Plain jQuery Mobile vs. APEX jQuery Mobile Smartphone vs. APEX Desktop

Oguz Ibram, Prof. Dr. Petra Sauer
Beuth Hochschule für Technik Berlin
Berlin

Schlüsselworte
Oracle APEX, jQuery Mobile, Rapid Application Development

Einleitung

Für Entwickler, die vorher nie mit Rapid Application Development Frameworks (RAD) gearbeitet haben, sondern alles per Hand gemacht haben, ist der Einstieg in RAD mit Hürden verbunden. Oft fragt sich der Umsteiger, was nach einem Mausklick im Hintergrund geschehen ist, da er es gewohnt ist, diesen Code selbst zu erstellen. Doch sind die Hürden einmal bewältigt, möchte man deklarative Werkzeuge gar nicht missen. Am Beispiel der seit Version 4.2 in APEX integrierten Smartphone-UI-Generierung soll im Beitrag eine Gegenüberstellung beider Entwicklungswege erfolgen und damit der Umstieg erleichtert werden. Dafür wird zuerst ein kurzer Einstieg in jQuery Mobile gegeben und danach demonstriert, wie das ganze in APEX mit ein paar Mausklicks geht. Der Beitrag soll damit klären, wieviel Arbeit APEX abnimmt, was nach einem Mausklick im Application Builder im Hintergrund geschieht, wieviel Freiraum der Entwickler innerhalb der APEX-Framework-Grenzen hat und wie mit APEX-Plugins und jQuery-Plugins die Grenzen erweitert werden können. Der Beitrag wird auch zeigen, welche APEX API-Funktionen und Prozeduren für die Anwendungs- und für die Plugin-Entwicklung sinnvoll verwendet werden können. Für Entwickler, die schon mit APEX gearbeitet haben, sollen die Unterschiede in APEX zwischen Desktop-Applikation und Smartphone-Applikation kurz angerissen werden. Der Beitrag ist gleichzeitig ein Erfahrungbericht einer Projektarbeit über mit APEX entwickelte standortbasierte, geodatenbasierte Dienste und wird Beispiele aus dem Umfeld des Projekts anzeigen und damit auch das Thema Geodaten und APEX anreißen.

Oracle Application Express - Grundlagen

Oracle Application Express (APEX) ist ein Tool zur deklarativen Erstellung von datenbanknahen Web-Applikationen. Die Kernidee hinter dem „Rapid Application Development“ (RAD) - Konzept ist die beschleunigte Anwendungsentwicklung durch das deklarative Einfügen von Funktionalität wie z.B. Forms und Reports, Navigationskontrollen, User Interface Themes. Oracle stellt mit APEX und ADF (Application Development Framework) mehrere RAD-Frameworks zur Verfügung. Während ADF der Erstellung von Java EE-basierten Middleware-Lösungen dient und Teil des Oracle Fusion Middleware ist, ist APEX datenbankzentriert und steht mit der installierten Oracle Datenbank (in jeder Edition) zur Installation auf dem Datenbankserver bereit. Datenbankzentriert bedeutet, dass die zu erstellenden Geschäftsprozesse Stored Functions, Stored Procedures, einfache DML-Anweisungen oder anonyme PL/SQL-Blöcke sind. Darüber hinaus kann eine APEX-Anwendung auch Web-Services konsumieren oder bereitstellen. Um HTTP-Requests entgegennehmen und beantworten zu können, entscheidet sich der APEX-Administrator zwischen dem APEX-Listener, der wiederum auf einem Java EE-Server laufen kann, dem Oracle HTTP-Server mit dem Modul `mod_plsql` und dem eingebetteten PL/SQL-Gateway. Abb. 1 zeigt die Architektur mit APEX-Listener.

Im Zentrum der Anwendungsentwicklung befindet sich die Entwicklung von Pages. Eine APEX-Anwendung ist eine Sammlung von miteinander verlinkten Webseiten, den „Pages“. Eine Page wird deklarativ durch deren „Page Definition“ erzeugt. Die Erstellung läuft Wizard-geführt ab. Dabei wählt

der Entwickler einen Page-Typ aus. Dies können Pages vom Typ Report, Form, Chart, Kalender uvm. sein.

