

AJAX

Ingo Ebel

8. Mai 2008

Inhaltsverzeichnis

1	Einführung	2
1.1	Web 2.0	2
1.2	Disruptive technology	2
2	Was ist Ajax?	2
2.1	Geschichte	3
2.2	Wozu wird AJAX gebraucht?	4
2.3	Funktionsweise	4
2.3.1	Unterschied zum Web 1.0	4
2.3.2	Basistechnologien	6
2.3.3	Client	6
2.3.4	Server	8
3	Sicherheit	8
3.1	Neue Probleme	8
3.2	Fazit	9
4	Zusammenfassung	9
4.1	Vorteile	9
4.2	Nachteile	9
A	Quellen	10
B	Lizenz	11

Abbildung 1: Viele soziale Netze



1 Einführung

1.1 Web 2.0

1.2 Disruptive technology

Das Internet prä Web 2.0 hatte ein Problem. Es führte alles nur in eine Richtung, vom Anbieter zum Konsumenten. Die Surfer konnten nicht mit den verschiedenen Services interagieren. Viele Technologien haben dazu beigetragen, dass sich das Web mehr zu einem Netz entwickelt, bei dem die User die Hauptrolle spielen. Beispiele sind hier RSS, bei dem sich News, aber auch Podcasts¹ aggregieren bzw. abonnieren lassen. Aber auch Blogs und Wikis, Soziale Netze und so weiter. All diese Dienste benutzen heute an irgendeiner Stelle auch Ajax.

2 Was ist Ajax?

AJAX war früher ein Akronym für Asynchronous JavaScript and XML und wurde daher auch großgeschrieben, heute ist es eher alleinstehend (ähnlich wie bei SOAP) und wird daher auch Ajax geschrieben. Außerdem wird heute auch nicht mehr überwiegend XML zum Transport benutzt, daher ist der Name nicht mehr gerechtfertigt. Der Hauptunterschied zu normalem HTTP ist die asynchrone Datenübertragung zwischen einem Server und dem Browser, das heißt, es können innerhalb einer HTML-Seite HTTP-Anfragen gestellt werden und Seitenteile ausgetauscht werden, ohne dass die komplette Seite neu geladen werden muss.

¹Podcasts sind Mediendateien (Audio oder Video), die über das Internet angeboten werden und mittels RSS-Feed abonniert werden können. Das Wort setzt sich aus iPod und Broadcasting zusammen.



Abbildung 2: inoffizielles Ajax-Logo

Ajax wird von allen gängigen Browsern (nativ) unterstützt:

- Konqueror (ab 3.2)
- Apple Safari (ab 1.2)
- Mozilla Firefox (1.0)
- Opera (8.0)
- IE (7.0)

2.1 Geschichte

Wo die Geschichte von Ajax beginnt, kann heute nicht mehr genau bestimmt werden, da Ajax ein Mix aus vielen Technologien ist. Auch der Namensursprung ist nicht mehr klar. Die erste Erwähnung des Namens kam jedoch in einem Aufsatz von Jesse James Gerrett "Ajax: A New Approach to Web Applications" vom 18. Februar 2005 vor. Dieser Aufsatz hat Ajax wohl zu den Anfangszeiten maßgeblich geprägt, auch weil der Aufsatz unter einer creative-commons-Lizenz veröffentlicht wurde und in mehrere Sprachen von verschiedenen Leuten übersetzt wurde.

Ajax ist, wie schon erwähnt, keine neue Technologie, sondern eine Mischung aus mehreren schon bekannten. So gab es von Microsoft erste Ansätze in den Jahren 1998/99. Ab 2005 war Ajax in den Medien stark präsent, gerade durch die verschiedensten Anwendungen des US-Unternehmens Google (Google Maps, Suggest). Kurze Zeit später, am 1. Februar 2006, gründete sich die OpenAjax Alliance. Über 15 Unternehmen gehörten zu den Gründern, unter anderem Global Players wie zum Beispiel Google, IBM, Mozilla Corporation, Novell, Oracle, Red Hat, Yahoo, Zend. Ziel der Vereinigung ist es, gemeinsame Standards zu schaffen und damit Alleingängen und Doppelentwicklungen vorzubeugen. Auch

das W3C² kümmert sich heute um Standardisierungen im Bereich von Ajax, um dieses voranzubringen.

