap out the -fx-background-color: value from -fx-control-inner-background; to the transparent; value. I'll also swap css code for the .app-content *.list-view: focused as well. These css code swaps over ride the previous value of -fx-control-inner-background; which referenced the hexadecimal value of #1e90ff in the caspian.css document.

After I select the "Run Main Project" control in the NetBeans IDE, this is the result of the css code swap. The background.png image can now be seen in the entire 1280 x 720 pixel canvas. The next ux control I would like to change the appearance of will be the focus selector in the vertical list view control. I will return to the NetBeans IDE and make some css code insertions.

```
     //desktop//    padding: 2 10 3 10;
31       -fx-padding: 6 30 9 30;
32       -fx-background-color: -fx-shadow-highlight-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
33       //desktop//    -fx-background-radius: 5, 5, 4, 3;
34       -fx-background-radius: 15, 15, 12, 9;
35       //desktop//    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
36       -fx-background-insets: 0 0 -3 0, 0, 3, 6;
37       -fx-text-fill: -fx-text-base-color;
38       -fx-vpos: CENTER;
39       -fx-graphic-vpos: CENTER;
40   }
41   .app-content *.button:focused {
42       -fx-background-color: -fx-focus-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
         //desktop//        -fx-background-radius: 6.4, 5, 4, 3;
44       -fx-background-radius: 19.2, 15, 12, 9;
45       //desktop//    -fx-background-insets: -1.4, 0, 1, 2;
46       -fx-background-insets: -4.2, 0, 3, 6;
47   }
48   .app-content *.list-view {
49       -fx-padding:  50 3 50 3;
50       -fx-background-color: transparent;
