Migration ADF 11g nach ADF 12c

Marcus Hammer
virtual7 GmbH
Karlsruhe

Schlüsselworte
Oracle Application Development Framework, Oracle Business Components, Internet Explorer, DB2

Einleitung

Das Application Development Framework in der Version 12c (12.1.3) ist bereits seit Juni 2014 verfügbar. Es ist bereits das zweite Release für die Version 12c, welches in erster Linie Fehler aus der Version 12.1.2 behoben hat. Trotzdem hat auch diese Version ihre Eigenheiten und aufgrund der größeren Änderungen gegenüber 11g auch viele Veränderungen mitgebracht.

Mit knapp 1,9 Millionen Versicherten, 3000 Mitarbeitern und Prämieneinnahmen von 5,7 Milliarden Schweizer Franken ist die Helsana-Gruppe der führende Kranken- und Unfallversicherer der Schweiz. Für die interne Aufgaben- und Dokumentenverwaltung wird eine Webanwendung auf Basis von Oracle Application Development Framework (ADF) und Oracle WebCenter Content (WCC) eingesetzt.

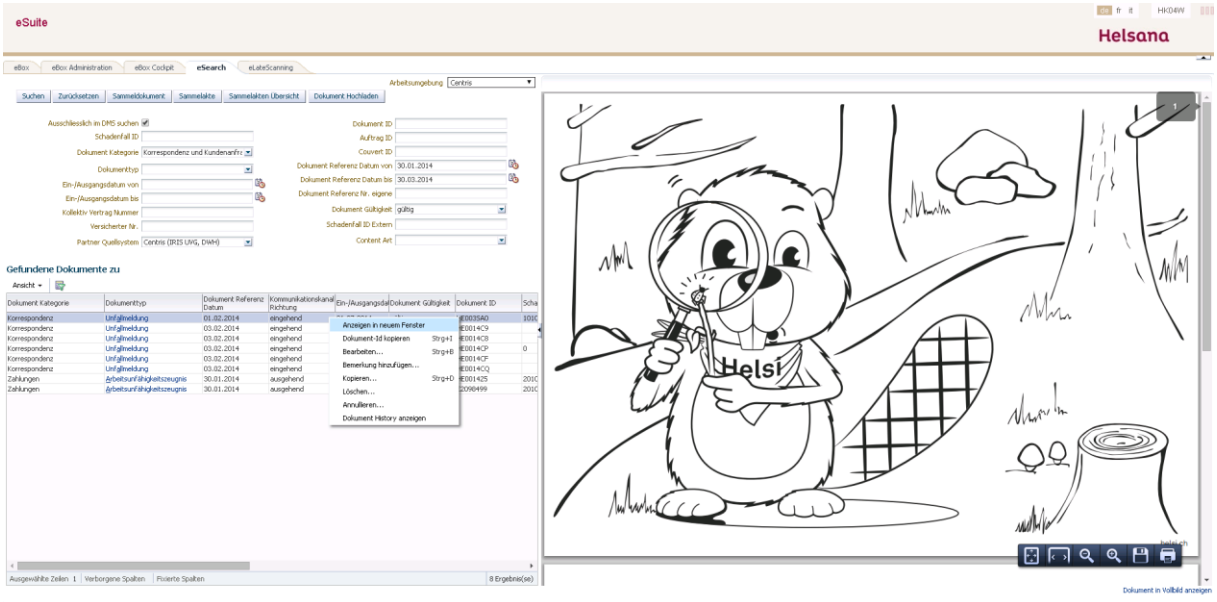

Abbildung 0.1 - eSearch mit ADF 12c

Der anstehende interne Umstieg auf den neuen Internet Explorer 11 zwang die Helsana von der ursprünglichen Version 11.1.1.5 auf eine neuere Version zu migrieren. Die Wahl fiel auf ADF 12c (12.1.3), damit gleich auf das neuste Major Release von Oracle gewechselt wird und nicht in kürzerer Zeit wieder migrieren muss.

Aus dieser und anderen Projekterfahrungen möchte ich hier die wichtigsten Punkte aufgliedern, die bei einer möglichen Migration von 11g nach 12c Beachtung finden sollten.

