Die APEX APIs Schatzkiste

Ulrike Brenner
click-click IT Solutions e.U.
Perchtoldsdorf

Schlüsselworte
APEX, API

Einleitung

APEX stellt seit der ersten Version APIs (Application Programming Interfaces) bereit, die im APEX-Umfeld eingesetzt werden können.

Es gibt dabei eine Vielzahl von Bereichen in denen uns APEX-APIs das Leben erleichtern: So können durch die Verwendung von APEX-Collections Spalten oder Zeilen temporär gespeichert werden, es kann durch eigene Logging-Meldungen die Fehlersuche vereinfacht werden oder der ganze Deployment-Prozess kann durch das Einspielen der Applikationen mit einem SQL-Tool vereinfacht werden, wobei notwendigen Parameter angepasst werden ohne den Export zu manipulieren.

In jeder APEX-Version kommen neue APIs bzw. neue Funktionsaufrufe dazu, allerdings kann es auch sein, dass Funktionalitäten bei einem neuen APEX-Release nicht mehr zur Verfügung stehen, wobei es hier meist Übergangsfristen gibt (siehe Deprecated and Desupported Features). Nicht nur deshalb ist es bei einem Upgrade sehr empfehlenswert, die Releasenotes zu lesen.

APEX_COLLECTION
Gibt es die Anforderung einen Workflow abzubilden, bei dem der Anwender zwischen den Seiten navigieren kann und erst nachdem alles bearbeitet wurde, entscheidet, ob die Eingaben in der Datenbank gespeichert werden sollen, oder ob doch alles verworfen werden soll, dann sind Collections die optimale Wahl.
Collections bieten die Möglichkeit sowohl ganze Zeilen aber auch einzelne Spalten während einer APEX-Session temporär zu speichern. Vergleichbar ist die Funktionalität mit temporären Datenbank-Tabellen, die, da APEX stateless ist, hier nicht verwendet werden können.

Eine Collection kann auch aus Performancegründen interessant sein, wenn eine zeitaufwändige Query innerhalb einer Session mehrfach ausgeführt werden muss.

Die Bearbeitung (insert, update und delete) von Daten einer Collection erfolgt mit dem PL/SQL-API APEX_COLLECTION.

Um eine Collection verwenden zu können muss diese zuerst mit CREATE_COLLECTION erstellt werden. Dabei gibt es verschiedene Varianten, bei denen die Collection gleich mit den Daten eines Selects erstellt wird (vergleichbar mit der Erzeugung einer Tabelle mit CREATE TABLE xy AS SELECT * FROM your_table), es ist aber auch möglich eine ‚leere' Collection zu erzeugen, die im Laufe des Prozesses befüllt wird. Wichtig dabei ist, dass der Name eindeutig ist. Die erlaubte Länge von 255 Zeichen für den Namen bietet dafür genügend Möglichkeiten.

Sobald eine Collection erzeugt wurde, werden die Datensätze mit der View APEX_COLLECTIONS (Achtung: PLURAL) in der eigenen APEX-Session abgefragt. Im Entwicklungsprozess können in der

Entwicklertoolbar unter dem Punkt Session alle in der aktuellen Session erstellten Collections selektiert werden (direkt bei der Einschränkung auf Page Items, Application Items etc.).
Innerhalb einer Collection ist die Spalte SEQ_ID der eindeutige Schlüssel für einen Datensatz aufgrund dieser alle Bearbeitungen der Zeile durchgeführt werden.
Bei der Verwendung von Collections ist zu beachten, dass die Attribute nicht indiziert sind.

Collections sind ein wunderbares Mittel um Prozesse abzubilden, jedoch vor allem wenn viele Collections in einer Applikation verwendet werden und diese gleichzeitig von viele Anwender verwendet wird, ist es wichtig zu beachten, dass die Daten in einer Datenbanktabelle gespeichert werden und der gewonnene Performancegewinn sich damit wieder aufheben kann.

