

Achieving the Perfect Layout with ADF Faces RC



Peter Koletzke
Technical Director &
Principal Instructor



quovera

Premise and Objective

- Every new technology uses a different strategy for UI layout
 - Oracle Forms, HTML, Java apps
- This strategy is often a well-kept secret
 - You often need to discover it or invent one
- The presentation discusses a strategy and the palette of tools you use with it
- It also provides some tips
 - All for ADF Faces RC, JDev 11g



quovera

2

Agenda

- The strategy
- The palette
- The tips and techniques

White paper includes
hands-on practices.



quovera

3

The Problem

Sixteen million colors
in your palette
are hard for any artist,
especially a beginner,
to turn down.

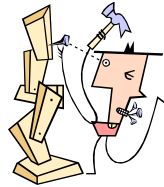
–Buffy Sainte-Marie
(1941-)

quovera

4

Strategy Principles

- Design the page using layout components
- Use facets in the layout components
- Set properties for behavior

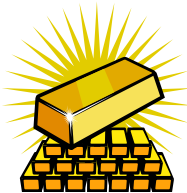


Most Important Principle

Work Declaratively

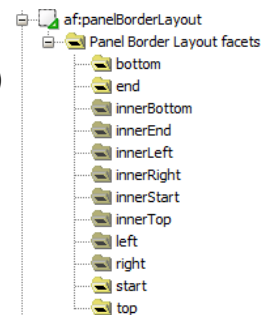
Know Thy Layout Components

- ADF Faces RC components that hold other components
 - Similar to `<table>` in HTML
 - A.k.a., “container components”
 - Maintain relative layout across browsers
- Allow for sophisticated layouts
 - Nest layout components within layout components
 - Virtually limitless possibilities
- Hide or display the contents using the layout component properties



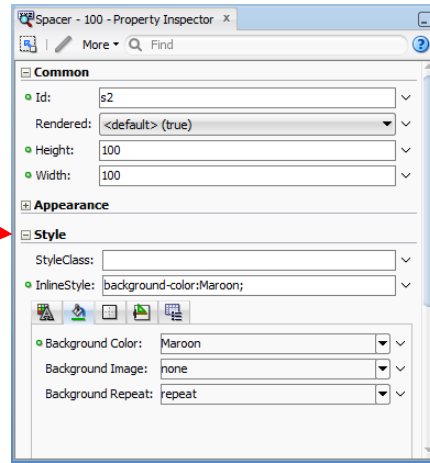
Know Thy Facets

- Subcomponent providing special functionality for layout components
 - `<f:facet>` component (JSF library)
 - Nested within other component
 - Like `af:panelBorderLayout`
- One common function: precise placement, for example:
 - Footer facet:
 - Contained components always appear at the bottom of the component
 - Center facet
 - Contained components always appear in the middle area of the component



Know Thy Properties

- Properties modify the behavior drastically
- Example, af:spacer
 - Height=100
 - Width= 100
- Don't forget the Style properties for layout components



Agenda

- The strategy
- The palette
- The tips and techniques



Palette Item: Layout Components

- Most are on the Layout page of the Component Palette
 - Use them to hold other components
- af:panelHeaderLayout
 - Region title
 - Stack components under it
- af:panelGroupLayout
 - Lay out components in a row or in a column

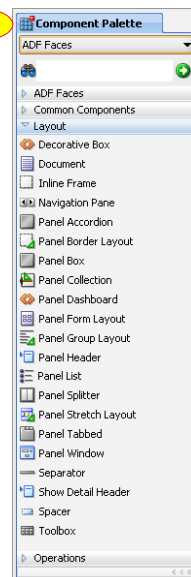


horizontal

vertical

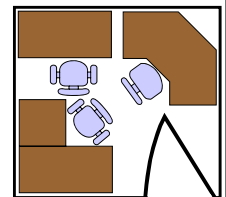


11.1.1



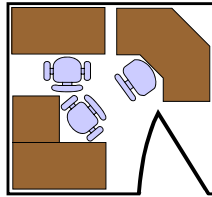
Some Layout Components

- af:panelAccordion
 - Like MS Outlook
 - Alternative to tabs
- af:panelStretchLayout
 - Expands contained components to fill width
 - Use if another container cuts a component

