

jQuery

Pascal Naujoks und Tobias Kästle

Zusammenfassung: In dieser Ausarbeitung wird das JavaScript Framework jQuery behandelt. Die Arbeit beginnt mit der Beschreibung der Problematik, welche die Notwendigkeit eines JavaScript Frameworks aufzeigt. Im Folgenden wird eine Auswahl an alternativen JavaScript Frameworks vorgestellt und erläutert warum jQuery hier besonders hervorsticht. Die wichtigsten Aspekte von jQuery werden in den Abschnitten jQuery Attributes, jQuery Events und jQuery Effects behandelt. Diese Aspekte werden jeweils mit konkreten Programmierbeispielen belegt. Abschließend wird die Community rund um jQuery vorgestellt. Dies beinhaltet das jQuery Plugin Repository, die Supportforen sowie Literaturempfehlungen.

1 Einleitung

Webseiten sind bereits seit mehr als 15 Jahren nicht mehr statisch. Moderne Content Management Systeme¹ sorgen dafür, dass sich Inhalte schnell und einfach anpassen lassen. Servlets und API's² zu verschiedenen Plattformen ermögliche es, dass sich Inhalte auch über viele verschiedene Webseiten hinweg einfach verteilen lassen. Seit ein paar Jahren gehen Webseiten weiter, sie werden dynamisch und verändern sich zur Laufzeit. Während früher für jede Veränderung innerhalb der Webseite ein klassischer HTTP-Request notwendig war (siehe Abbildung 1), können jetzt Inhalte für eine Webseite partiell mittels AJAX [KJ09] nachgeladen werden.

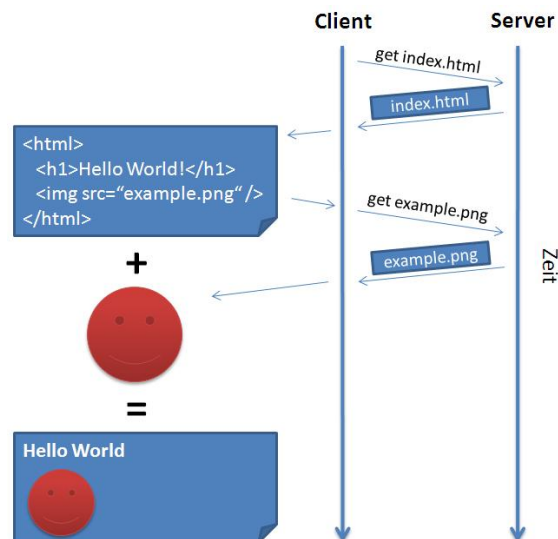


Abbildung 1: Standard HTTP-Request zwischen Client und Server

Die Dynamik innerhalb von Webseiten wird durch das schnelle Ein- und Ausblenden von Inhalten und die Einführung von Effekten, die früher nur dem Adobe Flash Player vorbehalten waren, noch gesteigert. Die Reduzierung der HTTP-Requests mittels AJAX

¹ Webseiten Backend-Systeme zur einfachen Verwaltung und Redaktionellen Steuerung von Webseiten

² Application Programming Interface. Schnittstelle zur Kommunikation zwischen Anwendungen

führt zu einer verringerten Last auf den Backend-Systemen, schnelleren Antwortzeiten und, zusammen mit dem gezielten Einsatz von Effekten und besseren User-Interface Elementen, zu einer erhöhten „User-Experience“ für den Besucher der Webseite.

1.1 Problematik beim Einsatz dynamischer Inhalte

Die Umsetzung der in Kapitel 1 gezeigten Möglichkeiten ist bereits seit 1995 mit der Einführung von JavaScript gegeben, war jedoch für Programmierer zu aufwändig, sie in die Webseiten zu integrieren. Zu den Hemmnissen gehörten an erster Stelle Inkompatibilitäten zwischen unterschiedlichen Browser Versionen. Diese machten es nötig Code bis zu vier Mal für verschiedene Browser-Engines zu implementieren. An folgendem Beispiel lässt sich dieser Umstand einfach veranschaulichen:

```
.transparent_class {
  filter:alpha(opacity=50);      /* Internet Explorer <= 8 */
  -moz-opacity:0.5;             /* K-HTML, Alter Safari */
  -khtml-opacity: 0.5;          /* Mozilla, Netscape */
  opacity: 0.5;                 /* FX, Safari, Opera */
  background:url(data:image/png;base64,iVBORw0KGgoAAAANSUheUgA(...)
    (...iVBORwACYII=);          /* Firefox 2, Alter Opera */
}
```

(1)

Durch den gezeigten CSS-Code (1) wird die Transparenz des Objekts „transparent_class“ auf 50% gestellt. Jede Zeile des Codes ist hierbei für die Ansprache eines anderen Browsers zuständig. Um die Transparenz eines Objekts (z.B. die eines Linkbuttons) von den anfänglichen 50% zur Laufzeit (ohne HTTP-Request) auf 100% erhöhen zu können, sind etliche Zeilen JavaScript Code notwendig. Hierzu gehören Fallunterscheidungen für Browser, Schleifen durch alle Instanzen des „transparent_class“ Objekts und die schlussendliche Zuweisung des neuen Transparenzwertes an alle Instanzen. Für neue Browser, wie zum Beispiel den kürzlich erschienenen Internet Explorer 9 und Firefox 4, müsste nachträglich Code ergänzt werden.

Die Arbeit mit AJAX ist in JavaScript ähnlich umständlich und ebenfalls in einigen Browsern unterschiedlich in der Handhabung. Die Instanziierung eines Asynchronen Requests, sowie die Abhandlung des Responses benötigen etwa 172 Zeilen JavaScript Code [AP06]. Mit jeder Funktion, die einen AJAX Request startet und den Response entgegen nehmen soll, steigt die Anzahl an Code Zeilen um mindestens ein Viertel des Instanziierungs-Skripts.

