Sessions mit PHP

Annabell Langs 2004

Sessions » Inhaltsverzeichnis

Sessions » Wozu Sessions? (1)

- Das HTT-Protokoll ist zustandslos (stateless) - Variablen und Daten sind nur pro Seite gültig
  - Request: Anfrage des Browser an den Webserver
  - Response: Antwort des Webservers auf die Anfrage

- Zustandslosigkeit ein Problem für komplexe Webseiten:
- Authentifizierung: Seitenübergreifende Benutzeridentifizierung
  - Onlineshops (Bsp.: Warenkorb) etc.
- Authorisierung: Überprüfung individueller Zugangsdaten eines wiedererkannten Benutzers und Gewährleistung damit verbundener Rechte
  - Administrationsbereiche etc.

- Keine praktikable Lösung wäre die Identifikation mittels IP (Proxies...)

- Praktikable Lösung: Anfragen eines Benutzers werden zu einer Session = Sitzung zusammengefasst
- Jede Session-ID steht für einen Sitzung und somit für einen Benutzer

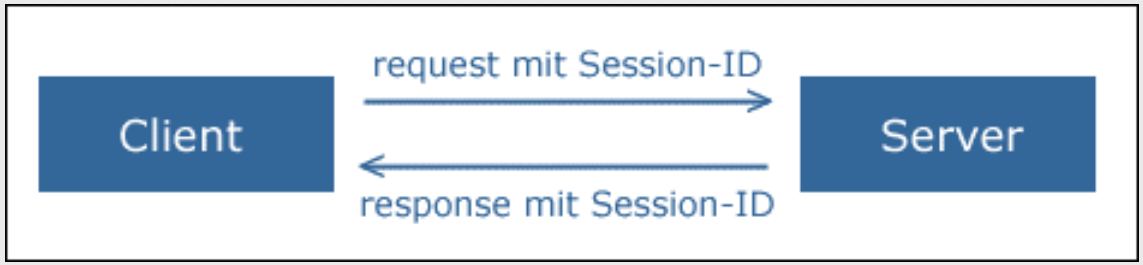
Sessions » Wozu Sessions? (2)

- Sessions gibt es seit PHP 4.0

- Session-ID = zufällige vom Server generierte (eindeutige) Zeichenfolge, die an den Browser geschickt wird, der sie dann speichert (→ Wie kann ich die Session-ID übergeben?)
  - Beispiel: 09fc391c2de9c00bb18ed1f26cd493f9 (steht z.B. nach PHPSESSID=)

- Bleibt für die Länge der Sitzung auf einer Webseite gültig

- Diese Session-ID wird nun bei jeder Kommunikation zwischen Server und Client zur Identifikation mitgeschickt



Sessions » Wie funktionieren Sessions? (1)

- Mit session_start() wird eine Session gestartet oder fortgesetzt
- Es darf keine andere Ausgabe vor diesem Befehl stehen
- Eine Sessiondatei wird auf dem Webserver angelegt. Diese gehört zur jeweiligen Session-ID dazu (→ Wie funktionieren Sessions? (3))

