



# Grundlagen Internet-Technologien

INF3171

## Cookies & Sessions

Version 1.0

10.07.2025





# Aktuelles

Aktuell kursieren Phishing-EMails, in denen vorgegaukelt wird, dass eine wichtige Datei über bwSync&Share zur Verfügung steht.

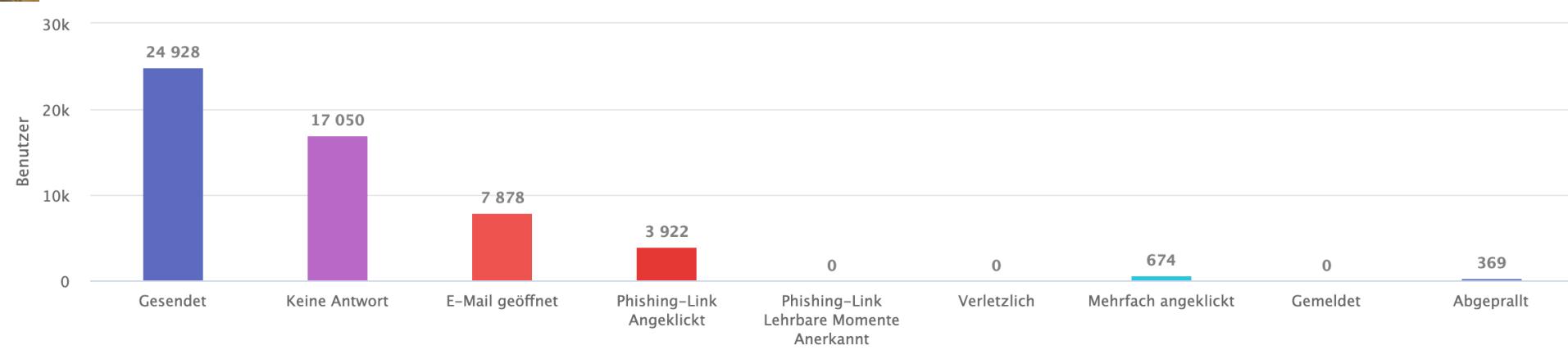
Hinter diesem Angriff stecken Cyberkriminelle, die die Domain „bwsyncandshare.de“ registriert haben. Die echte Adresse lautet bekanntlich „bwsyncandshare.kit.edu“.

Ich empfehle daher, die Domain „bwsyncandshare.de“ an jedem Standort zu sperren.

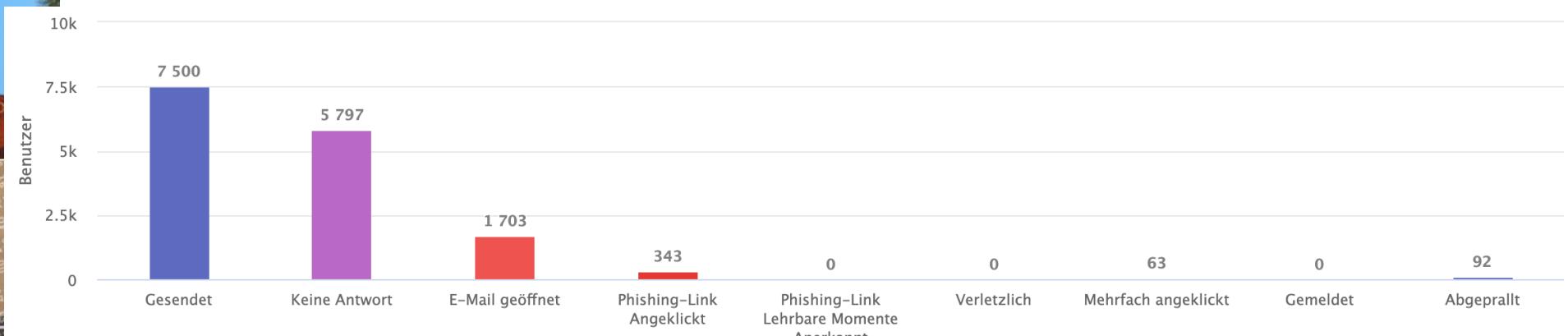


## Ergebnisse: Kampagne März 2025

Studierende: **49,8% Klickquote ("April, April")**



Beschäftigte: **20,1% Klickquote (Vgl. zu Januar: 24,4%, ↘ 4,3% verbessert)**





# Klausur

- Klausur  
Do 24.07.2025  
08:00 - 10:00  
Hörsaal N06 (Hörsaalzentrum Morgenstelle)
- Nachklausur  
Mi 08.10.2025  
16:00 - 18:00  
Hörsaal 21 (Kupferbau)



# Tour de France 2025 (09.07.)

FAHRER		ZEIT
1	 <b>T. POGACAR</b> UAE TEAM EMIRATES - XRG	17:22:58
2	 <b>R. EVENEPOEL</b> SOUDAL QUICK-STEP	+42
3	 <b>K. VAUQUELIN</b> ARKÉA - B&B HOTELS	+59
4	 <b>J. VINGEGAARD</b> TEAM VISMA-LEASE A BIKE	+1:13
5	 <b>M. JORGENSEN</b> TEAM VISMA-LEASE A BIKE	+1:22
6	 <b>M. VAN DER POEL</b> ALPECIN - DECEUNINCK	+1:28
7	 <b>J. ALMEIDA</b> UAE TEAM EMIRATES - XRG	+1:53
8	 <b>P. ROGLIC</b> RED BULL-BORA-HANSGROHE	+2:30
9	 <b>F. LIPOWITZ</b> RED BULL-BORA-HANSGROHE	+2:31



# Individualisierung im Web

- die klassische Web-Site ist **zustandslos**
- wesentliche Eigenschaft
  - die Website kennt **keine Vorgeschichte**
- ...für viele Anwendungen ist dies *nicht sinnvoll*
  - kein Anmelden, kein Warenkorb, ...