51       -fx-background-insets: 0,1;
52       -fx-pannable: true
53
54   }
55
56
57   .app-content *.list-view:focused {
         //desktop//    -fx-padding: 1;
59       -fx-padding: 50 3 50 3;
60       -fx-background-color: transparent;
61       //desktop//    -fx-background-insets: -1.4, 0, 1;
62       -fx-background-insets: -4.2, 0, 3;
63       //desktop//    -fx-background-radius: -1.4, 0, 0;
64       -fx-background-radius: -4.2, 0, 0;
65   }
66
67   .app-content *.list-cell:filled:selected {
68       -fx-background: transparent;
         -fx-background-color: linear (0%, 0%) to (0%, 100%) stops (0%, #bed5fa) (10%, #6a93e6) (50%, #5b7cb4) (51%, #274b88) (90%, #27589e) (95%, #4f92d7);
70       -fx-text-fill: -fx-selection-bar-text;
71       -fx-padding: 6 6 6 90;
72       -fx-scale-x: 140%;
73       -fx-scale-y: 140%;
74       -fx-background-insets: 0 118 0 0;
75       -fx-effect: dropshadow( gaussian, black, 10, 0.4, 0, 7);
76   }
77
78
```
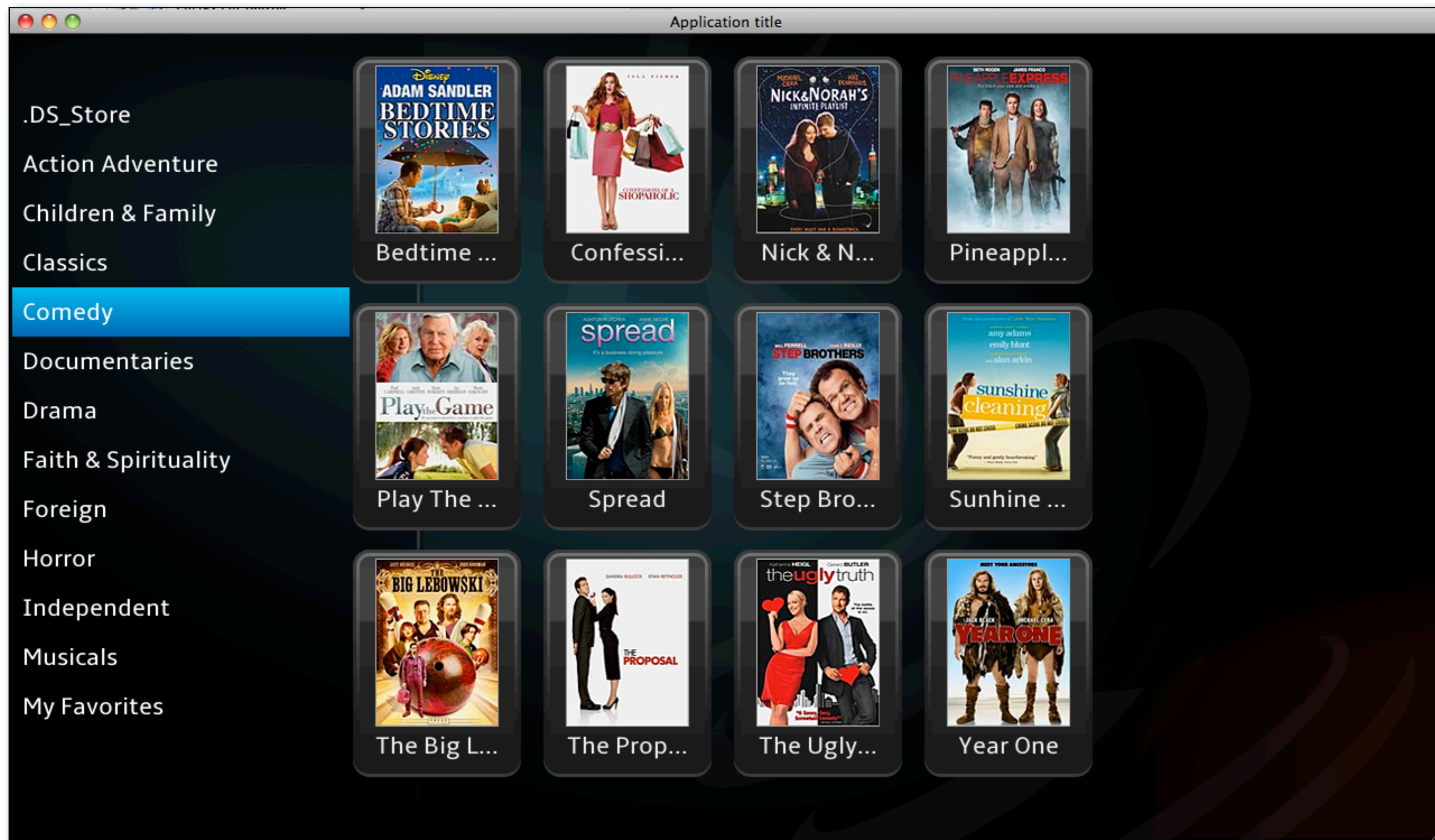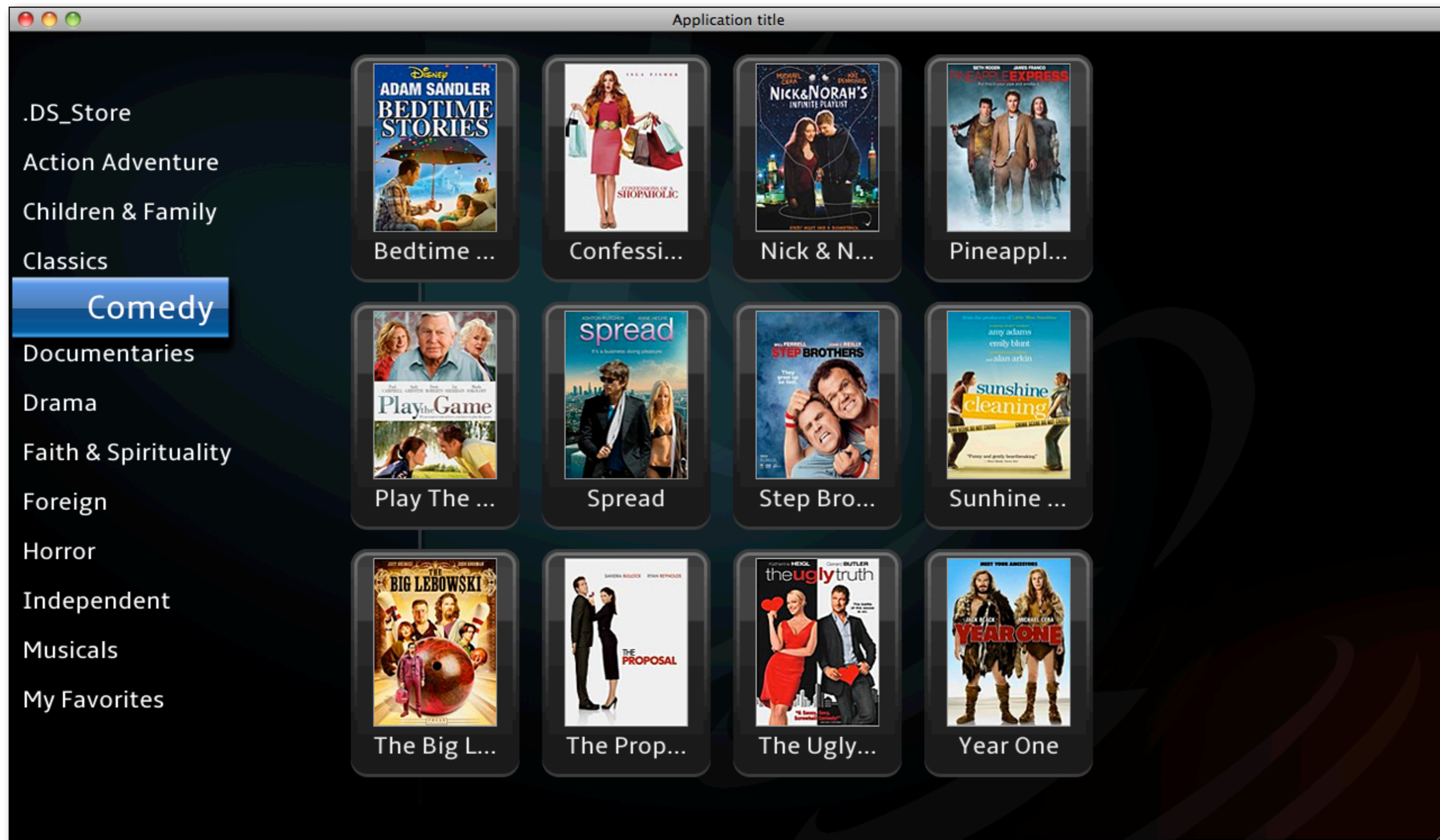
I've inserted some css code in the .app-content *.list-cell:filled:selected selector. The -fx-background-color: property has a new gradient value that will over ride the previous value of the focus selector that is being referenced in the caspian.css document. The focus selector and selected text will now be 40% larger. Please notice the -fx-scale-x and -fx-scale-y have been increased by 40%. There is also a new value of "90" in the -fx-padding: property that will effect the left side of the padding inside of the focus selector. It will shift the text inside of the focus selector over to the right. I have also assigned a drop shadow to the focus selector. Please notice the -fx-effect: dropshadow value.