1.2 Gründe für ein JavaScript Framework

Der Einsatz eines JavaScript Frameworks soll die Arbeit mit der Skriptsprache JavaScript hinsichtlich der in 1.1 gezeigten Problematik (und darüber hinaus) vereinfachen. Dazu zählen folgende Punkte:

- Sicherstellen von Cross-Browser Kompatibilität³
- Bereitstellung und Vereinfachung von häufig genutzten Funktionen
- Reduktion und Vereinfachung des zu schreibenden JavaScript Codes

³ Kurzform für „Kompatibilität zu allen auf dem Markt verfügbaren Browsern jetzt und zukünftig“

- Updatefähigkeit des Frameworks für zukünftige Browser
- Einsatz von Patterns zur sauberen Strukturierung von Code

Optional bietet das Framework dem Programmierer häufig genutzte User-Interface Bausteine bereits fertig programmiert an. Dazu gehören z.B. Kalender, Buttons, Slider, Dialoge, Tabs, Accordions, usw.

Die genannten Anforderungen erzeugen einen dementsprechenden Mehraufwand (Overhead) im Code des Frameworks und führen somit zu einer längeren Ladezeit der Webseite. Es sollte Ziel des Frameworks sein den eigenen Code so klein wie möglich zu halten und das eigene Framework idealerweise über ein Content Delivery Network (CDN⁴) anzubieten. Der Vorteil eines CDN liegt in der Architektur der Browser. Je nach Versionsstand können diese von einer Webseite nur eine bestimmte Anzahl an HTTP-Objekten gleichzeitig (aktuell zwischen 2 und 8) stapelweise herunterladen. Liegt das JavaScript Framework auf einer anderen Webseite (das CDN), so steigert sich die Anzahl an gleichzeitigen Downloads um eins für jedes von einem CDN angeforderte Objekt.

2 JavaScript Frameworks

Die Anzahl der im Internet verfügbaren JavaScript Frameworks wächst stetig. In einer Liste auf Wikipedia [WIKI1] sind aktuell 20 JavaScript Frameworks aufgeführt. Zu den am weitesten verbreiteten zählen laut „builtWith Trends“ [bWT1] jQuery, prototype, script.aculo.us, YUI und MooTools.

Die Differenzierung der einzelnen Frameworks wird in dieser Arbeit anhand der Dateigröße des voll einsatzfähigen Frameworks, der gebotenen Funktionalität und Erweiterbarkeit, sowie der aktuellen Verbreitung und Einsatzhäufigkeit der Frameworks im Internet vorgenommen.

2.1 Dateigröße der Frameworks

Mittlerweile ist das Surfen im Internet mit Downloadgeschwindigkeiten von bis zu 100 MBit pro Sekunde [KBW11] möglich. Dies ist allerdings nur ein theoretischer Wert und wird durch die Auslieferungsgeschwindigkeit von Webservern, sowie durch die in Kapitel 1.2 beschriebene maximalen Anzahl an parallelen Downloads pro Domain begrenzt. Desweiteren ist die Verbreitung von Breitband Internet noch längst nicht in alle Region vorgedrungen. *„Eine Breitbandquote von 50 Prozent wird heute nicht einmal in den gut entwickelten Ländern der Welt erreicht. Laut Fernmeldeunion liegt sie im Westen derzeit bei etwa 30 Prozent. In Schwellenländern könne aber nur jeder Zehnte über Breitband verfügen, in den ärmsten Ländern gar nur jeder hundertste (...)“*, sagte der Generalsekretär der Internationalen Fernmeldeunion (ITU), Hamadoun Touré, im September 2010 in New York [ITU10]. Demnach ist es nach wie vor, trotz vereinzelt verfügbarer Breitband Internetzugänge, nötig die auszuliefernde Anzahl an Dateien und Gesamtdateigröße so gering wie möglich zu halten. Aber auch wenn man davon ausgehen kann, dass nur Besucher mit Breitband Internetzugang die eigene Webseite besuchen muss auf einen möglichst geringe Auslieferungszeit geachtet werden. Nach einer Studie der Aberdeen Group im November 2008 [AG08] kann man

⁴ Bereitstellung von Inhalten über verschiedene redundant angelegte und geografisch stark verteilte Server und Backbones. Die Auslieferung der Inhalte erfolgt über den geografisch nächstgelegenen Server.

die Ladezeit einer Webseite direkt mit dem generierten Umsatz verknüpfen. Die folgende Grafik aus dieser Studie zeigt, dass die künstlich verzögerte Auslieferung einer Webseite um eine Sekunde 11% weniger Seitenaufrufe und 7% weniger Käufe zur Folge hat:

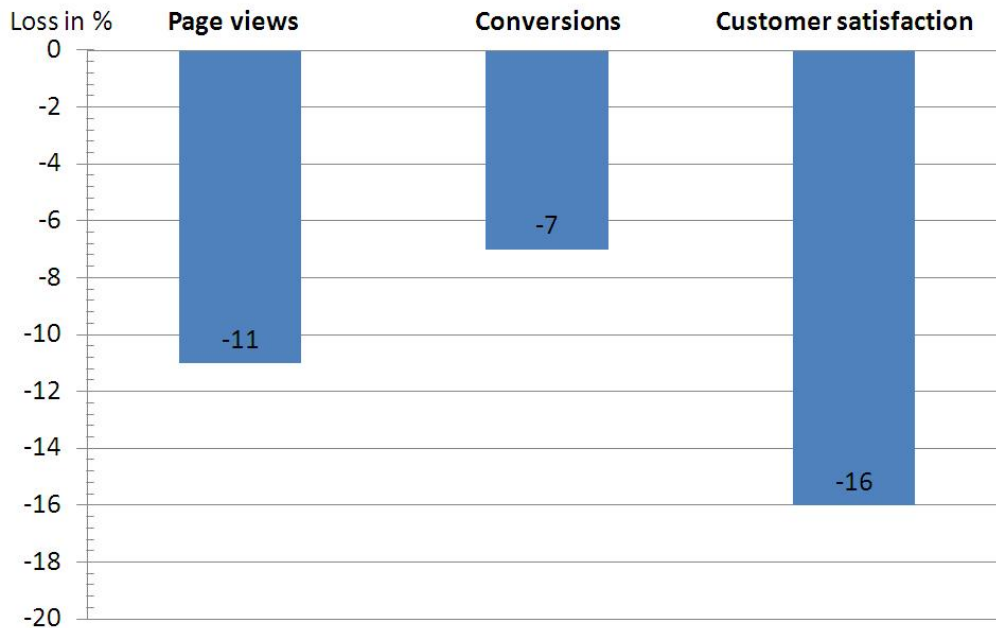


Abbildung 2: Verluste bei der verzögerten Auslieferung von Webseiten um eine Sekunde in %

Am Beispiel des Jahresgewinns 2010 von Amazon mit 1,15 Milliarden US Dollar [AMAZ11] wären das 126,5 Millionen US Dollar weniger.