2.2 Wozu wird AJAX gebraucht?

Ajax wird in vielen Bereichen eingesetzt. Hauptgrund ist, um Interaktivität zwischen Browser und Server zu ermöglichen, aber auch um Anwendungen bereitzustellen, die wie Desktop-Anwendungen benutzt werden können. Diese laufen zwar im Browser, der damit zum Rich Client wird, sie fühlen sich aber an, als würden diese Anwendungen lokal laufen. Dadurch wirken sie intuitiver, ein Klick ruft auch eine Aktion hervor und nicht ein Neuladen einer kompletten Seite; durch die asynchrone Übertragung wirken sie schneller.

exemplarisch einige Einsatzgebiete:

- soziale Software (xing, studivz, yigg, twitter.....)
- komplette Betriebssysteme (eyeOS)
- Mailprogramme (Roundcube...)
- Groupware (OpenXchange, Scalix..)
- Instant Messenger
- Textverarbeitung, Tabellenkalkulation
- Customer Relationship Management (CRM)
- Intranetanwendungen
- und vieles mehr...

2.3 Funktionsweise

2.3.1 Unterschied zum Web 1.0

- Früher: Ein Client stellt HTTP-Anfrage, der Webserver generiert die Seite und schickt Sie zum Client (synchrone Übertragung)
- Heute: Der Client kann Anfragen auch schicken, bei der nur Daten angefordert werden, die auch benötigt werden. Also eine asynchrone Übertragung. Der User kann weiter mit der Applikation arbeiten, während Daten vom Server übertragen werden.

²World Wide Web Consortium, internationales Konsortium, das sich um Webstandards kümmert

