Look & Feel - ein praktischer Leitfaden zur Layout-Gestaltung von ADF-Applikationen

Jürgen Menge
Oracle Deutschland B.V. & Co. KG

Schlüsselworte:

Oracle ADF, ADF Faces, Layout, Skinning

Einleitung

Bei der Entwicklung von Web-Applikationen werden zumeist hohe Anforderungen an das Layout gestellt. Gerade bei der Entwicklung eines Prototypen ist der Eindruck, den das Layout beim Auftraggeber hinterlässt, für die erfolgreiche Weiterführung des Projektes entscheidend. Verfolgt man den Ansatz, mit der entwickelten Applikation einen möglichst breiten Einsatzbereich abzudecken, sollte man auf eine weitgehende Unabhängigkeit:

- vom verwendeten Browser
- von der Bildschirm-Auflösung
- vom eingesetzten Endgerät

achten.
Da die Verantwortung für die Anzeige außerhalb der eigentlichen Applikation liegt, sollten deshalb alle Festlegungen vermieden werden, die diese Unabhängigkeit einschränken.
Dies ist ein wesentlicher Unterschied zu Oracle Forms-Applikationen, bei denen die Festlegung des Layouts zumeist direkt (explizit oder über Referenz) für jede sichtbare Komponente getroffen werden. Was zunächst die Arbeit des Entwicklers deutlich vereinfacht, ist aus langfristiger Sicht weit weniger produktiv, da in der Praxis beispielsweise unterschiedliche Varianten der Applikation für unterschiedliche Auflösungen (weiter-)entwickelt werden müssen.

Skinning

Skins beeinflussen das Look&Feel einer gesamten Applikation und können zur Laufzeit umgeschaltet werden. In Oracle ADF basieren die Skins auf dem Standard CSS 3.0. Zur Laufzeit werden aus den Definitionen CSS 2.0-Dateien für die darzustellenden Seiten generiert. Sie beinflussen das Verhalten des Renderers und ermöglichen die Festlegung von Farben, Fonts, Icons, Properties, sichtbaren Texten u.ä.. Mit Oracle ADF 11*g* werden einige vordefininierte Skins ausgeliefert (*blafplus-*, fusion, fusion-projector, simple*). Diese können um eigene Skins ergänzt werden.

Page Templates

Mit Oracle ADF 11*g* wurde ein leistungsfähiger Mechanismus zur Verwendung von Seitenvorlagen (Page Templates) implementiert, so dass Änderungen am Template unmittelbar zur Laufzeit wirksam werden. Durch die Verwendung von Page Templates kann die Seitenstruktur einer Applikation standardisiert werden. Änderungen am Layout können zentral vorgenommen werden und wirken sich unmittelbar auf die gesamte Applikation aus.

Layout Container

Die Oracle ADF Rich Client-Bibliothek enthält eine Vielzahl von Komponenten, die als Layout Container für andere funktionale Komponenten (z.B. Input Text-Felder) dienen. Layout Container sind in der Regel ineinander geschachtelt, beeinflussen sich gegenseitig und nutzen den Platz im Layout auf unterschiedliche Weise. Der Prozess, in dem die Komponenten untereinander den Platz und die Position im Layout aushandeln, wird als *Geometry Management* bezeichnet. Dieser Prozess wird durch eine Notifikation vom Browser über ein erforderliches Resizing nach einer Interaktion mit dem Benutzer (Load, Resize des Browser, Partial Page Rendering, Visibility On/off, Explizites Resize der Komponente) ausgelöst.

Stretchbare Komponenten (Stretching Components)
Die Layout-Komponente füllt einen zur Verfügung stehenden Bereich im Layout vollständig aus, d.h. passt sich in ihrer Größe diesem Bereich an. Diese Fähigkeit ist von der Komponente selbst und von der Parent-Komponente abhängig, sofern die zentrale Einstellung für das Dokument (*af:document*) *maximized=true* getroffen wurde. Komponenten, die als Parent das Stretching der untergeordneten Komponenten erlauben sind beispielsweise:

- panelSplitter
- paneStretchLayout
- panelDashboard

Bewegliche Komponenten (Flowing Components)
Die Layout-Komponente nutzt einen festgelegten Bereich im Layout, die Ansicht kann aber mittels Scrollbar innerhalb dieses Bereiches verschoben werden.