# Business-Applications

- Business-Applications brauchen ganz typisch einen *Zustand* wie
  - angemeldet sein
  - Rolle und Rechte
  - Warenkorb
- trivial-Ansatz:  
**hidden fields** im HTML-Formular
  - keine weitreichende Lösung
  - GET: in Browser History einsehbar



# hidden fields (I)

- HTML-Formulare können über **versteckte Felder** verfügen

`<INPUT type="hidden">`

- nicht sichtbar (werden nicht angezeigt)
- werden aber ganz konventionell als key-value-Paare im HTTP-Protokoll übertragen
  - Übertragung mit HTTP-Methoden **GET** und **POST**



# hidden fields (II)

- Vorteile der Technik:
  - absolut unabhängig von Clientsoftware/-konfiguration
  - Client wird nicht belastet
  - reine HTML-Technik
- Nachteile der Technik:
  - „anstrengend“, da nach dem ersten Formular alle Formulare dynamisch generiert werden und alle „versteckten“ Felder immer übertragen werden müssen
  - im HTML-Source-Code direkt sichtbar: "strg-u"



# bessere Ansätze

- Cookies und Sessions zur Personalisierung von Websites

- Beispiel:

neuer Browser, Besuch von amazon.de und dort suchen nach bestimmten Artikel

Browser schließen

neuer Besuch von Amazon:  
es werden passende Artikel angeboten

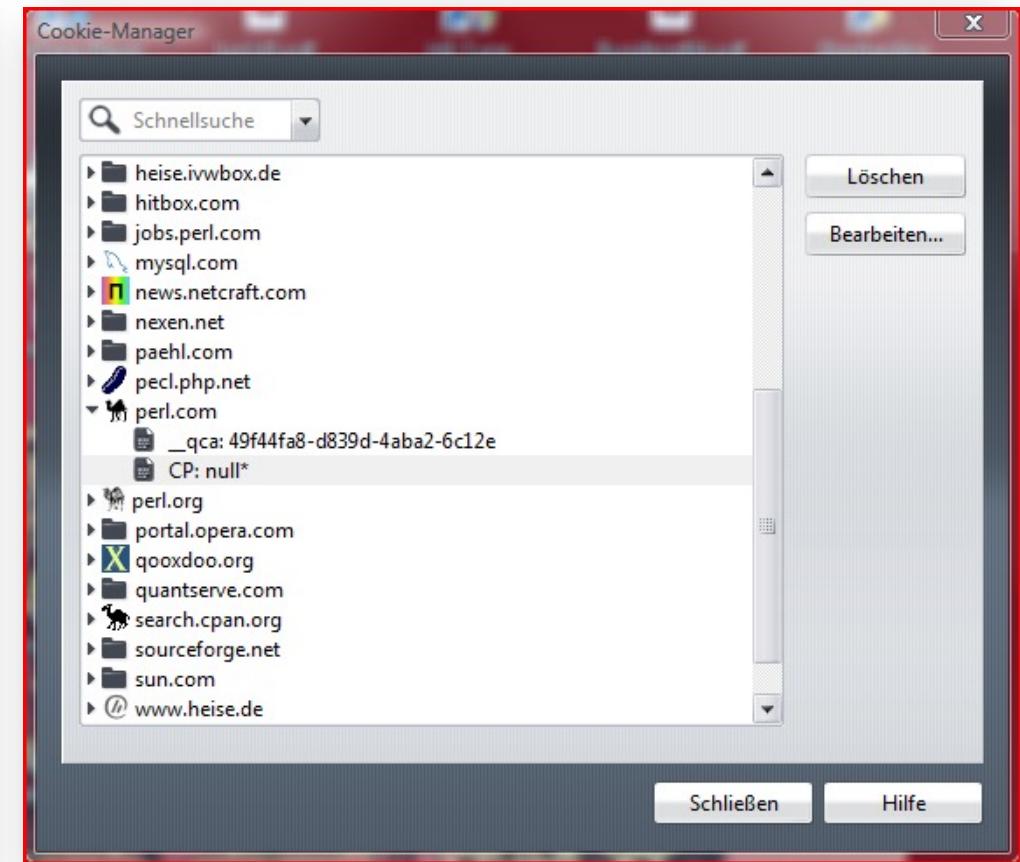
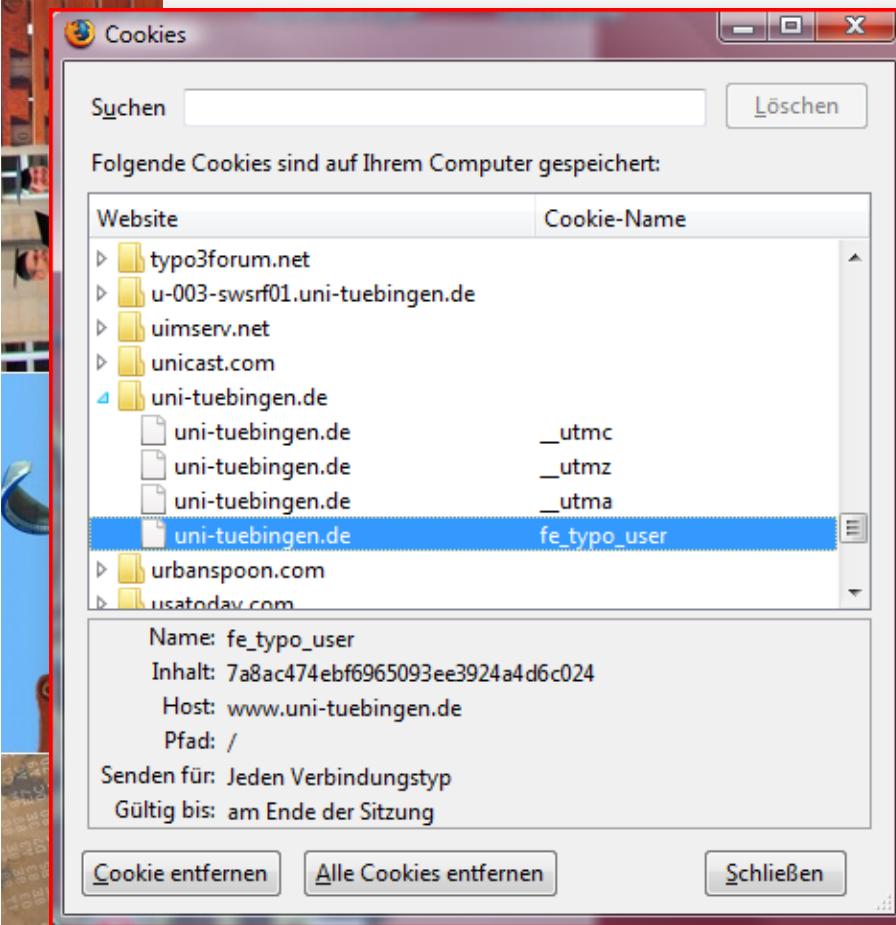