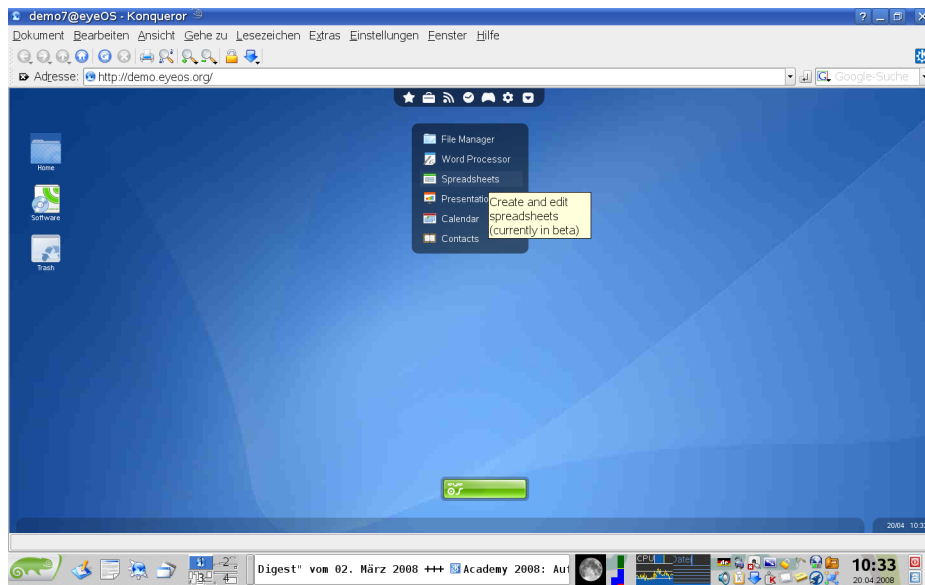


Abbildung 3: eyeOS, komplettes webbasiertes Betriebssystem

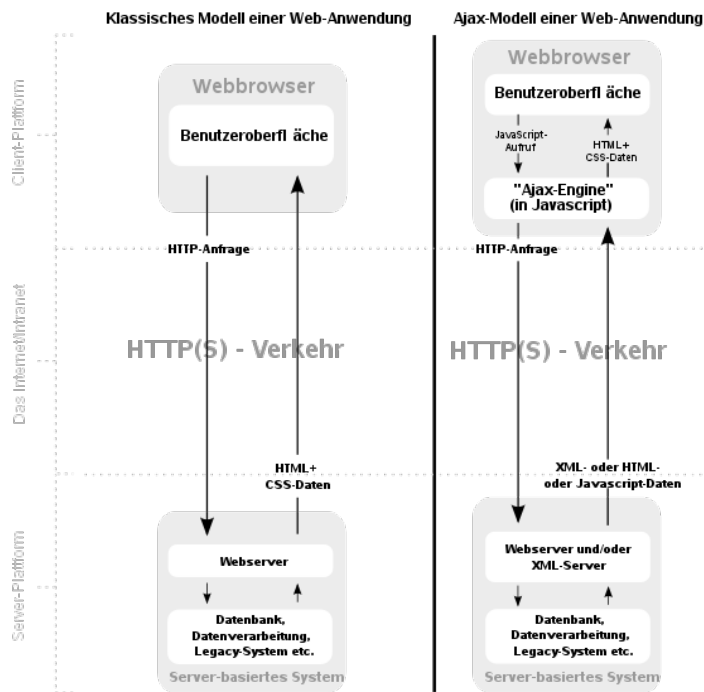


Abbildung 4: Unterschied zwischen einer HTTP-Verbindung mit und ohne Ajax

2.3.2 Basistechnologien

Eine Ajax-Anwendung basiert auf folgenden Web-Techniken:

- HTML- (oder XHTML-) Websites können aber auch in PHP, JSP, ASP etc. verfasst sein
- Document Object Model (DOM) zur Repräsentation der Daten oder Inhalte
- JavaScript zur Manipulation des Document Object Models und zur dynamischen Darstellung der Inhalte. JavaScript dient auch als Schnittstelle zwischen einzelnen Komponenten.
- Das XMLHttpRequest-Objekt, Bestandteil vieler Browser, um Daten auf asynchroner Basis mit dem Webserver austauschen zu können.
- Für die asynchronen Übertragungen können beispielsweise benutzt werden:
 - JSON (JavaScript Object Notation), eine Teilmenge des JavaScript-Sprachstandards, ist ein kompaktes Datenaustauschformat und wird heute häufig verwendet, da es schon JavaScript ist und zeitfressende Parsingvorgänge wie bei XML gespart werden können
 - Diverse proprietäre XML-Formate
 - SOAP (Austausch XML-basierter Nachrichten)
 - Plaintext
- meist auch verwendet für Ajax-Anwendungen, aber kein wirklicher Bestandteil von Ajax:
 - CSS zur Formatierung einer Webseite.
 - XSLT zur Datentransformation.

2.3.3 Client

Für Ajax-Anwendungen ist sowohl innerhalb des Webbrowsers als auch auf dem entsprechenden Server eine Komponente notwendig, die eine Ajax-basierte Kommunikation ermöglicht.

Auf der Clientseite ist das eine sogenannte Ajax-Engine. Zur Abwicklung der Ajax-Anfragen dient diese 'Ajax-Engine', eine in JavaScript geschriebene Komponente, die die clientseitige Arbeit übernimmt. Jede Benutzeraktion, die früher eine HTTP-Anfrage erzeugte, erzeugt nun einen JavaScript-Aufruf, der an die Ajax-Engine delegiert wird. Manche Aktionen wie Daten-Validierung³, oder sogar Navigieren kann die Engine beantworten, ohne Kontakt zum Server aufzunehmen. Sollten doch Daten vom Server benötigt werden um eine Aktion

³Was allerdings aus Security-Sicht lieber auf dem Server geschehen sollte

Ajax Modell einer Web-Anwendung (asynchrone Datenübertragung)

