APEX und JavaSript –

Pattern und Best Practices

Hendrik Gossens
OPITZ CONSULTING
Gummersbach

Schlüsselworte
Apex, JavaScript, jQuery

Einleitung
APEX-Anwendungen können durch JavaScript eine Aufwertung erfahren. Beispiele sind Tastaturnavigation, fixe Tabellenheader, erweitertes Logging und vieles mehr. Neben der Verwendung von Dynamic Actions ist in diesem Zusammenhang für die Erstellung von browserunabhängigem Quellcode das standardmäßig in APEX eingebundene JavaScript-Framework jQuery eine gute Option. Doch die Mächtigkeit von JavaScript birgt die Gefahr von Inkompatibilitäten. JavaScript-Quellcode, der in einer bestimmten APEX-Version lauffähig ist, funktioniert nicht auch zwangsläufig in späteren APEX Versionen. Bei der Erstellung von JavaScript-Code inner- und außerhalb von Dynamic Actions ist die Sicherstellung der Aufwärtskompatibilität des Quellcodes somit eine Herausforderung, die es zu meistern gilt. Der Vortrag stellt verschiedene Pattern und Best Practices vor, die die Lauffähigkeit des eigenen JavaScript Codes in zukünftigen APEX-Versionen positiv begünstigen und die Gefahr von Inkompatibilitäten minimieren können. Auf diese Weise ist es möglich, das Nutzererlebnis von APEX durch eine Prise JavaScript abzurunden und zu verfeinern, ohne zukünftig die Lauffähigkeit der Anwendung zu gefährden.

Verwendung von JavaScript - Optionen
Möchte man JavaScript in Apex einbinden, so gibt es verschiede Optionen:

- Verwendung von nativem Java Script
- Verwendung von jQuery
- Verwendung von Dynamic Actions

Oben aufgeführte Liste weist einen nach unten ansteigenden Abstraktionsgrad auf. Als Faustregel lässt sich festhalten, dass eine Abstraktion von nativem JavaScript Code die Wartbarkeit und browserunabhängiges Laufzeitverhalten garantieren. Daher sollte nach Möglichkeit weitestgehend auf natives JavaScript verzichtet werden. Abbildung 1 verdeutlicht diesen Sachverhalt:

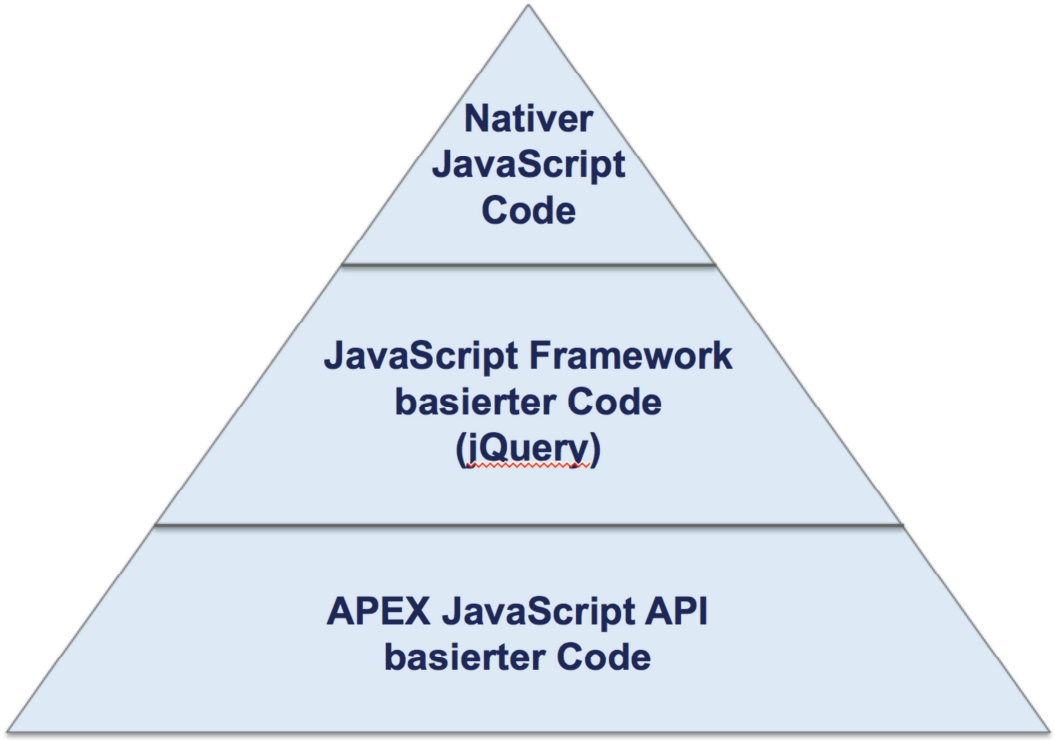

Abb. 1: Optionen für die Verwendung von JavaScript in APEX

Aus der Pyramide in Abbildung 1 lassen sich die Aufruf-Varianten von JavaScript in Apex ableiten. JavaScript Code integriert dabei am Besten mit Apex, wenn dieser Event gesteuert angesprochen wird. Hierzu zählt beispielsweise das Ausführen von JavaScript Code bei einem Button-Klick genauso wie das Abfangen von Tastaturereignissen.

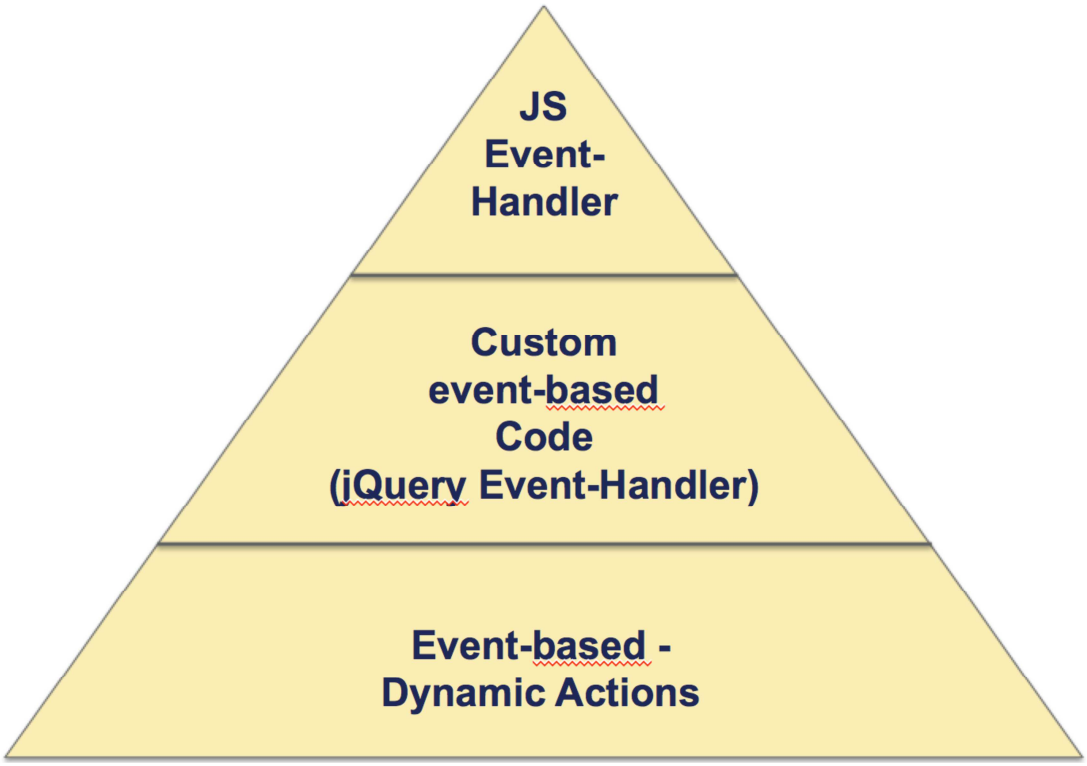

Abb 2: Aufrufvarianten

Use Cases

Aus bisherigen Kundenprojekten lassen sich vor allem drei Bereiche extrahieren, in denen die Verwendung von JavaScript das Standardverhalten von Apex sinnvoll ergänzen konnte:

1. Excelize your APEX
    a. Navigation in den Zellen über Tastaturpfeile und Enter-Taste
    b. Automatisches Ausfüllen von Zellen
    c. Vertikale Spaltenüberschriften
    d. Fixierte Spaltenüberschriften
2. Formsize your APEX: Tastatur-Shortcuts verwenden
3. Komfort- und Usability Funktionen:
    a. Validierungsverhalten ergänzen
    b. Smoothe Übergänge und Animationen
    c. Ergänzendes Hilfesystem

Mögliche Schwierigkeiten

Bei der Verwendung von JavaScript können verschiedene Schwierigkeiten die Entwicklung beeinträchtigen:

- **Datentypen:** JavaScript ist nicht typenstreng. Daher kann manuelles Casting von Datentypen erforderlich werden
- **Debugging:** Das Standard-Debugging-Verhalten von Browsern ermöglicht eine eher unkomfortable Codeüberwachung.
- **Wartbarkeit:** JavaScript Code kann an verschiedenen Stellen in Apex untergebracht werden. Aus diesem Grund ist nicht immer direkt ersichtlich, wie der Code zusammenhängt. Dies erschwert die Wartung und spätere Erweiterbarkeit
- **Browserabhängig unterschiedliches Verhalten desselben Codes**: Unterschiedliche Browser interpretieren JavaScript Code mit individuellen JavaScript Engines
- **Zusammenspiel einzelner Codeteile unklar (Event-Handler):** Da der Code Event-gesteuert angesprochen wird, ist nicht immer ersichtlich, wie die einzelnen Code-Teile zusammenspielen
- **Ein Syntaxfehler verhindert die komplette Ausführung (ohne sichtbare Fehlermeldung)**: Dies betrifft vor allem Code, in dem kein Exception Handling umgesetzt ist
- **Code spezifisch für spezielle APEX oder Browser-Version implementiert (Upgrade-Sicherheit?):** Der Code ist nicht genügend abstrahiert um in späteren Browser- oder Apex Versionen noch lauffähig zu sein.

Best Practices

Um das Risiko zu mindern, dass die Nachteile bei der Verwendung von JavaScript die Apex-Anwendung schlecht wartbar oder unausführbar machen, bietet es sich an, unternehmenseinheitliche / projektspezifische Richtlinien zu entwickeln, die den Einsatz von JavaScript in Apex sicherer machen:

- Debugging mit Toolunterstützung (Firebug / Chrome Developer Extensions)
- Eigene JavaScript Namespaces verwenden
- Exception-Handling mit JavaScript try-catch-finally Blöcken umsetzen
- Logging des Quellcodes
- Sinnvolle Kommentare verwenden
- Gegebenenfalls Dokumentier-Frameworks wie JSDoc verwenden

Fazit

JavaScript kann die Möglichkeiten von Apex sinnvoll ergänzen. Allerdings birgt dies auch einige Risiken, die sich durch die Verwendung eigener Richtlinien/Best Practices minimieren lassen. Dennoch sollte man zunächst prüfen, ob die spezifische Anforderung nicht durch Apex-Standardfunktionaliät abgedeckt werden kann. Zusammenfassend lässt sich somit folgende Faustregel für den Einsatz von JavaScript in APEX festhalten: JavaScript in Apex ja, aber man sollte dies nicht zu exzessiv betreiben.

Quellen

- http://docs.oracle.com/html/E24396_01/ejb3_overview_arch.html
- http://docs.oracle.com/cd/E28280_01/web.1111/b31974/bcquerying.htm
- http://openejb.apache.org/jpa-concepts.html

Kontaktadresse:
Hendrik Gossens
OPITZ CONSULTING Deutschland GmbH
Kirchstr. 6
D-51647 Gummersbach

Telefon:	+49 2102 30961-0
Fax:	+49 2102 30961-101
E-Mail	hendrik.gossens@mt-ag.com
Internet:	www.mt-ag.com