# Cookies

- Zusatzinformationen werden *lokal auf dem Client abgelegt* und müssen so nicht jedesmal übertragen werden
  - Erweiterung des ursprünglichen HTTP-Protokolls 0.9



# Cookie-Informationen





Cookies und Websitedaten

Website	Lokal gespeicherte Daten
www.hibike.de	3 Cookies
hiszilla.his.de	4 Cookies
wiki.his.de	4 Cookies
www.his.de	1 Cookie
www.hmt-rostock.de	1 Cookie
www.hna.de	Lokaler Speicher
www.hochschule-trier.de	1 Cookie
hosteurope.de	3 Cookies
www.hotel-wiegand.de	3 Cookies
hp.com	19 Cookies
h20000.www2.hp.com	1 Cookie
h20564.www2.hp.com	4 Cookies
h20565.www2.hp.com	3 Cookies
h20566.www2.hp.com	5 Cookies, Lokaler Speicher
www.hrk.de	Lokaler Speicher
login.hs-albsig.de	1 Cookie

Fertig



Cookies und Websitedaten

Website

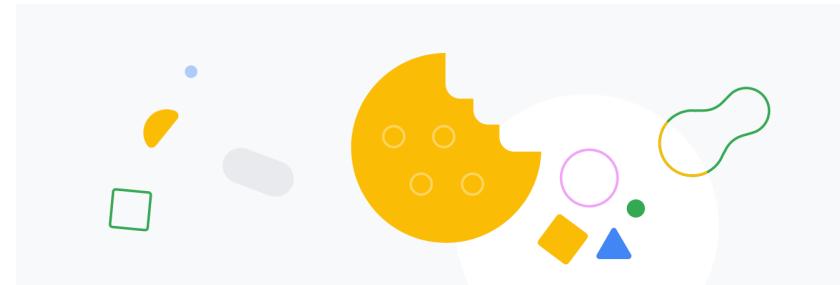
Lokal gespeicherte Daten

Alle entfernen

Cookies suchen

- u-003-sysmon.uni-tuebingen.... 2 Cookies
- www.unimuseum.uni-tuebing... Lokaler Speicher
- campus.verwaltung.uni-tuebi... 1 Cookie
- webmail.uni-tuebingen.de 4 Cookies
- webmail-test.uni-tuebingen.de 2 Cookies, Lokaler Speicher
- ceres.zdv.uni-tuebingen.de 3 Cookies
- e-antrag.zdv.uni-tuebingen.de 1 Cookie
- moodle02.zdv.uni-tuebingen.... 1 Cookie
- services.zdv.uni-tuebingen.de 2 Cookies
- u-003-service02.zdv.uni-tuebi... Lokaler Speicher
- u-003-service04.zdv.uni-tuebi... Lokaler Speicher
- www.unify.com Lokaler Speicher

Fertig



#### Allgemeine Einstellungen

##### Alle Cookies zulassen

- Websites können Cookies verwenden, um Ihnen das Surfen zu erleichtern; zum Beispiel, damit Sie angemeldet oder Artikel in Ihrem Einkaufswagen gespeichert bleiben
- Websites können Cookies verwenden, um Ihre Browseraktivitäten auf anderen Websites aufzuzeichnen und damit zum Beispiel Werbung zu personalisieren

##### Drittanbieter-Cookies blockieren

##### Alle Cookies blockieren (nicht empfohlen)

Cookies und Websitedaten beim Beenden von Chrome löschen



Bei Browserzugriffen eine "Do Not Track"-Anforderung mitsenden



Seiten vorab laden, um das Surfen und die Suche zu beschleunigen

Ruft Informationen von Webseiten im Voraus ab – auch von Seiten, die Sie noch nicht besucht haben.



Zu diesen Informationen gehören auch Cookies, wenn Sie diese zulassen.

#### Alle Cookies und Websitedaten anzeigen

Websites, die immer Cookies verwenden dürfen

Hinzufügen

Keine Websites hinzugefügt

Cookies immer löschen, wenn Fenster geschlossen werden

Hinzufügen

Keine Websites hinzugefügt

Websites, die nie Cookies verwenden dürfen

Hinzufügen





← Für www.heise.de lokal gespeicherte Daten

Alle entfernen



GED_PLAYLIST_ACTIVITY	v	x
GeizhalsConfcookie	v	x
_cb	v	x
_chartbeat2	v	x
_pubcid	v	x
_v__chartbeat3	v	x
j	v	x
tika	v	x
u2uforum_properties	v	x
wt3_eid	v	x
wt_ttv2_s_288689636920174	v	x
Dateisystem	v	x
Datenbankspeicher	v	x
Lokaler Speicher	v	x
Lokaler Speicher	^	x

Ursprung

<https://www.heise.de>

Dateigröße

3,2 MB

Zuletzt geändert

Freitag, 3. Juli 2020 um 12:38:43



← Für amazon.de lokal gespeicherte Daten

Alle entfernen

at-acbde	▼	X
authorizedDeviceId	▼	X
i18n-prefs	▼	X
lc-acbde	▼	X
s_dslv	▼	X
s_ev22	▼	X
s_nr	▼	X
s_vnum	▼	X
sess-at-acbde	▼	X
session-id	▼	X
session-id-time	▼	X
session-token	▼	X
sst-acbde	▼	X
ubid-acbde	▼	X
x-acbde	▼	X
x-wl-uid	▼	X



## Browserdaten löschen

Grundlegend

Erweitert

Zeitraum

Letzte Stunde



- Browserverlauf  
7 Elemente
- Downloadverlauf  
1 Element
- Cookies und andere Websitedaten  
Von 346 Websites
- Bilder und Dateien im Cache  
124 MB
- Passwörter und andere Anmeldedaten  
Keine

Abbrechen

Daten löschen



# Prinzip des Cookies

- ein Cookie ist ein **key-value-Paar**
- hat eine **begrenzte Haltbarkeit**
  - ...die aber lange sein kann
- ist Bestandteil des HTTP-Protokolls
- wird im **HTTP-Header übertragen**
- **einfache Textdatei** auf dem Client
- jedes Cookie gehört zu einer Website
- alle Browser speichern ihre Cookies getrennt
- für den jeweiligen Browser: **alle Browser-Tabs und Browser-Fenster haben die gleichen Cookies**
  - DOM: Cookie gilt für navigator-Objekt



# Datenstruktur des Cookies

- name, value (durch "=" getrennt)
  - version
  - expires
  - max-age
  - domain
  - path
  - port
  - comment
  - commenturl
  - secure
  - discard