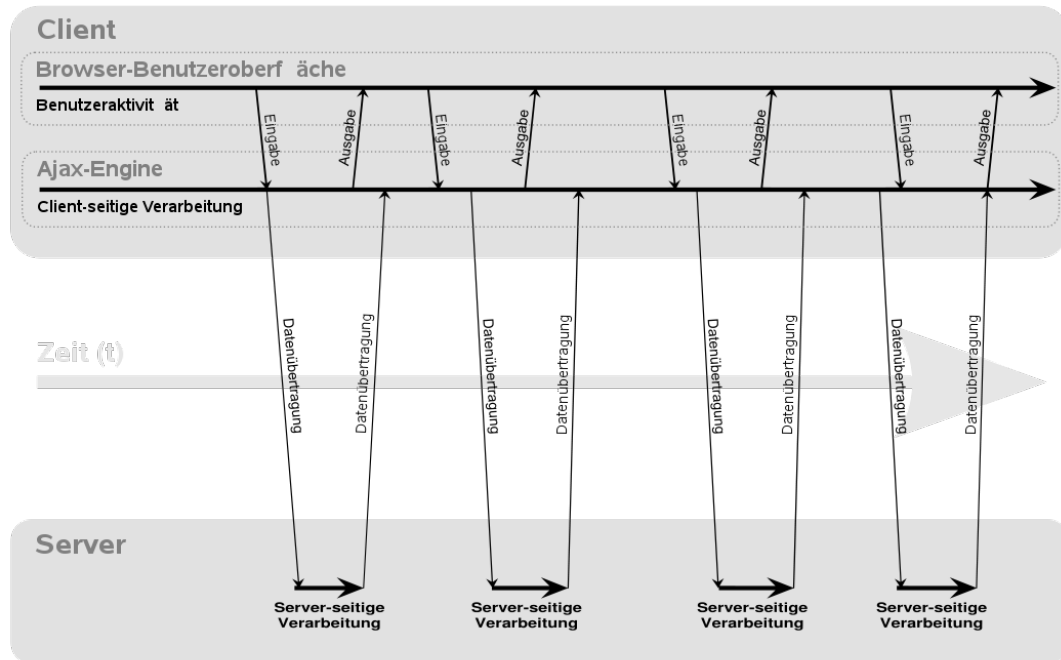


Abbildung 5: Der Ajax-Prozessfluss

ausführen zu können, zum Beispiel um Teile der Benutzeroberfläche nachzuladen oder neue Daten vom Server zu holen, geschieht dies asynchron meist mittels XML oder einer der anderen oben genannten Technologien. Die Interaktion des Benutzers mit der Webapplikation wird dabei nicht unterbrochen.

Umgesetzt wird die Ajax-Engine im Webbrowser mit der Hilfe einer umfangreichen Funktionalität auf der Basis von JavaScript und dem XMLHttpRequest-Objekt. Dabei gibt es zwei mögliche Implementierungen, die unterschieden werden. Entweder direkt dabei wird auf dem Client eine API zur direkten Kommunikation von Daten zur Verfügung gestellt oder indirekt, bei dem neue HTML-Fragmente vom Server an den Client gesendet werden, um die vorhandene Seite zu ergänzen oder Teile davon zu ersetzen. Die indirekte Variante ist einfacher zu implementieren und ist die "Thin client"-Herangehensweise; sie wird meist von legacy-Anwendungen, also Anwendungen, die über einen langen Zeitraum gewachsen sind, genutzt; dabei wird eher nur halbherzig auf Ajax umgestellt. Die direkt Variante ist zu bevorzugen, wenn eine Applikation von Grund auf neu geschrieben wird, da dies Serverressourcen schont.

