Öko APEX

Hybride Lösung für Smartphones

mit APEX und PhoneGap

Christian Rokitta
themes4apex
Utrecht, Niederlande

Schlüsselworte

Oracle Application Express, Apps, Smartphone, PhoneGap, HTML5, CSS, jQuery Mobile

Einleitung

Die Anzahl der Benutzer von mobilen Endgeräten mit permanentem Internetzugang ist in den letzten Jahren enorm gestiegen. Mobile Geräte bieten zwei verschiedene Möglichkeiten um dem Benutzer Information zugänglich zu machen: Apps, die man auf seinem Gerät installieren kann, oder als Web App, eigentlich eine Webseite, optimiert für den Browser auf dem Smartphone oder Tablet.

Als Oracle Application Express Entwickler habe ich keine Wahl: APEX generiert HTML und wird in Browser präsentiert. Oder doch nicht? In meiner Präsentation zeige ich, wie man mit Hilfe des PhoneGap Frameworks eine APEX Anwendung in eine auf dem Smartphone installierbare App umsetzt.

Web oder App?

Als Entwickler von mobilen Applikationen steht man vor der Herausforderung, eine Technologie zu wählen, die es möglich macht, so effektiv möglich zu entwickeln und alle möglichen Endgeräte der Zielgruppe unterstützt.

Native Apps

Native Apps, entwickelt im Framework und der Programmiersprache eines bestimmten Betriebssystem, wie Xcode für iOS von Apple oder Java für Android, haben den Vorteil, dass sie für das betreffende System optimaler funktionieren und, sicher bezüglich der Benutzeroberfläche, besser integrieren. Alle Funktionen des Smartphone, wie eingebaute Kamera oder GPS, sind für den Entwickler mittels integrierten Schnittstellen zugänglich.

Die Entscheidung für ein App mit eigenem Icon und Design kann auch . Für die Distribution und Installation von Apps stehen einem die üblichen Kanäle, wie Google Play und Apple App Store zur Verfügung.

Nachteil der „native“ Apps ist, dass diese für das jeweilige Betriebssystem angepasst werden müssen, was die Entwicklungszeit teurer und die Pflege komplexer macht.

HTML Web Apps

Web Apps sind eigentlich Webseiten, die, zum Beispiel mit jQuery Mobile, für die Benutzung auf mobilen Geräten optimiert sind. Web Apps können normalerweise mit jedem Gerät benutzt werden,

das über einen Internet Browser verfügt, ohne das spezielle Anpassungen an den Seiten vorgenommen werden müssen. Damit beschränkt sich die Entwicklung auf eine Version für alle Zielgeräte und verkürzt man den Entwicklungsprozess. Man braucht natürlich auch nicht das Knowhow der einzelnen Betriebssysteme eigenen Programmierframeworks.

Hybride Apps

Wenn man die Vorteile beider Möglichkeiten kombinieren will, Installation im Menü des mobilen Geräts und Benutzung der Distributionsmöglichkeiten (iTunes Store, Android Market, Amazon Market, BlackBerry App World, Windows Phone Marketplace, u.s.w....) aber in Standards wie HTML und CSS entwickeln wollen, gibt es die Möglichkeit um hybride Apps zu erstellen.

Abb. 1: Hybride App = Minibrowser

Im Prinzip ist eine hybride App nichts anderes als ein Webbrowser der den vollständigen Bildschirm füllt, ohne jegliche Menüs wie beim üblichen Browser. Es ist technisch derselbe Browser, der durch das Betriebssystem benutzt wird. Auf iOS ist dies der Objective-C UIWebView class; bei Android der android.webkit.WebView.

PhoneGap

Eines der bekanntesten Frameworks, die diese „Application Container“ Technik benutzen um aus HTML Web Applikationen installierbare Apps zusammen zu stellen, ist PhoneGap. PhoneGap stellt Webapplikationen native APIs mobiler Geräte zur Verfügung, so dass sich native Applikationen mit Webtechnik wie HTML5, JavaScript und CSS entwickeln lassen. PhoneGap unterstützt iOS, Android, Blackberry OS, WebOS, Symbian und Bada.

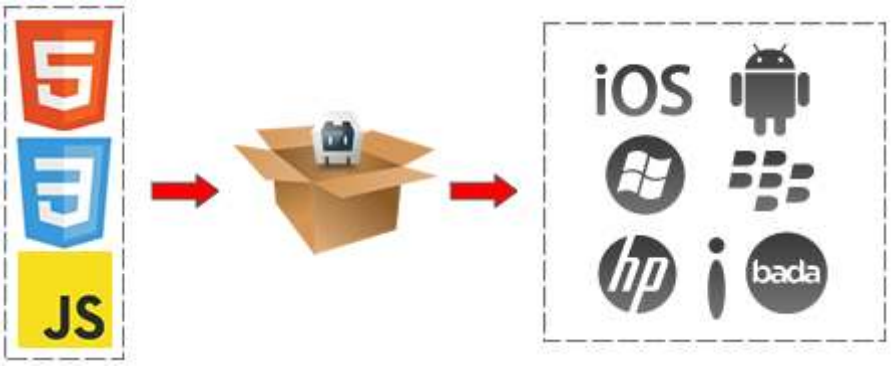


Abb. 2: Entwicklung in Webstandards. PhoneGap verpackt alles als App.

