Einführung in die AJAX Programmierung

Inhaltsverzeichnis

Ereignisbehandlung mit EventHandlern

Die HTTP-Anfragen werden durch JavaScript Ereignisse ausgelöst.
EventHandler sind das Bindeglied zwischen HTML und JavaScript.
Das W3-Konsortium hat sie als HTML-Attribut in den Sprachstandard aufgenommen.

Hier ist ein Handler definiert, der ausgelöst durch einen Mausklick die Funktion foo() aufruft.
An dieser Stelle wird davon ausgegangen, dass die Funktion foo() existiert.
Sie wird im Beispiel nachgereicht.

Quellcode

```
1. <div onclick="foo()">Container</div>
```

XMLHTTP als Kommunikationsschnittstelle

Das XMLHttpRequest Objekt ist eine API, die den HTTP-Zugriff auf Websites der eigenen Domain erlaubt.
Das W3C lieferte einen Vorschlag für eine Cross-Site-Extension in 'XMLHttpRequest 2' unter lists.w3.org/Archives/Public/public-webapi/2006Jun/0012.html.

Für die Kommunikation über das HTTP-Protokoll werden die beiden gebräuchlichsten Anfrage-Methoden POST und GET genutzt.
Wurde ein Event ausgelöst und eine Anfrage abgeschickt, übernimmt das XMLHttpRequest den Aufruf der Callback-Methode.

Aus historischen Gründen wird das XMLHttpRequest Objektes von den meisten Browser unterschiedlich initialisiert und sollte daher stets gekapselt werden.
Eine Implementierung für die gängigsten Browser findet man unter de.wikipedia.org/wiki/XMLHttpRequest

Callback Methoden

Die Callback Methode ist die 'Rückruf'-Funktion mit einer Antwort vom Server.
Das XMLHTTP Objekt initialisiert die Callback Methode und ruft sie nach jeder readyState-Änderungen auf.
Erst beim Status Vier steht das gesamte Dokument zur Verfügung.

Wann und ob die readyStates tatsächlich wechseln, kann auf Grund vieler Einflussfaktoren wie Bandbreite und Serverauslastung nie garantieren werden.
Die Callback Methode ist daher als eine void-Funktion ohne Rückgabewert zu implementieren.
Häufig nutzt man den Callback, als eine anonyme Methode.

Vollständiges Beispiel

Nun ist alles bekannt, was benötigt wird, um die Funktion foo() zu implementieren.
Sie soll eine AJAX Anfrage ausführen, um eine Liste von Namen nachzuladen.
Dazu wird die AJAX Klasse initialisiert und eine GET-Anfrage an backend.php geschickt.
Diese liefert im Beispiel folgendes XML Dokument.

Quellcode

```
1. <root>
2. <online>hans</online>
3. <online>peter</online>
4. <online>gunther</online>
```

5. </root>

In unserer anonymen Callback Methode, können wir die Daten mit DOM traversieren und in den Node mit appendChild direkt in den Website-Context einfügen.

Quellcode

```
1. function foo() {
2. var req = new XMLHttpRequest(); //mozilla implementierung
3. req.open('get', 'backend.php');
4. req.onreadystatechange = function() {
5. if ((req.readyState == 4) && (req.status == 200)) {
6. var domZiel = document.getElementById('context');
7. var xmlQuelle = req.responseXML.getElementsByTagName('online');
8. for(var i=0; i<xmlQuelle.length; i++) {
9. domZiel.appendChild(xmlQuelle[i]);
10. }
11. }
12. }
13. req.send(null);
14. }
```

Alles anzeigen

Probleme in AJAX

Das Arbeiten mit den am meisten verbreitetsten Web-Technologien HTML und JavaScript stellt die Entwickler vor eine neue Herausforderung.

Bei der normalen GUI-Programmierung kann die Anwendung bereits innerhalb der Entwicklungsumgebung debuggt und getestet werden.
Da JavaScript-Anwendungen aber erst beim Endbenutzer interpretiert werden,
benötigt der Entwickler verschiedene Testsysteme aus Betriebssystem- und Browserkombinationen.
Auch der komplette Klick-Lebenszyklus wird durch das besondere Verhalten der Callback-Methoden und die strikte Trennung zwischen der JavaScript-Frontend-Implementierung und der Backend-Implementierung schier unmöglich zu testen.
Es werden Testszenarien gebraucht die eine Serverüberlastung emulieren.

Selbst für den Benutzer ergeben sich Nachteile.
Denn das Navigieren über die Browserfunktionen oder das Neuladen der Seite können ohne ein ausgeklügeltes Konzept zu Fehlverhalten führen.