2.3.4 Server

Die eigentliche Programmlogik der Anwendung ist immer noch auf einem Server hinterlegt, obwohl ein Großteil der früheren Aufgaben des Servers zum Client übergegangen ist. Die Programmlogik kann z.B. in Form von EJBs, .NET-Komponenten aber auch Skriptsprachen wie Ruby auf dem Server liegen. Es spielt hier keine Rolle, welche Technik verwendet wird, es ist bei Ajax nicht definiert und so kann auf dem Server jede Technik angewendet werden, die den Client mit seinen Daten versorgen kann. Sowohl der Server als auch die Anwendungslogik werden im Ajax-Kontext als Server-Plattform bezeichnet. Die Aufgabe ist simpel: Die Plattform muss die vom Browser benötigten Komponenten bereitstellen und ggf. Proxy-Funktionen übernehmen, wenn beispielsweise Daten von anderen Servern als der, mit dem der Browser gerade kommuniziert, bereitgestellt werden sollen.

3 Sicherheit

3.1 Neue Probleme

Das größte Problem ist sicherlich, dass die die Sicherheitsprobleme aller beteiligten Technologien vor allem von JavaScript und PHP bei Ajax eine Rolle spielen. Zusätzlich kommen noch Probleme hinzu, die aus der Kombination dieser Technologien entstehen, z.B. fragt AJAX die Daten in kleinen Schritten ab. Ein Angreifer kann dies ausnutzen und die Art der Abfrage herausfinden, um an die Daten zu kommen.

Ein weiteres großes Thema ist XSS (Cross-Site-Scripting). Hierbei wird JavaScript in den aktuellen Seitenkontext eingeschmuggelt. Es können dabei beliebige Requests ausgeführt werden; z.B. können lokale Daten, wie Cookies, der Inhalt von Formularfeldern oder die Browserhistory ausgelesen werden.

Auch das XmlHttpRequest-Element ist nicht frei von Problemen, die Same-Origin-Policy, welche Sicherheit schaffen und garantieren soll, dass der User auch wirklich mit nur dem einen Server kommuniziert, den er ausgerufen hat, kann umgangen werden z.B. mit DNS-Pinning oder Proxy-Request-Spoofing.

Wenn JSON zur asynchronen Übertragung genutzt wird, ist auch das wieder ein zusätzlicher Angriffsvektor, denn JSON ist nun mal Javascript. Hier könnte versucht werden, JavaScript in die Übertragung einzuschleusen.

Neu hinzu kommen jetzt auch "Web 2.0 Viren" wie der MySpace-Wurm⁴ vom 4. Oktober 2005. Dies stellt einen komplett neuen Virustyp dar, der sich ausschließlich über die Webapplikation vorbereitet. Eine Infizierung erfolgt per XSS und AJAX und er verbreitet sich per AJAX, Formulare, Links oder sogar RSS.

Immer mehr Logik wandert in den Client, bis zu 100% von View und Controller können im Browser stattfinden, dies lässt den Browser zu einem neuen

⁴Innerhalb von nur rund 18 Stunden hat "Samy" mehr als eine Million "Freunde" angehäuft, bevor der Account endgültig vom Besitzer gelöscht wurde.

großen Angriffsziel werden, das oft nicht so stark geschützt ist wie die Serverlandschaften.

3.2 Fazit

Es gibt immer mehr Schnittstellen zwischen Client und Server und damit auch mehr Angriffsvektoren, die ausgenutzt werden. Die Programmierer und Administratoren müssen aufpassen und immer mehr Technologien wie Web Applikation Firewalls, Reverse Proxies und so weiter einsetzen, um ein Ausnutzen dieser Angriffsmöglichkeiten kleinzuhalten und den Ausbruch von neuen Virentypen zu verhindern.

4 Zusammenfassung

4.1 Vorteile

Ajax hat gegenüber Technologien wie Flash oder Java einen entscheidenden Vorteil. Es wird von allen gängigen Browsern nativ unterstützt d.h., es wird kein Plug-In benötigt, das man sich zusätzlich installieren muss.

Ein weiterer Vorteil ist die geringere Serverlast. Während bei herkömmlichen HTML-Seiten jedes Mal die komplette Seite übermittelt werden muss, muss bei Ajax nur der Teil übertragen werden, der auch wirklich vom Client gebraucht wird. Außerdem kann durch die Auslagerung von einiger Programmlogik auch der Client einen Teil der Arbeit abnehmen und damit Bandbreite und Serverkapazität sparen.