  BenutzerXY startet eine Session

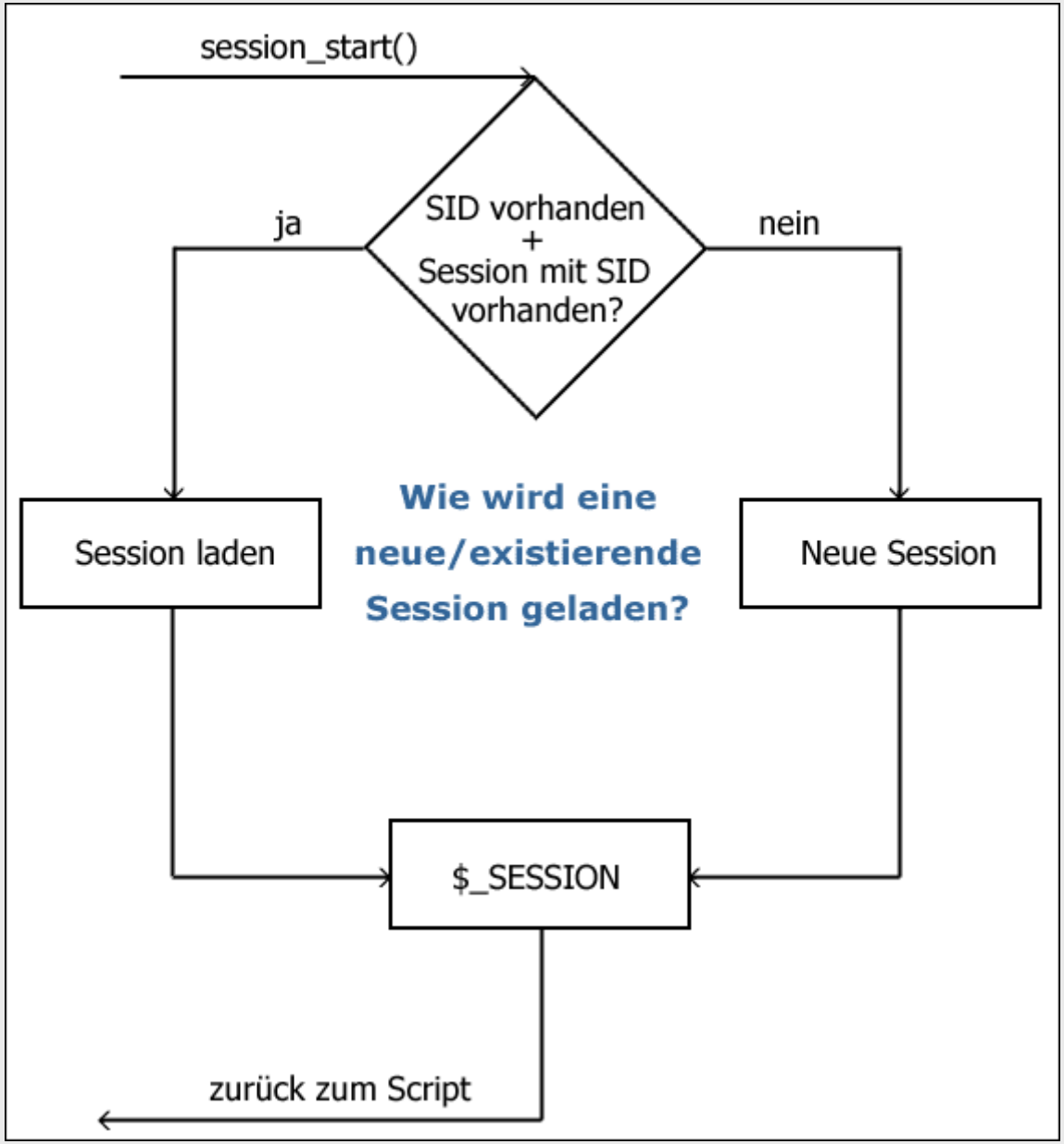
  Ihm wird die Session-ID 12345 zugewiesen

  Datei auf dem Server mit Namen sess_12345 wird erstellt

  (Diese Datei „gehört“ nun BenutzerXY – solange dieser Benutzer seine Session-ID „kennt“, hat er Zugriff auf Daten, die in seiner Sessiondatei gespeichert sind)

- Daten können nun in dieser Sessiondatei gespeichert werden und mit gleicher Session-ID abgerufen werden
- In PHP ab Version 4.1 stehen einem die Daten im assoziativen Array $_SESSION zur Verfügung

Sessions » Wie funktionieren Sessions? (2)



Sessions » Wie funktionieren Sessions? (3)

- Standardmäßig werden die Sessiondateien im Verzeichnis tmp gespeichert

- Nach start_session() wird eine Datei mit den Zugriffsrechten des Servers erstellt. Zu Anfang ist diese 0 Byte groß.
  - -rw------- 1 wwwrun www 0 Apr 10 18:00 sess_09fc391c2de9c00bb18...
- Werden Variablen deklariert, ändert sich die Größe dieser Sessiondateien
- Die Sessiondateien haben keine Größenbeschränkungen wie etwa die Länge einer URL o.Ä.
- Mittels session_destroy() sollte man die Sessiondateien am Ende einer Sitzung löschen

