window.XMLHttpRequest Objekt des Internet Explorer (ab IE 7)

Refrenz auf ein Objekt,
anhand dessen der XML-Datentransfer per HTTP möglich wird.
das erst erzeugt werden muss
ab IE 7, erweitert ab IE 8

Die XML-Datenbeschaffung über HTTP per Serverzugriff erfolgt, um vom Server empfangene Daten als XML von der Webseite verarbeiten zu lassen, wobei XML-Daten in den HTML-Kontext per XML-DOM konvertiert werden müssen (Daten als XML-DOM-Objekt lieferbar, die dann von der Webseite per Script anhand XML-DOM und HTML-DOM verarbeitbar sind). Man kann zur Erzeugung von HTML Extensible Stylesheet Language Transformations (XSLT) benutzen. Monderne Variante als Laufzeitsystem ist Ajax, das auch auf Server z.B. von Microsoft mit ASP-Zugang aufsetzen kann (je nach Ajax-Anbieter).

Will man das Objekt lokal testen, kann man das nur, wenn lokal ein Server läuft z.B. Apache, da der Serverdienst benötigt wird.

Hinweis: Mit window.XDomainRequest kann ein sicherer Datenservice für jede Seite bei anonymen Zugriff auf jeden Server zum Zweck des XML-Datenaustausches zwischen Domains implementiert werden.

Eigenschaften:

onreadystatechange	Eventhandler-Refenrenz für asynchronen HTTP-Zugriff setzen bzw. liefern
readyState	aktueller Status des Objektes z.B. des Zugriffes
responseBody	Körper der Antwortdaten liefern als Feld unsigned Bytes)
responseText	Körper der Antwortdaten liefern als String
responseXML	Körper der Antwortdaten liefern als eine Refrenz auf ein Objekt des XML-DOM
status	HTTP-Status-Code des Zugriffes liefern
statusText	HTTP-Status-Text des Zugriffes liefern
timeout	Timeout-Wert setzen bzw. liefern, erst ab IE 8

Methoden:

abort()	aktuellen Zugriff abbrechen
getAllResponseHeaders()	komplette Liste der Antwort-Kopf liefern
getResponseHeader()	einen bestimmten Anwort-Kopf liefern
open()	Zugriff creieren und öffnen, aber nicht starten Zuweisungmethode Ziel-Url Attribute des Zugriffes
send()	per geöffneten Zugriff Daten an den Server Senden und dann auf Antwort vom Server warten
setRequestHeader()	Benutzdefinierten HTTP-Kopf für einen Zugriff setzen

Event:

ontimeout	Event erzeugt, wenn Zugriffbeenddigung einen Timeout-Wert zeitlich überschritten hat erst ab IE 8

XMLHttpRequest bis zum IE 6 musste per Active-X-Control erfolgen:

```
var XMLHttpRequestObjekt = new ActiveXObject("MSXML2.XMLHTTP.3.0");
```

Beispiel für den IE7:

```
if (window.XMLHttpRequest)
{
  var XMLHttpRequestObjekt = new XMLHttpRequest();
  XMLHttpRequestObjekt.open("GET", "http://localhost/test.xml");
  XMLHttpRequestObjekt.send();
  alert(XMLHttpRequestObjekt.statusText);
}
```

Attribute

Attribut onreadystatechange

Eventhandler für asynchronen Zugriff zuweisen bzw. Referenz liefern

Syntax:

```
[ vHandler = ] XMLHttpRequestObjekt.onreadystatechange [ = vHandler ]

vHandler   Referenz auf Handler

lesen und schreiben
kein Standardwert
```

Beispiel:

```
function reportStatus()
```


```
{
   if (XMLHttpRequestObjekt.readyState == 4)
      alert('Transfer complete.');
}

var XMLHttpRequestObjekt = new XMLHttpRequest();
XMLHttpRequestObjekt.onreadystatechange = reportStatus;
XMLHttpRequestObjekt.open("GET", "http://localhost/test.xml", true);
XMLHttpRequestObjekt.send();
```

Attribut readyState

aktuellen Status des Zugriffes ermitteln
responseText und responseBody nur auswertbar, wenn Status 4 anliegt (ansonsten erzeugt ein Zugriff auf die beiden Eigenschaften eventuell einen Fehler)

Syntax:

[nState =] XMLHttpRequestObjekt.readyState

nState Integer
0 Objekt erzeugt und nicht initialisiert (open() bisher nicht aktiviert worden)
1 Objekt ist offen und bisher kein send() aktiviert worden
2 Objekt sendet (send() wurde aktiviert)
3 Objekt empfängt da Daten eingehen
4 Objekt hat empfangen beendet und alle Daten sind eingetroffen
responseText ist auswertbar
responseBody ist auswertbar

nur lesen
hat keinen Standardwert

Attribut responseBody

liefert den Antwort-Body als ein Feld aus unsigned Bytes

Syntax:

[vBody =] XMLHttpRequestObjekt.responseBody

vBody Feld aus unsigned Bytes

nur lesen
kein Standardwert

Attribut responseText

liefert den Antwort-Body als String

Syntax:

[sBody =] XMLHttpRequestObjekt.responseText

sBody String des Antwort-Body

nur lesen
kein Standardwert

Attribut responseXML

liefert den Anwortbody als XML-DOM-Objekt
Es können sämtliche Methoden zum Objekt laut XML-DOM aktiviert werden, da XML-DOM und HTML-DOM parallel nutzbar sind per Script.

Syntax:

[oXMLDocument =] XMLHttpRequestObjekt.responseXML

oXMLDocument Referenz aufXML-DOM-Objekt

nur lesen
kein Standardwert

Beispiel:

```
var XMLHttpRequestObjekt = new XMLHttpRequest();
XMLHttpRequestObjekt.open("GET", "http://localhost/books.xml", false);
XMLHttpRequestObjekt.send();
alert(XMLHttpRequestObjekt.responseXML.xml);
```

Attribut status

liefert den HTTP-Statuscode des Zugriffes

Syntax:

[nStatus =] XMLHttpRequestObjekt.status

nStatus Integer

100	Continue
101	Switching protocols
200	OK
201	Created
202	Accepted
203	Non-Authoritative Information
204	No Content
205	Reset Content
206	Partial Content
300	Multiple Choices
301	Moved Permanently
302	Found
303	See Other
304	Not Modified
305	Use Proxy
307	Temporary Redirect
400	Bad Request
401	Unauthorized
402	Payment Required
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable
407	Proxy Authentication Required
408	Request Timeout
409	Conflict
410	Gone
411	Length Required
412	Precondition Failed
413	Request Entity Too Large
414	Request-URI Too Long
415	Unsupported Media Type
416	Requested Range Not Suitable
417	Expectation Failed
500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Timeout
505	HTTP Version Not Supported

nur lesen
kein Standardwert

Beispiel:

```
if (XMLHttpRequestObjekt.status == 401)
   alert('Access denied.');
else
   alert(XMLHttpRequestObjekt.responseText);
```

Attribut statusText

liefert HTTP-Status des Zugriffes als Text

Syntax:

[sStatus =] XMLHttpRequestObjekt.statusText

sStatus String

nur lesen
kein Standardwert

Beispiel:

```
XMLHttpRequestObjekt.open("GET", "http://localhost/test.xml", false);
XMLHttpRequestObjekt.send()
if (XMLHttpRequestObjekt.statusText == "OK")
  alert(XMLHttpRequestObjekt.responseText);
else
  alert(XMLHttpRequestObjekt.statusText);
```


Attribut timeout erst ab Internet Explorer 8

setzen oder liefern des Timeout-Wertes als Wartezeit des Browsers auf Antwort des Servers.
Wird innerhalb der Wartetzeit keine Antwort an den Browser geliefert, dann wird responseText auf null-Zeiger gesetzt.
Wert nur setzbar nach open() und vor send()

Syntax:

[v =] XMLHttpRequestObjekt.timeout [= v]

v Integer
Miilisekunden als Wartezeit des Browser auf Antwort des Servers
Standard ist 0
Wert nur setzbar nach open() und vor send()

lesen und schreiben

Methoden

Methode abort()

aktuellen Zugriff abbrechen
aktuellen onreadystatechange-Event-Handler dekativieren und entfernen
readyState auf 0 setzen (Objekt ist uninitialisiert).

Syntax: XMLHttpRequestObjekt.abort();

liefert nichts

Methode getAllResponseHeaders()

liefert die komplette Liste der Antwort-Header
Liste ist Text
Jedes Paar aus Name und Wert ist begrenzt von CRLF (Return mit Linefeed)

Syntax:

sHeaders = XMLHttpRequestObjekt.getAllResponseHeaders();

liefert String

Methode getResponseHeader()

liefert einen bestimmten Antwort-Header

Syntax:

p = XMLHttpRequestObjekt.getResponseHeader(bstrHeader)

bstrHeader String
muss kodiert werden
Name des Antwort-Headers

liefert String

Methode open()

