

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 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

```
1. <html><head>
2. <title>AJAX: Timeout prüfen</title>
3. <script type="text/javascript">
4. <!--
5. // globale Variablen
6. var timeout = 2000; // in Millisekunden
7. var timer;
8. // AJAX Objekt initialisieren
10. try {
11. req = window.XMLHttpRequest?new XMLHttpRequest():
12. new ActiveXObject("Microsoft.XMLHTTP");
13. } catch (e) {
14. //Kein AJAX Support
15. }
16. /**
18. * zerstört den AJAX Request und schreibt eine Fehlermeldung
19. */
20. function timeoutCheck() {
21. req = null;
22. document.getElementById('j').innerHTML += 'error';
23. }
24. /**
26. * lädt Inhalt von status.php - inkl. Timeout Behandlung
27. */
28. function ajax() {
29. // Timer beginnen
30. timer = window.setTimeout("timeoutCheck()", timeout);
31. // Readystate behandlung
33. req.onreadystatechange = function() {
34. if ((req.readyState == 4) && (req.status == 200)) { // Hier Status ändern?
36. // Hier Timer beenden
37. window.clearTimeout(timer);
38. document.getElementById('j').innerHTML += 'readyGo';
39. }
40. }
41. // Request an URL senden
43. req.open('GET', 'status.php');
44. req.send(null);
```

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

Alles anzeigen