ORACLE®

This is the result of the build. The focus selector is now 40% larger as is the selected text. There is now a slick glossy texture to the focus selector that is more "TV Centric."   The selected font is now shifted to the right for a more dynamic look and feel.

The focus selector also has a drop shadow to add more depth and ground it to the movie genre text panel. The vertical negative space between the unselected movie genre text needs to be increased so that it doesn't look so crowded. It will also visually balance the vertical height of the selected text in the focus selector. The focus selector also needs to be enlarged to span the entire width of the movie genre text panel. The next code insertion I make in the NetBeans IDE will address those issues.

```css
       //desktop//    padding: 2 10 3 10;
31     -fx-padding: 6 30 9 30;
32     -fx-background-color: -fx-shadow-highlight-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
33     //desktop//    -fx-background-radius: 5, 5, 4, 3;
34     -fx-background-radius: 15, 15, 12, 9;
35     //desktop//    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
36     -fx-background-insets: 0 0 -3 0, 0, 3, 6;
37     -fx-text-fill: -fx-text-base-color;
38     -fx-vpos: CENTER;
39     -fx-graphic-vpos: CENTER;
40  }
41  .app-content *.button:focused {
42     -fx-background-color: -fx-focus-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
       //desktop//          -fx-background-radius: 6.4, 5, 4, 3;
44     -fx-background-radius: 19.2, 15, 12, 9;
45     //desktop//    -fx-background-insets: -1.4, 0, 1, 2;
46     -fx-background-insets: -4.2, 0, 3, 6;
47  }
48  .app-content *.list-view {
49     -fx-padding:  50 3 50 3;
50     -fx-background-color: transparent;
51     -fx-background-insets: 0,1;
52     -fx-pannable: true
53
54  }
55
56
57  .app-content *.list-view:focused {
       //desktop//    -fx-padding: 1;
59     -fx-padding: 50 3 50 3;
60     -fx-background-color: transparent;
61     //desktop//    -fx-background-insets: -1.4, 0, 1;
62     -fx-background-insets: -4.2, 0, 3;
63     //desktop//    -fx-background-radius: -1.4, 0, 0;
64     -fx-background-radius: -4.2, 0, 0;
65  }
66
67  .app-content *.list-cell:filled:selected {
68     -fx-background: transparent;
       -fx-background-color: linear (0%, 0%) to (0%, 100%) stops (0%, #bed5fa) (10%, #6a93e6) (50%, #5b7cb4) (51%, #274b88) (90%, #27589e) (95%, #4f92d7);
70     -fx-text-fill: -fx-selection-bar-text;
71     -fx-padding: 6 6 6 90;
72     -fx-scale-x: 140%;
73     -fx-scale-y: 140%;
74     -fx-background-insets: 0 118 0 0;
75     -fx-effect: dropshadow( gaussian, black, 10, 0.4, 0, 7);
76  }
77
78  .app-content *.list-cell {
79     -fx-cell-size: 24;
       //desktop//    -fx-padding: 3;
81     -fx-padding: 24;
82     -fx-background-color: transparent;
83  }
84
```
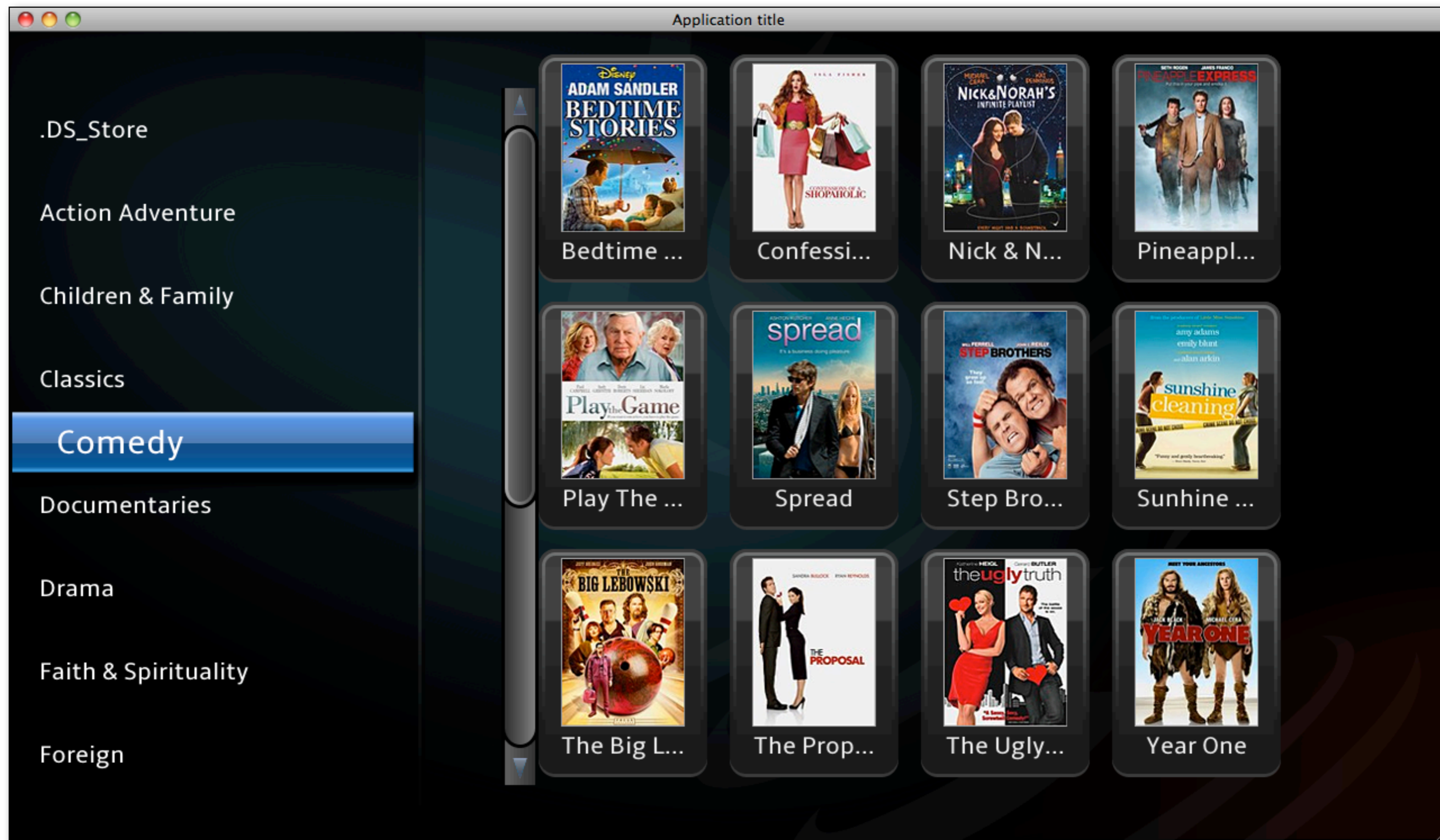
In the .app-content *.list-cell selector, the -fx-cell-size; psuedo-state has been assigned a new over ride value of "24." That new value will increase the vertical space between each of the movie genre options as well as increase the width of the focus selector to span the whole movie genre text panel. It will also move the movie grid control to the right and center it into correct alignment within our canvas.