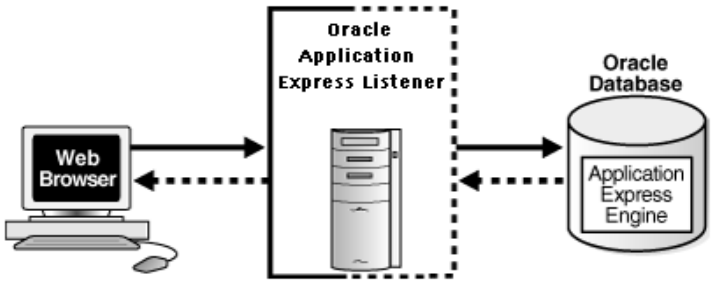

Abb. 1: Die Zwei-Schichten-Architektur von APEX (Quelle: OTN)

Dies sind out-of-the-box weitgehend fertig erstellte Webseiten für bestimmte Anwendungsfälle und ermöglichen eine Anwendungsentwicklung in kürzerer Zeit. Der Entwickler fügt mittels SQL-Queries und PL/SQL-Prozeduren oder -Blöcken die Seitenlogik ein. Die Abschnitte „Page Rendering“ und „Page Processing“ auf der „Page Definition“ gruppieren die Controls und Logiken übersichtlich nach Rendering und Processing / Submit-Funktion. Der Entwickler fügt hier Berechnungen für das Rendering (z.B. Währungsumrechnungen), PL/SQL-Prozeduren, Validierfunktionen, Dynamic Actions (deklaratives JavaScript) und Items ein. Items nehmen Benutzereingaben entgegen. Hinter ihnen verbergen sich die bekannten HTML-Formular-Elemente wie Radio-Buttons, Checkboxes, Textfelder etc. Da eine Page in Regions unterteilt ist, kann der Entwickler auch eine „Blank Page“ anlegen und seine Seite Region für Region konstruieren oder einer „fertigen Page“ wie z.B. einem Report weitere Regions z.B. weitere Reports hinzufügen. Die Darstellung der Pages oder Regions verbirgt sich hinter Templates. Ein Report wird gemäß dem dem Report zugewiesenen Report-Template erstellt. Der Entwickler erstellt mit HTML und CSS seine eigenen Templates oder nutzt eines der vielen out-of-the-box zur Verfügung stehenden Templates. Mehrere Templates bilden ein Theme. Seit der APEX-Version 4.2 wird zwischen „Desktop“-Themes und „jQuery Mobile Smartphone“-Themes unterschieden. Seit der Version 4.2 ist jQuery Mobile in APEX integriert, um auf den in diesem Abschnitt skizzierten Grundlagen aufbauend deklarativ Anwendungen nun auch für mobile Endgeräte zu erstellen.

jQuery und jQuery Mobile

jQuery ist eine JavaScript-Bibliothek mit der Grundidee, die JavaScript-Programmierung flexibler zu gestalten. Im Kern befindet sich das JavaScript-Objekt `jQuery`. Mithilfe dieses Objekts werden Elemente im HTML-Dokument auf flexible Art und Weise selektiert, um Methoden auf sie anzuwenden. Diese Methoden können einfache Effekte und Animationen ausführen, den HTML-Code oder den CSS-Code des gewählten Elements im Dokument dynamisch verändern oder Event Handler an HTML-Elemente binden. Durch Chaining können auf einem veränderten Element weitere Änderungen durchgeführt werden. Mit den jQuery AJAX-Methoden können Daten vom Server nachgeladen werden. Der Entwickler kann das `jQuery`-Objekt durch Plugins erweitern.

jQuery UI und jQuery Mobile bauen darauf auf. Bei beiden handelt es sich um Bibliotheken mit vorgefertigten UI-Interaktionen, Effekten, Widgets und Themes. Während jQuery UI auf Desktop-Oberflächen zugeschnitten ist, bietet jQuery Mobile Oberflächen und Funktionalitäten für kleine mobile Endgeräte an. jQuery Mobile wird in eine Seite eingebunden, indem u.a. die jQuery und jQuery-Mobile-Bibliotheken und eine CSS-Datei eingefügt werden. Die CSS-Datei liefert das „mobile Aussehen“, die JavaScript-Dateien erzeugen die Widgets und aktivieren die jQuery-Mobile API. jQuery Mobile nutzt intensiv das neue HTML5-Feature der `data-*`-Attribute. Die `data-*`-Attribute geben dem Entwickler die Freiheit, in jedes beliebige HTML-Element eigene Attribute einzubetten.

