Mit dem Itemtyp „Pop-up-List of Values (LOV)“ stellt Apex eine Möglichkeit zur Verfügung, um den Benutzer einen Wert aus dem Datenbestand zur Verwendung in einem Formular auswählen zu lassen.

Komfortable LOVs für Apex

Michael Schmid, Trivadis GmbH

Für viele einfache Anwendungsfälle ist der Mechanismus ausreichend. Im Vergleich zu der analogen Funktionalität von Oracle Forms wird der Entwickler allerdings die Möglichkeit, mehrere Werte an das aufrufende Formular zurückzugeben, vergeblich suchen. Dieser Artikel beschreibt eine Methode, wie sich komplexe und benutzerfreundliche Pop-up-LOVs in Apex realisieren lassen, die den Vergleich mit den LOVs von Oracle Forms nicht zu scheuen brauchen.

Bei der Erstellung benutzerfreundlicher Formulare ergibt sich häufig die Problematik, dass dem Benutzer zur Pflege einer Fremdschlüssel-Beziehung eine komfortable Möglichkeit zur Auswahl des entsprechenden Masterschlüssels geboten werden sollte. Zudem sollen fast immer im Formular zusätzliche Informationen zum jeweiligen Schlüssel angezeigt werden. Ein solches Apex-Formular, das auf dem EMP/DEPT-Schema basiert, könnte etwa wie in Abbildung 1 dargestellt aussehen.

Leider lässt sich das Füllen des „Deptno“-Schlüssels und der zusätzlichen Informationen („Dname“, „Loc“) dazu nicht mit dem im Lieferumfang von Apex befindlichen Pop-up-LOV-Mechanismus realisieren, denn damit ist es nur möglich, ein einziges Feld zu füllen. Außerdem lassen sich in dem Standard-Pop-up zwar einfache Filterungen durchführen, die Anzeige beschränkt sich jedoch auf nur eine einzige Spalte (siehe Abbildung 2).

Erstellen der Pop-up-Seite

Zur Lösung dieses Problems erstellen wir zunächst eine eigene Apex-Seite 210, die wir aus der Formularseite 200 heraus als Pop-up-Seite öffnen und die einen interaktiven Report (IR) enthält, der zur Auswahl des Master-Datensatzes dient. Die Filtermöglichkeiten der interaktiven Reports von Apex sind so umfangreich und komfortabel, dass damit auch anspruchsvolle Benutzer zufrieden sind. Die Abfrage, die dem Report zugrunde liegt, sollte alle Spalten selektieren, die wir später verarbeiten beziehungsweise zurückgeben wollen. In unserem Fall der DEPT-Tabelle wählen wir mit „select deptno, dname, loc from dept“ alle Spalten.

Als Page-Template nehmen wir für diese Seite „Popup“ und für die Report-Region empfiehlt sich das Region-Template „Region without Title“. Zusätzlich erstellen wir im IR eine „Link Column“, die den Aufruf der (noch von uns zu implementierenden) JavaScript-Funktion „passBack“ bewirkt und dabei die Werte des jeweiligen Datensatzes übergibt (siehe Abbildung 3).

Verbinden des Formulars mit der Pop-up-Seite

Um die Verbindung zwischen der Seite mit dem Formular und der Pop-up-Seite herzustellen, machen wir uns eine sehr nützliche JavaScript-Funktion namens „showModalDialog“ zunutze. Diese Funktion ist unter HTML 5 spezifiziert, funktioniert allerdings glücklicherweise schon seit geraumer Zeit in allen verbreiteten Browsern. So unterstützt der Internet Explorer diese Funktion schon seit Version 4, Firefox

Abbildung 1: Eingabemaske für einen EMP-Datensatz mit Zusatz-Informationen zum Department