Request öffnen, ermöglicht danach send()
Methode des Zugriffes zuweisen
gemischte Protokolle sind nicht erlaubt
Protokollwechsel nicht erlaubt
Quelle und Ziel des Datentransportes müssen identisches Protokoll benutzen
Ziel-Url des Zugriffes zuweisen
Domainbereich verlassen ist nicht erlaubt
nur Dateien innerhalb derselben Domaine
Quelle und Ziel des Datentransportes müssen identische Domain benutzen
Eigenschaften des Zugriffes zuweisen

Daten werden im Cache (Temporary Internet Files) des Internet Explorers
gepuffert bei Operation "GET"
nicht gepuffert bei Operation "POST"

Userkennung und Password (beide optional) werden erst dann übertragen, wenn die Verbindung bereits aktiv ist
Erst nach open() ist Senden möglich, da die Verbindung dann steht.

Syntax:

XMLHttpRequestObjekt.open(sMethod, sUrl [, bAsync] [, sUser] [, sPassword])

sMethod muss kodiert werden

HTTP-Methode zum Öffnern der HTTP-Connection
Gros-kleinschreibung egal
Quelle und Ziel des Datentransportes müssen identisches Protokoll benutzen

"GET"	Request URI
"POST"	Send data to server
"HEAD"	Request URI without body
"PUT"	Store data for URI
"DELETE"	Delete data for URI
"MOVE"	Move URI to to new location
"PROPFIND"	Request URI Properties
"PROPPATCH"	Update or Delete URI Properties
"MKCOL"	Create collection at URI
"COPY"	Create copy of URI
"LOCK"	Create Lock
"UNLOCK"	Remove Lock
"OPTIONS HTTP"	Request URI Options

sUrl muss kodiert werden
absolute oder relative URL der XML daten
des XML Web Services auf dem Server
Quelle und Ziel des Datentransportes müssen identische Domain benutzen

bAsync optional
boolean
true asynchroner Zugriff
Es muss eine Handler per onreadystatechange eingebunden sein, der feststellt, wann der Zugriff komplett ist (readyState benutzen)
Standardwert
false synchroner Zugriff
Browser wartet direkt auf Ende des Zugriffes, so dass der Browser weder Eingaben akzeptiert noch Ausgaben produziert.
Die HTTP-Verbindung sollte also zuverlässig und schnell sein.

sUser optional
String
Name des Users, der
den Zugriff auslöst
sich einloggen muss
wenn Name kodiert so einloggen mit Zugriff automatisch
wenn keine Name oder Leerkette "" kodiert so Login-Fenster angezeigt.

sPassword
optional
String
Password für Autentifizierung
wenn nicht kodiert oder Leerkette "" so kein Password z.B. per Loginfenster verlangt

liefert nichts

Methode send ()

Einen HTTP-Zugriff an den Server senden und auf Antwort des Servers warten.
Der Zugriff erfolgt je nach open() synchron oder ansynchron (Art des Wartens auf Anwort, siehe open())

Syntax:

XMLHttpRequestObjekt.send([varBody])

varBody optional
Körper der Meldung, die mit dem Zugriff gesendet wird
String oder Feld aus unsigned Bytes oder Zeiger auf XML-DOM-Objekt

liefert nichts

Methode setRequestHeader()

einen benutzerdefinierten Header dem Zugriff hinzufügen (für nächstes send() bereitstellen)

Syntax:

zeiger_auf_objekt.setRequestHeader(sName, sValue)

sName muss kodiert werden, 1. Teil des Paares
String
Name des Headers

sValue muss kodiert werden, 2. Teil des Paares
String

Wert des Headers

liefert nichts

Beispiel: HTTP Content-Type Header auf 'text/xml' setzen, bevor der Sendezugriff erfolgt (request body)

```
var XMLHttpRequestObjekt = new XMLHttpRequest();
XMLHttpRequestObjekt.open("POST", sURL, false);
XMLHttpRequestObjekt.setRequestHeader("Content-Type", "text/xml");
XMLHttpRequestObjekt.send(sRequestBody);
```

Events

Event ontimeout

Event ausgelöst, wenn
- ein Fehler im Zugriffverlauf erfolgt ist
- timeout-Periode endete bevor ein onload-Event ausgelöst werden konnte

Wenn Event ausgeöst, so ist responseText nicht verfügbar (Zugriff auf responseText erzeugt Fehler).

Syntax:

<SCRIPT FOR = object EVENT = ontimeout>

Event Information

Bubbles: No
Cancels: No

Beispiel:

```
<script type="text/javascript">
function timeo()
{
   alert("XDR ontimeout");
}
...
xdr.ontimeout = timeo;
```