← Für amazon.de lokal gespeicherte Daten

Alle entfernen

at-acbde



authorizedDeviceId



Name

authorizedDeviceId

Inhalte

27990937673479865

Domain

.amazon.de

Pfad

/

Senden für

Jede Verbindungsart

Zugänglich für Skript

Ja

Erstellt

Sonntag, 19. Februar 2017 um 17:43:22

Ablaufdatum

Dienstag, 13. Juni 2028 um 12:32:44



← Für amazon.de lokal gespeicherte Daten

Alle entfernen

at-acbde

^ | X

Name

at-acbde

Inhalte

Atza|lwEBIO9OZ5wTDWsAlbQ7U-DjRp81D2L8pC-  
PvBVRgogwX57FYDC5XtcV0hMJSa9S8BZGfMfqDe6dy8357qpAOFK5TVePStt4U3wthQkZ260DI-  
MGE4MC5PMqQcQYncL-MZ9J-6RhP2zaR8nAuf5C-IQj98Spuom1PdPVZeNrP7vo2Tjjb685dF9G6RCSI-  
7NSi1TihzhM4N9fpa3nb1sNWw1kZM-BImXStCs9OB0hCDMQ-  
dAt\_2kUjKv2p\_ylZkgYOGKyl6ES9zKChoV2tynYEVAz3qZbJ0gTomP5h0JeTn5kyt98ct8HME2lszdI0zzrN  
KsWVKgd07b5PQFMk7-laB06bsjKzYCTfhqlkj2BqKL4PT-  
\_7oC\_rpXqR7iq4cp0YF6g8STy68kEJyWCtIMjINhfC

Domain

.amazon.de

Pfad

/

Senden für

Nur sichere Verbindungen

Zugänglich für Skript

Nein (nur HTTP)

Erstellt

Sonntag, 5. April 2020 um 22:28:24

Ablaufdatum

Samstag, 31. März 2040 um 22:28:24



# Cookies und Sicherheit

- keine direkte Gefährdung des Clients durch das Cookie selbst
  - aber: "Ausspionieren" des Clients (auch von fremden Sites)
- einfache Manipulation des Cookies
  - Gefahr für *Diensteanbieter*





# Cookies in PHP

- Prinzipiell: Cookie ist Bestandteil von HTTP
  - Cookie wird vor dem Dokumentenbeginn <HTML> gesetzt
- PHP-Methode **setcookie**(...) setzt Cookie in PHP
  - Argument: key-value-Paar
  - „klassisches PHP“: Variable mit Bezeichner **\$key** und Belegung **value** zur Verfügung
- einfacher Zugriff auf Cookies über das PHP-Superglobal

**\$\_COOKIE**



# Syntax setcookie

- **bool setcookie ( string name [, string value [, int expire [, string path [, string domain [, bool secure]]]] ] )**
- name, value, path klar
- expire: Unix-timestamp, wann Cookie ungültig wird
- domain: (Sub-)Domainen, die das Cookie benutzen können
- secure: wenn true, dann Cookie-Informationen nur über https übertragbar



# setcookie

(PHP 4, PHP 5, PHP 7, PHP 8)

setcookie — Sendet ein Cookie

## Beschreibung

```
setcookie ( string $name , string $value = "" , int $expires = 0 , string $path = "" , string  
$domain = "" , bool $secure = false , bool $httponly = false ) : bool
```

Alternative Signatur, verfügbar ab PHP 7.3.0:

```
setcookie ( string $name , string $value = "" , array $options = [] ) : bool
```

**setcookie()** definiert ein mit den HTTP-Header-Informationen zu übertragendes Cookie. Wie andere Header auch, müssen Cookies **vor** jeglicher Ausgabe Ihres Skriptes gesendet werden (dies ist eine Einschränkung des Protokolls). Das bedeutet, dass Sie diese Funktion vor allen anderen Ausgaben, einschließlich der Ausgabe von `<html>`- oder `<head>`-Tags sowie jede Art von Whitespaces, aufrufen müssen.

Sind die Cookies einmal gesetzt, können Sie beim nächsten Seitenaufruf anhand des `$_COOKIE`-Arrays auf diese zugreifen. Die Cookie-Werte können auch in `$_REQUEST` vorhanden sein.



# Beispiel: Zähler, Dauer

< > C ⌂ ⚠ 134.2.6.167/~zrvwa01/php/cookie.php

⊕ 📸 ✅ ➤ ❤️ ⏪ ⏴

## Grundlagen Internet-Technologien

### Cookies mit PHP

Zähler: 4

Startzeitpunkt: 1625388873

Dauer der Session in Sekunden: 13



```
<?php
/*
 * Grundlagen der Internet-Technologien
 *
 * Beispiel fuer Cookies mit PHP:
 * - Cookie "start" merkt sich den Zeitpunkt des Seitenzugriffs
 * - Cookie "count" die Zugriffe vom gleichen Client
 * - wenn count nicht gesetzt ist, werden zunaechst beide gesetzt
 * - alle Cookies bekommen eine "Lebensdauer" von 600 Sekunden
 * - Verwendung des Superglobals $_COOKIE
 */

// Setzen des Cookies
if(!isset($_COOKIE['start'])) {
    $count = 1;
    $start = time();
    setcookie("start", $start, time()+600);           // setzen des Cookie "start"
} else
    $count = $_COOKIE['count'] + 1;      // Wert (value) des Cookies 'count' + 1

setcookie("count", $count, time()+600);           // Setzen des count-Cookies
?>
<!DOCTYPE html>
<html>
    <head>
        <title>Grundlagen Internet-Technologien: Cookies mit PHP</title>
        <link rel="stylesheet" type="text/css" href="../css/webkompendium.css">
        <link rel="shortcut icon" href="/favicon.ico">
    </head>
    <body>
        <center>
            <H2>Grundlagen Internet-Technologien</H2><H3>Cookies mit PHP</H3><HR><H4>
Z&auml;hler:
<?php if (isset($_COOKIE['count'])) echo $_COOKIE['count']; else echo "0"; ?>
<BR>Startzeitpunkt:
<?php if (isset($_COOKIE['start'])) echo $_COOKIE['start']; else echo "0" ?>
<BR>Dauer der Session in Sekunden:
<?php
    if (isset($_COOKIE['start'])) $duration = time() - $_COOKIE['start'];
    else $duration = "0";
    echo "$duration";
?
</H4><HR></center>
        </body>
    </html>
```



# Cookies in anderen Sprachen

- Python: Modul Cookie
- Perl: Modul CGI::Cookie
  - `my $cgi = new CGI;`  
`my $cookie = cgi->cookie(-name=>'name' ,`  
`-value=>'wert');`
- Ruby: ähnlich Perl



# ...und dann ist da noch...

## Diese Webseite verwendet Cookies

Wir verwenden Cookies, um Inhalte und Anzeigen zu personalisieren, Funktionen für soziale Medien anbieten zu können und die Zugriffe auf unsere Website zu analysieren. Außerdem geben wir Informationen zu Ihrer Verwendung unserer Website an unsere Partner für soziale Medien, Werbung und Analysen weiter. Unsere Partner führen diese Informationen möglicherweise mit weiteren Daten zusammen, die Sie ihnen bereitgestellt haben oder die sie im Rahmen Ihrer Nutzung der Dienste gesammelt haben.