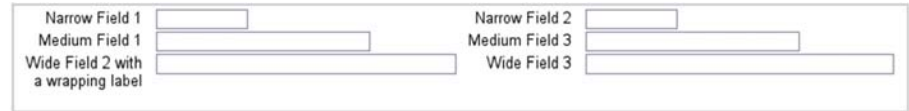


More Layout Components

- af:popup
 - Drop in an af:dialog or af>window
 - Drop af:showPopupBehavior into an action item (button or menu choice)
- af:menuBar
 - Panel Menu Bar
 - Creates menu area, drop in af:menu then af:menuItem
- af:panelLabelAndMessage
 - Provides a prompt for a group of objects
 - FirstName and LastName fields with a prompt of “Name”
 - Use inside af:panelFormLayout



Panel Form Layout

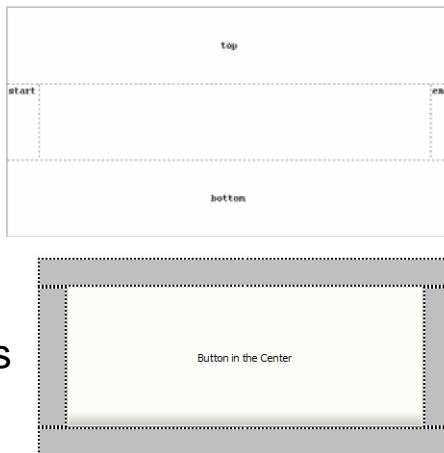


- Lay out fields in rows and columns
 - Perfect for most input forms
- Right justifies prompts
- Left justifies fields
- Tab order is down the first column, then across to the second column
 - Not necessarily intuitive
 - Workaround discussed later



Panel Border Layout

- af:panelBorderLayout
 - Predefined layout areas
 - Uses facets to hold the contents of each area
 - Start, end, top, bottom (and more)
- Center area stretches its contents to fill the area



Even More Layout Components

- af:panelSplitter
 - Split pane control
 - Optional: user can move the drag bar
 - Horizontal or
 - Vertical
- af:calendar
 - MS Outlook style
- af:carousel
 - Good for visual browsing



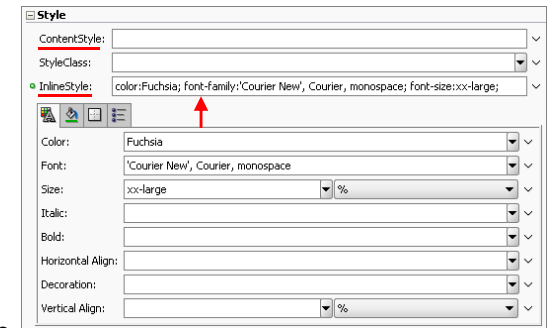
Important Property Setting

- `styleClass=AFStretchWidth`
 - Use this so container components (like `af:table`) fill their width
- Do not use `width=100%`
 - Not as portable



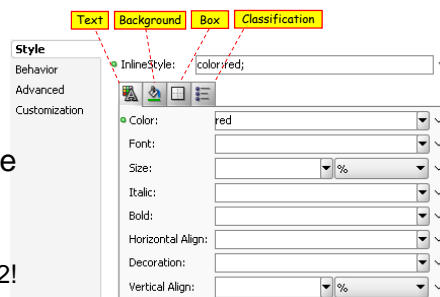
Modifying the Visual Aspects

- Skins
 - First and foremost – get this right
- Properties - secondary
 - *ContentStyle*
 - For data inside the component (foreground)
 - *InlineStyle*
 - Set from tab area below it
 - Or just type it in
 - Last resort, though



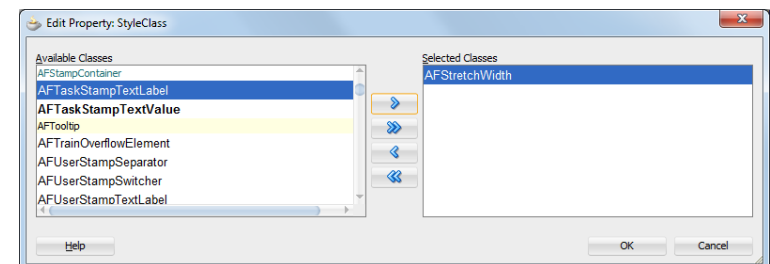
inlineStyle == Last Resort?

- Reasons
 - Dialog is the same for all components
 - Some components ignore some or all settings
 - Other properties may achieve the goal
 - Not reusable
 - Consider a skin change instead
 - Will apply universally
 - New skin editor in 11.1.2!