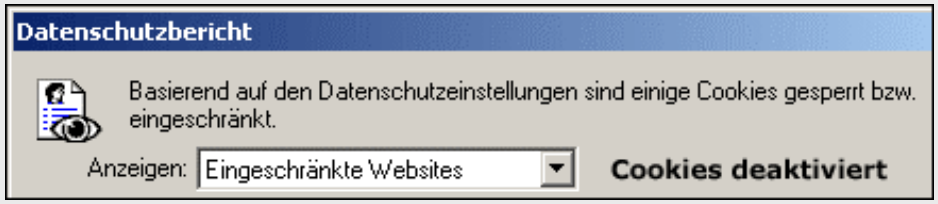
Sessions » Wie kann ich die Session-ID übergeben?

- Es gibt drei Arten, die Session-ID bei dem jeweiligen Benutzer zu speichern/weiterzugeben:
  - Per temporärem Session-Cookie (Standard)
  - Per $_GET in der URL
    forum/viewtopic.php?t=250&start=0&postdays=0&postorder=asc&highlight=&sid=6568ed7fd94f05c120b1fa0822c12b63
  - Per $_POST über Formulare (unsichtbare Felder mit Session-ID als vordefinierter Wert)
- Für die $_GET Methode muss „session.use_cookies“ in der php.ini auf 0 gesetzt werden
- Variante mit Cookies ist sicherer als per URL-Weitergabe (Session-ID im Klartext)

Sessions » Sicherheit

- Wird die URL inklusive Session-ID (SID) übergeben, so erlangt man Zugriff auf die Daten

- Cookies sind sicherer, aber nicht jeder hat „Akzeptieren von Cookies“ im Browser aktiviert (Fallback…)



Sessions » Überblick: Wichtige Funktionen (1)

→ Basierend auf PHP 4.1.0 werden die Funktionen session_register(), session_unregister und session_is_registered() nicht benötigt

- Session_start()
  - Erzeugt eine Session oder setzt eine bisherige Session fort
  - Funktion gibt immer true zurück
- Unset($_SESSION['Variable'])
  - Aufheben der Registrierung einer Variablen mit $_SESSION und deaktiviertem register_globals
- Session_destroy()
  - Löscht alle in Verbindung mit der aktuellen Session-ID bestehenden Daten (nicht aber mit der Session zusammenhängende globale Variablen und Sessioncookies)
  - Funktion gibt true oder false (bei Fehler) zurück

Sessions » Überblick: Wichtige Funktionen (2)

- Session_name([string])
  - Liefert und/oder setzt den Namen der aktuellen Session
  - Session_name muss bei jeder Anfrage aufgerufen werden und muss vor session_start stehen
  - Standardname: phpsessid

- Session_ID([string ID])
  - Liefert und/oder setzt die aktuelle Session-ID
  - Wird eine ID angegeben, wird die vorhandene durch diese ersetzt (zu diesem Zweck muss session_ID vor session_start aufgerufen werden)

Sessions » Beispiel für die Nutzung von Sessions (1)

→ Anwendung von Sessions am Beispiel eines Logins

- Login Formular:

```
<form action="logincheck.php" method="post">
Passwort:
<input name="passwort" size="10" type="password"><br>
<input name="login" type="submit" value="login">
</form>
```

Passwort mittels POST in $_POST['passwort'] speichern

- Logincheck:

```
<?php
session_start();

//dummy PW
$kpw = "1234";
$passwort = $_POST['passwort'];
$login = $_POST['login'];

if($login && $passwort==$kpw)
{
  $_SESSION['passwort']=$passwort;
}

?>
```

Zunächst wird eine neue Session gestartet

Wenn das Formular abgeschickt wurde und das eingegebene Passwort mit dem Zugangspasswort übereinstimmt, wird dieses in der Sessionvariablen $_SESSION['passwort'] gespeichert

Sessions » Beispiel für die Nutzung von Sessions (2)