Migration mit Refactoring?

Die erste Frage die man sich stellen sollte ist, ob man den existierenden Code lediglich mit ADF 12c zum Laufen bringen soll, oder ob man im Zuge dessen auch ein Refactoring der bestehenden Anwendung verbinden kann. Oftmals liegt beides in Kombination nahe, da man einmal die einzelnen Elemente bearbeitet und dann auch alte Fehler bereinigt werden können!

Für die Helsana war der Umstieg auf die neue ADF Version auch die Chance, alle Altlasten aus den Anwendungsteilen zu eliminieren und sowohl auf ein neues Datenbank Modell als auch auf eine klarere ADF Architektur umzustellen.

Veränderungen von 11g nach 12c

Mit ADF 12c wurde der Support von HTML5, CSS3, Java EE 6 und JSF 2.0 eingeführt. Diese Technologiesprünge ziehen neue Komponenten oder neue Möglichkeiten mit sich.

Facelets

Der erste Punkt ist, dass Seiten und Seitenteile in 12c nun auf Facelets basieren. Da 11g noch auf JSP basierten, gibt es hierfür einen Converter im JDeveloper 12c. Dieser verändert die Tags in den Seitenteilen im Seitenkopf, passt am ViewController Projekt die Bibliotheken an und verändert gegebenenfalls die Dateiendung (von jspx zu jsf).

Neue Kopfzeile einer Facelets basierten Seite:

```
<ui:composition xmlns:af=http://xmlns.oracle.com/adf/faces/rich
xmlns:f="http://java.sun.com/jsf/core"
xmlns:dvt="http://xmlns.oracle.com/dss/adf/faces"
xmlns:ui="http://java.sun.com/jsf/facelets">
```

Alte Kopfzeile einer JSP basierten Seite:

```
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.1"
xmlns:af="http://xmlns.oracle.com/adf/faces/rich"
xmlns:f="http://java.sun.com/jsf/core"
xmlns:dvt="http://xmlns.oracle.com/dss/adf/faces"
xmlns:ui="http://java.sun.com/jsf/facelets">
```

Dieser Converter funktioniert soweit gut, passt sogar bei konvertierten Login-Seiten die web.xml an.