Dies wird von der jQuery Mobile-Bibliothek intensiv ausgenutzt. jQuery Mobile erkennt bei der Verarbeitung eines HTML-Dokuments seine eigenen `data-*`-Attribute, selektiert sie und fügt in diese Stellen jQuery Mobile-Code und CSS-Klassen hinein. Daraus resultiert, dass der Entwickler der mobilen Webseite weniger Aufwand aufbringen muss. Er schreibt seinen HTML-Code nach Möglichkeit mit den neuen semantischen HTML5-Tags anstelle von `div`-Tags, er fügt seinen Elementen eines der `data-*`-Attribute hinzu, den Rest erledigt die jQuery Mobile-Bibliothek, indem in diese Stellen CSS-Klassen und weitere Markups generiert werden. Zu den `data-*`-Attributen von jQuery Mobile gehören `data-role, data-rel, data-transition, data-direction` uvm. Ein wichtiges Konzept von jQuery Mobile ist das Konzept der internen Seiten. Interne Seiten sind Teil desselben HTML-Dokuments. Sie sind mit `data-role="page"` und einer `id` gekennzeichnet. Die Verlinkung auf eine interne Page geschieht wie gehabt durch `#id`. So existieren im Minimalfall nur ein HTML-Dokument, aber beliebig viele mobile Webseiten, die alle in diesem einen HTML-Dokument untergebracht sind. Somit dauert der erste Aufruf einer mobilen Webseite etwas länger, da das Dokument heruntergeladen werden muss, nicht aber die nachfolgenden Aufrufe, da sich das HTML-Dokument bereits im mobilen Endgerät befindet und Netzzugriff nicht mehr benötigt wird. jQuery Mobile bietet nach ähnlichem Funktionsprinzip weitere mobile Features an wie z.B. einen Dialog (ein aufpoppendes Fenster statt einer Seite), Navigation und History durch das `location.hash`-Objekt, animierte Transitions. Durch eine Vielzahl von Events reagiert jQuery Mobile auf Berührungen, Orientation Changes oder Scrolls. Weitere Events beziehen sich auf die Übergänge zwischen den internen Pages (z.B. `pagebeforeshow, pagebeforehide`). Die Event Listener werden seit jQuery 1.7 nur noch mit `on()` an die Elemente gebunden (früher `bind(), live()` und `delegate()`).

Was kann nun alles auf einer (internen oder externen) Seite platziert werden? Bei mobilen Anwendungen dominiert das „List View“. Es handelt sich dabei um eine Liste, die nur wenig Information über die Datensätze anzeigt. Durch das Aufklappen eines Listenelements erscheint mehr Information zum Datensatz. Der jQuery-Mobile-Entwickler fügt den `ul / ol` und `li`-Elementen einfach ein `data-role="listview"` hinzu. Den Rest erledigt das Framework. Mit `data-role="divider"` wird ein `li`-Element zum Listentrenner, `class="ui-li-icon"` signalisiert jQuery Mobile, dass dem `li`-Element ein Icon hinzugefügt werden soll, `class="ui-li-count"` signalisiert, dass hier Datensätze gezählt werden, die zu einem Listenelement gehören. Dargestellt wird dies durch eine eingekreiste Zahl in einem Listenelement. Auf ähnliche Weise – d.h. durch das Zusammenspiel von `data-*`-Attributen und CSS-Klassen – werden weitere Seitenelemente hinzugefügt wie z.B. Navigation-Bars, Seiten-Header und –Footer, die unabhängig von Scrollvorgängen immer angezeigt werden können (`data-position="fixed"`), für Berührungen geeignete Buttons (`data-role="button"`) und für Berührungen geeignete Checkboxes und Radio-Buttons, uvm.

Das Theming-System von jQuery Mobile trennt das Layout (also das Theme) von den Farbtönen und Texturen, die das Layout haben kann. Die Farbtöne und Texturen sind die Swatches. Ein Theme besitzt also mehrere Swatches. Ein Swatch wird mit `data-theme="[a-z]"` gewählt. Das Default-Theme von jQuery Mobile besitzt 5 Swatches: `a, b, c, d, e`. Der nächste Abschnitt wird kurz skizzieren, wie APEX seit der Version 4.2 den Entwickler bei der Erstellung von jQuery-Mobile-Seiten deklarativ unterstützt.