Another Visual Aspects Property

- *StyleClass*
 - Equivalent to the HTML *class* property
 - Apply existing style sheet selectors
 - Can apply more than one to a component



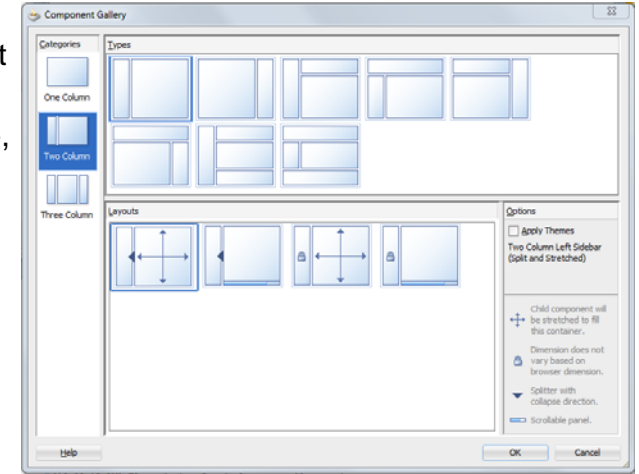
Agenda

- The strategy
- The palette
- The tips and techniques



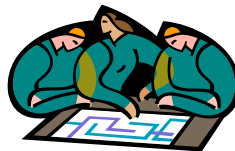
Don't Invent a Layout

- Quick Start Layouts: prebuilt functionality
- Access this window when creating the page, template, or page fragment
- Find by category, type, and layout.
- Each QSL has a "name."
- Copy the tags to existing pages



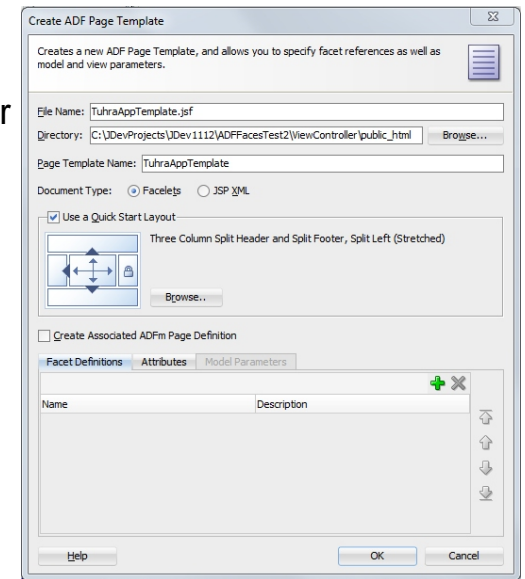
More Perfect Layout Tips

- Design page fragments, not separate pages
- Use af:spacer to fine tune placement
 - Sparingly however!
- For scrollbars, use af:panelGroupLayout with *Layout = scroll*
- Generalize layout into a template
 - Once per project/enterprise
 - Consistently repeatable
 - Easy to update



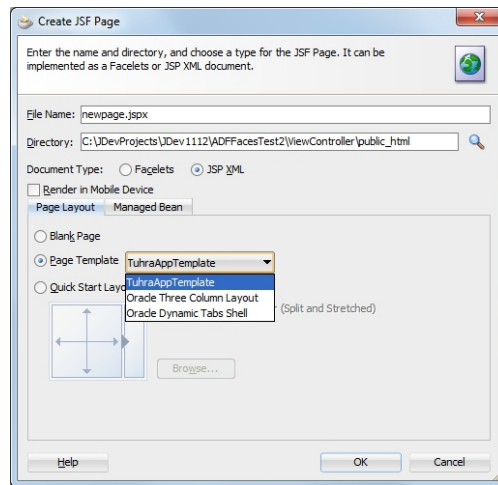
Create a Template

- New gallery item for JSF page template
- Define facets
 - Your own layout areas
- Add attributes
 - Can transfer data from page to template
- Add container components



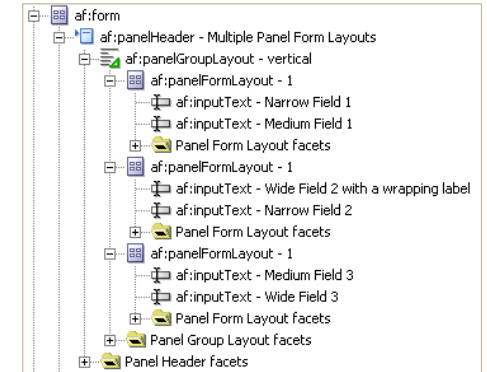
Use the Template

- Application's templates appear in the JSF Page dialog
 - Page Template pulldown
- The layout elements are referenced from the template