Im Folgenden ist die Dateigröße verschiedener JavaScript Frameworks gegenübergestellt:

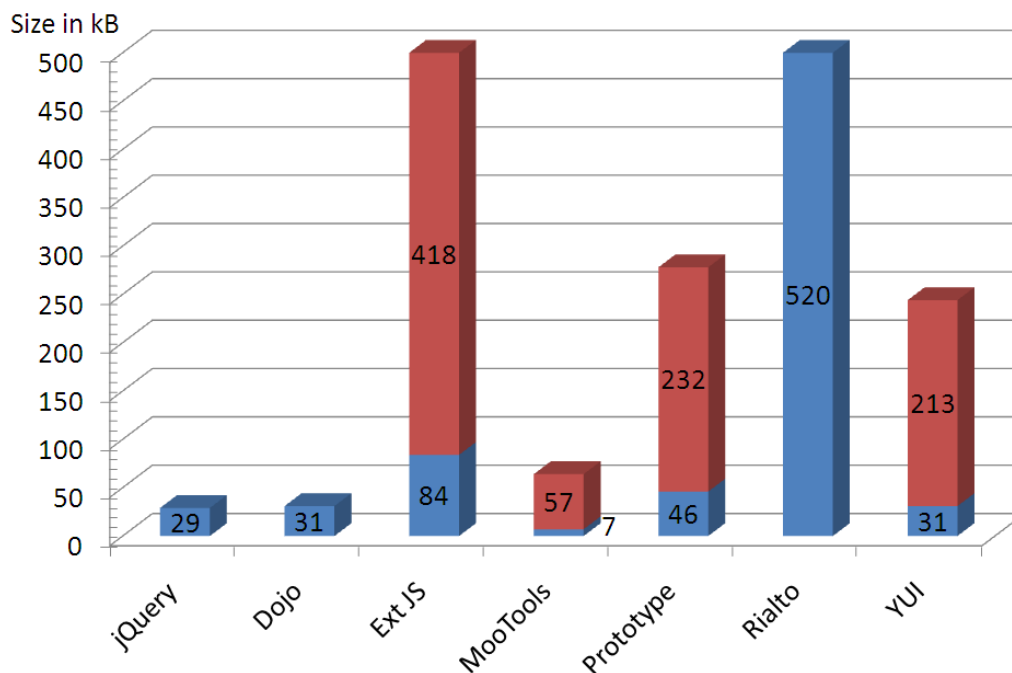


Abbildung 3: Dateigrößen von JavaScript Frameworks in kB

Die Balken zeigen die Dateigröße des Frameworks im komprimierten Zustand. Manche Frameworks unterstützen das Erstellen einer speziell für den jeweiligen

Anwendungszweck zugeschnittenen Version. Die Basisversion (*Core*) des Frameworks ist in diesem Fall im Diagramm blau dargestellt und die komplette Version des Frameworks (*Full Featured*) rot.

jQuery bietet eine zusätzliche Bibliothek an (jQuery UI), die von der Core Datei abhängig, aber getrennt zu beziehen ist. In der jQuery UI befinden sich gängige User-Interface Elemente wie z.B.: Tabs, Dialoge, Buttons, Drag&Drop, Accordions, Autocomplete, Datepicker, Slider, weitere Effekte usw.

Zu den in Abbildung 3 gezeigten Production-Distributionen stellen die Entwickler auch Development-Distributionen ihrer Frameworks bereit. Production-Distributionen werden in Produktivsystem eingesetzt und wurden von Kommentaren, unnötigen Leerzeichen und Zeilenumbrüchen befreit. Zusätzlich werden in den meisten Fällen auch Variablen- und Funktionsnamen auf eins bis zwei Alphanumerische Zeichen gekürzt. Hier ein Auszug aus der aktuellen Productive-Distribution von jQuery 1.5.2:

```
(function(a,b){function ci(a){return d.isWindow(a)?a:a.nodeType===9?a.defaultView||a.parentWindow:!1}func
```

Development-Versionen dienen zu Weiterentwicklungs- und Debugging-Zwecken und sollten nur in Testsystemen eingesetzt werden. Development-Versionen enthalten den Quellcode des Frameworks im unkomprimierten Originalzustand. Dies beinhaltet Kommentare, Funktionsbeschreibungen und (je nach Framework) weitere Dokumentation zur Handhabung und Einsatz des Frameworks.

2.2 Funktionsumfang und Erweiterbarkeit

Der Funktionsumfang der meisten großen Plugins wie Prototype, MooTools, Dojo, YUI, Ext Js und jQuery hält sich im Core und den Standarderweiterungen die Waage. Verschiedene Selektoren und Filter, DOM⁵-Manipulation, Event-Handling, Animationen, AJAX und eine Plugin API sind in allen genannten Frameworks schon seit langem Standard.

Der Funktionsumfang kann an der Anzahl der zur Verfügung stehenden Plugins (siehe Abbildung 4) gemessen werden. Leider gibt es nicht für alle Frameworks eine Übersicht über alle zur Verfügung stehenden Plugins oder eine Auswahl an Plugins ist in den Core migriert worden (In diesem Fall wurden keine Daten erhoben, der Wert ist 0).