ORACLE®

This is the result of the build. The width of the focus selector has now been enlarged to cover the entire span of the movie genre text panel. The movie grid has also been moved to the right and centered within our canvass. The vertical negative space between the unselected movie genre text has been increased to visually balance the vertical height of the focus selector and selected movie genre text. Now that the vertical space between each of the movie genre text options has been increased, all of the movie genre text selections do not fit within the border of the 720 px. tall canvas. The application code has evoked a vertical scroll bar class. This is as it should be, but it doesn't look very "TV Centric." The next css code insertions will address that issue.
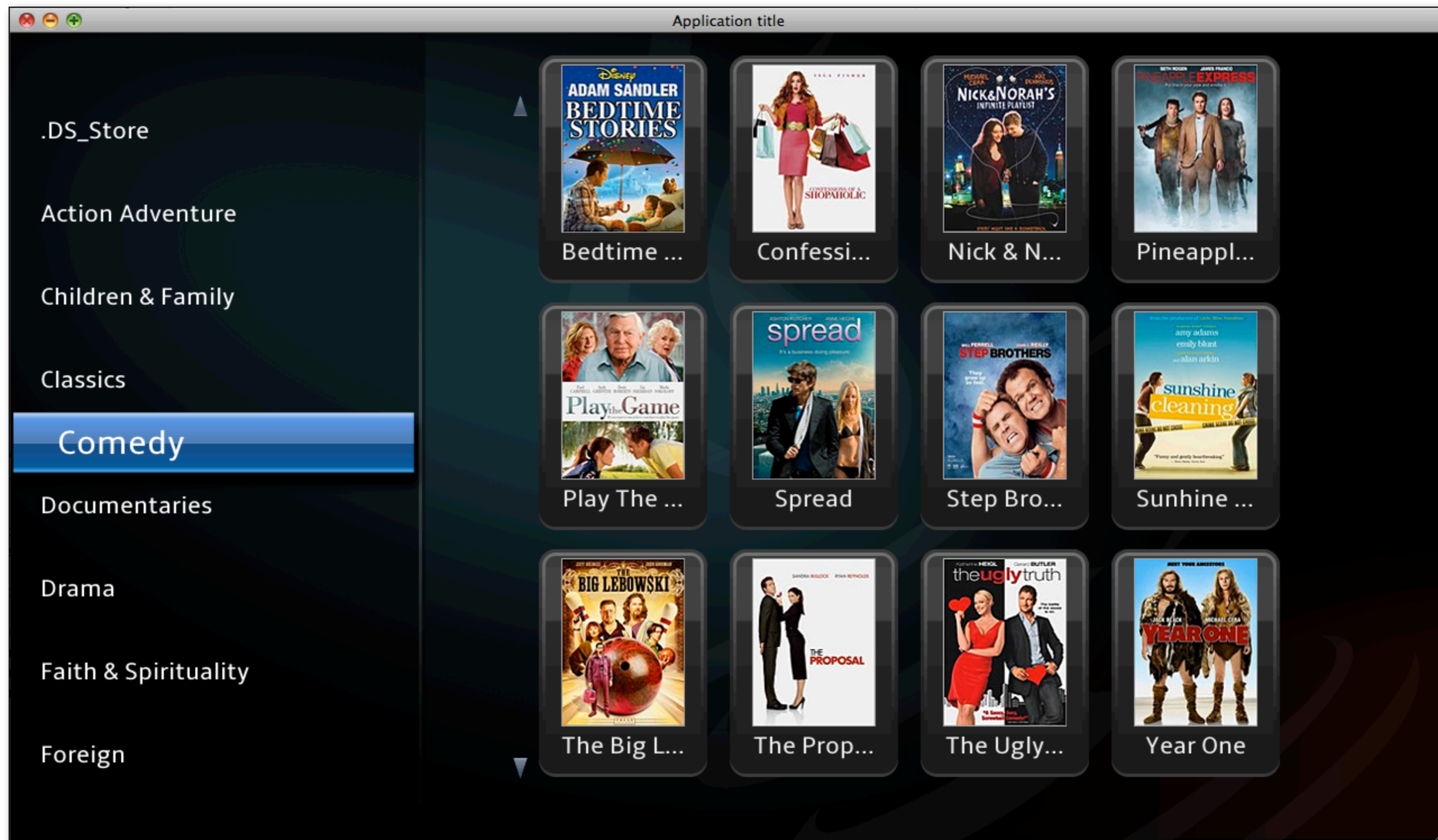
```
59        -fx-padding: 50 3 50 3;
60        -fx-background-color: transparent;
61        //desktop//    -fx-background-insets: -1.4, 0, 1;
62        -fx-background-insets: -4.2, 0, 3;
63        //desktop//    -fx-background-radius: -1.4, 0, 0;
64        -fx-background-radius: -4.2, 0, 0;
65    }
66
67    .app-content *.list-cell:filled:selected {
68        -fx-background: transparent;
        -fx-background-color: linear (0%, 0%) to (0%, 100%) stops (0%, #bed5fa) (10%, #6a93e6) (50%, #5b7cb4) (51%, #274b88) (90%, #27589e) (95%, #4f92d7);
70        -fx-text-fill: -fx-selection-bar-text;
71        -fx-padding: 6 6 6 90;
72        -fx-scale-x: 140%;
73        -fx-scale-y: 140%;
74        -fx-background-insets: 0 118 0 0;
75        -fx-effect: dropshadow( gaussian, black, 10, 0.4, 0, 7);
76    }
77
78    .app-content *.list-cell {
79        -fx-cell-size: 24;
        //desktop//    -fx-padding: 3;
81        -fx-padding: 24;
82        -fx-background-color: transparent;
83    }
84    .app-content *.scroll-bar:vertical {
85        -fx-background-color:transparent;
        //desktop//    -fx-background-insets: 0, 1;
87        -fx-background-insets: 0, 3;
88    }
89
90
91    .app-content *.scroll-bar:vertical *.thumb {
92        -fx-background-color:transparent;
        //desktop//    -fx-background-radius: 6, 5, 4;
94        -fx-background-radius: 18, 15, 12;
95        //desktop//    -fx-background-insets: 0, 1, 2;
96        -fx-background-insets: 0, 3, 6;
97    }
98
99
100   .app-content *.scroll-bar:vertical *.track {
101       -fx-background-color:transparent;
        //desktop//    -fx-background-radius: 6, 5;
103       -fx-background-radius: 18, 15;
104       //desktop//    -fx-background-insets:  0, 1;
105       -fx-background-insets: 0, 3;
106   }
107
```

I have inserted code that pertains to the .app-content *.scroll-bar:vertical class that over rides the previous attributes referenced in the Caspian.css document to a value of transparent.