Workaround for Tab Order Issue

- `af:panelGroupLayout`
 - Orientation = vertical
 - `af:panelFormLayout`
 - `af:inputText`
 - `af:inputText`
 - `af:panelFormLayout`
 - `af:inputText`
 - `af:inputText`
 - `af:panelFormLayout`
 - `af:inputText`
 - `af:inputText`
- Set *fieldWidth* and *labelWidth* for all `af:panelFormLayout` components the same
- White paper contains hands-on practice with steps



For This Example
 fieldWidth = 300px
 labelWidth = 100px
 Inline Style = width:800px

Result

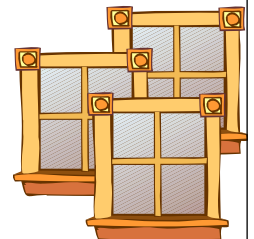
Narrow Field 1	<input type="text"/>	Medium Field 1	<input type="text"/>
Wide Field 2 with a wrapping label	<input type="text"/>	Narrow Field 2	<input type="text"/>
Medium Field 3	<input type="text"/>	Wide Field 3	<input type="text"/>

- Problems
 - Width of the browser window does not affect the container
 - Width of the column in the container is not based on the width of the widest field in that column
 - Not immune to the user increasing the font size in the browser
 - Takes a bit of experimentation

Hands-on practice in the white paper

Dialogs and Windows

- `af:dialog`
 - A window with preassigned buttons – modal or not
 - OK/Cancel
 - Yes/No/Cancel
 - Yes/No
 - Cancel
- `af:noteWindow`
 - Floating window containing read-only information
- `af:panelWindow`
 - A bordered box (“window”)
- `af:decorativeBox`
 - Window with prebuilt “themes” (color gradient schemes)



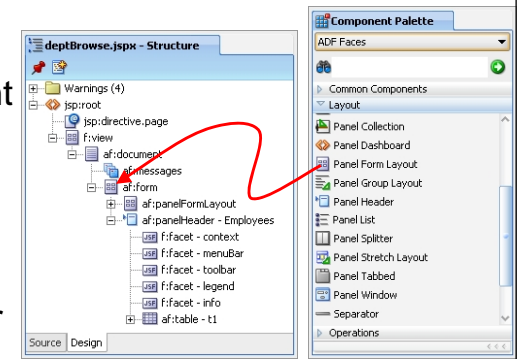
Popup Window Example

- Add an af:popup
 - Anywhere in the form
 - The visual editor will change to show just that window
 - Click in another node to switch out of that window editor
- Add an af:panelWindow
 - Add contents to the window
- Drop an af:showPopupBehavior onto a button or link



Tip: Use the Structure Window

- Drop on top of container into which you want the component to appear
- Much more accurate
- Other options
 - Click the component after selecting the Structure window node
 - Use the bread crumbs in the editor



Design This Container

- An option in the pulldown menu for the layout component



- Allows you to resize container elements visually
 - This action changes the property values
- Another tip: Assign color to the borders so you can see the containers



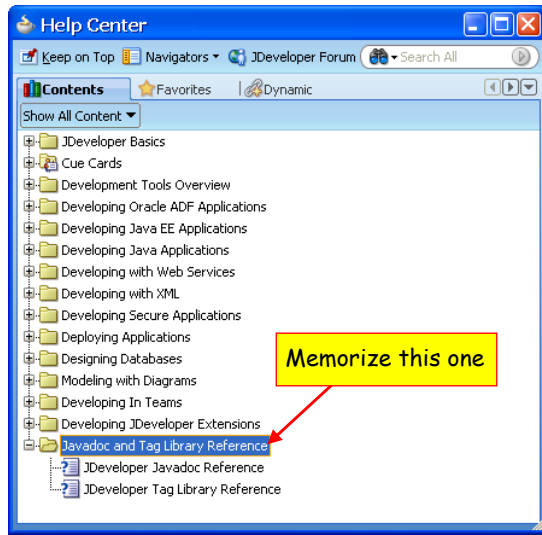
Approaching a Layout

- Use Quick Start Layouts
- Use templates
- Familiarize yourself with the palette
 - Components, facets, properties
- Know where to find information
- Above all:

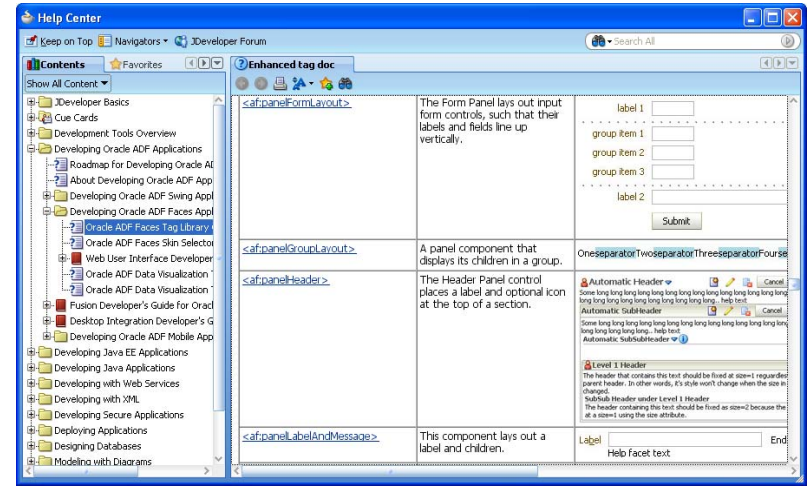
Work Declaratively

JDeveloper Help Center

- Help | Table of Contents
- Search engine
- Link to JDev forum
- Favorites tab
- Dynamic tab
 - Context-sensitive list based on the task at hand
- Opening a topic opens another tab

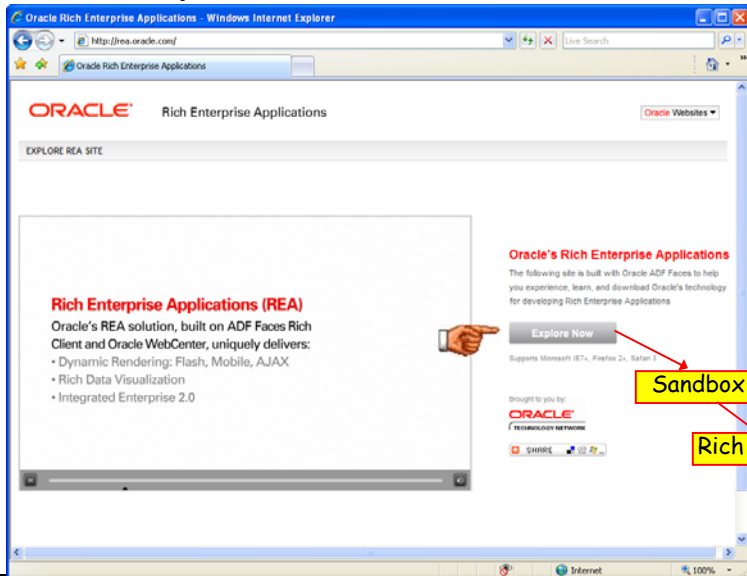


Visual Component Guide

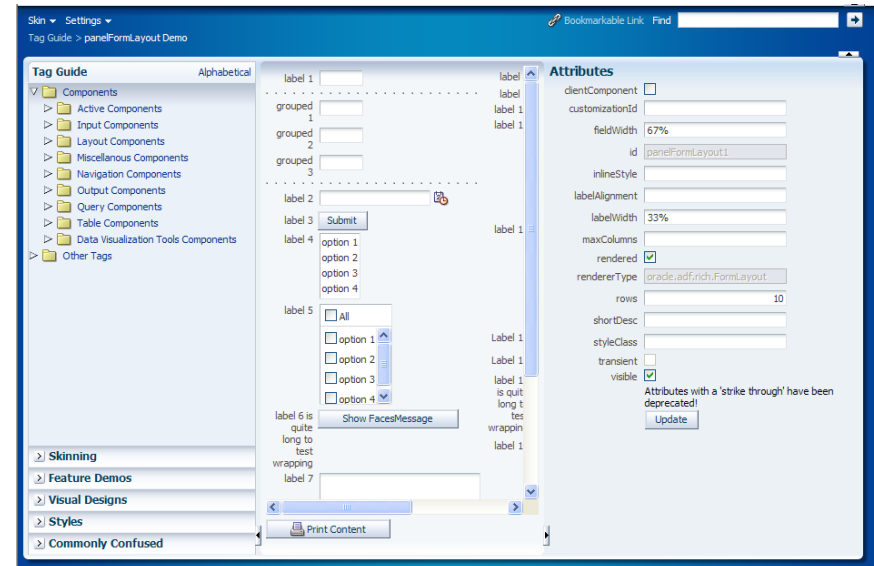


- JDev Help Center (help system)
 - Search for “enhanced tag doc” (local doc)