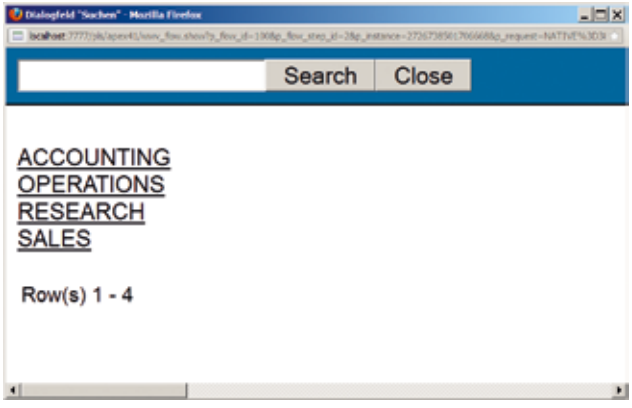

Abbildung 2: Apex „Pop-up LOV“

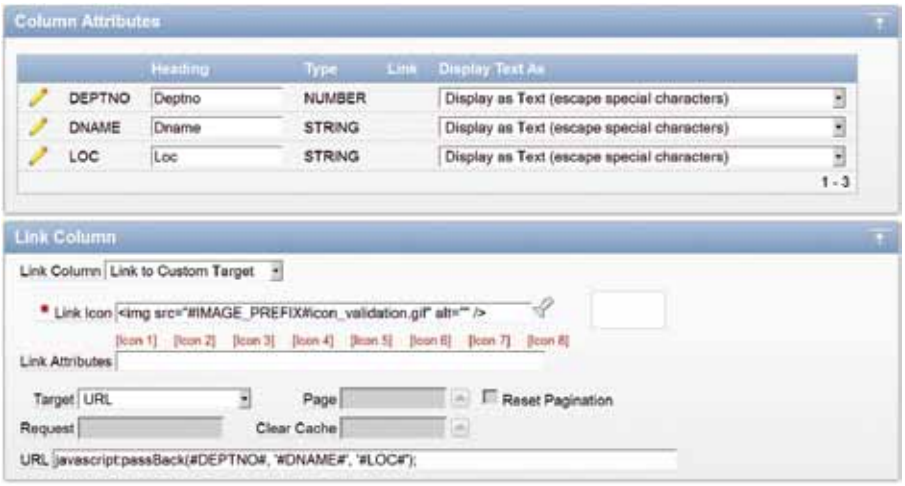

Abbildung 3: Die Einstellungen für die „Link Column“ des interaktiven Reports

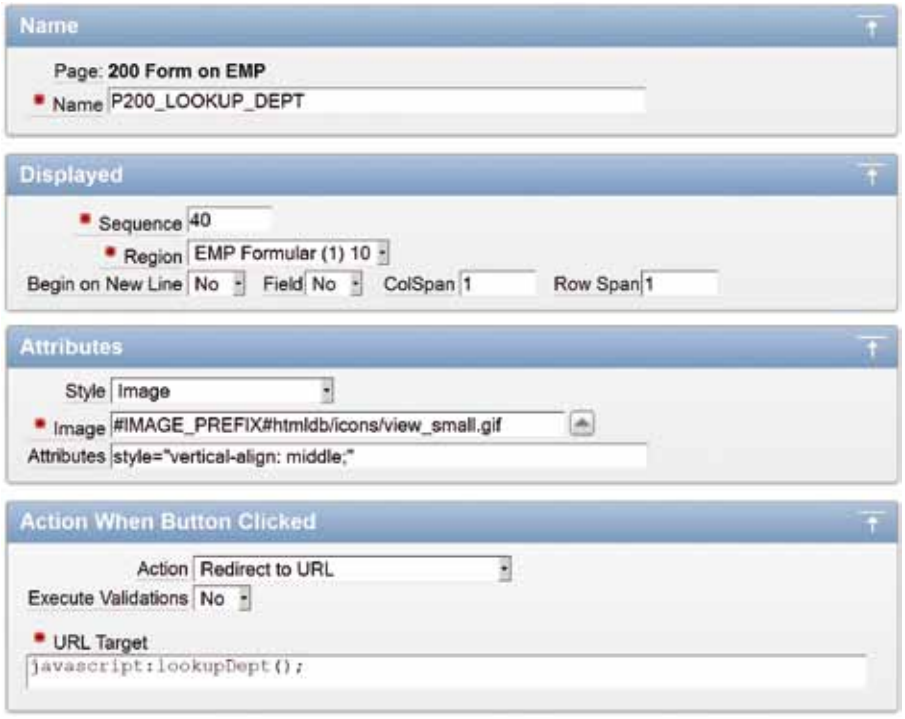

Abbildung 4: Die Einstellungen für den Button zum Öffnen der Pop-up-Seite

```
function lookupDept() {
  var ret =
    window.showModalDialog(
      „f?p=“ + $v(„pFlowId“) + „:210:“ + $v(„pInstance“),
      null,
      „dialogWidth: 620px; dialogHeight: 420px; „ +
      „resizable: yes“
    );
  if(ret) {
    $s(„P200_DEPTNO“, ret.deptno);
    $s(„P200_DNAME“, ret.dname);
    $s(„P200_LOC“, ret.loc);
  }
}
```

Listing 1: JavaScript-Funktion zum Öffnen des Pop-up-LOVs

beherrscht sie seit Version 3 und aktuelle Versionen von Chrome und Safari kommen damit ebenfalls zurecht (siehe auch die Links am Ende dieses Artikels).

