**Lektion 12: Sessions (Sitzungen)**

Wenn Sie eine Webseite besuchen, machen sie viele verschiedene Sachen. Sie surfen von einer Seite zur anderen. Vielleicht füllen Sie auch ein Formular aus oder kaufen irgendetwas ein.

Für Webentwickler sind solche Informationen sehr wichtig, um erfolgreiche Webanwendungen zu erstellen.

Nehmen wir z.B. an, dass Sie ein Projekt verwirklichen wollen, bei dem einige Seiten mit einem Benutzernamen und Passwort geschützt sind. Um diesen Schutz effektiv umzusetzen, sollten die geschützten Seiten Zugang zu der Information haben, ob sich der Nutzer bereits angemeldet hat. Sie müssen sich im Prinzip also "erinnern" können, was der Nutzer früher gemacht hat.

Dies ist genau das, worum es hier in dieser Lektion geht - wie Sie Sessions in PHP dazu benutzen können, Informationen während des Besuches eines Benutzers zu speichern und abzufragen.

**Session (Sitzung)**

Die PHP-[session](http://www.php.net/manual/en/book.session.php) erlaubt es Ihnen Informationen über eine bzw. während einer Besuchsinstanz eines Nutzers zu verwalten. Sie können 'kluge' Applikationen schreiben, die Benutzer erkennen und Informationen über sie sammeln können.

Eine Session kann auf drei verschiedene Arten beginnen. Wir werden und nicht mit technischen Details aufhalten, uns aber auf den Fall konzentrieren, wobei eine Session damit beginnt, dass ein Wert gespeichert wird. Eine Session endet (engl. Fachbegriff: "dies" - stirbt), wenn ein Benutzer innerhalb eines bestimmten Zeitfensters (standardmäßig 20 Minuten) keine Seiten mehr aufgerufen hat. Natürlich können Sie jederzeit Ihre Session in Ihrem Skript beenden (engl. Fachbegriff: "kill" - töten).

Nehmen wir an, 50 Leute klicken zur selben Zeit auf der gleichen Webseite herum, z.B. in einem Onlineshop. Die Angaben, was jeder von ihnen in seinem Einkaufskorb hat werden am besten in einer Session gespeichert. Um die einzelnen Nutzer identifizieren zu können, verwendet der Server eine individuelle Benutzer-ID (User-ID), die in einem sog. Cookie gespeichert wird. Ein Cookie ist eine kleine Textdatei, die auf dem Computer des Benutzers gespeichert wird. (Mehr über Cookies in [Lektion 13](http://de.html.net/tutorials/php/lesson13.php)). Aus diesem Grunde verlangen Session oft die Zulässigkeit von Cookies im Browser des Benutzers.

**Ein Beispiel zur Nutzung von Sessions**

Als Sie diese Seite aufgerufen haben, haben wir die aktuelle Zeit in einer Session gespeichert. Dies haben wir gemacht, um Ihnen an einem Beispiel zeigen zu können, wie eine Session funktioniert.

Dieser Eintrag ist mit "StartTime" (also Startzeit) benannt und wurde durch die folgenden Zeile, die unserem Code hinzugefügt wurde, gespeichert:

 <?php

 session\_start();

 $\_SESSION["StartTime"] = date("r");

 ?>

Dabei wurde eine Session gestartet. Wie oben beschrieben, bekommt jede Session vom Server eine ID.

Ihre Session hat die folgende ID: **5f3cd0a18b06b2c9d3e4d39940a27403**

Jederzeit kann die "StartTime" aus der Session, mit den folgenden Zeilen, abgerufen werden:

 <?php

 session\_start();

 echo $\_SESSION["StartTime"];

 ?>

Dies enthüllt, dass die Seite zu diesem Zeitpunkt - Sat, 12 Jan 2013 14:58:01 +0100 - aufgerufen wurde. (zu Grunde liegt hier die Zeiteinstellung unseres Webservers).

Das Interessante aber ist, dass die Zeit auch in der Session gespeichert bleibt, auch nachdem Sie diese Seite verlassen haben. Diese Information wird Sie 'verfolgen', bis Ihre Session endet.

Standardmäßig dauert eine Session solange, bis der Benutzer den Browser schließt, dann endet sie automatisch. Wenn Sie aber eine Session beenden möchten, kann dies immer mit dem folgenden Befehl geschehen:

 <?php

 session\_destroy();

 ?>

Lassen Sie uns noch ein weiteres Beispiel betrachten, bei dem mit Sessions gearbeitet wird: eine Passwort-Lösung.

**Login-System mit Sessions**

Im folgenden Beispiel, werden wir ein sehr einfaches Login-System erstellen. Wir werden viele Sachen anwenden können, die wir in vergangenen Lektionen gelernt haben.

Zuerst brauchen wir ein Formular, in das man Benutzernamen und Passwort eingeben kann. Es könnte wie folgt aussehen:

 <html>

 <head>

 <title>Login</title>

 </head>

 <body>

 <form method="post" action="login.php">

 <p>Benutzername: <input type="text" name="username" /></p>

 <p>Passwort: <input type="text" name="password" /></p>

 <p><input type="submit" value="Lass mich ein!" /></p>

 </form>

 </body>

 </html>

Dann erstellen wir die Datei: login.php.

In dieser Datei überprüfen wir, ob eine korrekte Kombination aus Benutzername und Passwort eingegeben wurde. Wenn dies der Fall ist, erstellen wir einen Eintrag in der Session der besagt, dass der Benutzer sich ordnungsgemäß angemeldet hat.

 <html>

 <head>

 <title>Login</title>

 </head>

 <body>

 <?php

 // Überprüfe, ob Benutzername und Passwort korrekt sind

 if ($\_POST["username"] == "php" && $\_POST["password"] == "php") {

 // Wenn korrekt, setze Sessioneintrag auf YES

 session\_start();

 $\_SESSION["Login"] = "YES";

 echo "<h1>Sie sind jetzt ordnungsgemäß eingeloggt!</h1>";

 echo "<p><a href='document.php'>Link zur geschützten Datei</a><p/>";

 }

 else {

 // Wenn nicht korrekt, setze Sessioneintrag auf NO

 session\_start();

 $\_SESSION["Login"] = "NO";

 echo "<h1>Sie sind NICHT ordnungsgemäß eingeloggt!</h1>";

 echo "<p><a href='document.php'>Link zur geschützten Datei</a><p/>";

 }

 ?>

 </body>

 </html>

In der geschützten Datei müssen wir überprüfen, ob der Benutzer ordentlich eingeloggt ist. Wenn dies nicht der Fall ist, dann wird er zurück zum Login-Formular geleitet. Und so wird diese Sicherung programmiert:

 <?php

 // Startet die PHP-Session

 session\_start();

 // Wenn der Benutzer nicht eingeloggt ist, Weiterleitung zum Login-Formular

 if ($\_SESSION["Login"] != "YES") {

 header("Location: form.php");

 }

 ?>

 <html>

 <head>

 <title>Login</title>

 </head>

 <body>

 <h1>Dieses Dokument ist geschützt!</h1>

 <p>Sie können es nur sehen, weil Sie eingeloggt sind.</p>

 </body>

 </html>

[Klicken Sie hier, um das Login-System zu testen](http://de.html.net/tutorials/php/lesson12_ex1.php)

Jetzt haben Sie Sessions kennengelernt. In der nächsten Lektion werden wir weiter auf diesem Gebiet arbeiten, aber wir werden uns Cookies genauer anschauen.