AJAX Ladegrafik

Welches Ziel soll erreicht werden?

Der Benutzer startet eine beliebige AJAX Aktion. Das können zum Beispiel sein: Formular absenden, Seite wechseln, ...
Die Ladegrafik soll dem Benutzer ein Feedback geben, dass noch etwas geschieht - dass wir für ihn arbeiten und sich das Warten lohnt.
Die Ladegrafik soll entsprechend dynamisch sein, damit der Benutzer nicht das Gefühl hat, dass die Seite langsam ist.
Wir sollten verhindern, dass der Benutzer den Link x-mal drückt und so unnötig Last erzeugt.
Außerdem muss die Grafik natürlich klein sein. Die Seite soll nicht schneller geladen werden als die Grafik

Inhaltsverzeichnis

Grafik erstellen

Als Vorbereitung müsst ihr euch eine Grafik beschaffen. Ihr könnt die Grafik entweder mit GIMP, Photoshop & Co selbst erstellen oder aber den kostenlosen Webservice von ajaxload.info nutzen.
easy-coding.de/Attachment/487/…ba9011f510d9c98ec10399143

AJAX Request bauen

Sobald ihr die Grafik habt und anfangen wollt zu programmieren, solltet ihr erst sicherstellen, dass ihr einen funktionierenden Prototypen habt. Erledigt niemals mehrere Schritt gleichzeitig, es sei denn ihr wisst was ihr tut.
Ich stelle euch hier einen einfachen Prototypen zur Verfügung.
Zum testen habe ich eine PHP Datei "demo-wait-for-3sec.php" mit künstlicher Wartezeit erstellt. Diese ruft die Funktion sleep(3) auf.

Quellcode

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" dir="ltr" xml:lang="de">
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<title>AJAX Loading Icon</title>
<script type="text/javascript"><!--
function sendRequest(linkref, target) {
var req;
try {
req = window.XMLHttpRequest?new XMLHttpRequest():
new ActiveXObject("Microsoft.XMLHTTP");
} catch (e) {
// no AJAX Support
}
req.onreadystatechange = function() {
if ((req.readyState == 4) && (req.status == 200)) {
document.getElementById(target).innerHTML = req.responseText;
}
}
req.open('post', 'demo-wait-for-3sec.php');
req.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
req.send('key=val');
return false; // return false to avoid reload/recentering of the page
}
//-->
</script>
</head>
```

```
30. <body>
31. <div id="targetDiv">
32. "Lorem ipsum dolor sit amet, consectetuer sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et
    dolore magna aliquyam erat, sed diam voluptua."
33. <p><a href="#" onclick="return sendRequest(this, 'targetDiv')">weiter</a></p>
34. </div>
35. </body></html>
```

Alles anzeigen

Bild in Ziel zeigen

Als nächstes wollen wir erreichen, dass die Ladegrafik im Zielcontainer eingehängt wird. Dazu erstellen wir eine Grafik im globalen JavaScript Scope. Von hier ist sie überall erreichbar.
Sobald der Link geklickt wird hängen wir das Bild in den Zielcontainer ein. Um den hässlichen grauen Rahmen um den Link zu verhindern, wenden wir außerdem die Funktion blur() auf diesem Objekt an.

Quellcode

```
1. var loadingImg = document.createElement('img');
2. loadingImg.src = 'ajaxload.gif';
3. function sendRequest(linkref, target) {
4. linkref.blur();
5. document.getElementById(target).appendChild(loadingImg);
6. var req;
7. try {
8. ...
9. }
10. }
```

Alles anzeigen

Ladegrafik wieder entfernen

Sobald die Inhalte fertig geladen wurden muss die Ladegrafik wieder entfernt werden. Das ist der einfachste Job dieses Tutorials, denn eigentlich müsst ihr nichts tun.
Da der Inhalt des Ziel Containers mit der Antwort des AJAX Requests überschrieben wird und die Ladegrafik zum Inhalt gehört hat, wird diese automatisch überschrieben,

Quellcode