http://rea.oracle.com



af:panelFormLayout Demo Page

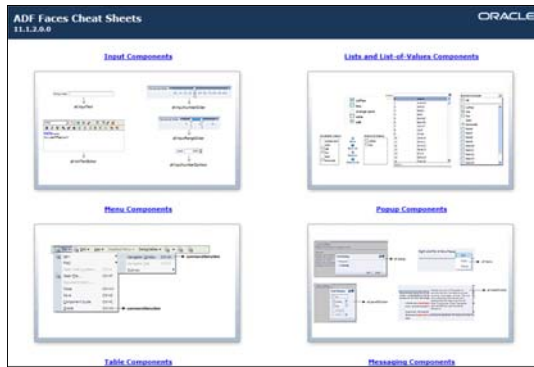


ADF Faces RC Website

- <http://www.oracle.com/technetwork/developer-tools/adf/overview/index-092391.html>

★ • *The Web UI Developer's Guide for Oracle ADF*

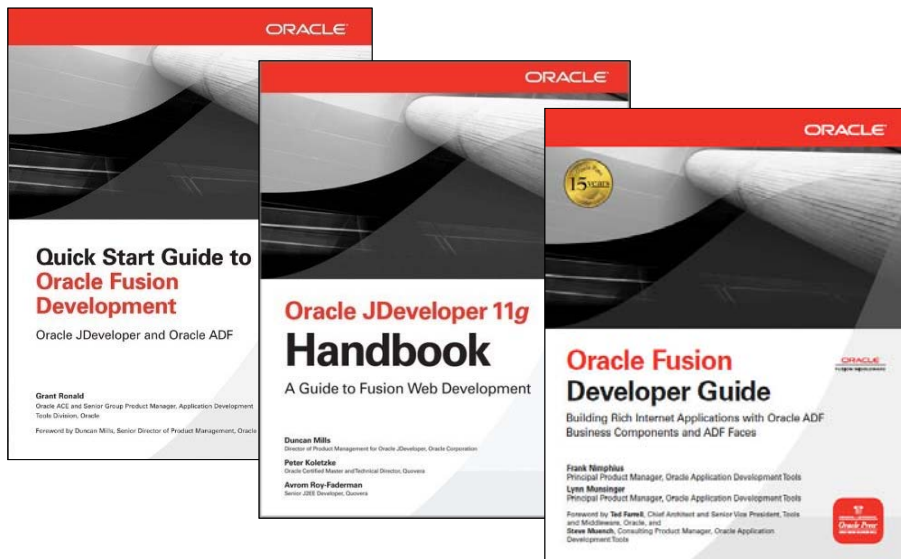
- Tutorials
- Demos
- Cheat sheets



Useful Oracle Blogs

- Martin Deh
 - martindeh.blogspot.com
 - “ADF Layout Overview and Best Practices”
- Steve Muench
 - blogs.oracle.com/smuenchadf
- Frank Nimphius
 - thepeninsulasedge.com/frank_nimphius/
- Shay Schmeltzer
 - blogs.oracle.com/shay

Oracle Press Resources



I'd Hammer in the Morning

All parts should go together
without forcing...
By all means,
do not use a hammer.

— IBM Maintenance Manual (1925)

Summary

- The “secret” strategy:
 - Know your palette: Layout components, facets, properties
 - Work declaratively
 - Collect tips and techniques
- Follow the hands-on practices in the white paper
- Your layouts **can** achieve perfection!



QUOVERA

41

JW MARRIOTT
SAN ANTONIO HILL COUNTRY

ODTUG
Kscope12

SAN ANTONIO, TEXAS * JUNE 24-28

www.kscope12.com



- www.quovera.com for files
- Books co-authored with Dr. Paul Dorsey, Avrom Roy-Faderman, & Duncan Mills

QUOVERA

<http://www.quovera.com>

- Founded in 1995 as Millennia Vision Corp.
- Profitable for many years without outside funding
- Consultants each have 10+ years industry experience
- Strong High-Tech industry background
- 200+ clients/300+ projects
- JDeveloper Partner
- More technical white papers and presentations on the web site

QUOVERA

43