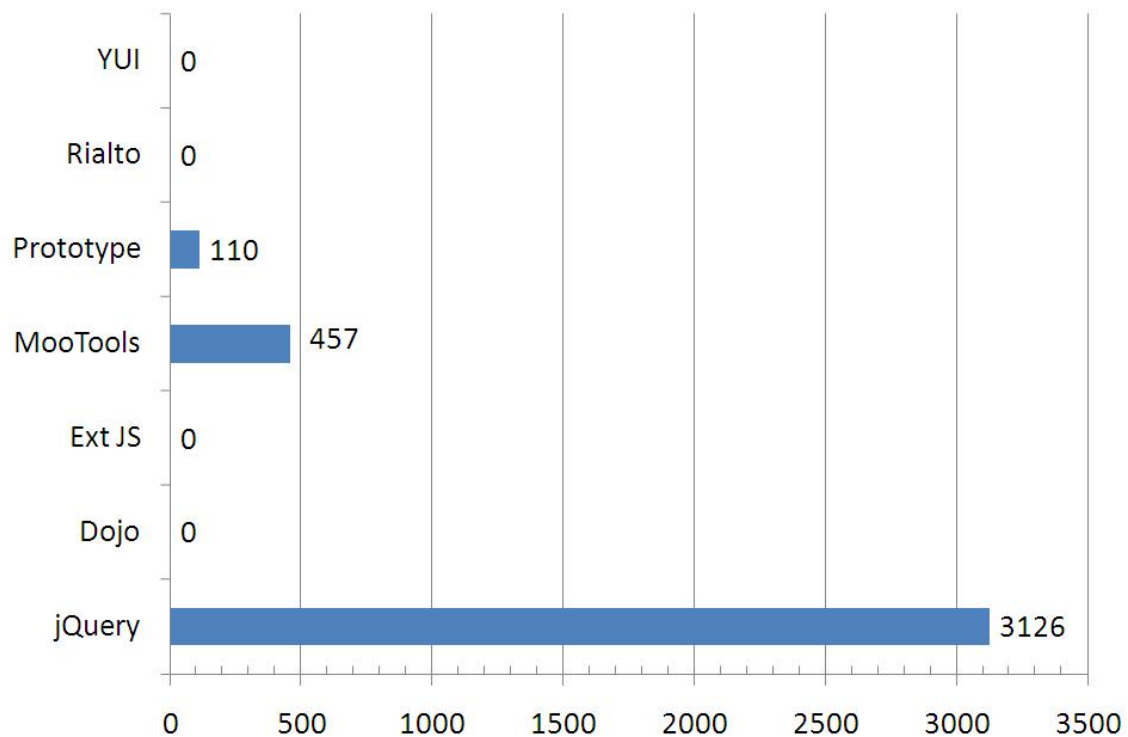


Abbildung 4: Anzahl an Plugins von verschiedenen JavaScript Frameworks

Gibt es eine derart große Anzahl an Plugins ist davon auszugehen, dass die Community das Framework gerne unterstützt und das Projekt weiter voran treiben will. Dies dient mitunter der Nachhaltigkeit des Frameworks.

⁵ „Document Object Model“. Bezeichnet die Struktur der HTML Elemente in einer Webseite

2.3 Verbreitung und Trends

Soll ein Projekt lange bestehen und auch zu einem späteren Zeitpunkt weiterentwickelt werden können, ist auch die Nachhaltigkeit der gewählten Projektbestandteile für den dauerhaften Erfolg entscheidend. Daher sind bei der Wahl einer bestimmten Programmiersprache oder Technik die Aktualität und der Verbreitungsgrad ein wichtiges Kriterium. Die Webseite „builtWith Trends“ [bWT1] befasst sich mit der prozentualen Verbreitung von Programmiersprachen im Web. Ebenfalls abrufbar ist der Einsatz verschiedener JavaScript Snippets im Web. Die Webseite zählt JavaScript Frameworks dabei zu solchen Snippets. Die letzte Auswertung vom April 2011 zeigt folgende Verbreitung:

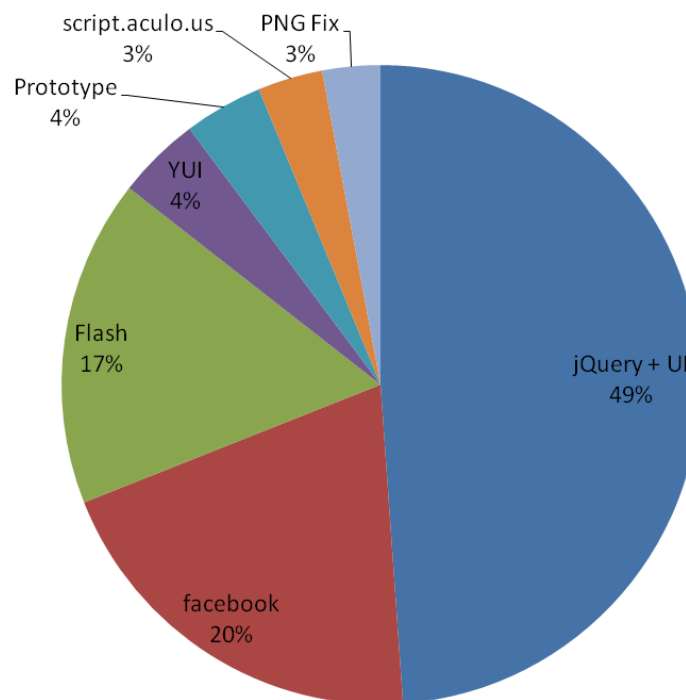


Abbildung 5: Prozentuale Verbreitung von JavaScript Frameworks und Snippets im Web

Nach eigenen Angaben von builtWith Trends werden wöchentlich ca. 90 Millionen Webseiten [bWT2] nach deren verwendeten Webtechnologien durchsucht und ausgewertet. Das in Abbildung 5 gezeigte Diagramm fasst jQuery und die Erweiterung von jQuery „jQuery UI“ zusammen und zeigt, dass 49% aller im Web gefundenen JavaScript Snippets mit jQuery geschrieben wurden. Das facebook Plugin (facebook connect oder der „Gefällt mir“ Button) sind mit 20% vertreten. Das SFW-Objekt und Adobe Active Content ist in diesem Diagramm unter „Flash“ mit 17% zusammengefasst. Erst danach kommen mit weniger als 5% weitere JavaScript Frameworks. Dazu gehören die Yahoo User Interface Library (YUI), Prototype und script.aculo.us. Der PNG Fix ist unabhängig davon zu sehen. Er dient lediglich dazu den Internet Explorer 5.5 und 6.0 um die Funktion zu erweitern transparente PNG's darzustellen.