ORACLE®

This is the result of the build. The vertical scroll bar class has been assigned a new value of transparent so it is no longer visible. The decrement and increment arrow classes still remain, but they look odd in their current position. The next css code insertion will align them to be above and below the movie genre text options within the vertical list view control.

```
 81        -fx-padding: 24;
 82        -fx-background-color: transparent;
 83    }
 84    .app-content *.scroll-bar:vertical {
 85        -fx-background-color:transparent;
 ⚠      //desktop//    -fx-background-insets: 0, 1;
 87        -fx-background-insets: 0, 3;
 88    }
 89
 90
 91    .app-content *.scroll-bar:vertical *.thumb {
 92        -fx-background-color:transparent;
 ⚠      //desktop//    -fx-background-radius: 6, 5, 4;
 94        -fx-background-radius: 18, 15, 12;
 95        //desktop//    -fx-background-insets: 0, 1, 2;
 96        -fx-background-insets: 0, 3, 6;
 97    }
 98
 99
100    .app-content *.scroll-bar:vertical *.track {
101        -fx-background-color:transparent;
 ⚠      //desktop//    -fx-background-radius: 6, 5;
103        -fx-background-radius: 18, 15;
104        //desktop//    -fx-background-insets:  0, 1;
105        -fx-background-insets: 0, 3;
106    }
107    .app-content *.scroll-bar:vertical *.increment-arrow {
 ⚠      //desktop//    -fx-padding: 4 6 0 0;
109    //    -fx-padding: 15 20 0 0;
110        -fx-background-color: -fx-mark-highlight-color, -fx-mark-color;
111        //desktop//    -fx-background-insets: 0 0 0 0, 1;
112    //    -fx-background-insets: 10 0 -3 0, 0;
113    //    -fx-shape: "M346.5,335.5L14,141-28-0.5L346.5,335.5z"
114        -fx-translate-y: 40;
115        -fx-translate-x: -270;
116        -fx-scale-x: 200%;
117    }
118    .app-content *.scroll-bar:vertical *.decrement-arrow {
 ⚠      //desktop//    -fx-padding: 4 6 0 0;
120     //  -fx-padding: 15 20 0 0;
121        -fx-background-color: -fx-mark-highlight-color, -fx-mark-color;
122        //desktop//    -fx-background-insets: 0 0 0 0, 1;
123     // -fx-background-insets: 10 0 -3 0, 0;
124     // -fx-shape: "M346.5,335.5L14,141-28-0.5L346.5,335.5z"
125        -fx-translate-y: -40;
126        -fx-translate-x: -270;
127        -fx-scale-x: 200%;
128    }
129
```