APEX als Hybride Apps?

Typische Web Apps, gebaut mit HTML, CSS und JavaScript und bestehen idealerweise aus seiner einzigen Multi-Page HTML Seite, die vollständig im App Package eingebettet wird, inklusive aller benötigten Ressourcen wie Bild-, CSS- und JavaScript Dateien. Ein PhoneGap Applikation dient als Client die mit einem Applikationsserver kommuniziert um Daten zu empfangen. Auf dem Applikationsserver befindet sich die Business Logik und Datenbank. Daten können eventuell lokal im Smartphone mit Hilfe des HTML5 Local Storage festgehalten werden. Damit ist sogar ein offline Betrieb der Applikation möglich.

Dieses Prinzip unterscheidet sich vollständig von der daten-zentrischen Art und Weise die wir von der Entwicklung mit Oracle Application Express gewöhnt sind. Eine APEX Applikation besteht üblicherweise aus mehreren Seiten, die zum Zeitpunkt des Anrufs dynamisch aufgebaut werden. Alle Anrufe werden von vom APEX Engine abgehandelt und als Datenbank benutzen wir natürlich die Oracle Datenbank selbst.

Trotzdem ist es möglich diese scheinbar gegensätzlichen Konzepte miteinander zu vereinigen. Letztendlich generiert APEX HTML Seiten die vollständig auf Standarden basiert sind. Die Kombination von Oracle APEX und jQuery Mobile, das in Version 4.2 integriert angeboten wird, beweist, dass Application Express auch sehr gut geeignet ist um Applikationen für mobile Endgeräte zu entwickeln.

Wie man mit Hilfe von jQuery Mobile in APEX mobile Applikationen entwickelt, erläutere ich in meiner zweiten Präsentation auf der DOAG Konferenz 2012, zu der ich gerne als Zuhörer einladen möchte.

In diesem Vortrag will ich das PhoneGap Framework, Schritt für Schritt, an Hand einiger Beispiele im Android SDK erläutern und schließlich eine APEX Web App in eine mobile Hybrid-App Android umsetzten. Dabei gehe ich auf die Besonderheiten ein, die bei der Einbindung von APEX in PhoneGap zu beachten sind.

Wer mit PhoneGap Apps für verschiedene Betriebssysteme entwickeln will, muss immer noch zumindest einen WebView Container pro Betriebssystem erstellen, in dem zu einer Webapplikation verwiesen wird. Obwohl umständlich, ist es wahrscheinlich eine einmalige Investierung, wenn man sich weiterhin beim Entwickeln der App auf das WebApp Teil der Applikation beschränkt. Es ist dann nicht notwendig neue Versionen des PhoneGap Container App an Benutzer zu distribuieren.

Im Laufe meines Vortrags werde ich am Beispiel des Android SDK den Bau einer solcher PhoneGap Container App für eine Application Express Applikation demonstrieren.

Native API

Was native Apps für Entwickler (und Benutzer) so Interessant macht, ist die Möglichkeit die Hardware des Gastgerätes vollständig benutzen zu können. Man denke zum Beispiel an den Accelerometer (Nike+), GPS (Foursquare) oder die Kamera (Instagram). Web Apps können, und das auch nur teilweise, lediglich die GPS Funktion benutzen.