Eine weitere Quelle, die die Popularität einer Programmiersprache oder Technik belegen kann ist Google. Google hat mit seinem Dienst „Google Insights for Search“ ein Tool zur Verfügung gestellt, mit in dem man die Häufigkeit von Suchanfragen zu einem bestimmten Begriff über die Zeit abrufen kann. Im Folgenden ein Vergleich von jQuery zu YUI, Prototype, MooTools und Dojo von 2004 bis Ende 2010:



Abbildung 6: Suchanfragen zu JavaScript Frameworks bei Google Insights for Search⁶

Zu erkennen ist, dass Prototype Mitte 2005 das erste gefragte JavaScript im Internet war. Obwohl John Resig ebenfalls 2005 mit der Entwicklung von jQuery begonnen hat (Release von Version 1.0 Anfang 2006), gelang der Durchbruch von jQuery erst Mitte bis Ende 2007 mit der Veröffentlichung der Erweiterung jQuery UI (*jQuery User Interface*). Der „Break Even Point“ für jQuery ist im ersten Drittel 2008 erreicht und geht seit dem stetig nach oben.

3 jQuery Philosophie

Das Grundprinzip von jQuery ist einfach gehalten:

1. Finde HTML im DOM-Baum
2. Mache etwas damit

Der Code hierzu folgt dabei dem gleichen Prinzip

```
$('#someElement').hide();
```

Der Selektor „\$“ läutet eine jQuery Funktion ein. Alternativ kann auch der String „jQuery“ an dessen Stelle verwendet werden, um die Kompatibilität mit anderen JavaScript Frameworks sicherzustellen. Der Selektor in den runden Klammern findet das entsprechende Element im DOM-Baum (siehe Kapitel 4) und übergibt es der darauf folgenden Funktion `.hide()`. Dieses Grundprinzip kann um die in den Kapiteln 4 bis 6 erläuterten Methoden, Ereignisse und Funktionen erweitert werden.

⁶ Die Marker A und B sind von Google Trends hinzugefügte News bzw. Events, die in Verbindung mit den Suchanfragen gebracht wurden. A: „Microsoft Adopts jQuery Javascript Library“
B: „Microsoft renews vows with JQuery Javascript“. Siehe auch [MINO]

4 jQuery Attributes

Alle Elemente in HTML besitzen diverse Attribute, die Informationen über das Element beinhalten. Dazu gehören Universalattribute, wie CLASS, ID, STYLE, TITLE, LANG und DIR, wobei es sich bei den letzten zwei um Sprachattribute zur Definition der Schriftrichtung und der Sprache handelt. Diese Attribute können in fast allen einleitenden HTML-Tags aufgeführt werden. Zusätzlich zu den Universalattributen gibt es noch Elementattribute, sozusagen Attribute, welche nur in bestimmten Elementen gültig sind, beispielsweise das ‚href‘ Attribute bei Hyperlinks oder das ‚action‘ Attribut bei Formularen. Die jQuery Attributes Funktionen dienen dazu, diese Eigenschaften und Attribute von HTML-Elementen auszulesen und zu verarbeiten. Zum einen können sogenannte Generalattribute aus den einleitenden HTML-Tags genutzt werden, wie beispielsweise die ID eines Elements oder der Wert (VALUE) von einem Eingabefeld, zum anderen kann auch der HTML-Inhalt des Elements selbst für eine DOM-Manipulation genutzt werden. Des Weiteren erlaubt jQuery auch eine Definition von HTML fremden Generalattributen. Das heißt es können Attribute in Elementen definiert und gelesen werden, welche vom HTML-Interpreter selbst nicht verstanden werden. Somit kann beispielsweise in einem hierarchischen System einem Element zusätzlich zur eigenen ID noch ein Attribut PARENTID definiert bekommen, welches die ID des Vaterknoten besitzt und welches dann per jQuery Attributes Funktion ausgelesen werden kann. Alle jQuery Attributes-Funktionen können sowohl zum Auslesen von Informationen (getter), als auch zum Ändern dieser Informationen (setter) genutzt werden.

Getter	Kommentar	Setter	Kommentar
<code>.val()</code>	Liest den Wert aus einem Eingabefeld	<code>.val('WERT')</code>	Setzt den Wert eines Eingabefeldes auf 'WERT'
<code>.html()</code>	Liest den HTML-Inhalt des Elements aus	<code>.html('Hallo')</code>	Setzt den HTML-Inhalt des Elements auf 'Hallo'
<code>.attr('id')</code>	Liest die ID eines Elements aus	<code>.attr('id', 'id1')</code>	Setzt die ID eines Elements auf 'id1'

Tabelle 1: Beispiele der jQuery Getter und Setter

5 jQuery Events

Da eine Internetseite nur einmal zur Darstellung vom Browser geladen wird und dieser nach dem Laden normalerweise erst wieder mit dem Server kommuniziert, wenn man auf einen Link oder ähnliches klickt, müssen Verhalten definiert werden, welche aktiviert werden, wenn der Benutzer mit dem Browser interagiert. HTML bietet hierfür bereits Attribute, über welche Events wie z.B. ein Klick (onclick) oder ein Tastendruck (onkeypress) ausgelöst werden. Wenn ein Event eintritt, wird eine Handler-Funktion aufgerufen, welche diesen Event dann verarbeitet. Da die Bindung von Events als HTML-Attribute aufwendig und redundant ist und eine unschöne Mischung von Javascript und HTML ergibt, ist es per jQuery Events möglich, diese Events per Javascript zu definieren und an eines oder mehrere Elemente zu binden.

