

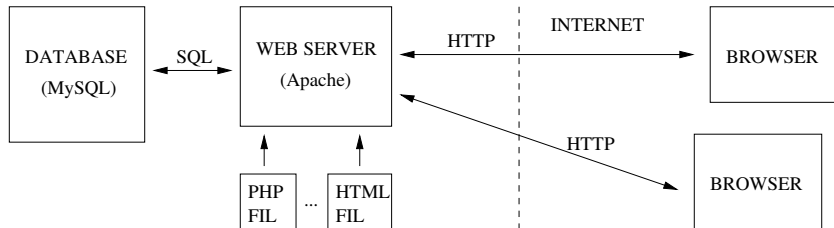
Scripting, Databaser og Systemarkitektur, F2008

Forelæsning

- ▶ Hvad har vi lært indtil nu?
- ▶ Cookies
- ▶ Sessions, ultra kort
- ▶ Spørgetime, hvad skal vi lave?
- ▶ Introduktion til øvelsen
- ▶ Har vi god tid laver vi en ad hoc applikation.

Hvad har vi lært indtil nu?

Oversigt:



En PHP-fil:

```
<body>
<?php
    $hello = "<b>Hello</b>";
    $world = "<i>world!</i>";
    echo $hello.$world;
?>
</body>
```

Hvad har vi lært indtil nu?

En anden PHP-fil:

```
<?php
    $w = array('Hello ', 'world!');
    $t = array('b', 'i');
    $str='';
    for($i=0; $i<sizeof($w); $i++){
        $str.='<'.$t[$i].'>'.$w[$i].'</'.$t[$i].'>';
    }
    echo $str;
?>
```

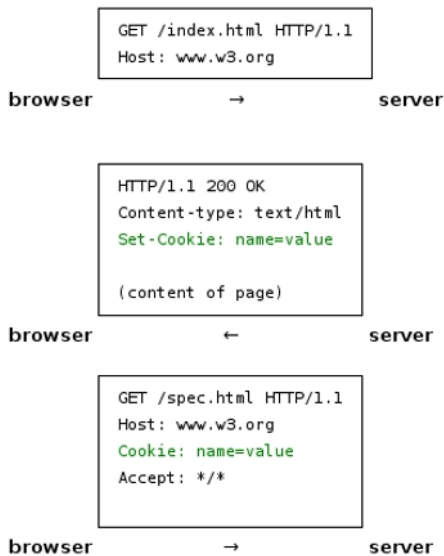
Hvad har vi lært indtil nu?

- ▶ Variabler, tal, strenge og arrays
- ▶ Beregninger
- ▶ if-sætninger og løkker
- ▶ Funktioner og kodegenbrug
- ▶ Indhentning af data fra brugere med forms
- ▶ Check af brugerindtastninger med regulære udtryk
- ▶ SQL (create, update, insert, delete, join, foreign key references....)
- ▶ Databasetilgang via PHP
- ▶ SQL forespørgsler via PHP
- ▶ Afsending af emails
- ▶ Datoer og beregning af tid

Hvad er Cookies

- ▶ En cookie er en tekst-streng som en web-server sender til en browser og som browseren returnerer uændret når browseren igen besøger det samme site. Se evt.

`wp.netscape.com/...`
Denne tekst-streng kan vi bruge til at opbevare værdier



Billedet er taget fra Wikipedia

Kort fortalt:

- ▶ Det betyder vi kan holde styr på brugeren uden at skulle sende diverse værdier rundt med alle links.
- ▶ Normalt holder vi styr på brugere ala:
 - ▶ `page.php?userid=1&password=not-so-secret`
 - ▶ der hentes med `$_REQUEST['userid']` og `$_REQUEST['password']`
- ▶ Med cookies kan vores url'er så således ud:
 - ▶ `page.php`
 - ▶ og brugerinfo kan så hentes ind via `$_COOKIE` arrayet (forudsat vi har sat en cookie)
 - ▶ `$_COOKIE['userid']` og `$_COOKIE['password']`

