Ajax

Am 18.02.2005 veröffentlichte Jesse James Garrett von Adaptive Path einen Artikel über ein neues Konzept zur Entwicklung von Web-Anwendungen, welches man bei seiner Firma auf den Namen “Ajax” getauft hatte. Der Artikel steht inzwischen unter einer Creative Commons License und darf u.a. frei weiterverbreitet werden, dennoch habe ich mir am 23.07.2006 eine offizielle Erlaubnis von Mr. Garrett zu dieser Übersetzung eingeholt. Die Übersetzung steht natürlich ebenfalls unter der o.g. CCL.

Ajax - ein neuartiger Ansatz für Web-Anwendungen

Wenn etwas bei der Gestaltung von Schnittstellen zur Zeit als “schick” gilt, dann ist es die Entwicklung von Web-Anwendungen. Mal ehrlich, wann hörten Sie zuletzt jemanden vom Schnittstellendesign eines Produktes schwärmen, welches *nicht* im Web zu finden war (vom iPod vielleicht einmal abgesehen)? Alle innovativen neuen Projekte sind online.

Trotzdem können Webdesigner nicht leugnen, dass sie ein wenig neidisch auf die Entwickler von Desktop-Software schauen. Deren Anwendungen besitzen eine Reichhaltigkeit und eine Ansprechbarkeit, die für das Web einfach unerreichbar schien. Dieselbe Einfachheit, die die schnelle Verbreitung des Webs ermöglicht hat, erzeugt eine Diskrepanz zwischen der Benutzerführung, die wir anbieten können und der Benutzerführung, die die User von einer Desktop-Anwendung erwarten dürfen.

Diese Diskrepanz wird nun immer mehr geschlossen. Schauen Sie auf “Google Suggest” und auf die Art und Weise, wie die vorgeschlagenen Begriffe sich nahezu sofort anpassen, während Sie tippen. Schauen Sie auf “Google Maps”, zoomen und scrollen Sie die Karte ein wenig. Auch hier geschieht alles nahezu in Echtzeit, ohne dass Sie lange auf das Nachladen der Seite warten müssen.

Google Suggest und Google Maps sind zwei Beispiele für einen neuen Ansatz für Web-Anwendungen, den wir bei Adaptive Path auf den Namen “Ajax” getauft haben, eine Abkürzung für “Asynchronous JavaScript And XML”. Ajax repräsentiert eine fundamentale Weiterentwicklung dessen, was im Web möglich ist.

Wie wird Ajax definiert?

Ajax ist keine einzelne Technologie. Ajax beinhaltet mehrere Technologien, jede mit ihrer ganz besonderen Daseinsberechtigung, die auf neue und mächtige Weise miteinander verbunden wurden. Im einzelnen handelt es sich um:

- standardgerechte Präsentation mit XHTML und CSS
- dynamische Anzeigen und Interaktivität mittels des Document Object Models (DOM)
- Datenaustausch und -manipulation mit XML und XSLT
- asynchrone Datenabfrage unter Verwendung von XMLHttpRequest
- und schließlich JavaScript, das all dies zusammenbringt

Das klassische Modell von Web-Anwendungen arbeitet wie folgt: Benutzeraktionen lösen einen HTTP-Request zum Webserver aus. Der Server führt daraufhin bestimmte Prozesse aus, z.B. Daten sammeln, Berechnungen durchführen, andere Services abfragen, und liefert schließlich eine HTML-Seite an den Client aus. Es ist ein Modell in Anlehnung an die ursprüngliche Form des Web als Hypertextmedium, aber wie Fans von “The Elements of User Experience” wissen: Was das Web zu Hypertext befähigt, ist nicht notwendigerweise gut für Software-Anwendungen.

Das klassische Modell macht in technischer Hinsicht Sinn, aber nicht im Hinblick auf gute Benutzerführung. Was macht der User, während der Server arbeitet? Genau, warten. Und bei jedem weiteren Schritt wartet er wieder.

Wenn wir das Web für Anwendungen völlig neu designen könnten, würden wir den User natürlich nicht ständig warten lassen. Wenn ein Interface einmal geladen wurde, warum sollte die Interaktion jedesmal zum Stillstand kommen, wenn die Anwendung etwas vom Server benötigt? Warum sollte der User überhaupt etwas von dieser Kommunikation mitbekommen?

Was ist anders an Ajax?