**Cookies zulassen**

**Auswahl erlauben**

**Nur notwendige Cookies verwenden**



**Notwendig**



**Präferenzen**



**Statistiken**



**Marketing**

Details zeigen ▾





## Aktuelles Urteil zu Cookies: BGH sagt Einwilligung muss sein



maxsim - Fotolia.com

**Rechtsanwalt Sören Siebert**

Letzte Bearbeitung: 07. April  
2021



★★★★★ (73 Bewertungen, 3.90 von 5)

**Erst der EuGH, jetzt auch der BGH: Die Gerichte sind sich einig, dass Webseitenbetreiber eine aktive Einwilligung der Besucher benötigen, wenn sie Cookies setzen wollen. Diese Einwilligung muss vom Nutzer ausgehen, eine schon vorher ausgewählte Checkbox im Cookie-Banner genügt nicht. Wir erklären, was dieses Urteil für Webseitenbetreiber und Webdesigner bedeutet.**



# Cookiebot

< > C ■ | 🔒 [www.cookiebot.com/de/](https://www.cookiebot.com/de/)

Instagram X YouTube Heart Download Print

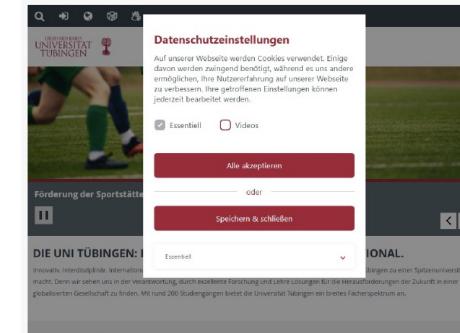
The screenshot shows the homepage of cookiebot.com/de. The page has a blue-tinted background featuring a smartphone, a laptop keyboard, and a person's hands. The Cookiebot logo is at the top left, and a menu icon is at the top right. A sub-header reads: "COOKIEBOT™ HILFT IHNEN, DIE VERWENDUNG VON COOKIES UND DAS ONLINE-TRACKING AUF IHRER WEBSITE KONFORM ZU GESTALTEN." The main title "Ist meine Website konform?" is prominently displayed in white. Below it, text explains the purpose of the service: "Die Allgemeine Datenschutz-Grundverordnung der EU (DSGVO) betrifft alle Websites mit Benutzern aus der EU. Überprüfen Sie, ob die Verwendung von Cookies und Online-Tracking auf Ihrer Website der DSGVO und der ePrivacy-Richtlinie (ePR) entspricht. Sehen Sie, welche Daten Ihre Website sammelt und ob Daten an Dritte übermittelt werden - auch zur Einhaltung des California Consumer Privacy Act (CCPA) nützlich." At the bottom, there is a white input field for "Ihre Webseitenadresse" and a blue button labeled "JETZT MEINE WEBSITE PRÜFEN".



# Cookie scan report

## Summary

Scan date: 05/07/2021  
 Domain name: www.uni-tuebingen.de  
 Server location: Germany  
 Cookies, in total: 1



## Scan result

1 cookies were identified.

**The result is based on a scan of up to 5 website pages and therefore not complete. To perform a complete scan, create a Cookiebot subscription for your domain.**

## Category: Necessary (1)

Necessary cookies help make a website usable by enabling basic functions like page navigation and access to secure areas of the website. The website cannot function properly without these cookies.

COOKIE NAME	PROVIDER	TYPE	EXPIRY
ROUTEID	uni-tuebingen.de	HTTP	Session
<b>First found URL:</b> <a href="https://uni-tuebingen.de/">https://uni-tuebingen.de/</a>			
<b>Cookie purpose description:</b> This cookie is used in context with load balancing - This optimizes the response rate between the visitor and the site, by distributing the traffic load on multiple network links or servers.			
<b>Initiator:</b> Webserver			
<b>Source:</b> uni-tuebingen.de			
<b>Data is sent to:</b> Germany (adequate)			
Adequate country under GDPR (EU)			



# Datenschutzreform: Britische Regierung will Cookie-Banner abschaffen

Die "lästigen" Pop-ups zur Einwilligung in Datennutzungen sollen mit dem geplanten "Data Reform Bill" durch ein Browser-basiertes Modell ersetzt werden.

Lesezeit: 5 Min. In Pocket speichern

93

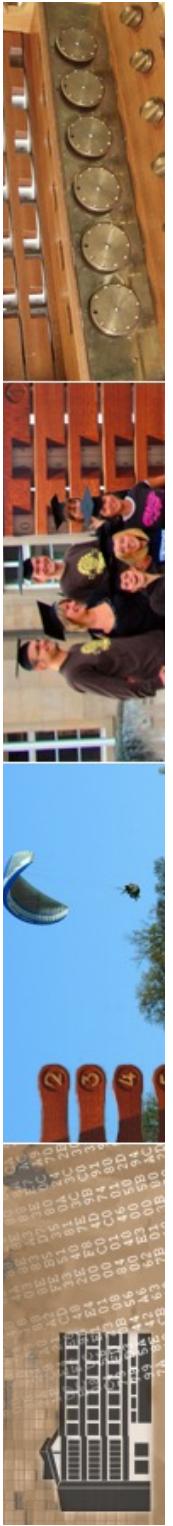


(Bild: Datenschutz-Stockfoto/Shutterstock.com)

19.06.2022 13:28 Uhr

Von Stefan Krempf

Die britische Regierung hat ihre Pläne für ein neues Gesetz zur verstärkten Nutzung von Daten und für den Schutz personenbezogener Informationen ausgeführt. Mit dem vorgesehenen "Data Reform Bill" sollen unter anderem Pop-ups und Banner beim Surfen im Internet reduziert werden. Die Rede ist hier von "den lästigen Kästchen, die Nutzer derzeit auf jeder Webseite sehen".