Bemærk: Selvom vi gør det her, er det ikke det rigtige at gemme passwords i cookies

- ▶ Cookies er specielt nyttige til håndtering af tilstande — session tracking:
 - ▶ Fokusering af banner ads — hvad var en bruger interesseret i sidst? (tracking)
 - ▶ Adgangskontrol — login mekanisme
- ▶ I de fleste browsere kan man se hvilke cookies der er aktive i filen `cookies.txt`:
 - ▶ **XP, FireFox:** C:/Documents and Settings/"USERNAME"/Application Data/Mozilla/Firefox/
 - ▶ **OS X:** /Users/"USERNAME"/Library/Application Support/Firefox/Profiles/
 - ▶ **Linux:** /home/\$USER/.mozilla/firefox/
- ▶ Andre browsere gemmer hver cookie i en separat fil
 - ▶ **XP, Explorer:** C:/Documents and Settings/"USERNAME"/Cookies

Cookies, sådan ser de ud

```
# HTTP Cookie File
# http://www.netscape.com/newsref/std/cookie_spec.html
# This is a generated file! Do not edit.
# To delete cookies, use the Cookie Manager.

.facebook.com TRUE / FALSE 1208877194 h_user cd61099412fa
.facebook.com TRUE / FALSE 1218790794 login_x user%40domain.dk
.facebook.com TRUE / FALSE 1209395589 ABT c7bf99ee1...
.facebook.com TRUE / FALSE 1211382789 datr 7cabb4c...
```

Hver linie uden # repræsenterer en cookie

Ingredienser i en cookie

.facebook.com TRUE / FALSE 1218790794 login_x user%40domain.dk

- ▶ .facebook.com
 - ▶ Domæne hvor cookien er tilgængelig
- ▶ TRUE
 - ▶ Kan subdomæner læse cookie'en (eks test.facebook.com)
- ▶ /
 - ▶ Path på serveren der kan læse cookien. (her alle).
/test ville begrænse det til facebook.com/test
- ▶ FALSE
 - ▶ Kræves der en sikker forbindelse for at sende cookie'en?
- ▶ 1218790794
 - ▶ Unix timestamp, hvornår udløber cookien
- ▶ login_x
 - ▶ Cookie'ns navn, dvs \$_COOKIE['login_x']
- ▶ user%40domain.dk
 - ▶ Cookie'ns værdi, dvs det echo \$_COOKIE['login_x'] udskriver

- ▶ Almindelige browsere understøtter kun omkring 20 cookies pr. site
- ▶ Almindelige browsere understøtter kun omkring 300 cookies totalt
- ▶ En cookie kan højest være 4 kilobytes stor
- ▶ Brugeren kan slå cookies fra
- ▶ Sikkerhedsproblemer med Cookies
 - ▶ Cookies kan true privatlivet — “third-party cookies”
 - ▶ “cookie theft”
 - ▶ Det er stadig indhold fra brugeren, vi kan ikke stole på det

Se evt. http://en.wikipedia.org/wiki/HTTP_cookie

Der er en grund til at nogle personer vælger at slå cookies fra i deres browser:

- ▶ Søgemaskiner viser ads for hvad man søgte efter sidste gang
 - ▶ Problematisk hvis man bliver kigget over skulderen af arbejdsgiveren
 - ▶ eller af konen?
- ▶ Google ved hvor du bor
 - ▶ Tredie parter kan følge din færden
- ▶ Tilstandsproblemer:
 - ▶ Forskelle mellem tilstand implementeret med cookies og tilstand implementeret med formvariabler — “Back”-knappen
 - ▶ Problemer med cookies når flere browsere benyttes samtidig

Cookies kan stjæles og manipuleres

- ▶ Cookies gemmes som tekst på brugerens maskine, så fysisk adgang eller en ubeskyttet computer kan lade skurken kopiere cookien.
- ▶ Sendes cookien ukrypteret, eksempelvis via åbne trådløse net, er det en forholdsvis simpel sag at opsnappe cookien når den sendes.
- ▶ Kan man stjæle en cookie, kan man stjæle en identitet

Løsning?