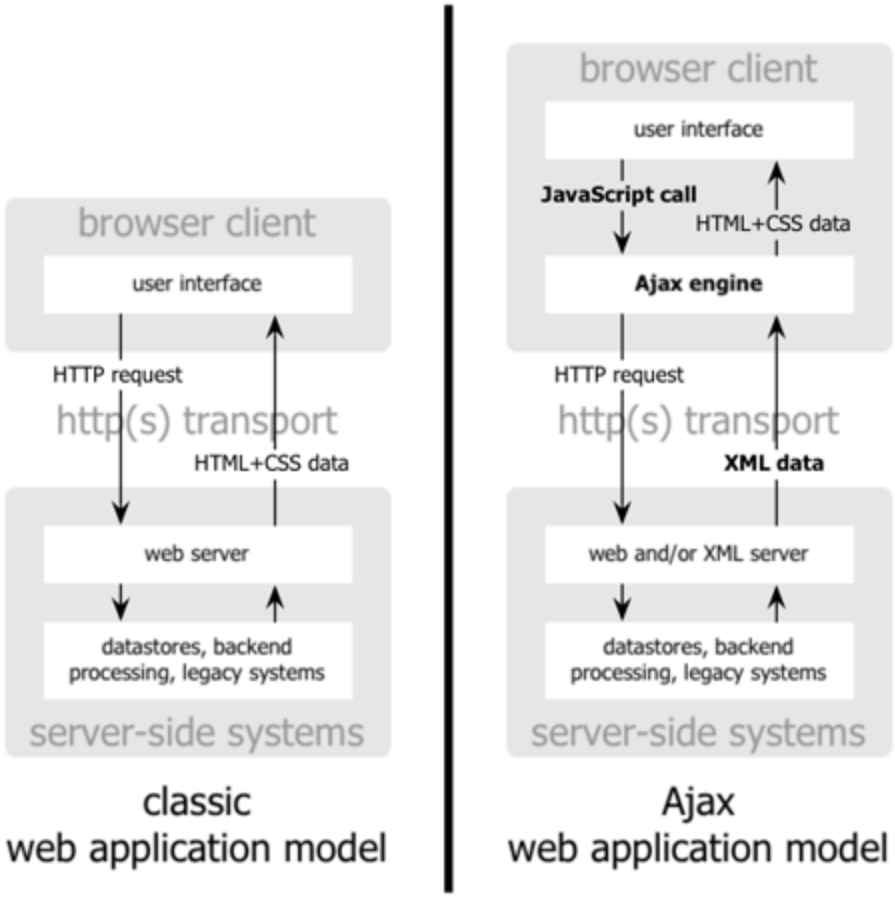

Abbildung 1: Das traditionelle Modell für Web-Anwendungen (links) im Vergleich zum Ajax-Modell (rechts)

Eine Ajax-Anwendung eliminiert die “start-stop-start-stop”-Methode bei Interaktionen im Web, indem sie einen Intermediär, die Ajax-Engine, zwischen Client und Server schaltet. Man könnte meinen, die Hinzunahme einer weiteren Ebene müsste die Anwendung weiter verlangsamen, doch das Gegenteil ist der Fall.

Anstatt einer Webseite lädt der Browser zu Beginn einer Session eine Ajax-Engine, geschrieben in JavaScript und für gewöhnlich in einem versteckten Frame untergebracht. Diese Engine kümmert sich sowohl um die Darstellung des Interfaces, als auch um die Kommunikation mit dem Server (im Auftrag des Users). Diese beiden Vorgänge erfolgen asynchron, die Interaktion des Users mit der Anwendung ist also unabhängig von der Kommunikation mit dem Server. Daher muss der Benutzer niemals auf ein leeres Fenster mit einem Sanduhr-Icon starren, während er auf Rückmeldung vom Server wartet.