Beispiel: Button

Soll beispielsweise ein Button auf ein Klick-Event reagieren, muss das Element einen Event-Handler zugewiesen bekommen, eine Funktion, die aufgerufen wird, wenn der Klick-Event eintritt. So muss erst dieser Button per Selektor definiert werden und anschließend die jQuery Event-Methode `.click()` aufgerufen werden, welche dem Button einen Event-Handler zuweist, der bei einem Mausklick auf dieses Event reagiert.

```
$('#button').click(handler);
```

Die Handler-Funktion kann entweder als Funktionsaufruf übergeben werden oder direkt als Parameter in der Methode `.click()` implementiert werden. Will man beispielsweise, dass unser Button einen Wert von einem Textfeld in ein zweites kopiert, wird die Handler-Funktion mitübergeben:

```
$('#button').click(function() {
    $('#id1').val( $('#id2').val() );
});
```

5.1 Ereignisgruppen

Grob gesehen unterscheidet man zwischen drei großen Ereignisgruppen, die im Folgenden näher erläutert werden.

5.1.1 Mouse-Events

Mouse-Events sind Events, die mit der Maus ausgelöst werden. Entweder durch die momentane Position der Maus, evtl. über einem Element, oder alternativ durch eine Geste mit der Maus, beispielsweise ein Klick oder ein Doppelklick. Wird ein Mouse-Event ausgelöst, so enthält das Eventobjekt zusätzlich Eigenschaften wie die momentane Position der Maus in x-Achse (`event.pageX`) und y-Achse (`event.pageY`)

Event	Kommentar
<code>.click(handler)</code>	Bindet einen Handler an das Element, der bei einem Mausklick ausgeführt wird
<code>.dblclick(handler)</code>	Bindet einen Handler an das Element, der bei einem Doppelklick ausgeführt wird
<code>.toggle(handler, handler, [handler])</code>	Bindet zwei oder mehrere Handler an das Element, welche bei verschiedenen Mausklicks ausgeführt werden
<code>.mousedown(handler)</code> <code>.mouseup(handler)</code>	Bindet einen Handler an das Element, welcher beim Drücken und beim Loslassen der Maustaste ausgeführt wird
<code>.mouseover(handler)</code> <code>.mouseout(handler)</code>	Bindet einen Handler an das Element, welcher ausgeführt wird, wenn der Mauszeiger in das Element oder ein Kindelement bewegt, oder das Element bzw. das Kindelement verlässt
<code>.mouseenter(handler)</code> <code>.mouseleave(handler)</code>	Bindet einen Handler an das Element, welcher ausgeführt wird, wenn der Mauszeiger in das Element bewegt, oder das Element verlässt. Diese Events werden anders als <code>.mouseover()</code> und <code>.mouseout()</code> nur für das Element selbst, nicht für Kindelemente definiert
<code>.mousemove(handler)</code>	Bindet einen Handler an das Element, der bei einer Bewegung

	der Maus ausgeführt wird
--	--------------------------

Tabelle 2: Übersicht der Mouse-Events

5.1.2 Keyboard-Events

Diese Events werden durch die Tastatur ausgelöst. Dabei wird zwischen den drei Zuständen Taste *gedrückt*, Taste *losgelassen* und Taste *gehalten* unterschieden. Wird ein Keyboard-Event ausgelöst, so enthält das Eventobjekt den Schlüsselcode der Taste, welche das Ereignis ausgelöst hat (`event.which`)

Event	Kommentar
<code>.keydown(handler)</code>	Bindet einen Handler an das Element, der bei einem drücken einer Taste ausgeführt wird
<code>.keyup(handler)</code>	Bindet einen Handler an das Element, der bei einem loslassen einer Taste ausgeführt wird
<code>.keypress(handler)</code>	Bindet einen Handler an das Element, der bei einem halten einer Taste ausgeführt wird

Tabelle 3: Übersicht der Keyboard-Events

5.1.3 Form-Events

Form-Events sind Ereignisse, die bei der Interaktion mit Formularen auftreten. Wenn sich beispielsweise der Wert eines Elements ändert, oder das Formular abgeschickt wird.

Event	Kommentar
<code>.blur(handler)</code>	Bindet einen Handler an das Element, der beim Verlassen des Elements (z.B. des Eingabefelds) ausgeführt wird
<code>.change(handler)</code>	Bindet einen Handler an das Element, der beim Verlassen des Elements ausgeführt wird, wenn sich was an seinem Wert geändert hat
<code>.focus(handler)</code>	Bindet einen Handler an das Element, der beim Fokussieren ausgeführt wird
<code>.select(handler)</code>	
<code>.submit(handler)</code>	Bindet einen Handler an das Formular-Element, der beim Absenden des Formulars ausgeführt wird

Tabelle 4: Übersicht der Form-Events

5.2 Event Objects

Werden Events ausgelöst, so wird immer auch ein Eventobjekt erstellt, welches verschiedene Informationen über das Ereignis enthält. Die Eventobjekte können von Browser zu Browser variieren, wodurch das Eventhandling unnötig verkompliziert wird. jQuery normalisiert dieses Objekt anhand des W3C-Standards. Dieses Eventobjekt wird dann optional als Parameter der Handler-Funktion übergeben.

Je nach dem um was für einen Event es sich handelt, werden unterschiedliche Eigenschaften einem Eventobjekt zugewiesen. Es sind zwar auch z.B. bei Keyboard-Events die Eigenschaften wie `event.pageX` definiert, allerdings enthalten diese in

diesem Fall keinen Wert. Abgesehen von den Ereignisspezifischen Eigenschaften existieren auch Standardeigenschaften, welche für alle Eventgruppen definiert sind⁷.