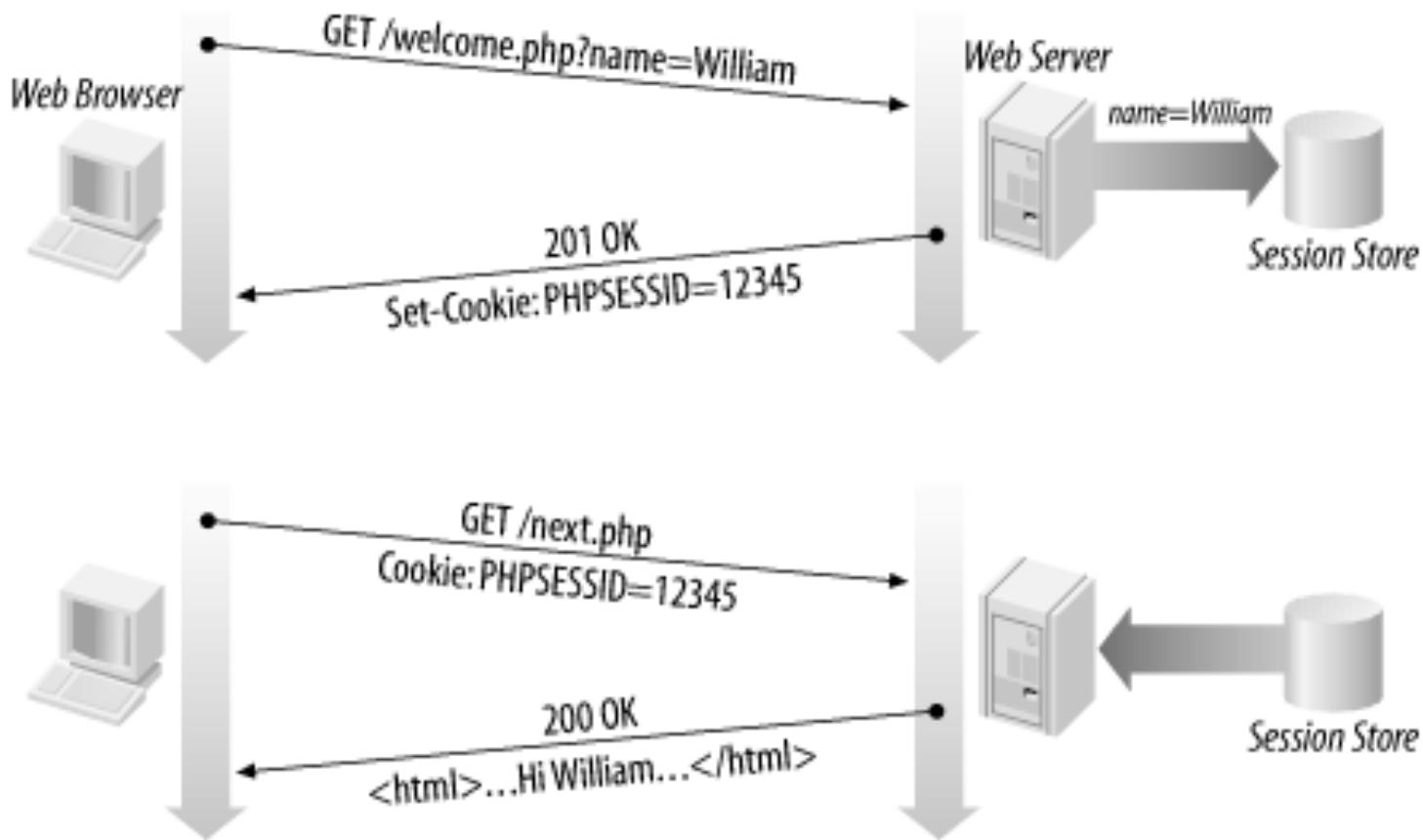


# Sessions

- Cookie: Daten liegen *auf dem Client* - und können dort direkt verändert werden
- Session: Daten liegen **auf dem Server**, werden über eine Session-ID der Browserinstanz zugeordnet
  - Kennzeichnung der Session über Session-ID
  - **diese Session-ID ist dann typischerweise ein Cookie auf dem Client**



# Prinzip





# Sessions in PHP

- die eigentlichen Session-Daten liegen auf dem Webserver
  - üblicherweise in /tmp, konfigurierbar über `session.save_path`
  - Daten sind dort sicher
- neue Session nur möglich vor der ersten Ausgabe
- Beginn einer Session:
  - `session_start()`
- Session-Daten in Subglobal
  - `$_SESSION`
- Ende einer Session:
  - `session_destroy()`



# session\_start

(PHP 4, PHP 5, PHP 7, PHP 8)

session\_start — Erzeugt eine neue Session oder setzt eine vorhandene fort

## Beschreibung

```
session_start ( array $options = [] ) : bool
```

**session\_start()** erzeugt eine Session oder nimmt die aktuelle wieder auf, die auf der Session-Kennung basiert, die mit einer GET- oder POST-Anfrage oder mit einem Cookie übermittelt wurde.

Wenn **session\_start()** aufgerufen wird oder eine Session automatisch startet, ruft PHP die Öffnen- und Lesen-Routinen der Session-Speicherfunktion auf. Dies ist entweder eine eingebaute Speicherfunktion, die standardmäßig mitgeliefert wird oder von Erweiterungen (wie z.B. SQLite oder Memcached) zur Verfügung gestellt wird oder eine eigene Funktion, die mittels [session\\_set\\_save\\_handler\(\)](#) definiert wurde. Die Lesen-Routine ruft alle vorhandenen Sessiondaten (abgespeichert in einem speziellen serialisierten Format) ab und deserialisiert sie, um damit die `$_SESSION`-Superglobale zu füllen.

Um eine benannte Session zu verwenden, rufen Sie [session\\_name\(\)](#) auf, bevor Sie **session\_start()** aufrufen.

Wenn [session.use\\_trans\\_sid](#) aktiviert ist, registriert die Funktion **session\_start()** eine interne Ausgaberoutine für das Umschreiben von URLs.