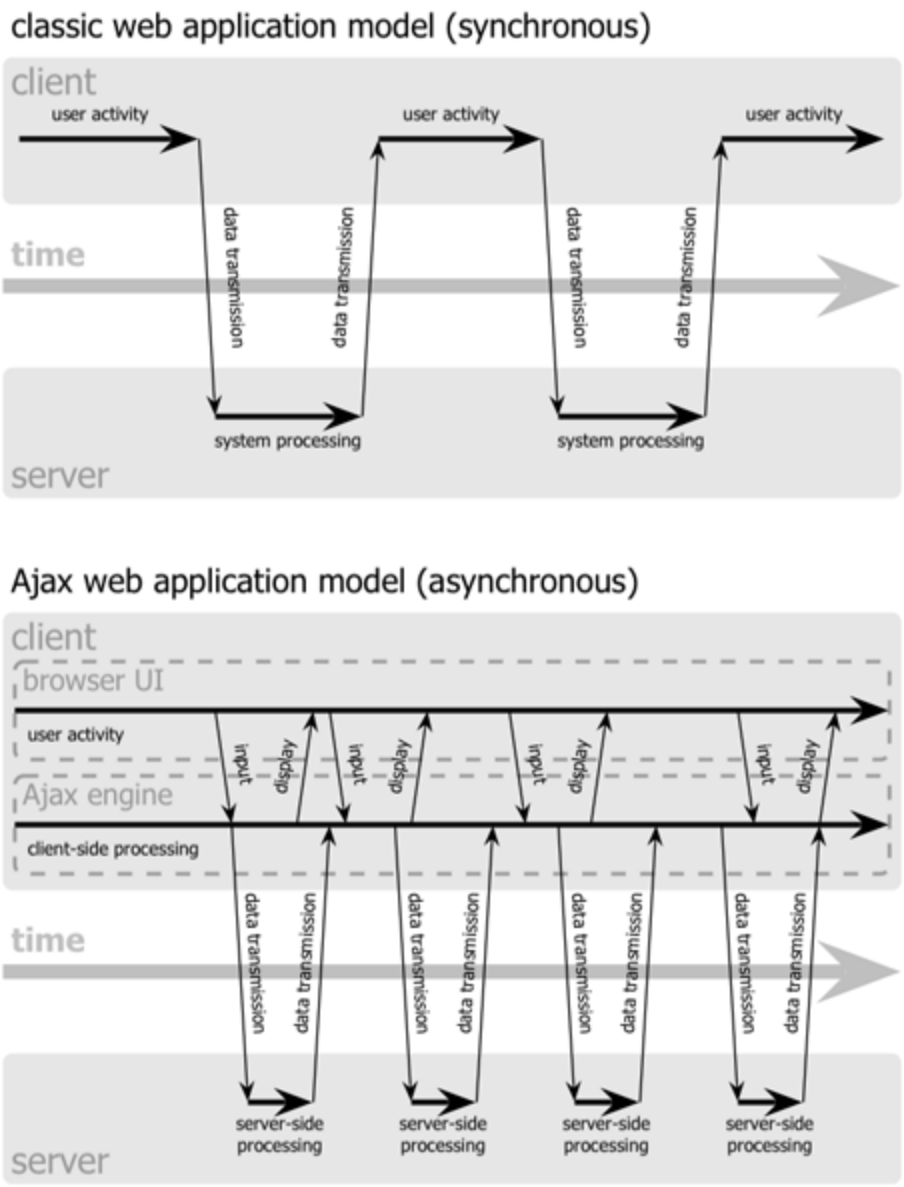

Abbildung 2: Das synchrone Interaktionsmuster einer traditionellen Web-Anwendung (oben) im Vergleich zum asynchronen Muster einer Ajax-Anwendung (unten)

Jede Benutzeraktion, die normalerweise einen HTTP-Request zur Folge hat, nimmt stattdessen die Form eines JavaScript-Aufrufes an die Ajax-Engine an. Jede Reaktion auf eine Benutzeraktion, die nicht unbedingt eine Anfrage an den Server erfordert (z.B. Datenvalidierung, Datenbearbeitung im Speicher, einfache Navigation), bearbeitet die Engine selbst. Wenn die Engine für die Antwort etwas vom Server benötigt (sei es für die Weiterverarbeitung von Daten, das Nachladen von weiterem Interface-Code oder das Abfragen aktualisierter Daten), geschieht dies asynchron mittels XML, ohne dass die Interaktion zwischen User und Anwendung gestört wird.

Wer benutzt Ajax?

Google tätigt hohe Investitionen in die Entwicklung des Ajax-Ansatzes. Alle größeren Google-Produkte des letzten Jahres - Orkut, Google Mail, Google Groups, Google Suggest und Google Maps - sind Ajax-Anwendungen. (Für einen weitergehenden Einblick in das technische Einmaleins dieser Ajax-Implementationen sei auf die folgenden exzellenten Analysen verwiesen: Google Mail, Google Suggest, Google Maps) Andere folgen diesem Beispiel: Viele der Features, die die Leute an flickr so mögen, basieren auf Ajax. Und Amazon's Suchdienst A9.com verwendet ähnliche Techniken.

Diese Projekte demonstrieren, dass Ajax nicht nur ein technisches Spielzeug, sondern äußerst praktikabel für reale Anwendungen ist. Es ist eben keine weitere Technologie, die nur im Labor funktioniert. Und Ajax-Anwendungen können beliebig groß sein, vom einfachen Google Suggest mit nur einer Funktion bis hin zum sehr komplexen und anspruchsvollen Google Maps.

Bei Adaptive Path nutzen wir Ajax seit einigen Monaten für unsere Arbeit und wir beginnen zu erkennen, dass wir bislang nur an der Oberfläche der Möglichkeiten von Ajax gekratzt haben. Ajax ist eine bedeutende Entwicklung für Web-Anwendungen, und seine Bedeutung steigt immer weiter. Da bereits sehr viele Entwickler diese Technik kennen und beherrschen erwarten wir, dass viele weitere Organisationen dem Beispiel von Google folgen und den Wettbewerbsvorteil ausschöpfen, den Ajax bietet.

Wie geht es weiter?

Die größten Herausforderungen bei der Entwicklung von Ajax-Anwendungen sind nicht technischer Natur. Die technischen Kernkomponenten sind ausgereift, stabil und wohlverstanden. Stattdessen wird die größte Herausforderung für die Designer sein, zu vergessen was sie über die Beschränkungen des Web zu wissen glauben, und sich einen größeren, reichhaltigeren Bereich der Möglichkeiten vorzustellen.

Das wird sicher Spaß machen.

Aktualisiert am 14. NOVEMBER 2008