AJAX Hintergrundwissen

AJAX Hintergrundwissen

AJAX steht für Asynchronous JavaScript and XML.

Vermutlich weiß jetzt niemand mehr als zuvor, doch es sollte immerhin klar geworden sein,
dass es ein Begriff aus der Informatik ist und man für dessen Verwendung weder das gleichnamige Waschmittel, noch den Fußballverein kennen muss.

AJAX ist keine Programmiersprache, sondern beschreibt das Konzept von drei Technologien:
DOM zur Manipulation von Web-Seiten,
JavaScript zur Ereignisbehandlung und
HTTP zum Nachladen von Daten.

Inhaltsverzeichnis

Das Zusammenspiel dieser drei Technologien erlaubt dynamische Websites,
die durch asynchrone Verbindungen in einer interaktiven Umgebung beliebige Datenmengen bewältigen können.

Der Client schickt HTTP-Anfragen an den Server, ohne dass die eigentliche Seite neu geladen wird, oder der Benutzer währenddessen warten muss.

Document Object Model (DOM)

Das Document Object Model (kurz DOM) ist eine vom W3-Konsortium verabschiedete Norm für die Definition einer Baumstruktur und ihrer Traversierungsfunktionen in XML.
Mit DOM lässt sich also ein komplettes HTML Dokument in ein dynamisches Objekt binden.

Das Beispiel zeigt den DOM-Baum einer Standard-Web-Seite, mit Head- und Body-Bereich und mehreren Unterknoten.
easy-coding.de/Attachment/492/…bf0cd7cfe47e8aea8e67a14c0

In einer Client-Server-Architektur lässt sich DOM sowohl im Backend zur Dokumentengenerierung,
als auch und im Frontend zum dynamischen Manipulieren nutzen.

JavaScript

Um Verwechslungen vorzubeugen, sei an dieser Stelle gesagt, dass es sich bei JavaScript weder um Java handelt, noch dass konzeptionelle Ähnlichkeiten zwischen den beiden Sprachen existieren.
JavaScript Code wird nicht für die Ausführung in einer VirtualMaschine kompiliert, sondern zur Interpretation im Klartext an den Browser übertragen.
Die Ausführung des Codes läuft somit nicht auf dem Webserver, sondern auf dem eigenen Rechner ab.

JavaScript läuft in einer Sandbox und schränkt darüber den Zugriff auf Netzwerkfunktionen oder das Dateisystem ein.

Skripte werden in das HTML-Dokument eingebettet, um auf Ereignisse zu reagieren oder die DOM Repräsentierung der Website dynamisch zu manipulieren.

Ein simples Beispiel ist der Aufruf der alert-Funktion.

Quellcode

```
1. <html>
2. <head><title>Test</title>
3. <script type="text/javascript">
4. alert("Hello World!");
5. </script>
6. </head><body>
```

7. </body></html>

easy-coding.de/Attachment/493/…bf0cd7cfe47e8aea8e67a14c0

Für die Programmierung mit JavaScript gibt es ein umfangreiches Online-Standardwerk unter de.selfhtml.org.

HTTP

HTTP ist ein zustandsloses Protokoll zur Datenübertragung zwischen zwei Computern.
Ein Computer formuliert eine Anfrage.
Der andere Computer beantwortet diese mit einem Inhalt.

Normale Websites laden in einem Rutsch.
Dazu formuliert der Client eine HTTP-Anfrage mit Hostnamen und Pfad.
Wird die URL foo.com/ajax.html im Browser aufgerufen, wird die folgende Anfrage gesendet:

Quellcode

1. GET /ajax.html HTTP/1.1
2. Host: www.foo.com

Der Server beantwortet die Anfrage mit einem Status-Code und den angeforderten Daten:

HTML-Quellcode

1. HTTP/1.1 200 OK
2. Content-Type: text/html
3. <html>
5. <head>
6. <title>Hello World!</title>
7. </head>
8. <body>
9. Hello World!
10. </body>
11. </html>

Alles anzeigen

Klickt man auf einen Link werden wieder und wieder neue HTTP-Anfragen geschickt und das komplette Dokument neu übertragen.

Das folgende Sequenzdiagramm beschreibt den Ablauf von Anfrage und Antwort.
easy-coding.de/Attachment/494/…bf0cd7cfe47e8aea8e67a14c0

Asynchrones HTTP

AJAX erweitert dieses starre Konzept und erlaubt das asynchrone Senden und Empfangen von Daten über HTTP.
Das bedeutet, dass das Dokument nicht komplett geladen werden muss,
sondern es stattdessen während der Laufzeit nachgeladen werden kann.
Während des Ladevorgangs kann der Benutzer weiter mit der Seite interagieren,
da der Browser dafür sorgt, dass die asynchrone Anfrage in einem eigenen Thread abläuft.

Zudem ist keine Socketverbindung auf Ebene von TCP oder UDP notwendig.
Das Einfügen in den DOM Baum erledigt dann JavaScript.

So funktioniert das Zusammenspiel der drei Technologien:
Die Ajax-Engine reagiert auf einen Klick in der Benutzeroberfläche,
schickt daraufhin eine HTTP-Anfrage an den Server und sobald dieser antwortet,

wertet sie die Antwort aus und fügt die Informationen mit DOM-Manipulationen in das eigentlich Dokument wieder ein.
easy-coding.de/Attachment/495/…bf0cd7cfe47e8aea8e67a14c0