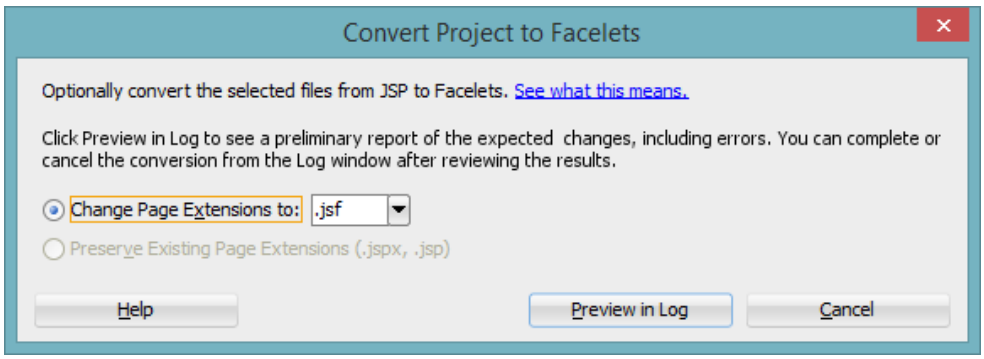

Abbildung 0.1 - Converter

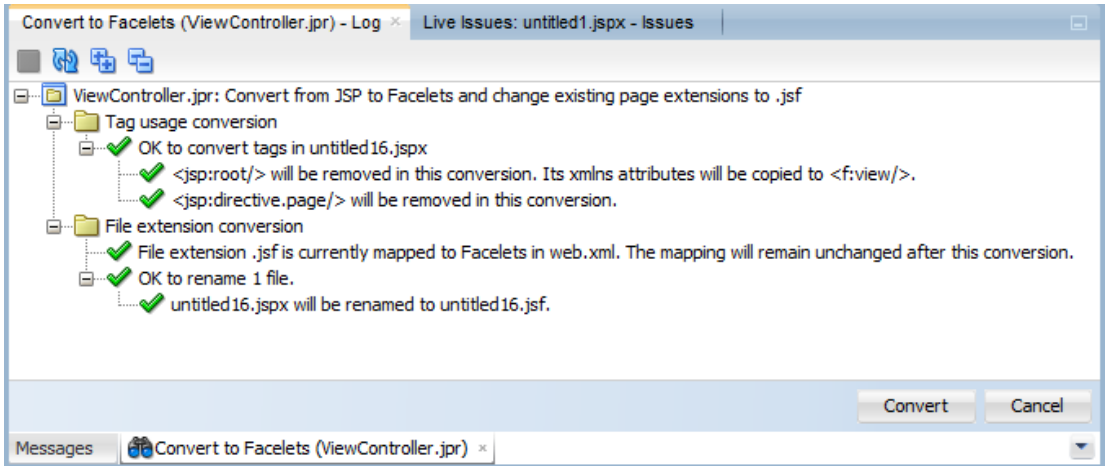

Abbildung 0.2 - Converter Log

Neue UI Komponenten

Aufgrund der neuen Facelets werden nun neue UI Komponenten eingeführt. Es erfolgt eine Konsolidierung der verschiedenen Buttons und Links, wie sie noch in 11g existent waren. In 12c gibt es nur noch af:button und af:link. Alle andere sind deprecated.

Beim Umstieg auf die neuen Komponenten ist zu beachten, dass sich der Default-Wert für partialSubmit gedreht hat. Wurde hier kein expliziter Wert gesetzt, so verändert sich ein partialSubmit=false auf partialSubmit=true. Dies kann zur Folge haben, dass sich diverse Sachen nun aufgrund eines fehlenden PPRs oder Full-Submits nun nicht mehr aktualisieren oder sich Popups mit „autoCancel=true“ nicht mehr schließen.

Image innerhalb Link

Es ist mit der neuen Komponente af:link nicht mehr möglich ein af:image unterhalb einzubetten.

InvokeAction deprecated

In 11g war es noch möglich invokeActions im Binding-Layer einzusetzen. Die Idee dahinter war, dass diese die Ausführung einer Action oder einer MethodAction aufgrund eines Events (Betreten der Seite) ausführt. Diese Funktion ist nun deprecated.

Um diese Actions nun auf einem anderen Weg beim „Laden der Seite“ auszuführen, sollten die Task-Flows verwendet werden. Auf dem Control Flow hin zu der Seite können verschiedene MethodActions ausgeführt werden, bevor die Seite final erreicht und angezeigt wird. Dies sollte mögliche Vorfilterungen und andere Verarbeitungen, die vorher mit invokeAction realisiert wurden, abbilden können.

Designtime erkennt Scopes nicht

Die Scopes einer Bean-Instanz an Taskflows

- backingBeanScope
- viewScope
- pageFlowScope

werden vom JDeveloper nicht mehr korrekt erkannt. Möchte man z.B. an eine Action eines Buttons eine Methode einer Bean hängen, so ist im Expression Builder keine Bean in diesen Scopes selektierbar. Selbst wenn man den EL Ausdruck von Hand eingibt, so wird die Methode zur Design-Time nicht erkannt. Zur Laufzeit allerdings funktioniert der Methodenaufruf ohne Probleme.

Dies wirkt sehr stark wie ein Bug, der zu mindestens in 12.1.2 und 12.1.3 noch existent ist.

ShowPopupBehaviour zuerst

Wenn man ein Popup mittels showPopupBehaviour öffnet und an dem Button/Link noch eine Action oder einen ActionListener aht, der z.B. Daten für das zu öffnende Popup aufbereitet, so hat sich die Ausführungsreihenfolge nun verändert.

Das ‚Popup wird nun zuerst angezeigt und erst danach werden die Action/ActionListener ausgeführt.

Das hat zur Folge, dass die Daten im Popup nicht mehr dem entsprechen, wie in 11g. Um das zu umgehen, sollte das Popup programmatisch am Ende der Action-Methode geöffnet werden, um die Ausführung des ActionListeners und der Action gezielt vor das Öffnen das Popups zu erzielen.