APEX_DEBUG
APEX bietet viele verschiedene Möglichkeiten zur Fehlersuche, jedoch möchte man als Entwickler manchmal zusätzlich eigene Informationen aus PL/SQL-Blöcken dargestellt bekommen (z.B.: das Teil-Ergebnis einer Berechnung oder der Wert einer Variable im Verlauf des PL/SQLs).
Mit APEX_DEBUG hat man die Möglichkeit in den Debug-Output von APEX eigene Informationen einzufügen. Dies funktioniert sowohl von PL/SQL-Prozessen, die direkt in APEX codiert sind, als auch von PL/SQL-Prozeduren, die in der Datenbank gespeichert sind und in APEX aufgerufen werden. Vergleichbar ist diese Funktionalität mit DBMS_OUTPUT.PUT_LINE in PL/SQL-Code, der direkt in einem SQL-Entwicklungstool in der Datenbank ausgeführt werden kann.

Dieses API bietet aber auch die Möglichkeit das Debuggen mit einem speziellen Debug-Level zu enablen bzw. zu disablen. Vor APEX 5.0 war dies eine wichtige Variante, um beim „Laden“ jeder Seite Debug-Messages zu bekommen (also auch dann, wenn man z.B. in einem Interactive Report Filter setzt) – ab 5.0 werden diese AJAX-Calls automatisch mit geloggt, wenn Debug aktiviert ist.

Die eigenen Messages können entweder direkt in der Anzeige des Debug-Outputs gefunden werden (Entwicklertoolbar/View Debug), oder man selektiert sie in der View APEX_DEBUG_MESSAGES.
Achtung: In APEX 4.2 wurde das APEX_DEBUG_MESSAGE Package in APEX_DEBUG umbenannt.

APEX_IR
Der Interactive Report ist ein tolles und mächtiges Tool mit dem u.a. dem Anwender die Möglichkeit gegeben wird, selbst zu entscheiden, welche Datensätze in welcher Form angezeigt werden soll.
Man kann den IR aber auch als ‚Filter’ für Folgeprozesse verwenden (z.B. anstelle einer eigenen Filter-Seite), d.h. der User hat die Möglichkeit genau jene Datensätze zu filtern, mit denen danach ein Prozess gestartet werden soll. Dabei ist für den Entwickler unabdingbar zu wissen, welche Filter der Anwender gesetzt hat. Das Query kann man mittels dem PL/SQL-Package APEX_IR bekommen.

Diese Information kann aber auch genutzt werden, wenn es bei einem speziellen IR immer wieder zu Performance-Problemen kommt. Dafür kann z.B. ein Button direkt in der Applikation eingebaut werden, auf den der Anwender klickt, wenn die Performance nicht wie erwartet ist. Das Query kann dabei vom Entwickler in einer eigenen Tabelle gespeichert und nachträglich analysiert werden.

Zusätzlich ist es möglich mit diesem API viele weitere Punkte rund um den Interactive Report zu behandeln: Hinzufügen/Löschen von Filter, Clear/Reset eines Reports, sodass die Verwendung des Interactive Reports für den Anwender komfortabler gestaltet werden kann.

Ein Beispiel zu APEX_IR finden Sie in den Packaged Application „Sample Reporting“ auf Seite 11.

APEX_APPLICATION_INSTALL
Spätestens wenn die Applikation fertig entwickelt ist stellt sich die Frage, wie der Deploymentprozess am Besten gestaltet werden kann. Dafür gibt es zwei Möglichkeiten: erstens der Upload der Applikation über GUI oder Ausführen des Export-Files (der Export ist ein PL/SQL-Code, der in einem SQL-Tool direkt gestartet werden kann). Beim Import müssen aber meist Attribute von Test auf PreProd bzw. Prod geändert werden. Diese werden beim Upload auf GUI automatisch angepasst bzw. können eingegeben werden.
Aber was ist beim Einspielen mittels SQL*PLUS oder einem anderen SQL-Tool?
Genau für dieses Zweck bietet das Package APEX_APPLICATION_INSTALL einige sehr nützliche Funktionen, ohne das Export-File zu manipulieren.