Die Funktion „showModalDialog“ ermöglicht es, eine Web-Seite als modalen Dialog zu öffnen. Die URL der zu öffnenden Seite wird als erster Parameter übergeben. Mithilfe des zweiten Parameters lässt sich ein (komplexes) JavaScript-Objekt übergeben und mit dem dritten Parameter kann das Erscheinungsbild des neuen Fensters beeinflusst werden. Als Rückgabewert kann zum Beispiel sowohl ein String als auch ein komplexes Objekt geliefert werden, das die Pop-up-Seite generiert hat.

Auf der Formularseite 200 erstellen wir zum Öffnen der Pop-up-LOV-Seite 210 unter „Function and Global Variable Declaration“ eine Funktion „lookupDept“ mit dem entsprechenden Code aus Listing 1.

Die Bedeutung der Item-Namen auf der Formularseite 200 lässt sich leicht erschließen. Durch den Aufruf von „showModalDialog“ wird die Seite 210 in einem modalen Fenster geöffnet, das heißt der Benutzer kann erst wieder zum Formular zurückkehren, nachdem das Pop-up-Fenster verlassen wurde.

Zusätzlich legen wir die Größe des neuen Fensters mit 620 x 420 Pixeln fest und gestatten dem Benutzer, die Fenstergröße anzupassen. Da wir in diesem Fall von der Möglichkeit, ein Objekt zu übergeben, keinen Gebrauch machen, ist der zweite Parameter auf „null“ gesetzt. Die Ausführung des JavaScript-Codes blockiert am Aufruf von „showModalDialog“ und wartet, bis das Pop-up-Fenster geschlossen wird.

Das Rückgabeobjekt wird in der lokalen Variablen „ret“ gespeichert. Falls ein Wert zurückgegeben wurde, lesen wir die Werte „deptno“, „dname“ und „loc“ aus dem Objekt aus und legen sie in den entsprechenden Items ab.

Der Aufruf der Funktion „lookupDept“ erfolgt über einen Item-Button, den wir beispielsweise direkt neben dem Item „P200_DEPTNO“ platzieren (siehe Abbildungen 1 und 4).

Rückgabe der gewählten Werte

Nun müssen wir uns noch um die Rückgabe der Werte von der Pop-up-Seite 210 zurück zur Formularseite 200 kümmern. Dazu erstellen wir auf der Pop-up-Seite unter „Function and Global Variable Declaration“ eine Funktion namens „passBack“, auf die wir ja schon in unserem IR in der „Link Column“ referenzieren (siehe Listing 2).

Die Funktion erstellt aus den übergebenen Parametern ein JavaScript-Objekt und legt es in dem Property „returnValue“ des Window-Objekts ab. Anschließend wird das Pop-up-Fenster geschlossen. Der Wert des Property „returnValue“ wird an den Aufrufer zurückgegeben – in unserem Fall also an die lokale Variable „ret“ der Funktion „lookupDept“ auf der Formularseite. Empfehlenswert ist zusätzlich noch die Erstellung eines Buttons zum Schließen des Pop-up-Fensters, ohne eine Auswahl treffen zu müssen (siehe Abbildung 5). Diesen Button platzieren wir rechts neben der Suchleiste des IR (siehe Abbildung 6).