ValidateRegExpr validiert auch Leerfelder

Bei JSF 2.0+ ist es nun üblich, dass „validateRegExpr“ auch Leerfeldern von dem regulären Ausdruck ausgewertet werden. Das hat zur Folge, dass diese nun gefüllt sein müssen, um die Validierungsmeldung zu verhindern.

Hierfür gibt es zwei Lösungen:

1. Der reguläre Ausdruck muss auch Leerfelder akzeptieren (z.B. ein „+“ zu einem „*“ verändern)
2. Das Validieren von Leerfeldern über einen Kontext-Parameter in der web.xml global verhindern

```
<context-param>
<param-name>javax.faces.VALIDATE_EMPTY_FIELDS</param-name>
<param-value>false</param-value>
</context-param>
```

SetPropertyListener später

Wird versucht mittels eines af:setPropertyListener einen Wert zu setzen und über diesen Wert in der Methode eines übergeordneten ActionListeners auszulesen, so bekommt man in der Java Methode nur ein Null zurück. Begründet ist das Ganze damit, dass der ActionListener nun ausgeführt wird, bevor der setPropertyListener seine Arbeit tut und die Werte übergibt.

Neue transiente Attribute untrusted

Werden neue transiente Attribute in Entiy oder ViewObject erstellt und eine Groovy Expression hinterlegt, so wird diesem ein Property „untrusted“ gegeben. Dieses bewirkt, dass das transiente Attribut nicht wie geplant funktioniert. Zieht man z.B. die Sequence über einen Groovy Ausdruck, so wirft dieser auf „untrusted“ keine neue ID aus der Sequence!

Man muss in die Source von dem VO/EO gehen und dort das „untrusted“ auf „trusted“ verändern. Dann geht wieder alles wie gewohnt.

```
<ViewAttribute
  Name="Id"
  IsNotNull="true"
  PrecisionRule="true"
  EntityAttrName="Id"
  EntityUsage="RbwFaelleEntity"
  AliasName="ID">
  <TransientExpression
    trustMode="untrusted"><![CDATA[(new oracle.jbo.server.SequenceImpl("EMPLOYEES_SEQ",adf.object.getDBTransaction())).getSequenceNumber()]]>
</ViewAttribute>
```

Abbildung 0.3 - Groovy untrusted

Iteratoren Verhalten verändert

Die „ChangeEventPolicy“ der Iteratoren ist nun standardmäßig auf „PPR“ eingestellt. Dies verursacht, dass sich Komponenten auf der Seite bei der Veränderung am Iterator von allein aktualisieren. Klingt in der Theorie ganz gut, verursacht aber unnötig viele Refreshs auf der Seite, was man nicht mehr in der Hand hat.

Um das ursprüngliche Verhalten wieder herzustellen, stellt man das Default für die Gesamtanwendung von PPR wieder auf NONE in der adf-config.xml.

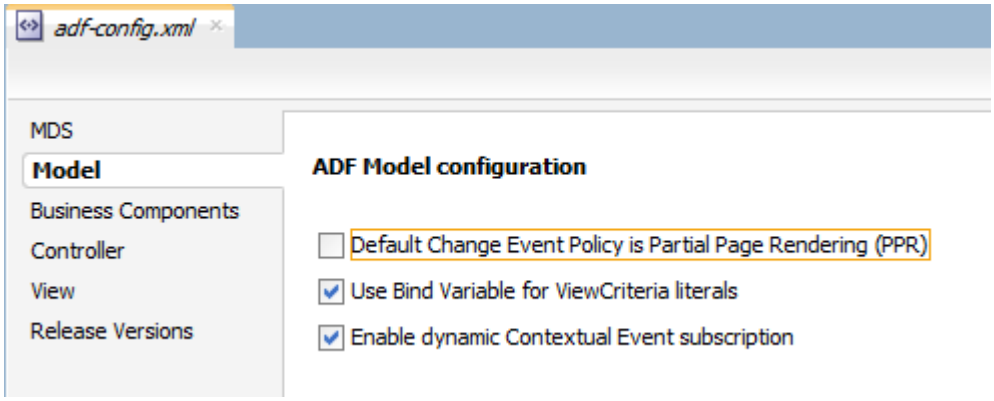

Abbildung 0.4 - Change Event Policy

Anwendung Stretch nicht mehr?