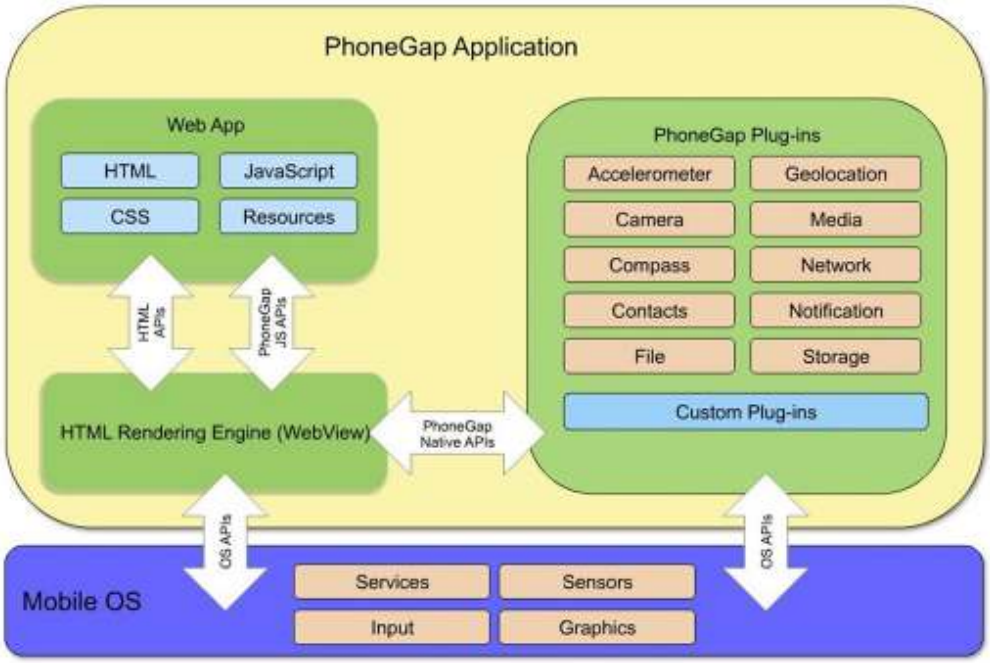


Abb. 3: PhoneGap Architektur.

PhoneGap bietet eine JavaScript Bibliothek, die es ermöglicht einen Teil der Funktionen des Betriebssystems an zu rufen. In einem Demo werde ich die Kamerafunktion eines Android Smartphone mit Hilfe der PhoneGap Bibliothek ansprechen und ein Foto mit Hilfe einer APEX Applikation vom Smartphone in die Oracle Datenbank laden.

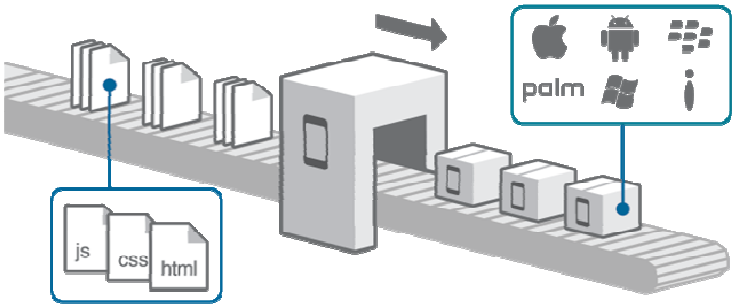
Hier ein HTML/PhoneGap Beispiel in dem mit getImage() ein Bild vom Gerät geladen und als multipart Formulier auf einen Server geladen wird:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
   <title>File Transfer Example</title>
   <script type="text/javascript" charset="utf-8" src="phonegap-1.2.0.js"></script>
   <script type="text/javascript" charset="utf-8">

        // Wait for PhoneGap to load
        document.addEventListener("deviceready", onDeviceReady, false);

        // PhoneGap is ready
        function onDeviceReady() {
            // Do cool things here...
```

```
        }

        function getImage() {
            // Retrieve image file location from specified source
            navigator.camera.getPicture(uploadPhoto, function(message) {
               alert('get picture failed');
            },{
               quality: 50,
               destinationType: navigator.camera.DestinationType.FILE_URI,
               sourceType: navigator.camera.PictureSourceType.PHOTOLIBRARY
              }
            );
        }

        function uploadPhoto(imageURI) {
            var options = new FileUploadOptions();
            options.fileKey="file";
            options.fileName=imageURI.substr(imageURI.lastIndexOf('/')+1);
            options.mimeType="image/jpeg";

            var params = new Object();
            params.value1 = "test";
            params.value2 = "param";

            options.params = params;
            options.chunkedMode = false;

            var ft = new FileTransfer();
            ft.upload(imageURI, http://yourdomain.com/formhandler
                              , win, fail, options);
        }

        function win(r) {
            console.log("Code = " + r.responseCode);
            console.log("Response = " + r.response);
            console.log("Sent = " + r.bytesSent);
            alert(r.response);
        }

        function fail(error) {
            alert("An error has occurred: Code = " = error.code);
        }

        </script>
</head>
<body>
    <button onclick="getImage();">Upload a Photo</button>
</body>
</html>
```

Adobe PhoneGap Build: Es kann noch einfacher?

Seit einiger Zeit bietet Adobe, die vor einem Jahr PhoneGap Entwickler Nitobi übernommen haben, einen Dienst an, mit dem es möglich wird die Dateien, die man normalerweise in einem lokal installierten SDK aufnehmen und kompilieren muss, hoch zu laden und durch den Adobe PhoneGap Build Cloud Service in ein fertiges App verpacken lassen. Besser noch, man kann mit ein und demselben Projekt, Apps für verschiedene Zielsysteme erstellen lassen.



Leider ist dieser Service, im Gegensatz zu PhoneGap selbst, das als Open Source Projekt jetzt in der Apache Software Foundation fortgeführt wird, nicht kostenfrei.

Kontaktadresse:
Christian Rokitta
themes4apex
Agnietenhove 5
NL 3834XA Leusden

Telefon: +31 (0) 6-41754763
E-Mail c.rokitta@themes4apex.com
Internet: www.themes4apex.com