5.3 Individuelle Events

Abgesehen von den vordefinierten Events ist es auch möglich, eigene Events zu definieren und auszulösen. Dazu kann die Funktion `.bind(name, handler)` genutzt werden. Prinzipiell werden alle Events über diese Funktion definiert, indem z.B. `.click()` intern die Bind-Funktion aufruft:

```
$('#button').bind('click', handler);
```

Hat man sich auf diese Weise einen eigenen Event definiert, so kann dieser über die Funktion `.trigger(event)` ausgelöst werden:

```
// Definition des Events 'foobar' an den Button
$('#button').bind('foobar', handler);
// Auslösen des Events
$('#button').trigger('foobar');
```

5.4 Event-Handler Attachment

Wenn man ein umfangreicheres Skript hat, welches auch den DOM-Tree manipuliert, Elemente hinzufügt, entfernt oder verändert, ist es teilweise nötig auch an diese nachträglich veränderten Elemente an Event-Handler zu binden. Um dies zu realisieren gibt es die `.live()` Methode. Diese Methode bindet nicht nur die momentan vorhandenen Events, sondern auch alle zukünftigen Elemente der Seite.

```
$('#button').live('click', handler);
```

Ein einfaches Beispiel dazu ist wenn in einem Formular Benutzerinformationen gesammelt werden und der Benutzer optional mehrere E-Mail Adressen angeben kann. Das erste Feld existiert bereits und prüft bei der Eingabe auf die korrekte Formatierung der E-Mail Adresse. Kommen nun weitere Felder hinzu können mit der `.live()` Methode die neuen Felder an denselben Handler gebunden werden:

```
$('.email').live('change', validateEmail);
```

Die `.live()` Methode geht allerdings immer vom Dokument selber aus, was bedeutet, dass eventuell auch andere Elemente der Seite betroffen sein können, welche eigentlich nicht gebunden werden sollen. Soll dies verhindert werden, kann das Binden auch an andere Elemente delegiert werden. Dadurch werden Handler nur noch an Unterelemente von dem delegierten Element gebunden. Auch diese Bindungen werden über die Methode `.delegate()` für momentane und zukünftige Elemente durchgeführt:

```
$('#form').delegate('.email', 'change', validateEmail);
```

⁷ Eine List mit diesen Eventeigenschaften befindet sich unter:
<http://api.jquery.com/category/events/event-object/>

6 jQuery Effects

Mit nativem JavaScript ist es nur sehr umständlich möglich, Elemente zu bewegen und zu animieren. jQuery bietet verschiedene Möglichkeiten um Animationen auf der Webseite zu implementieren. Mit den jQuery Effects Methoden lassen sich viele Standard-Animationen und Effekte einfach nutzen. Die Bibliothek beinhaltet unter anderem folgende Effekte:

Methode	Kommentar
<code>.fadeIn()</code> , <code>.fadeOut()</code> , <code>.fadeTo()</code>	Verändert die Deckkraft des Elements stetig entweder auf 100%, auf 0% oder auf einen angegebenen Wert
<code>.hide()</code>	Blendet ein Element von der linken oberen Ecke aus
<code>.show()</code>	Blendet ein Element von der linken oberen Ecke ein
<code>.slideUp()</code> , <code>.slideDown()</code>	Blendet ein Element von oben nach unten ein, oder von unten nach oben aus
<code>.delay()</code>	Wartet eine gewisse Zeitspanne ab, bevor eine Animation durchgeführt wird

Tabelle 5: Übersicht der jQuery Effects Methoden

Diese Standard-Animationen bieten viele Möglichkeiten Elemente zu animieren. Zusätzlich kann auch die Geschwindigkeit der jeweiligen Animation als Parameter übergeben werden. Dies kann man entweder als String mit den Werten ‚slow‘, ‚fast‘ oder durch Angabe der Millisekunden geschehen, wobei die Werte ‚slow‘ für 600 Millisekunden und ‚fast‘ für 200 Millisekunden stehen. Zusätzlich ist es auch bei Standard-Animationen möglich, eine Funktionskette zu definieren, welche von der ersten zur letzten Funktion abgearbeitet wird. Ein Beispiel:

```
$('#div').slideUp(500).delay(500).slideDown(500);
```

Neben diesen Standard-Animationen ist es außerdem möglich, komplett individuelle Effekte zu erzeugen. Dabei wird als einziger erforderlicher Parameter für die Animation eine Menge von CSS-Eigenschaften definiert, welcher der Methode `.animate()` übergeben wird. Als optionale Parameter stehen, wie auch bei den Standard-Animationen, die Geschwindigkeit und eine Callback-Funktion zur Verfügung, welche aufgerufen wird, wenn die Animation beendet wurde.

7 Die Community

jQuery bietet eine große Anhängerschaft, welche nicht nur in Foren, Chats und Blogs eine große Hilfestellung bietet, sondern auch durch Erweiterungen einige Arbeit abnehmen kann. In den Foren können andere Programmierer nach ihren Erfahrungen befragt werden, oder auch bereits beantwortete Fragen nachgeschlagen werden. In den Blogs werden immer wieder die Neuerungen vorgestellt, beispielsweise wenn ein neues Release erscheint. Am Hilfreichsten ist allerdings bei weitem die jQuery API Reference.

7.1 jQuery API Reference

Die jQuery API Reference ist die größte Hilfestellung bei der Entwicklung mit Hilfe des Frameworks. Zum einen bietet er für Einsteiger eine umfassende Anleitung, wie man jQuery nutzen kann, zum anderen beinhaltet sie auch einige Anleitungen bei denen nach

und nach die Entwicklungsschritte beschrieben werden. Außerdem gibt es eine Sammlung mit häufig gestellten Fragen, welche hilfreich beantwortet wurden.

Aber auch für fortgeschrittene jQuery Entwickler ist die API Reference, durch die umfassende Sammlung der implementierten Funktionen und die Erklärung der Nutzung an Beispielen, eine große Hilfe.

Jede Funktion und ihre Parameter werden erklärt und anhand von meistens mehreren funktionierenden Beispielen dargestellt, wodurch der Entwickler sofort sieht wie sich die Funktion verhält und was es für weitere Möglichkeiten gibt.

7.2 jQuery Plugins

Neben dem Kern von jQuery existieren inzwischen unzählige Erweiterungen für das Framework, welche sich ohne Probleme nutzen lassen und meist von Nutzern von jQuery selbst entwickelt wurden. Mittlerweile gibt es 21 Kategorien, in welche die jQuery Plugins aufgeteilt werden. Diese beinhalten unter anderem Funktionen wie AJAX-Plugins, Slideshows, diverse Menüs, Tab-Navigationen oder auch Popups.

7.3 jQuery UI

Ähnlich wie die jQuery Plugins bietet das jQuery User Interface eine Erweiterung für das Framework. Während allerdings bei den Plugins die optische Darstellung im Hintergrund steht (beispielsweise bei AJAX-Plugins), wird diese bei dem User Interface in den Vordergrund gestellt. Das UI beinhaltet beispielsweise Funktionen wie einen Datepicker, einen Colorpicker, Slider oder auch Fortschrittsanzeigen. Nicht anders als die Plugins werden die UI Funktionen größtenteils von Nutzern des Frameworks entwickelt.