ORACLE®

The new css code insertions pertain to the .app-content *.scroll-bar:vertical *.increment-arrow and .app-content *.scroll-bar:vertical *.decrement-arrow selectors and contain new -fx-translate-y: and-fx-translate-x: values that will align them into proper positions above and below the movie text options panel within the vertical list view control. The code insertions also contain a psuedo-state -fx-scale-x: that over rides their previous width value to 200%.
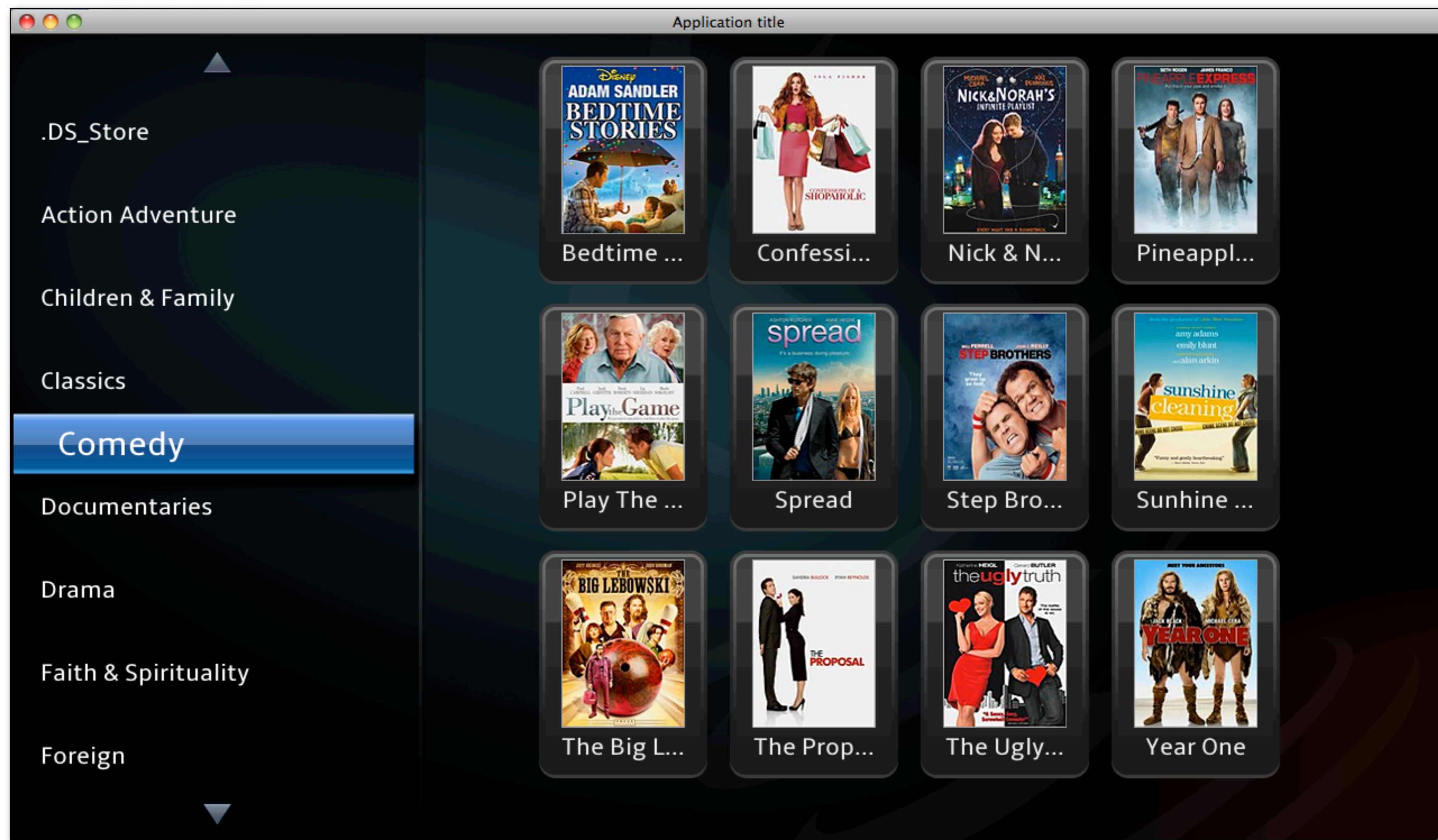
This is the result of the build. The decrement and increment arrows are now properly aligned within the movie genre text panel within the vertical list view control. The list view control is looking pretty good. Now it's time to focus on the movie grid control. The movie grid control looks a bit crowded. To increase the negative space between the individual movie button images, the movie button images will be reduced by 30%. The rounded corners of the movie button images should also be squared off to match the corners of the movie images.   The movie button images should also have a drop shadow assigned to them to add a sense of depth and anchor them to the background image.
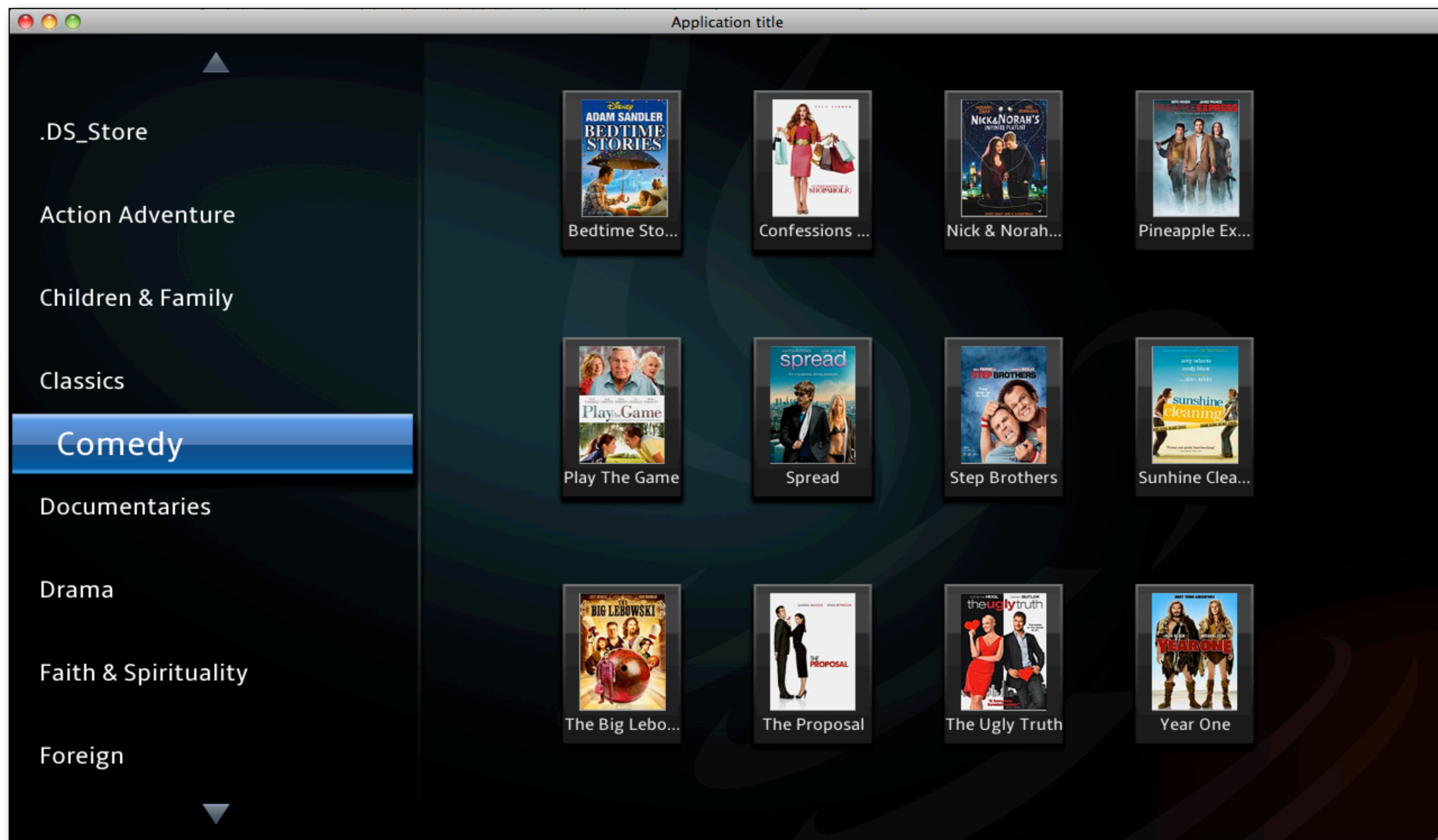
Left image:

```
 8
 9    */
10
11
12    .app-background {
13        -fx-background-image: url("background.png");
14        -fx-background-repeat: no-repeat;
15        -fx-background-position: left top;
16        -fx-padding-top: 0px;
17        -fx-margin-bottom: 0px;
18
19    }
20
21
22    .app-content *.button:armed {
23
24        -fx-color: red;
25        -fx-text-fill: white;
26
27    }
28
29    .app-content *.button {
30        //desktop//    padding: 2 10 3 10;
31        -fx-padding: 6 30 9 30;
32        -fx-background-color: -fx-shadow-highlight-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
33        //desktop//    -fx-background-radius: 5, 5, 4, 3;
34        -fx-background-radius: 15, 15, 12, 9;
35        //desktop//    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
36        -fx-background-insets: 0 0 -3 0, 0, 3, 6;
37        -fx-text-fill: -fx-text-base-color;
38        -fx-vpos: CENTER;
39        -fx-graphic-vpos: CENTER;
40    }
41    .app-content *.button:focused {
42        -fx-background-color: -fx-focus-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
43        //desktop//        -fx-background-radius: 6.4, 5, 4, 3;
44        -fx-background-radius: 19.2, 15, 12, 9;
45        //desktop//    -fx-background-insets: -1.4, 0, 1, 2;
46        -fx-background-insets: -4.2, 0, 3, 6;
47    }  .app-content *.list-view {
48    .app-content *.list-view {
49        -fx-padding:  50 3 50 3;
50        -fx-background-color: transparent;
51        -fx-background-insets: 0,1;
52        -fx-pannable: true
53
54    }
55
56
```