- ▶ Benyt en sikker forbindelse hvis muligt
- ▶ Tjcek brugerinput
- ▶ Lad **aldrig** cookien indeholde fortrolige informationer
 - ▶ Generelt gemmer vi ikke passwords og lignende i cookies, vi gemmer en tilfældigt genereret streng som vi bruger til at slå op i databasen med

Registrering af cookies

- ▶ For at registrere at en cookie skal sættes i browseren kan PHP-funktion `setcookie()` benyttes:
 - ▶ `setcookie(name, value, expire, path, domain, secure)`
 - ▶ Fra PHP5 har `setcookie()` et 7. parameter der angiver om JavaScript må læse cookien, men det er ikke alle browsere der understøtter det. IE7 og FF2 gør.
- ▶ Kortere former er tilladte:
 - ▶ `setcookie(name, value)`
 - ▶ Eksempel: `setcookie("a", "5")`
- ▶ Cookies er en del af browserens "headers" og skal sættes/slettes/rettes før der sendes noget til browseren (med f.eks. `echo`)

Eksempel 1.

```
setcookie('language', 'Dansk', time()+3600, '/', '.itu.dk', 0);
```

Vores cookie "language" med værdien "Dansk",
udløber "time()+3600", dvs om 1 time
path er sat til "/" så den gælder for alle niveauer på sitet
domain er sat til .itu.dk

Dvs. webmail.itu.dk, sysadm.itu.dk, people.itu.dk, etc.
Secure sættes til false (true ville være '1')

Eksempel 2.

```
setcookie('language', 'Engelsk', time()+36000, '/mail/');
```

Vores cookie "language" med værdien "Engelsk", udløber "time()+36000", dvs om 10 timer path er sat til "/mail/" så den gælder fra mail mappen og dybere Eks. a.dk/mail, a.dk/mail/junk

```
vi lader domain og secure være tomme,  
default domain: nuværende domæne (eks. a.dk)  
default secure: false
```

Læsning af cookies i PHP

- ▶ For at læse en cookie sendt med en forespørgsel kan et PHP-script kigge i et cookie-array:
 - ▶ `$mycookie = $_COOKIE["mycookie"];`
 - ▶ `echo $_COOKIE["language"];`
- ▶ Vi kan tjekke om en cookie findes, ligesom vi tjekker om en variabel findes.

```
if(@$_COOKIE['mycookie']!= ''){  
    // Hvad er det nu @ gør?  
    // Hvorfor bruger vi den her?  
}
```

```
if( isset($_COOKIE['mycookie']) ) {  
    // Hvad er det nu isset() gør?  
}
```

Sletning af Cookies

- ▶ Cookies slettes ved at sætte en cookie med udløb i fortiden!

```
//vi skal benytte næsten samme parametre som da vi satte den  
//en cookie sat med:
```

```
setcookie('mycookie', 'sweet', time()+100, '/', '.a.dk', 1);
```

```
//skal fjernes med
```

```
setcookie('mycookie', '', time()-100, '/', '.a.dk', 1);
```

I eksemplet sætter vi cookies med `secure-parameteret = 1`, dvs når vi fjerner den skal `secure` være 1 igen, det sammen med `name`, `path` og `domain`.

```
<?php
    $count = @$_COOKIE["count"];
    if ( $count == "" ) {
        $count = 0;
    }
    // vi sætter ikke "expire" på cookien, den forsvinder
    // når browseren lukkes
    setcookie("count", $count + 1);

    echo "<h2>CookieCount: $count</h2>
        <a href=\"cookiecount.php\">Up</a>";
?>
```

Bemærk:

- ▶ Der sendes ikke formvariabler med til scriptet i Up-linket!
- ▶ Hvordan virker scriptet når flere browsere er åbne på samme client?