7.4 jQuery Books

Es existiert neben den Onlinedokumentationen zum jQuery Framework auch diverse Literatur zu diesem Thema.

Ein gutes Buch dazu ist das „jQuery Kochbuch“ von Cody Lindley (O'REILLY) [ORJQ]. Es liefert eine gute Einführung in die Thematik und ist sowohl für Einsteiger, wie auch für Fortgeschrittene jQuery Entwickler geeignet. Es beschreibt sowohl die Syntax von jQuery und liefert zusätzlich auch einige Programmierbeispiele, die im täglichen Gebrauch durchaus hilfreich sein können. Zusätzlich dient es auch hervorragend als Nachschlagewerk.

Der Nachteil an Fachliteratur zu diesem Thema ist, dass sie durch stetig neue Veröffentlichungen des Frameworks meistens nur für eine kurze Periode aktuell ist.

8 Zusammenfassung

Möchte man moderne Webseiten effizient und Browserübergreifend programmieren ist die Verwendung eines JavaScript Frameworks unabdingbar (siehe Kapitel 1.1). Für welches man sich entscheidet ist größtenteils Geschmackssache, da die Basisfunktionalitäten für einen Großteil der Anwendungsfälle in allen Frameworks vorhanden sind (siehe Kapitel 2.2). Bei der Arbeit mit jQuery fällt besonders die einleuchtende Syntax, die sehr übersichtliche und mit Beispielen belegte Dokumentation (siehe Kapitel 6.1), sowie die Vielzahl an verfügbaren Plugins (siehe Kapitel 2.2) auf. Zusammengefasst ist jQuery ein ausgesprochen starkes JavaScript-Framework, das vor allem durch eine leicht zu lernende Syntax, einfache DOM-Manipulation über CSS-Selektoren (siehe Kapitel 3) und die umfangreiche Funktionsbibliothek die Arbeit in der Webentwicklung erleichtert. Durch die zusätzlichen Plugins, die haus eigene Erweiterung jQuery UI und die große Community (siehe Kapitel 6) stehen viele sehr einfach zu implementierende Funktionen bereits zu Verfügung.

Viele große Webseiten setzen bereits auf jQuery (siehe Abbildung 7) und die aktuellen Entwicklungen zeigen, dass weitere Softwareriesen, wie Microsoft, Nokia [MINO] und Adobe [ADOBE] in Ihren Entwicklungen zukünftig auf jQuery setzen werden.



Abbildung 7: Webseiten und Unternehmen, die jQuery einsetzen (unvollständige Liste)

Für die Zukunft will sich jQuery mit der Entwicklung von jQuery Mobile [JQMOB] verstärkt auf den Einsatz von jQuery auf mobilen Endgeräten fokussieren. Die aktuelle Version 1.0 Alpha 4.1 wirbt bereits mit der Unterstützung von Android, iOS, blackberry und weiteren Betriebssystemen für Smartphones. Dies ist eine enorme Erleichterung in der „App-Entwicklung“, da die Entwickler Ihre Anwendungen nicht mehr für jedes Betriebssystem getrennt entwickeln müssten. Die Vor- und Nachteile dieser Verfahrensweise sind ein weiteres interessantes Thema.

Literatur

- [KJ09] Kai Jäger (2009): AJAX in der Praxis. Begriffserklärung „AJAX“ auf Seite 7. ISBN 978-3-540-69333-8. Hochschule der Medien, Springer-Verlag Berlin Heidelberg
- [AP06] Adam Presly, 2006: <http://www.adampresley.com/>
Snippet siehe: <http://www.bytemycode.com/snippets/snippet/329/>
Abgerufen am 07.04.2011
- [AMAZ11] Quelle: <http://phx.corporate-ir.net/phoenix.zhtml?c=176060&p=irol-newsArticle&ID=1521090> Abgerufen am 13.04.2011
- [KBW11] Quelle: <http://www.kabelbw.de/kabelbw/cms/InternetUndTelefon/>
Abgerufen am 08.04.2011
- [WIKI1] Wikipedia, April 2011: https://secure.wikimedia.org/wikipedia/en/wiki/Comparison_of_JavaScript_frameworks
Abgerufen am 08.04.2011
- [bWT1] builtWith Trends, April 2011: <http://trends.builtwith.com/javascript>
Abgerufen am 08.04.2011
- [bWT2] Quelle: <http://trends.builtwith.com/faq.aspx>
Abgerufen am 13.04.2011
- [ITU10] Internationalen Fernmeldeunion, 19. September 2010:
http://www.itu.int/net/pressoffice/press_releases/2010/33.aspx
Deutsche Übersetzung und Zitat (Abgerufen am 08.04.2011):
<http://www.tagesschau.sf.tv/Nachrichten/Archiv/2010/09/20/International/UNO-Breitband-Internet-fuer-die-Menschheit>
- [AG08] Aberdeen Group, November 2008: http://www.gomez.com/wp-content/downloads/Aberdeen_WebApps.pdf Seite 4
Abgerufen am 08.04.2011
- [ORJQ] Verfügbar unter http://books.google.de/books?id=3JtNrqEZK_cC
Abgerufen am 13.04.2011
- [MINO] Quelle siehe <http://www.heise.de/newsticker/meldung/Nokia-und-Microsoft-setzen-auf-jQuery-208664.html> und Marker in Abbildung 6.
Abgerufen am 13.04.2011
- [ADOBE] Quelle siehe <http://www.heise.de/newsticker/meldung/Adobe-stellt-Creative-Suite-5-5-vor-1225285.html> und
http://www.adobe.com/ch_de/products/creativesuite/web/features.html
Abgerufen am 14.04.2011
- [JQMOB] jQuery Mobile: <http://jquerymobile.com/>
Abgerufen am 13.04.2011
- [JQAPIR] jQuery API Reference: <http://api.jquery.com/>
Abgerufen am 23.03.2011