APEX 4.2 UI: jQuery Mobile Smartphone

APEX hat mit der Version 4.2 jQuery Mobile integriert. Der Entwickler bekommt deklarative Unterstützung in Form von Wizards. Die Erstellung einer Anwendung läuft im Grunde nach dem gleichen im ersten Abschnitt skizzierten Prinzip ab. Einige Wizards sind allerdings anders aufgebaut. Der Entwickler kommt erstmals bei der Auswahl eines User Interface für seine Anwendung mit

„jQuery Mobile Smartphone“ in Kontakt. Hier kann er zwischen dieser Option und „Desktop“ wählen. Bei jeder neuen Page-Erstellung kann der Entwickler zwischen diesen beiden Optionen wählen. Das bedeutet, dass eine Anwendung aus Desktop-Pages und/oder mobilen Pages bestehen kann. Eine Page sowohl als mobile als auch Desktop-Page anzubieten ist ohne einige Redundanz nicht möglich, d.h. die Page muss dupliziert werden. Die Redundanz wird aber stark reduziert, wenn die Geschäftslogik in PL/SQL-Prozeduren oder Web-Services ausgelagert wird. Gerade im Umfeld der unten skizzierten Projektarbeit wurde aber deutlich, dass eine Seite nicht immer in beiden Versionen existieren muss. So gibt es Außendiensttätigkeiten (z.B. Baumkontrollen), die in der mobilen Version vorliegen müssen, bei denen eine Desktop-Ansicht aber keinen Sinn macht, da Außendiensttätigkeiten nie im Büro stattfinden. Dagegen gibt es stationäre Tätigkeiten wie z.B. Sachbearbeitungsaufgaben, die in der Desktop-Version vorliegen sollten, bei denen es aber nicht schadet, wenn sie zusätzlich auch in der mobilen Version existieren. Für Anwendungen, die je nach Endgerät das Aussehen der selben Seite anpassen sollen, sollte ein Responsive Design-Konzept erarbeitet werden. Mit dem Desktop-Theme 25 bietet APEX eine solche Oberfläche bereits an. Der Entwickler kann das Theme 25 verändern oder erweitern.