Right image:
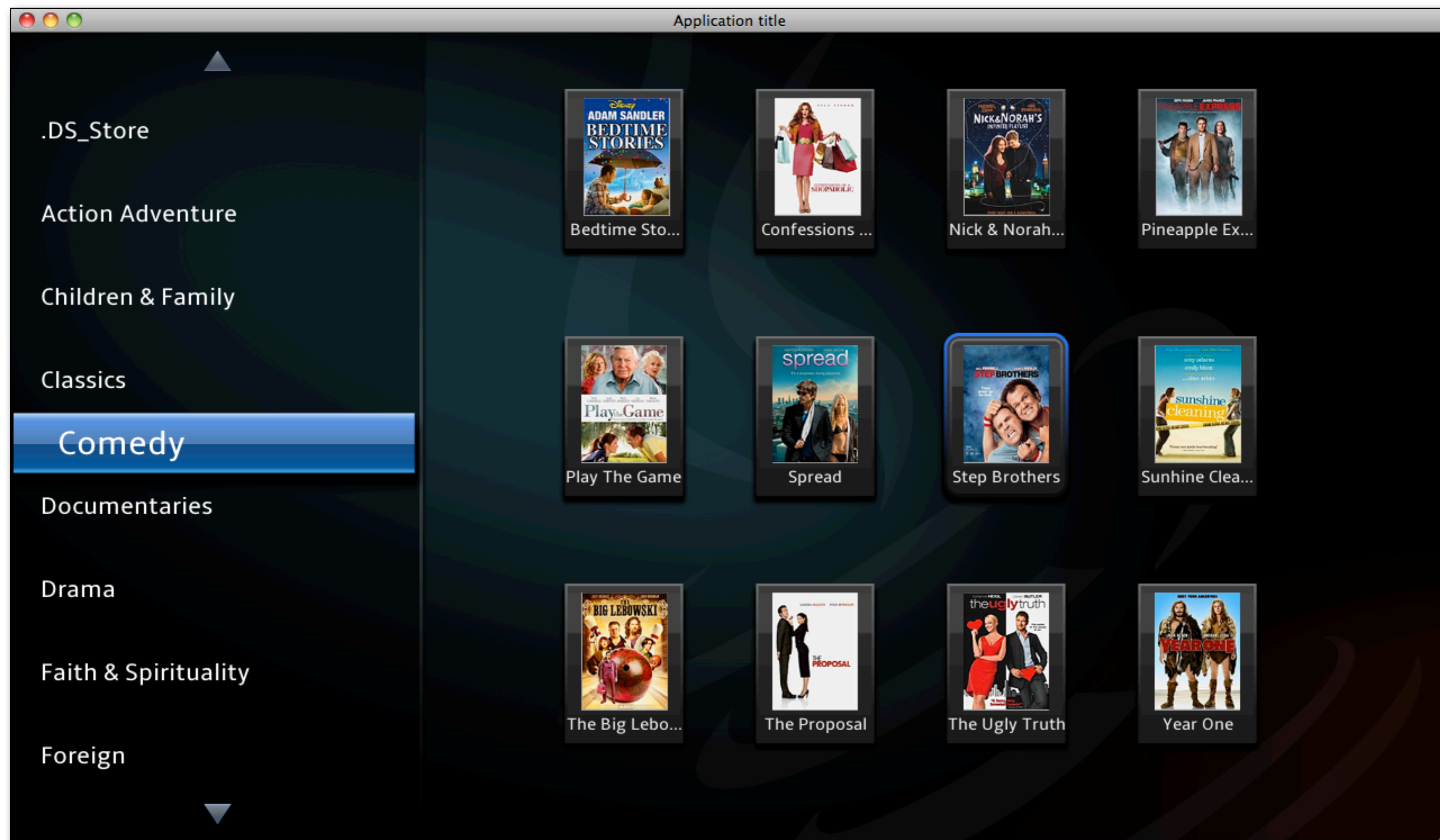
```
 8
 9    */
10
11
12    .app-background {
13        -fx-background-image: url("background.png");
14        -fx-background-repeat: no-repeat;
15        -fx-background-position: left top;
16        -fx-padding-top: 0px;
17        -fx-margin-bottom: 0px;
18
19    }
20
21
22    .app-content *.button:armed {
23
24        -fx-color: red;
25        -fx-text-fill: white;
26
27    }
28
29    .app-content *.button {
30        //desktop//    padding: 2 10 3 10;
31        -fx-padding: 2 0 0 0;
32        -fx-background-color: -fx-outer-border, -fx-inner-border, -fx-body-color;
33        //desktop//    -fx-background-radius: 5, 5, 4, 3;
34        -fx-background-radius: 0, 0, 0, 0;
35        //desktop//    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
36        -fx-background-insets: 0 0 -3 0, 0, 3, 6;
37        -fx-text-fill: -fx-text-base-color;
38        -fx-vpos: CENTER;
39        -fx-hpos: RIGHT;
40        -fx-graphic-vpos: CENTER;
41        -fx-scale-x: 70%;
42        -fx-scale-y: 70%;
43        -fx-effect: dropshadow( gaussian, black, 10, 0.4, 0, 7);
44    }
45
46    .app-content *.button:focused {
47        -fx-background-color: -fx-focus-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
48        //desktop//        -fx-background-radius: 6.4, 5, 4, 3;
49        -fx-background-radius: 19.2, 15, 12, 9;
50        //desktop//    -fx-background-insets: -1.4, 0, 1, 2;
51        -fx-background-insets: -4.2, 0, 3, 6;
52    }
53    .app-content *.list-view {
54        -fx-padding:  50 3 50 3;
55        -fx-background-color: transparent;
56        -fx-background-insets: 0,1;
```

I'm going to swap in some new css code that will replace the selector -fx-background-radius: value from 15, 15, 12, 9: (left image) to 0, 0, 0, 0; (right image.) This will square off the corners of the movie button images. I will also insert -fx-scale-x: and -fx-scale-y: selectors that have a value of 70% that will over ride the previous value of 100%. This will reduce the size of the movie button images by 30%. An -fx-effect: selector with a dropshadow value has also been inserted into the app.css document. (left image.)