Ajax setzt zudem auf Standardtechnologien, die schon lange genutzt werden wie JavaScript und XML. Diese Technologien können gemeinhin als ausgereift gelten. Außerdem sind sie durch jahrelanges Benutzen den potentiellen Anwendern bekannt.

4.2 Nachteile

Neben den schon unter Sicherheit genannten Problemen, die beim Einsatz von Ajax auftreten können, spielen auch noch weitere Nachteile eine Rolle. So ist zwar auf der einen Seite die Serverlast eine geringere, auf der anderen Seite aber auch eine viel höhere, bedingt durch die andere Nutzungsweise. So werden Inhalte ständig nachgeladen. Teilweise sind dies große Mengen an Daten wie zum Beispiel Karten- oder Satellitendaten. Viele interaktive Inhalte sind zwar reizvoll für den Nutzer, es kann jedoch auch geschehen, dass damit die Seiten überfrachtet werden und die Serverinfrastruktur nicht mitskaliert. So müssen bei größeren Diensten wie Twitter⁵ zeitweise einige Add-ons abgeschaltet werden, damit der Dienst überhaupt noch benutzbar bleibt.

Auch sind umfangreiche Tests der Applikationen unabdingbar. Nicht nur um eventuelle Sicherheitsprobleme aufzudecken, sondern auch um die verschiedenen

⁵Twitter ist ein soziales Netzwerk und ein Mikro-Blogging-Dienst.

Clients (sprich Browser) zu testen. So verhalten sich die Browser teilweise unterschiedlich, früher war sogar teilweise eine andere Schreibweise nötig, damit die Applikation überhaupt funktioniert.

Intuitiv sollen die Ajax-Anwendungen sein. Das sind sie auch im Gegensatz zu alten Webanwendungen, aber die User haben sich daran gewöhnt, mit der "Zurück-Schaltfläche" des Browsers auch ihre letzte Aktion rückgängig machen zu können. Dies ist bei dieser Art dynamischer Webseiten, die nur Teile ihrer Inhalte neu laden, jedoch ein Problem. Der Browser erkennt nur die Seite als Ganzes und würde zur vorhergehenden Seite springen, was der User nicht erwartet. Hierzu gibt es jedoch schon Lösungsansätze und einige Frameworks haben diese auch schon integriert.

Das Problem mit den States, also wo sich der User gerade befindet, welche Kette von Aktionen er genommen hat, betrifft nicht nur die Zurück-Schaltfläche, sondern auch Lesezeichen/Bookmarks, Search Engine Optimization (SEO) und vieles mehr. Außerdem sind Ajax-Seiten per se nicht barrierefrei. Und wenn ein User JavaScript ausgeschaltet hat, wird eine Ajax-Seite auch nicht zu verwenden sein. All dies sind Probleme die gelöst werden müssen oder derer sich ein Programmierer solcher Websites bewusst sein sollte. Bei den letztgenannten hilft eigentlich nur, eine zweite Seite ohne Ajax bereitzustellen.

A Quellen

- AJAX mit PHP, Ralph Steyer, Verlag Addison Wesley, 2006
- <http://www.openajax.org>
- [http://de.wikipedia.org/wiki/Ajax_\(Programmierung\)](http://de.wikipedia.org/wiki/Ajax_(Programmierung))
- <http://de.wikipedia.org/wiki/SOAP>
- <http://praegnanz.de/essays/submit-buttons-waren-gestern-heute-ist-ajax>
- <http://ajax.org>
- http://www.mayflower.de/images/media/downloads/AJAX_in_Action_Web20_Security.pdf
- http://www.distinguish.de/?page_id=46
- <http://www.json.org/>
- <http://www.heise.de/newsticker/XSS-Wurm-legt-MySpace-lahm-/meldung/65039>

B Lizenz

Dieses Dokument unterliegt der Creative Commons Lizenz CC-BY-NC-SA DE. Informationen, welche Rechte damit eingeräumt werden, können unter <http://creativecommons.org/licenses/by-nc-sa/2.0/de/> nachgelesen werden.