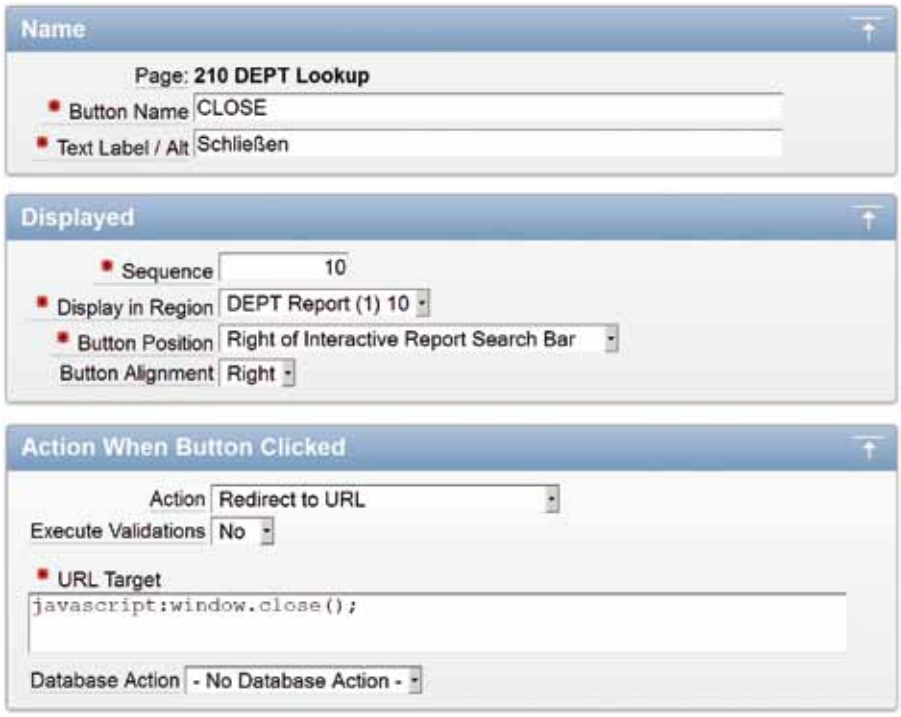

Abbildung 5: Die Einstellungen für den Button zum Schließen der Pop-up-Seite

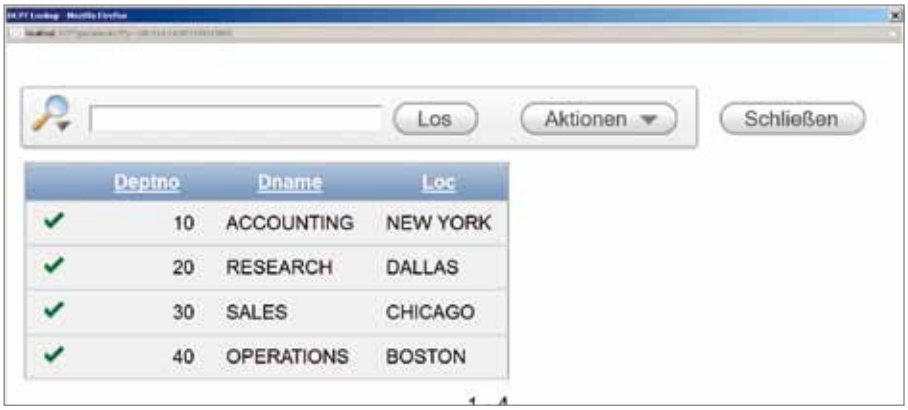

Abbildung 6: Die Pop-up-Seite mit IR und Schließen-Button

Weitere Ausbaumöglichkeiten

Die Interaktion zwischen unserer Formularseite und der Pop-up-LOV-Seite ist nun komplett implementiert – zusätzlich könnten wir unsere Pop-up-LOV-Seite noch um die Fähigkeit erweitern, neue Departments anzulegen. Ein solches Feature ist bei den Anwendern gern gesehen. Da es sich bei dieser Seite 210 um eine vollwertige Apex-Seite handelt, erstellen wir dazu eine Formularregion („Neues DEPT“) mit den benötigten Items („P210_DEPTNO“, „P210_DNAME“, „P210_LOC“) und legen einen entsprechenden Prozess („Process Row of DEPT“) an, der die eigentliche Speicherung übernimmt. Ferner ist noch ein Branch zurück zur selben Seite nötig (siehe Abbildung 7).