Ein weiteres neues Setting im JDeveloper 12c, welches allerdings nur bei einem neuen Projekt in 12c gesetzt wird, ist die Dimension, an der sich die Komponenten inital orientieren für das Stretching.

Der Context Parameter „DEFAULT_DIMENTSIONS“ steht initial auf „auto“. Der ehemals korrekte Wert von 11g war hier „parent“. Wenn das äußerste Stretch Layout einfach nicht stretchen will, dann hilft es hier wieder auf „parent“ zu gehen.

Probleme mit JDeveloper/ADF

Auditing verursacht langsamen JDeveloper

Es wirkt so, als wenn das Auditing, sofern es aufgrund von vielen Quellcode Anpassungen ständig angestoßen wird, den JDeveloper schnell immer träger werden lässt. Eventuell bietet es sich an das Auditing komplett zu deaktivieren und auf Compiler-Ergebnisse zu warten.

Fast Deployment nicht praktikabel

Versucht man nach einem erfolgreichen Deployment die Anwendung mittels „RUN“ neu zu deployen auf dem integrierten Weblogic, so geht dies sehr schnell!

Allerdings bekommt der JDeveloper dieses Hot Deployment nicht korrekt verarbeitet, so dass Klassen injeziert werden, die dann nicht mehr mit dem restlichen Deployment harmonieren. Man erhält dann z.B. ClassCastExceptions von einer Klasse auf exakt dieselbe Klasse o.ä. Verhaltensweisen, die dieses Feature komplett unbrauchbar machen.

Was am Ende wirklich hilft ist das vollständige Undeployment der Anwendung (geht zum Glück schnell), bevor man diese neu einspielt.

Spezielle DB2 Probleme

Da mit dem JDeveloper 12c nun der SQL Developer integriert wurde, nutzt nun auch der Wizard für das Abfragen von Tabellen oder Views bei Business Components eine neue Logik. Wo eine DB2 auf z/OS bei JDeveloper 11g noch Problemlos kommuniziert hat, zwingt es den 12c nun mit einem Fehler in die Knie.

DB2 baut auf bestimmten Katalogen auf, die eindeutige Bezeichnungen besitzen. Hier wurde auf einen falschen Katalognamen zugegriffen vom Framework (SYSIBM.SQLTABLES statt SYSIBM.SYSTABLES).

Auch hat das Updaten von Attribute Mappings bei angepasstem SQL nicht korrekt funktioniert und stellenweise das View Object unbrauchbar gemacht.

➔ Für beide Probleme sind nach SRs Patches erhältlich

Spezielle IE 11 Probleme

- Nach Öffnungen eines Popups auf Basis eines TaskFlow Dialogs, funktionieren keine Buttons mehr
- Sonderbare Dateinamen nach Export über ADF Tabellen Export in CSV (=_UTF-8_Q_VeranstalterExport_=)
- Zeilenzähler in der PanelCollection Statusbar funktionier fehlerhaft

➔ Alle drei Probleme sind nach SRs nun Patches erhältlich

Fazit

Wir haben mehrere Migrationsprojekte erfolgreich durchgeführt, allerdings war keins ohne Holpersteine.

Die aktuelle ADF Version 12 ist durchaus verwendbar und bietet viele Neuerungen und Unterstützungen. Sie ist aber leider noch nicht fehlerfrei und nachdem nun über ein Jahr vergangen ist wird es an der Zeit für eine neue Version mit der Integration aller Auffälligkeiten, die einem aktuell noch das Leben etwas erschweren.

Kontaktadresse:
Marcus Hammer
virtual7 GmbH
Zeppelinstraße, 2
D-76185 Karlsruhe

Telefon:	+49 (0) 721-619 017 50
Fax:	+49 (0) 721-619 017 29
E-Mail	hammer@virtual7.de
Internet:	www.virtual7.de