AJAX: Timeout prüfen

Im Gegensatz zu Socket-Objekten, die man aus vielen Programmiersprachen kennt, besitzt das XMLHttpRequest Objekt, das für den AJAX Datenaustausch genutzt wird, kein Attribut zum Setzen der Timeout Zeit.
Das Problem kann man leicht lösen, indem man zu Beginn des AJAX Aufrufs einen Timer setzt. Der Timer ruft dann eine Funktion auf, die aussagt, dass der Timeout abgelaufen ist.
Wenn der AJAX Request vorher fertig geladen ist, wird der Timer abgebrochen.

In diesem Beispiel ist der Timeout als der Zeitpunkt definiert, zu dem dem der Inhalt des AJAX Request fertig geliefert wurde. Alternative Status sind:
0 = uninitialized
1 = loading
2 = loaded
3 = interactive
4 = complete

Die zu ändernde Stelle ist im Quelltext markiert.

Quellcode

```
<html><head>
<title>AJAX: Timeout prüfen</title>
<script type="text/javascript">
<!--
// globale Variablen
var timeout = 2000; // in Millisekunden
var timer;
// AJAX Objekt initialisieren
try {
req = window.XMLHttpRequest?new XMLHttpRequest():
new ActiveXObject("Microsoft.XMLHTTP");
} catch (e) {
//Kein AJAX Support
}
/**
* zerstört den AJAX Request und schreibt eine Fehlermeldung
*/
function timeoutCheck() {
req = null;
document.getElementById('j').innerHTML += 'error';
}
/**
* lädt Inhalt von status.php - inkl. Timeout Behandlung
*/
function ajax() {
// Timer beginnen
timer = window.setTimeout("timeoutCheck()", timeout);
// Readystate behandlung
req.onreadystatechange = function() {
if ((req.readyState == 4) && (req.status == 200)) { // Hier Status •ndern?
// Hier Timer beenden
window.clearTimeout(timer);
document.getElementById('j').innerHTML += 'readyGo';
}
}
// Request an URL senden
req.open('GET', 'status.php');
req.send(null);
```

```
45. }
46. //-->
47. </script>
48. </head>
49. <body onload="ajax()">
50. <div id="j">Loading...</div>
51. </body>
52. </html>
```

Alles anzeigen