ORACLE®

This is the result of the build. The movie button images have new been reduced by 30% and now the movie button grid doesn't appear to be crowded. The corners of the movie button images have also been squared off to match the corners of the movie images. There is also a drop shadow assigned to the movie button images that adds depth to the scene and anchors the movie button images to the background.

The selected state of the movie button image still retains the rounded corner. This needs to be changed to match the squared off corners of the unselected movie button images. The selection of the movie button image would also appear more dynamic if the selected movie button image were enlarged upon selection. The next css code insertions will address those issues.

```
29    .app-content *.button {
      //desktop//     padding: 2 10 3 10;
31    -fx-padding: 2 0 0 0;
32    -fx-background-color: -fx-outer-border, -fx-inner-border, -fx-body-color;
33    //desktop//    -fx-background-radius: 5, 5, 4, 3;
34    -fx-background-radius: 0, 0, 0, 0;
35    //desktop//    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
36    -fx-background-insets: 0 0 -3 0, 0, 3, 6;
37    -fx-text-fill: -fx-text-base-color;
38    -fx-vpos: CENTER;
39    -fx-hpos: RIGHT;
40    -fx-graphic-vpos: CENTER;
41    -fx-scale-x: 70%;
42    -fx-scale-y: 70%;
43    -fx-effect: dropshadow( gaussian, black, 10, 0.4, 0, 7);
44    }
45
46    .app-content *.button:focused {
47    -fx-background-color: -fx-focus-color, -fx-outer-border, -fx-inner-border, -fx-body-color;
      //desktop//         -fx-background-radius: 6.4, 5, 4, 3;
49    -fx-background-radius: 19.2, 15, 12, 9;
50    //desktop//    -fx-background-insets: -1.4, 0, 1, 2;
51    -fx-background-insets: -4.2, 0, 3, 6;
52    }
53    .app-content *.list-view {
54    -fx-padding:   50 3 50 3;
55    -fx-background-color: transparent;
56    -fx-background-insets: 0,1;
57    -fx-pannable: true
58
59    }
60
61
62    .app-content *.list-view:focused {
      //desktop//    -fx-padding: 1;
64    -fx-padding: 50 3 50 3;
65    -fx-background-color: transparent;
66    //desktop//    -fx-background-insets: -1.4, 0, 1;
67    -fx-background-insets: -4.2, 0, 3;
68    //desktop//    -fx-background-radius: -1.4, 0, 0;
69    -fx-background-radius: -4.2, 0, 0;
70    }
```

```
29    .app-content *.button {
      //desktop//     padding: 2 10 3 10;
31    -fx-padding: 2 0 0 0;
32    -fx-background-color: -fx-outer-border, -fx-inner-border, -fx-body-color;
33    //desktop//    -fx-background-radius: 5, 5, 4, 3;
34    -fx-background-radius: 0, 0, 0, 0;
35    //desktop//    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
36    -fx-background-insets: 0 0 -3 0, 0, 3, 6;
37    -fx-text-fill: -fx-text-base-color;
38    -fx-vpos: CENTER;
39    -fx-hpos: RIGHT;
40    -fx-graphic-vpos: CENTER;
41    -fx-scale-x: 70%;
42    -fx-scale-y: 70%;
43    -fx-effect: dropshadow( gaussian, black, 10, 0.4, 0, 7);
44    }
45    .app-content *.button:focused {
46    -fx-background-color: -fx-focus-color, -fx-pressed-base;
      //desktop//         -fx-background-radius: 6.4, 5, 4, 3;
48    -fx-background-radius: 0, 0, 0, 0;
49    //desktop//    -fx-background-insets: -1.4, 0, 1, 2;
50    -fx-background-insets: -4.2, 0, 3, 6;
51    -fx-scale-x: 100%;
52    -fx-scale-y: 100%;
53    }
54
55    .app-content *.list-view {
56    -fx-padding:   50 3 50 3;
57    -fx-background-color: transparent;
58    -fx-background-insets: 0,1;
59    -fx-pannable: true
60    }
61
62    .app-content *.list-view:focused {
      //desktop//    -fx-padding: 1;
64    -fx-padding: 50 3 50 3;
65    -fx-background-color: transparent;
66    //desktop//    -fx-background-insets: -1.4, 0, 1;
67    -fx-background-insets: -4.2, 0, 3;
68    //desktop//    -fx-background-radius: -1.4, 0, 0;
69    -fx-background-radius: -4.2, 0, 0;
70    }
```

I've swapped in the .app-content *.button:focused css code that over rides the previous value of -fx-background-radius: 19.2, 15, 12, 9; (left image) with a new value of 0, 0, 0, 0;. (right image.) This will square off the rounded corners of the selected movie button images. A new value of -fx-pressed-base; has also over ridden the previous value of -fx-body-color; linked to the -fx-background-color: selector. This will over ride the previous -fx-body-color; to a darker hue. The -fx-scale-x: and -fx-scale-y: values have been over ridden to a value of 100%. This will give the illusion that the selected movie button image is enlarging upon selection, but it is really maintaining its original size. The unselected movie button images have been reduced by 30%.

ORACLE®

The new build has the selected movie button image enlarged from the unselected state. It also has the squared off corners to match the movie image. The body color of the selected movie image is also a darker hue.

By modifying and inserting the css code into the simple tv app.css document, we have changed the appearance of the ux controls to a more "*TV Centric*" presentation without ever touching the source code of the application. That is pretty powerful!!!

Using CSS to Style JavaFX UI Controls
The Visual Designer's Experience

{ **Eileen Bugée and Tim Dunn** }
Principal Visual Designers | Oracle Corporation

Our e-mail addresses:
eileen.bugee@oracle.com
timothy.dunn@oracle.com

Our slides are posted here:
http://openworld.vportal.net/