```
1. req.onreadystatechange = function() {
2. if ((req.readyState == 4) && (req.status == 200)) {
3. document.getElementById(target).innerHTML = req.responseText;
4. }
5. }
```

Optimierung

Schauen wir uns die Ziele an, die erreicht werden sollten, können wir noch nicht mit uns zufrieden sein. Wir wollen dem Benutzer ein besseres Feedback geben und verhindern, dass er weitere Angaben innerhalb dieses Containers machen kann. Dazu machen wir aus dem Bild ein Blockelement, skalieren es über die volle Höhe des Zielcontainers und machen es halbtransparent, damit die alten Inhalte noch durchschimmern.

Quellcode

```
1. .loadingIcon {
2. display:block;
```

```
3. position:absolute;
4. left:0px;
5. top:0px;
6. width:100%;
7. height:100%;
8. opacity:0.75;
9. filter:alpha(opacity=75);
10. background-color:#6a7c87;
11. background-image:url(ajaxload.gif);
12. background-position:center center;
13. background-repeat:no-repeat;
14. }
```

Alles anzeigen

Damit das ganze funktioniert geben wir dem Zielcontainer noch die Eigenschaft **position:relative**.

Ergebnis

Der fertige Quellcode sieht nun folgendermaßen aus:

Quellcode

```
1. <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
2. <html xmlns="http://www.w3.org/1999/xhtml" dir="ltr" xml:lang="de">
3. <head>
4. <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
5. <title>AJAX Loading Icon</title>
6. <script type="text/javascript"><!--
7. /**
8. * @var loadingImg - precache the loading gif
9. */
10. var loadingImg = document.createElement('img');
11. loadingImg.className = 'loadingIcon';
12. /**
13. * sends a request under usage of a loading graphic - targetDiv has to be positioned relative
14. *
15. * @param linkref - reference to the link element (will be blured)
16. * @param target - reference to the target div, the content will be put in
17. */
18. function sendRequest(linkref, target) {
19. linkref.blur();
20. document.getElementById(target).appendChild(loadingImg);
21. var req;
22. try {
23. req = window.XMLHttpRequest?new XMLHttpRequest():
24. new ActiveXObject("Microsoft.XMLHTTP");
25. } catch (e) {
26. // no AJAX Support
27. }
28. req.onreadystatechange = function() {
29. if ((req.readyState == 4) && (req.status == 200)) {
30. // it's not necessary to delete the img, cause the content
31. // of the target div will be overwritten in the next line anyway
32. // document.getElementById(target).removeChild(loadingImg);
33. document.getElementById(target).innerHTML = req.responseText;
34. }
35. }
36. req.open('post', 'demo-wait-for-3sec.php');
```

```
41. req.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
42. req.send('key=val');
43. return false; // return false to avoid reload/recentering of the page
45. }
46. //-->
47. </script>
48. <style type="text/css">
49. <!--
50. .loadingIcon {
51. display:block;
52. position:absolute;
53. left:0px;
54. top:0px;
55. width:100%;
56. height:100%;
57. opacity:0.75;
58. filter:alpha(opacity=75);
59. background-color:#6a7c87;
60. background-image:url(ajaxload.gif);
61. background-position:center center;
62. background-repeat:no-repeat;
63. }
64. /* demostyle - not necessary to make it work*/
65. body{font-size:11pt;font-family:Verdana,Arial,Sans}
66. #targetDiv {width:300px;height:150px;border:1px solid #000;background-color:#efefef;padding:10px;}
67. //-->
68. </style>
69. </head>
70. <body>
72. <div id="targetDiv" style="position:relative;">
73. "Lorem ipsum dolor sit amet, consectetuer sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua."
74. <p><a href="#" onclick="return sendRequest(this, 'targetDiv')">weiter</a></p>
75. </div>
76. </body></html
```

Alles anzeigen

Demo

Eine Online-Demo findet ihr unter demo.easy-coding.de/ajax/ajax-loading-icon/. Den Download aller Dateien gibt es unter demo.easy-coding.de/ajax/ajax-loading-icon/download.zip
easy-coding.de/Attachment/488/…ba9011f510d9c98ec10399143