Zusätzlich sind noch zwei Änderungen beziehungsweise Erweiterungen am Page-Template „Popup“ vorzunehmen. Um zu verhindern, dass sich beim Internet Explorer durch das Abschicken des Formulars ein neues Fenster öffnet, stellen wir sicher, dass

```
function passBack(deptno, dname, loc) {
  window.returnValue =
    { deptno: deptno, dname: dname, loc: loc};
  window.close();
}
```

Listing 2: JavaScript-Funktion zur Rückgabe der Werte

```
<div class="success" id="success-message" style="width: 400px;">
  <img src="#IMAGE_PREFIX#delete.gif" onclick="$x_Remove(,success-
message')" style="float:right;" class="remove-message" alt="" />
  #SUCCESS_MESSAGE#
</div>
```

Listing 3: Sub-Template „Success Message“

der HTML-Head-Tag den Code „<base target="_self">“ enthält. Zum Schluss passen wir noch die Sub-Templates „Success Message“ (siehe Listing 3) und „Notification“ (siehe Listing 4) an. Damit ist schließlich unser Pop-up-LOV für die Departments fertig (siehe Abbildung 8).

Fazit

Mit der beschriebenen Methode ist es möglich, benutzerfreundliche LOVs in Apex zu erstellen und der wahrscheinlich größte Vorteil dabei ist: Überall in unserer Applikation, wo wir ein Look-up zum Beispiel für ein Department benötigen, können wir dieselbe Pop-up-Seite wiederverwenden und auch der Aufruf mit „showModalDialog“ ließe sich in eine JavaScript-Bibliothek auslagern. Einzig die weitere Verarbeitung des zurückgegebenen Objekts ist seitenspezifisch zu implementieren. Und dies eröffnet die Möglichkeit, die Resultate etwa in Items oder in Felder eines tabellarischen Formulars zu schreiben oder andere komplexere Verarbeitungsschritte mit JavaScript durchzuführen.

Weitere Informationen

[1] Dokumentation zur JavaScript-Funktion „showModalDialog“ im Mozilla Developer Network: https://developer.mozilla.org/en/DOM%3Awindow.showModalDialog

[2] Dokumentation zur JavaScript-Funktion „showModalDialog“ im Microsoft Developer Network: http://msdn.microsoft.com/en-us/library/ms536759(VS.85).aspx

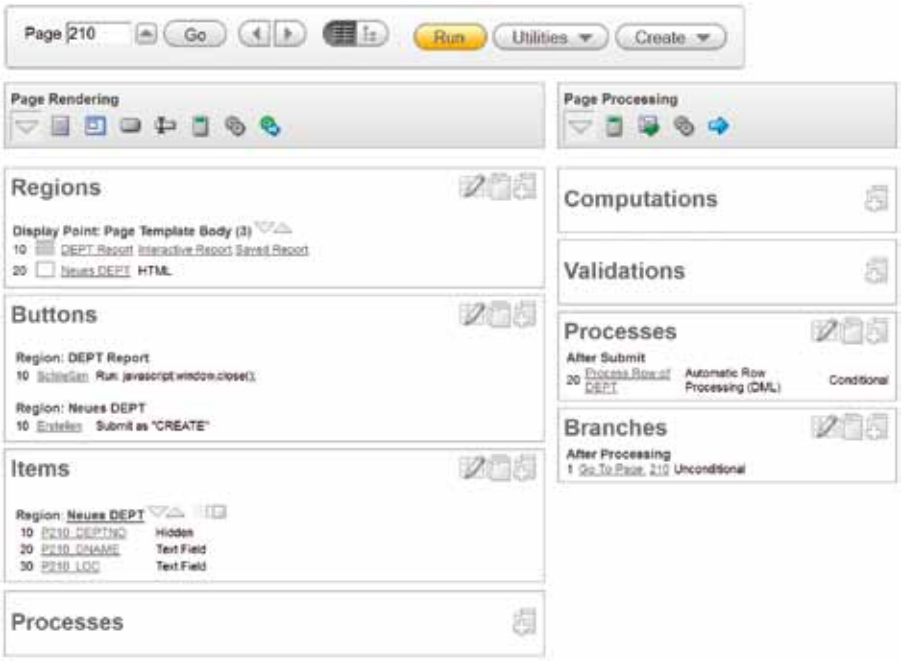

Abbildung 7: Die Seitendefinition der Pop-up-Seite mit den benötigten Komponenten zum Erstellen eines Departments

Abbildung 8: Das fertige Pop-up-LOV nach der Erstellung eines neuen Departments

Michael Schmid
Trivadis GmbH
michael.schmid@trivadis.com

```
<div class="notification" id="notification-message" style="width:
400px;">
  <img src="#IMAGE_PREFIX#delete.gif" onclick="$x_
Remove('notification-message')"  style="float:right;" class="remove-
message" alt="" />
  #MESSAGE#
</div>
```

Listing 4: Sub-Template „Notification“