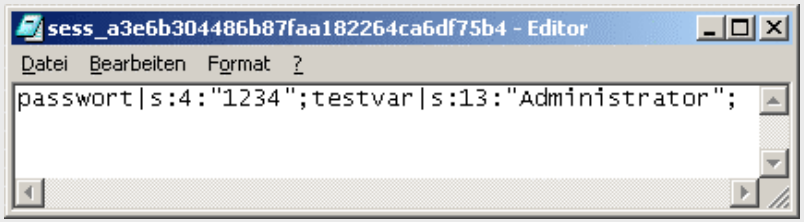
- Logincheck:

```php
<?php

if($_SESSION['passwort']==$kpw) {
  echo "Erfolgreich eingeloggt!<br>Hier geht es weiter
        zu einer <a href=\"loggedin.php\">geschützten
        Beispielseite</a>.";

  //Beispiel
  $_SESSION['testvar']="Administrator";
  echo "<br><br>Variablen kann man nun ganz einfach
        weitergeben, wie zB \"Administrator\" in der
        \"SESSION['testvar']\"";
}
else {
  echo "Hier kommst du nit rein! Gib das richtige
        Passwort ein!";
}

?>
```

Wenn das PW übereinstimmt, ist man eingeloggt.

Beispiel: „Administrator“ in $_SESSION['testvar'] speichern

Falsches Passwort eingegeben, hier geht es nicht weiter



- Logged-in (Beispielseite):

```php
<?php
session_start();
?>
```

Session wird fortgesetzt

(steht vor jeder anderen Ausgabe)

Sessions » Beispiel für die Nutzung von Sessions (3)

- Logged-in (Beispielseite):

```php
<?php

if (isset($_SESSION['passwort'])) {
  echo "Es klappt! Das Passwort war
         ".$_SESSION['passwort']."<br>";
  echo "Die SESSION ID ".session_id()." und somit
         alle gespeicherten Daten der  Sitzung sind
         noch vorhanden";

  //Beispielübergabe
  echo "<br><br>Auch der Inhalt unserer 'testvar'
         bleibt erhalten! -> ".$_SESSION['testvar'];

  //Logout
  echo "<br><br><a href=\"logout.php\">Hier
         klicken</a> f&uuml;r den Logout um die
         Session zu l&ouml;schen";
}

else { //wenn nicht vorher eingeloggt
  echo "Hier kommst du nit rein!<br>Bitte logge dich
         vorher <a href=\"login.php\">hier</a>
         ein.";
}

?>
```

Ob man eingeloggt ist, kann man überprüfen, indem man kontrolliert, ob die Sessionvariable gesetzt (isset) ist

Der Zugriff auf übergebene Variablen geschieht über $_SESSION['varname']

Sessions » Beispiel für die Nutzung von Sessions (4)

- Logout:

```php
<?php
   session_start();
   echo "Die Session ID war ".session_id();
   session_destroy();
   echo "<br><br>Logout erfolgreich!";
?>
```

Mit session_destroy kann eine Session gelöscht werden (Session Datei auf dem Server wird gelöscht)

→ Wichtige Erkenntnisse:

- In diesem Beispiel wird die Session-ID in einem Cookies (Standard) gespeichert.
- Mittels session_destroy wird die Session-Datei auf dem Server gelöscht und sollte daher bei einem Logout immer angewandt werden
- Session_start startet nicht nur eine neue Sitzung, sondern setzt auch eine vorhandene fort

Sessions » Häufiger Fehler und Schlusswort

- Warning: Cannot send session cookie - headers already sent...
  - Eine Session muss vor der Ausgabe jeglicher Zeichen gestartet werden, also am besten als einer der allerersten Befehle eines Scriptes.

Weitere Fragen? Jetzt könnt ihr fragen ;-)

Quellen:

- PHP Manual
- http://www.weigl.de/seminar/php4/sessions.htm
- http://www.webmaster-resource.de/tricks/php/login-mit-sessions.php
- http://www.free2code.net/tutorials/programming/php/4/sessions.php
- http://tut.php-q.net/sessions.html
- http://www.netz-id.de/article936.html
- [...]