Verwendet ein Benutzer `ob_gzhandler` oder ähnliches mit [ob\\_start\(\)](#), dann ist die Reihenfolge der Funktionen wichtig für eine korrekte Ausgabe. Zum Beispiel muss `ob_gzhandler` vor Beginn der Session registriert werden.



```
zrvwa01@infodienste => more session1.php
<?php
/*
 * Grundlagen Internet-Technologien
 *
 * Sessions in PHP: Anlegen einer Session
 */

    // das fuehrende "@" unterdrueckt moegliche Fehlermeldungen

    @session_start();

    $_SESSION["s_userName"]      = "Thomas Walter";
    $_SESSION["s_userPermissions"] = "keine :-(";
?>
<!DOCTYPE html>
<HTML>
    <HEAD>
        <TITLE>Grundlagen Internet-Technologien: Sessions mit PHP</TITLE>
        <link rel="stylesheet" type="text/css" href="../css/webkompendium.css">
        <link rel="shortcut icon" href="/css/favicon.ico">
    </HEAD>
    <BODY>
        <CENTER>
            <HR><H2>Grundlagen Internet-Technologien</H2>
            <H3>Sessions in PHP</H3>
            <HR><H4>Session in PHP gestartet!</H4><HR>
        </CENTER>
    </BODY>
</HTML>
```





```
[zrvwa01@infodienste => more session2.php
<?php
/*
 * Grundlagen Internet-Technologien
 *
 * Sessions in PHP: Verwendung der Session
 */

@session_start();

$name  = $_SESSION['s_userName'];
$recht = $_SESSION['s_userPermissions'];

// wird in kein session_unregister() oder
// session_destroy() ausgefuehrt, bleiben die Daten erhalten!
?>
<!DOCTYPE html>
<HTML>
    <HEAD>
        <TITLE>Grundlagen Internet-Technologien: Sessions mit PHP</TITLE>
        <link rel="stylesheet" type="text/css" href="../css/webkompendium.css">
        <link rel="shortcut icon" href="/css/favicon.ico">
    </HEAD>
    <BODY>
        <CENTER>
            <HR><H2>Grundlagen Internet-Technologien</H2>
            <H3>Sessions in PHP</H3>
            <HR><H4>Hallo <?php echo $name; ?><BR>
            Ihre Rechte: <?php echo $recht; ?></H4><HR>
        </CENTER>
    </BODY>
</HTML>
```



# Grundlagen Internet-Technologien

## Sessions in PHP

Session in PHP gestartet!

# Grundlagen Internet-Technologien

## Sessions in PHP

Hallo Thomas Walter  
Ihre Rechte: keine :-(



```
[zrvwa01@infodienste => more session3.php
<?php
/*
 * Grundlagen Internet-Technologien
 *
 * Sessions in PHP
 */

@session_start();

if (!isset($_SESSION['start'])) {
    $_SESSION['start'] = time();
    $_SESSION['count'] = 0;
}

?>
<!DOCTYPE html>

<HTML>
    <HEAD>
        <TITLE>Grundlagen Internet-Technologien: Sessions mit PHP</TITLE>
        <link rel="stylesheet" type="text/css" href="../css/webkompendium.css">
        <link rel="shortcut icon" href="/css/favicon.ico">
    </HEAD>
    <BODY>
        <CENTER>
            <HR><H2>Grundlagen Internet-Technologien</H2>
            <H3>Sessions in PHP</H3>
            <HR><H4>Anzahl der Zugriffe: <?php echo ++$SESSION['count']; ?><BR>
            Dauer der Session: <?php echo (time() - $_SESSION['start']); ?><BR>
            Session-ID: <?php echo session_id(); ?> </H4><HR>
        </CENTER>
    </BODY>
</HTML>
```