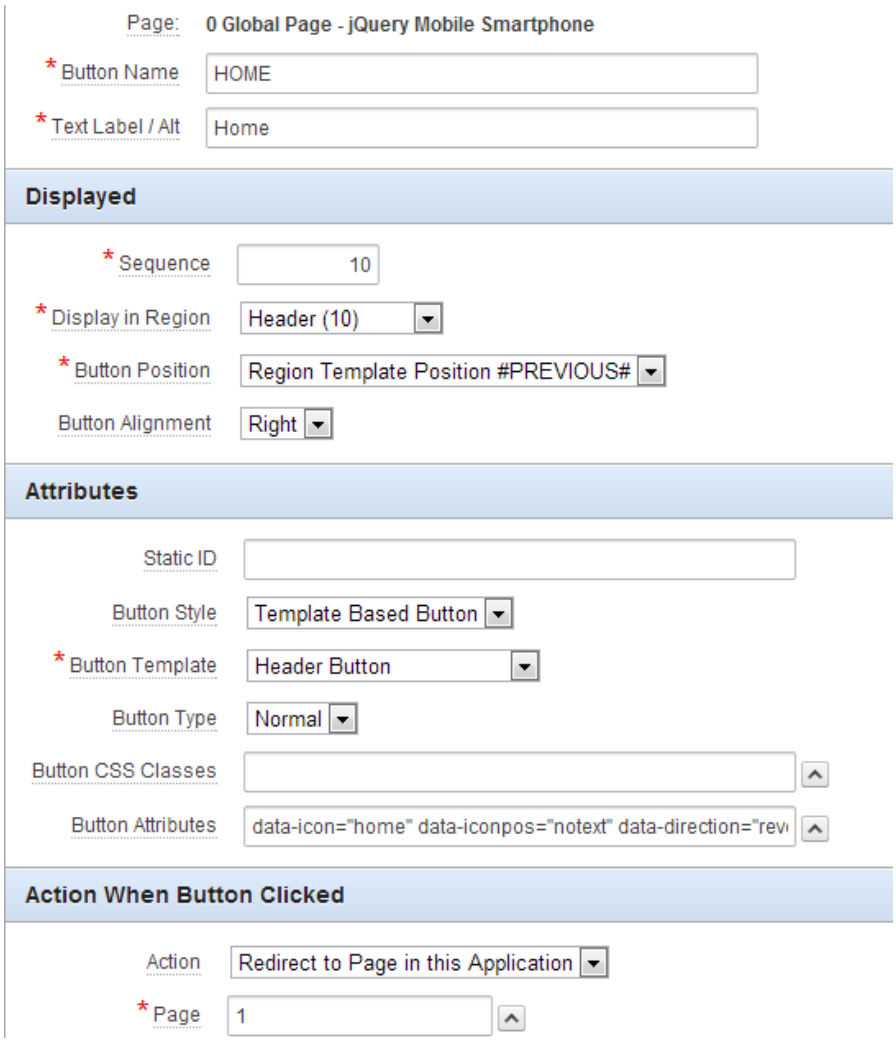

Abb. 2: jQuery Mobile-Angaben zu einem Button

Wie sieht nun eine Page-Erstellung aus? Der Entwickler bestimmt den Page-Typ wie gehabt (Report, Form, Chart, Kalender etc). Beim Report stellt er fest, dass „Interactive Report“ wegen schlechter Handhabbarkeit auf einem kleinen Gerät nicht mehr zur Auswahl steht. Dafür steht der neue Reporttyp „List View“ zur Verfügung. Zur Erstellung von Reports für mobile Geräte werden List-Views intensiv verwendet. „Classic Report“ steht zwar auch zur Verfügung, für den Anwender dürfte es aber viel Scrollen bedeuten, einen umfangreichen Bericht auf dem Smartphone durchzulesen.
Eine „leere“ mobile APEX-Anwendung kommt bereits mit drei Pages: der Global Page, der Login Page und der Home Page. In der Global Page sind die jQuery Mobile Header- und Footer platziert. Diese Elemente können durch Auswahlfelder z.B. als `fixed` eingestellt werden (Auswahl des Templates `Header Toolbar (Fixed)`). In Abb. 2 ist anhand des Home-Buttons ersichtlich, wie die Unterstützung durch APEX aussieht. Damit der Button zu einem jQuery-Mobile-Button wird, werden das Template (hier: `Header Button`) und weitere `data-*`-Attribute benötigt, die aber APEX generiert (siehe Feld `Button Attributes`). Im Template ist entweder `data-role="button"` angegeben oder es handelt sich wie in diesem Beispiel schlicht um einen Anchor Link, auf den jQuery Mobile selbst `data-role="button"` anwendet. Nach diesem Muster läuft auch bei anderen Elementen mobiler Pages die deklarative Unterstützung durch APEX ab. Die o.g. für mobile Anwendungen typischen Events wie Berührungen, Orientation Changes oder Transitions können in APEX durch Dynamic Actions deklarativ erstellt werden. Mit der Version 4.2 erhielt das Dynamic Action-Framework hierfür nötige 13 neue deklarative Events u.a. `orientationchange, scrollstart, tap, swipe,` usw. Wie auch bei der jQuery Mobile-Entwicklung lassen sich mit ThemeRoller CSS-Dateien generieren, um der APEX-Anwendung ein neues Aussehen zu verleihen.

Das Projekt MobiBaum

Im Projekt wird ein Baumkataster erstellt, das mobile und andere Dienste anbietet. Mit den Diensten kann ein Baumbestand verwaltet, kontrolliert und gepflegt werden. Die Entwicklungsarbeit erfolgte mit APEX 4.2 und mit Oracle Spatial für die Speicherung der Geodaten. Die Bäume sind mit ihren Koordinaten als Punktgeometrien vom Datentyp `SDO_GEOMETRY` gespeichert. Die Dienste zur Baumkontrolle wurden mit APEX und jQuery Mobile erstellt.

Quellen

[Hart13] Hartman, Roel; Rokitta, Christian; Peake, David: Oracle Application Express for Mobile Web Applications, New York 2013

[Reid11] Reid, Jon: jQuery Mobile – Building Cross-Platform Mobile Applications, Sebastopol 2011

Danksagung

Die Entwicklungsarbeit wurde im Rahmen des Projektes Forschungsassistenz VI durch die Senatsverwaltung für Wirtschaft, Technologie und Forschung mit Mitteln des Europäischen Sozialfonds gefördert.

Kontaktadresse:
Dipl.-Inf. Oguz Ibram
Beuth Hochschule für Technik Berlin
Seestraße 64
D-13347 Berlin

Telefon: +49 (0) 30-4504 3882
E-Mail oibram@beuth-hochschule.de

Prof. Dr. Petra Sauer
Beuth Hochschule für Technik Berlin
Luxemburger Straße 10
D-13353 Berlin

Telefon: +49 (0) 30-4504 2691
E-Mail sauer@beuth-hochschule.de
Internet: prof.beuth-hochschule.de/~sauer