Übergangs-Komponente (Transition Components)
Für den Übergang zwischen äußeren stretchbaren Komponenten und inneren beweglichen Komponenten wird eine Komponente für den Übergang benötigt, d.h. die Komponente ist selbst stretchbar, erlaubt aber ihren Kindern nur das Scrollen im festgelegten Bereich. Dies kann beispielsweise durch die Verwendung des *PanelGroupLayout* mit der Einstellung *layout=scroll* oder *layout=vertical* erreicht werden.

Der äußere Seitenaufbau sollte zunächst mit stretchbaren Komponenten beginnen, um den im Browser verfügbaren Platz zu nutzen. Mit dem Einsatz der Übergangs-Komponente wird der Übergang zu beweglichen Komponenten eingeleitet. Diese bilden feststehendene Inseln im Layout, wobei der Bildausschnitt jeder Komponente durch Scrollbars verschoben werden kann.

Unbedingt vermieden werden sollte:

- die Verwendung von stretchbaren Komponenten innerhalb von beweglichen Komponenten.
- die Verwendung von beweglichen Komponenten innerhalb von anderen beweglichen Komponenten mit Scrollbars, da ineinandergeschachtelte Bereiche mit mehreren Scrollbars die Benutzer verwirren.
- die Verwendung von Inline Style-Attributen (Höhe, Breite, ..) für Komponenten.

Da das Geometry Management Ressourcen bindet, sollte die Zahl der unterhalb einer Parent-Komponente mittels Geometry Management verwaltenen Child-Komponenten möglichst gering gehalten werden. Einen guten Ausgangspunkt für das Layout erhält man durch die Verwendung eines *Quick Start Layouts* beim Anlegen einer neuen Seite mit Hilfe des *JSF Page Wizards* im *Oracle JDeveloper*.

Das folgende Beispiel zeigt die Darstellung einer Tabelle innerhalb eines stretchbaren *PanelGroupLayouts* (*Layout=vertical*). Während die *Panel Group* sich der verfügbaren Breite anpasst, ist die Tabelle als Child innerhalb der Group beweglich und mit einem Scrollbar versehen.

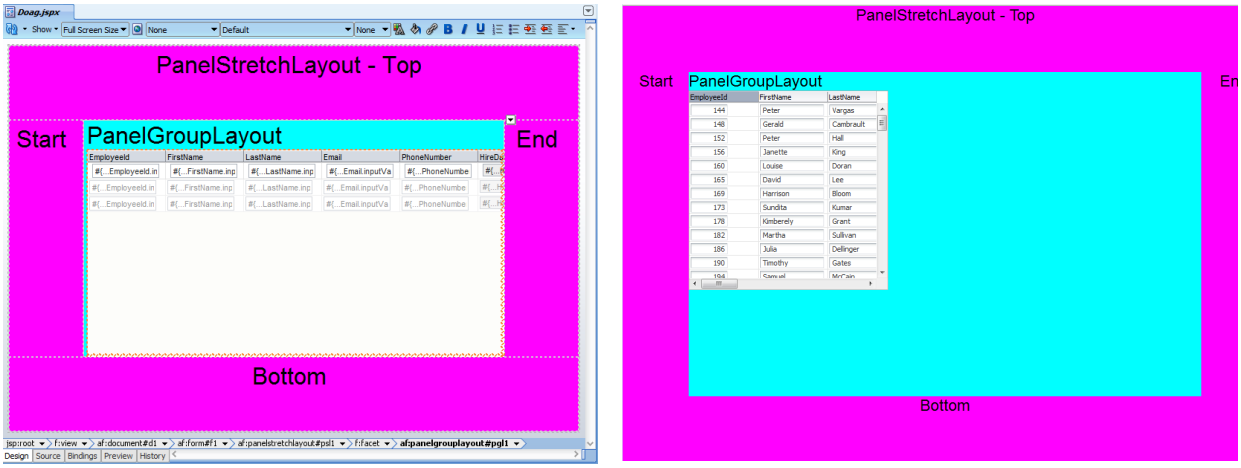

Abb. 1: Stretchbares PanelGroupLayout mit beweglicher af:table (Oracle JDeveloper und Ergebnis im Browser)