< > C ☰ | ▲ 134.2.6.167/~zrvwa01/php/session3.php

+ 📸 ✅ ⏪ ❤️ | ⏴ ⏵

# Grundlagen Internet-Technologien

## Sessions in PHP

Anzahl der Zugriffe: 3

Dauer der Session: 9

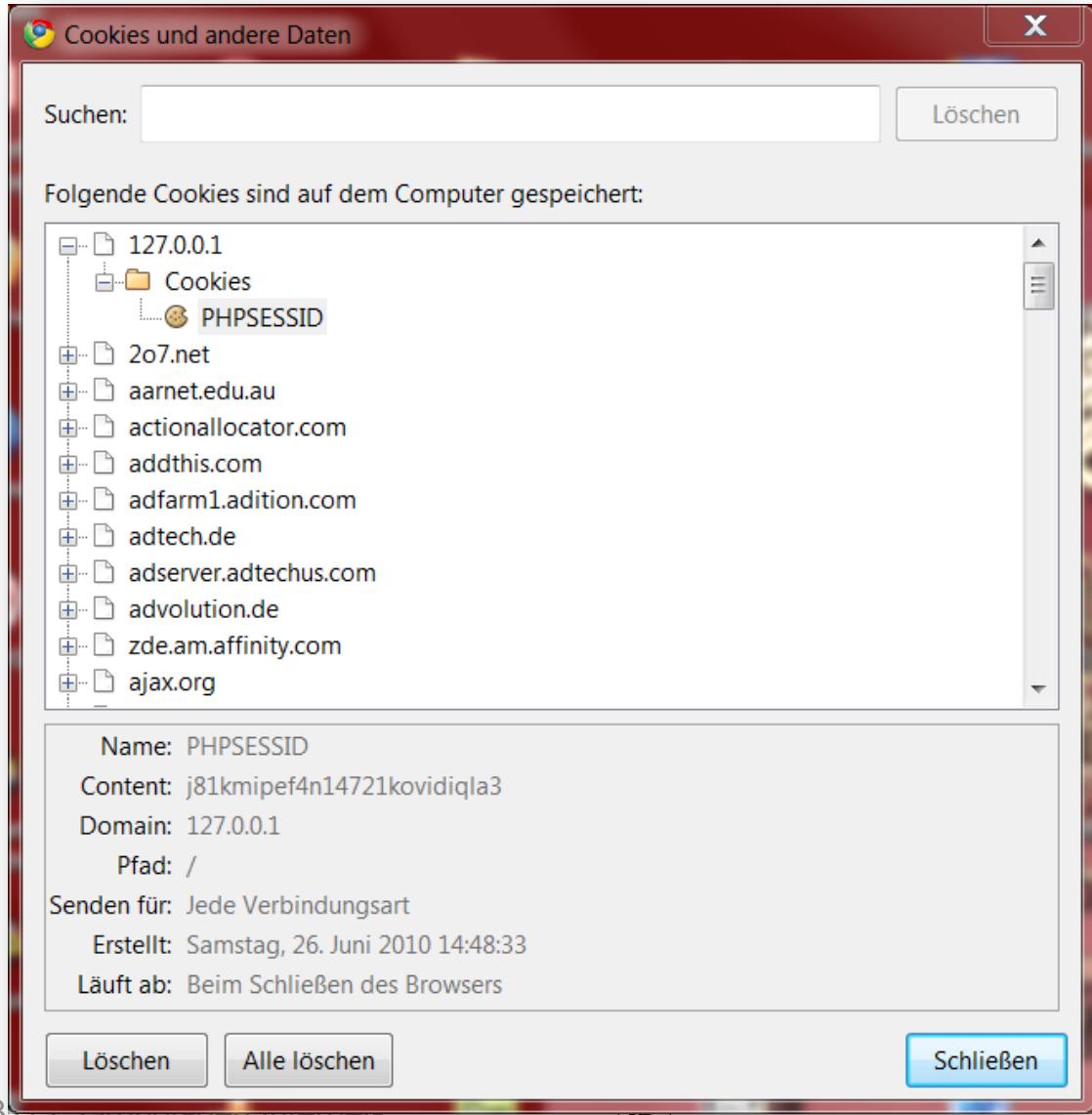
Session-ID: 9a4f21292pgvucgil0bbdu1tt8





# die Session-ID

- ist ein Cookie
- mit dem Namen PHPSESSID





← Für 134.2.6.146 lokal gespeicherte Daten

[Alle entfernen](#)

S  
Ä  
T  
N



PHPSESSID

^

X

Name

PHPSESSID

Inhalte

te8irfdspi7rmf512s7rurliu6

Domain

134.2.6.146

Pfad

/

Senden für

Jede Verbindungsart

Zugänglich für Skript

Ja

Erstellt

Sonntag, 5. Juli 2020 um 11:24:10

Ablaufdatum

Beim Beenden der Browsersitzung



# PHP-Sessions ohne Cookies

- es gibt auch Alternativen ohne Cookies:
  - Session-ID an die URL anfügen:  
**Session->url()**
  - wieder als verstecktes Formular-Feld:  
**Session->hidden()**



# Cookies & Sessions in Python

- Python3-Modul `http.cookies` erlaubt einfache Verwaltung von Cookies
- Cookie als Python-Objekt
- auf dieser Basis Implementierung von Sessions



## 22.23. `http.cookies` — HTTP state management

Source code: [Lib/http/cookies.py](#)

The `http.cookies` module defines classes for abstracting the concept of cookies, an HTTP state management mechanism. It supports both simple string-only cookies, and provides an abstraction for having any serializable data-type as cookie value.

The module formerly strictly applied the parsing rules described in the [RFC 2109](#) and [RFC 2068](#) specifications. It has since been discovered that MSIE 3.0x doesn't follow the character rules outlined in those specs and also many current day browsers and servers have relaxed parsing rules when comes to Cookie handling. As a result, the parsing rules used are a bit less strict.

The character set, `string.ascii_letters`, `string.digits` and `!#$%&'*+-.^_`|~:` denote the set of valid characters allowed by this module in Cookie name (as `key`).

*Changed in version 3.3:* Allowed ':' as a valid Cookie name character.

**Note:** On encountering an invalid cookie, `CookieError` is raised, so if your cookie data comes from a browser you should always prepare for invalid data and catch `CookieError` on parsing.

### `exception http.cookies.CookieError`

Exception failing because of [RFC 2109](#) invalidity: incorrect attributes, incorrect *Set-Cookie* header, etc.



# Sessions in Perl

- es gibt mehrere (viele) Möglichkeiten, Sessions in PERL (CGI) zu implementieren
  - Session-Funktionalität über verschiedene PERL-Module
  - Suche in CPAN



# Sessions in Python

- für Python Funktionalität über Sessions object
- Teil von Requests



## Requests: HTTP for Humans™

<https://2.python-requests.org/en/master/>





# Session Objects

The Session object allows you to persist certain parameters across requests. It also persists cookies across all requests made from the Session instance, and will use `urllib3`'s [connection pooling](#). So if you're making several requests to the same host, the underlying TCP connection will be reused, which can result in a significant performance increase (see [HTTP persistent connection](#)).

A Session object has all the methods of the main Requests API.

Let's persist some cookies across requests:

```
s = requests.Session()

s.get('https://httpbin.org/cookies/set/sessioncookie/123456789')
r = s.get('https://httpbin.org/cookies')

print(r.text)
# '{"cookies": {"sessioncookie": "123456789"}}'
```

Sessions can also be used to provide default data to the request methods. This is done by providing data to the properties on a Session object:

```
s = requests.Session()
s.auth = ('user', 'pass')
s.headers.update({'x-test': 'true'})

# both 'x-test' and 'x-test2' are sent
s.get('https://httpbin.org/headers', headers={'x-test2': 'true'})
```

Any dictionaries that you pass to a request method will be merged with the session-level values that are set. The method-level parameters override session parameters.



# Modul CGI-Session

- aktuelle Version: 3.95
- Autor: Sherzod Ruzmetov
- die Session-Informationen können serverseitig wie schon in PHP verschieden abgelegt werden
  - File (default)
  - Database
  - ...
- Modul CGI::Session ist davon unabhängig, spezifischer Treiber wird dazugeladen wie bei DBI

# CGI::Session::File

- Standard-Fall: Session-Information in Datei auf Server
- notwendiger Parameter: Verzeichnis für die Ablage der Datei
  - `use CGI::Session;`  
`$session = new CGI::Session("driver:File,`  
`undef, {Directory=>'C:/temp'} );`
- jede Session in separater Datei
  - Defaultname: `cgisess_%s` mit `_%s` : session-id



# Clientseitig: Cookie

- wie in PHP wird Clientseitig die session-ID im Cookie abgelegt:
  - `$cookie = $cgi->cookie  
(CGISESSIONID =>$session->id) ;`
  
  - `print $cgi->header (-cookie=>$cookie) ;`



# ...damit...

- verliert die Website die Eigenschaft "Zustandslos"
- wesentliche Erweiterung unserer bisherigen Fähigkeiten und Möglichkeiten!



# ...und nun...

- haben wir zwei wesentliche Erweiterungen für die Web-Programmierung kennen gelernt:  
Cookies & Sessions