Wird die Applikation im selben Workspace mit der selben Workspace-ID wieder importiert, dann kann das Export-File direkt in SQL*Plus aufgerufen werden:

```
@f1234.sql
```

Wird die Applikation 1234 in den Workspace 'MY_WS_PROD' auf Produktion eingespielt, wobei das Schema 'MY_PROD_SCHEMA' verwendet werden soll (die Workspace-IDs auf Produktion und Entwicklung sind unterschiedlich), kann dies mit folgenden Befehlen durchgeführt werden.

```
DECLARE
    vWorkspaceID    NUMBER;
BEGIN
    SELECT WORKSPACE_ID
      INTO vWorkspaceID
      FROM APEX_WORKSPACES
     WHERE WORKSPACE = 'MY_WS_PROD'
    ;
    --
    APEX_APPLICATION_INSTALL.SET_WORKSPACE_ID( vWorkspaceID );
    APEX_APPLICATION_INSTALL.GENERATE_OFFSET;
    APEX_APPLICATION_INSTALL.SET_SCHEMA( 'MY_PROD_SCHEMA' );
    APEX_APPLICATION_INSTALL.SET_APPLICATION_ALIAS( 'MY_PROD_APP' );
END;
/

@f1234.sql
```

Ein anderer Anwendungsfall ergibt sich, wenn Sie für verschiedene Anwender in eigenen APEX-Workspaces Applikationen zur Verfügung stellen wollen, z.B. für Schulungs-/Übungszwecke, dann kann dies mit APEX_APPLICATION_INSTALL weitgehend automatisiert und somit leicht mehrfach wiederholt werden.

APEX_ZIP
Das mit APEX 5.0 neu hinzugekommene Package bietet die Möglichkeit Dateien in APEX zu zippen, neue Dateien zu zip-Containern hinzuzufügen und diese auch wieder zu unzippen.

APEX_UTIL
Wie der Name schon vermuten lässt bietet dieses API eine Vielzahl von Funktionen, die oft in Zusammenhang mit anderen APEX-APIs Verwendung finden. So kann man damit u.a. den Session-State holen bzw. setzen, die Berechtigung von APEX-Usern prüfen und Cache Informationen auslesen bzw. den Cache leeren.

Eine besonders interessante Methode sind die USER-Funktionen, mit denen APEX-User angelegt und bearbeitet werden können. So kann einem „Super“-User eine Maske zur Verfügung gestellt werden, mit welcher er in einem Schritt User genau für seinen Bedarf anlegt, vorausgesetzt die Applikation arbeitet mit APEX-Usern. Dabei ist zu beachten, dass dieser „Super“-User in APEX Administrator Rechte benötigt.

APEX_ITEM
Mit diesem API können Elemente dynamisch generiert werden. Auf den ersten Blick scheint es für dafür keinen Anwendungsfall (außer APEX selbst) zu geben, falls aber der Aufbau einer Seite in einer Datenbanktabelle konfiguriert ist, kann mit diesem API jede Seite dynamisch aufgebaut werden.

Dabei gibt es neben Checkboxes, Datumsfelder, Radiogroups, Textfeldern auch die Möglichkeit Hidden Items zu erzeugen.

Dies ist nur ein kleiner Auszug aus allen APEX-APIs, die mit APEX 5.0 zur Verfügung gestellt werden. Diese Packages bieten viele Möglichkeiten die Applikationen zu erweitern und es macht auf jeden Fall Sinn nach weiteren Funktionalitäten in der Dokumentation zu stöbern.
(https://docs.oracle.com/cd/E59726_01/doc.50/e39149/toc.htm)

Kontaktadresse:
Ulrike Brenner
click-click IT Solutions e.U.
Aspettenstraße 48
A-2380 Perchtoldsdorf

Telefon:	+43 (0) 1 3119425-30
E-Mail	ulrike.brenner@click-click.at
Internet:	www.click-click.at