Wird die Tabelle direkt im zentralen Facet des Panel Stretch Layout platziert, nutzt sie die gesamte zur Verfügung stehende Breite im Layout.

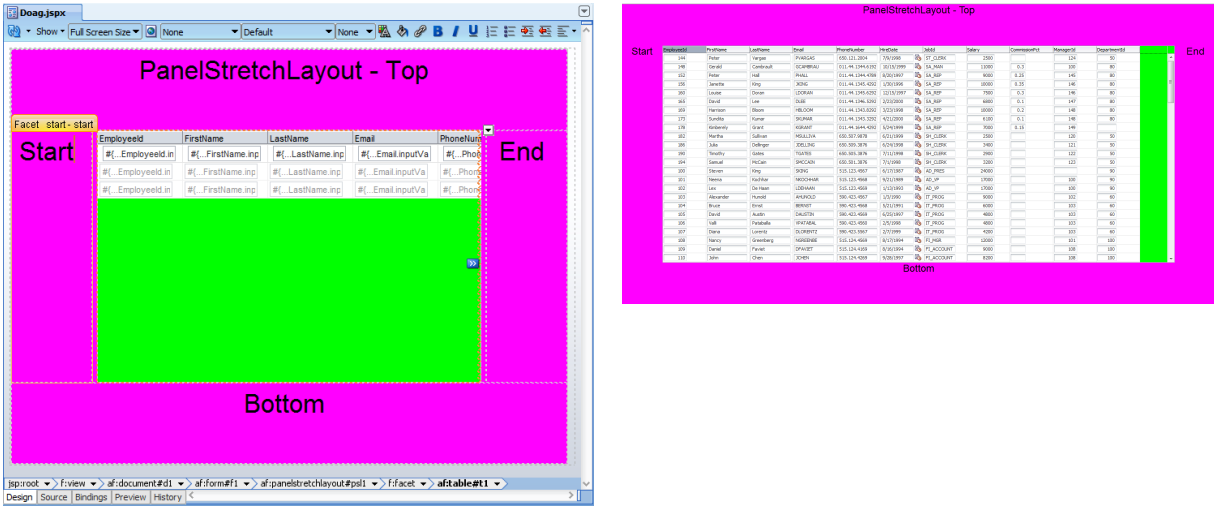

Abb. 2: PanelStretchLayout mit af:table (Oracle JDeveloper und Ergebnis im Browser)

Für die praktische Arbeit im Layout hier noch zwei Tipps:

- Durch Einfärben der Komponenten wird die tatsächliche Ausdehnung innerhalb des hierarchischen Designs sowohl im *Oracle JDeveloper* als auch im Browser sichtbar. So lassen sich die z.T. komplexen Wechselwirkungen bei der Verwendung bestimmter Komponenten besser nachvollziehen.
- Änderungen im Layout (z.B. Hinzufügen eines Layout Containes) erfordern kein erneutes Kompilieren der Seite. Es reicht aus, die Änderungen im JDeveloper zu speichern und die laufende Applikation durch Reload im Browser erneut zu starten. Dies erspart viel Zeit, da der Zyklus von Änderung und Kontrolle der Ergebnisse bei der Gestaltung des Layouts häufig durchlaufen werden muss.

Kontaktadresse:

Dr. Jürgen Menge
Oracle B.V. & Co. KG
Riesstr. 25
D-80992 München

Telefon: +49 (0) 89-1430 2239
Fax: +49 (0) 89-1430 2150
E-Mail juergen.menge@oracle.com
Internet: www.oracle.com