Eksempel: Ulovlige handlinger

Eksempel 1:

```
<html>.....  
<?php  
    //vi sætter en cookie der udløber om en time  
    setcookie("language", "da", time()+3600);  
?>
```

Eksempel 2:

```
<?php  
    echo "Hej";  
    //vi fjerner cookien "language"  
    setcookie("language", "", time()-3600);  
?>
```

Hvorfor er de to eksempler ”ulovlige”?

Vi tager den i browseren

Eksempel: `cookiebasket.php`

Kildekode: `index_show.php?file=cookiebasket.php`

Eksamen ligger den 3. juni 2008

Vi mødes den 29 maj kl 17.

Hvad skal vi snakke om?

- ▶ Arrays, løkker, variable, html, SQL, Reg Expr ????
- ▶ Vi kan gennemgå et længere eksempel ????
- ▶ Mail jeres forslag (MEGET gerne i grupper) inden den 22/5

Eksempel: `http://itu.dk/people/holbech/cookiestealer/`

Opgavesæt 11 er igen en “åben øvelse”

Hvis du har en ide til en lille web-service har du nu mulighed for at bygge den!

Næste gang gennemgår vi et tidligere eksamenssæt, lav det evt. inden, så ved du hvad det drejer sig om og hvad du har svært ved

Så er det også lettere at stille spørgsmål

Der er ikke øvelser næste gang, men jeg har bedt hjælpelærerne om at være der

PHP har også en anden måde at gemme "state" på kaldet `sessions` det kan du læse lidt om på de næste slides, det er **IKKE** en del af pensum

Sessions mener nogle er lettere at gå til, og så kan de blandt andet gemme arrays direkte

- ▶ Med sessions gemmes tilstand på webserveren og kun et *sessionid* gemmes som en cookie på clienten.
- ▶ PHP har god support for sessions:
 - ▶ Funktionen `session_start()` ser om en *sessionid* allerede er tilstede som en cookie på browseren. Ellers genereres et nyt *sessionid* som gemmes på browseren som en cookie.
 - ▶ Når en session er startet (med `start_session()`) kan data hentes *OG* gemmes fra et session-array:
 - ▶ `echo $_SESSION["myvar"]`
 - ▶ `$_SESSION["myvar"]='myval'`

- ▶ Funktionen `session_destroy()` kan benyttes til at ryde op efter en session, dvs, lager frigives og `sessionid` cookie slettes på klienten.
- ▶ For at "ødelægge" brugerens session med `session_destroy()` skal `session_start()` være kaldt!!!
- ▶ Funktionen `session_start()` skal ligesom `setcookie()` kaldes *før* der sendes noget til browseren

Potentielle problemer med sessions

- ▶ Cookie-problematik
- ▶ Det er ikke klart hvornår web-serveren kan antage at sessionlageret kan genbruges eller smides væk

Løsningen: sessionid kan sendes som URL parameter, se IPMA s. 317-318

Implementation af en simpel indkøbskurv ved brug af sessions.

```
<?php session_start();
if ( @$_REQUEST['submit'] == "Empty Basket" ) { // destroy basket
    session_destroy();                          // destroy session
    header("Location: basket.php"); exit;        // reload and exit
}
$kurv = @$_SESSION["kurv"];
if ( @$_REQUEST['new'] != "" ){
    $kurv[] = $_REQUEST['new']; // maybe add new stuff
}
$_SESSION["kurv"] = $kurv;
//forsættelse følger .....
```

Eksempel: Session Basket — basket.php, del 2

```
//del 2
echo "<html><title>Session Basket</title>
    <body><h2>Session Basket</h2>
        <form action=\"basket.php\">
            <ul>";
// loop igennem arrayet
for ( $i = 0 ; $i < count($kurv) ; $i++ ) {
    echo "<li>$kurv[$i]</li>";
}
echo "<li> <input type='text' name='new'>
    <input type='submit' value='Add'>
    <input type='submit' name='submit' value=\"Empty Basket\">
    </ul></form></body></html>";
?>
```

basket.php source