# AJAX und Sessions

## Kurzer Test - Erfolg?

Wir testen das ganze indem wir eine SESSION Variable von der Hauptdatei an den AJAX Call übergeben werden. Dazu benutzen wir die bekannte ajaxPost Funktion.

### Quellcode

```
function ajaxPost(url, postData, callback) {
var req;
try {
req = window.XMLHttpRequest ? new XMLHttpRequest(): new ActiveXObject("Microsoft.XMLHTTP");
} catch (e) {
// browser does not have ajax support
}
req.onreadystatechange = function() {
if (req.readyState == 4 && req.status == 200) {
callback = document.getElementById(callback);
callback.innerHTML = req.responseText;
}
};
req.open('POST', url, true);
req.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
req.send(postData);
return false;
}
```

Alles anzeigen

... und wir erstellen das Beispiel: Statische und dynamische Inhalte sollten beim Aufruf den selben Inhalt zeigen.

### Quellcode

```
<?php
session_start();
$_SESSION['url'] = 'http://www.easy-coding.de';
?><!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" dir="ltr" xml:lang="de">
<head>
<title>Session Test</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<script type="text/javascript" src="ajax.js"></script>
</head>
<body onload="ajaxPost('session.php','','ajaxResponse')">
<h3>Static</h3>
<?php print_r($_SESSION); ?>
<h3>Dynamic</h3>
<div id="ajaxResponse"></div>
</body>
</html>
```

Alles anzeigen

Die aufgerufene session.php startet einfach nur eine Session und gibt den Inhalt der $_SESSION Superglobalen zurück

### Quellcode

```
<?php
```

## Inhaltsverzeichnis

```
2. session_start();
3. print_r($_SESSION);
4. ?>
```

## Deaktivierte Cookies?

Werden Cookies vom Browser abgelehnt erschwert sich die Abfrage. Um das Verhalten zu testen wird der Firefox Browser so konfiguriert, dass er keine Cookies annimmt.
easy-coding.de/Attachment/505/

Rufen wir das Beispiel nun auf sehen wir, dass das dynamische Array keine Daten liefert.
Die einzige Möglichkeit die Session über mehrere GET Requests aufrecht zu erhalten ist es die SESSION ID per URL zu übergeben.

Dazu nutzen wir die Konstante "SID", um den den aktuellen Namen und die Session-ID als Zeichenkette passend zum Anhängen an URLs zu erhalten. Sind Sessions im Browser akzeptiert ist die Konstante leer und wir müssen nichts manipulieren.

Weil AJAX über JavaScript funktioniert speichern wir uns diesen String in einer globalen JavaScript Variablen.

### Quellcode

```
1. <script type="text/javascript"> //<![CDATA[
2. var SID = '<?php echo SID?>';
3. //]]>
4. </script>
```

In meinem Beispiel sieht das Ergebnis so aus:

### Quellcode

```
1. <script type="text/javascript"> //<![CDATA[
2. var SID = 'PHPSESSID=337fb9a607abacaf86f7dba4f1b003fa';
3. //]]>
4. </script>
```

Diesen String müssen wir nun innerhalb der ajaxPost Funktion an die URL anhängen. Um möglichst flexibel zu bleiben fragen wir die URL ab und prüfen ob schon andere GET Parameter gesetzt wurden. In dem Fall hängen wir den String mit einem "&" hinten an. Andernfalls müssen wir die Session ID als ersten GET Parameter mit einem Fragezeichen kennzeichnen.

### Quellcode

```
1. if(typeof SID != 'undefined') url += (url.indexOf('?')>0 ? '&' : '?') + SID;
2. req.open('POST', url, true);
```

## Demo

Die fertigen Dateien finden Sie unter demo.easy-coding.de/ajax/sessions/download.zip.
Die beiden Beispiele können Sie online nachvollziehen. Einfach ohne Korrektur und weiter mit Korrektur.