Begrynder til at lave log ind system

Hej

Vil jeg gerne lave en lille programmering forklare til hvordan du laver din helt egen lille start på at log ind system du kan altid lave videre på det eller bygge videre på det :)

Som sagt i den her log ind system som vi skal bygge på og lave på

Skrevet den **25. Mar 2012** af **tobrukDk** I kategorien **Programmering / PHP** | ★★☆☆☆

Som det første før man kan logge ind i et system er man nødt til at have oprettet en bruger. Så først vil vi se på hvordan vi får oprettet en bruger i inde MySQL-tabel.

Allerførst må vi have en simpel forståelse for hvordan en MySQL-database er skruet sammen. Databasen består af tabeller og tabeller består af rækker og kolonner. Kolonnerne defineres når du opretter tabellen. De fleste steder der understøtter MySQL har ligeledes installeret phpMyAdmin, som er en brugervenlig måde at håndtere dine databaser og tabeller på.

og har du ikke lege med database før nu så kan det her måske være en utrolig god start for dig som ikke har lege eller prøve det før!, :) Der findes utrolig mange local server som du kan hente utrolig mange steder og den som jeg vil sige til dig at du klart skal bruge det er den her ; http://www.apachefriends.org/en/xampp.html

Vi opretter først en tabel til vores brugere. Denne tabel kalder vi simpelt nok "brugere". Den skal indeholde følgende 3 kolonner (felter):

"id" af typen INT, under ekstra vælges "auto_increment" og der sætte et hak ved den lille firkant med nøglen (Primær). som gør talet UNIK! og det er utroligt vigtigt for dig og brugerne!.

"brugernavn" af typen VARCHAR og længden 255
"password" af typen VARCHAR og længden 255

Følgende SQL-kode kan også bruges til at oprette tabellen (dette gøres i phpMyAdmin under fanen SQL):

```
CREATE TABLE `brugere` (
`id` INT NOT NULL AUTO_INCREMENT PRIMARY KEY ,
`brugernavn` VARCHAR( 255 ) NOT NULL ,
`password` VARCHAR( 255 ) NOT NULL
)
```

Du kan også valg at bare sige 100 af varchar men det er klart bedste at bruge alle sammen f.eks hvis en bruger ind taster et password på 101 så kommer det sidste ikke med!.. så bruge dem alle det er klart det som jeg vil gøre hvis jeg var dig..

```
<html>
<head>
<title>Opret bruger</title>
</head>
<body>
<h1>Opret bruger</h1>
<form action="indsaet.php" method="post">
Brugernavn: <input type="text" name="brugernavn">
<br>
Password: <input type="password" name="password">
<br>
Gentag password: <input type="password" name="gentag">
<br>
<input type="submit" value="Opret">
</form>
</body>
</html>
```

At bedre brugeren indtaste de oplysninger der skal bruges til at logge ind med er ganske simpel html og kunne se sådan her ud:

Det første man skal lægge mærke til er "action" her defineres den adresse man kommer hen til når man klikker på "Opret"-knappen. "method=post" definerer måden hvorpå de indtastede data sendes videre til serveren, det kommer vi til at se nærmere på om lidt.

indsaet.php

```
<?
$brugernavn = $_POST["brugernavn"];
$password = $_POST["password"];
$gentag = $_POST["gentag"];
$errorCount = 0;
if($brugernavn == "")
{
echo "Du skal indtaste et brugernavn.<br>";
$errorCount++;
}
if($password == "")
{
echo "Du skal indtaste et password.<br>";
$errorCount++;
}
if($gentag == "" || $gentag != $password)
{
echo "De to passwordfelter skal have ens indhold.<br>";
```

```
$errorCount++;
}
?>
```

Det første vi lægger mærke til her, er en ny type variabel-navn. $_POST["navn"] er en global variabel. $_POST variablen indeholder alt data sendt fra en html-form med "method=post". Den måde man finder de rigtige data på er klammerne bagefter. Husker vi på html-delen havde hvert felt en "name="-attribut. Det der står efter lighedstegnet er det vi skal have ind i vores [""]-klammer.
$_POST["brugernavn"] indeholder altså det brugeren har indtastet i "brugernavn"-inputfeltet.

Indsæt i databasen
Vi har nu udtrykket de data brugeren indtastede i systemet og mangler altså kun at indsætte brugere i databasen.
Til filen **indsaet.php** tilføjes:

```
mysql_connect("host","brugernavn","password");
mysql_select_db("database");
if($errorCount == 0)
{
$password = MD5($password);
$insert = mysql_query("INSERT INTO brugere (brugernavn,password) VALUES
('$brugernavn','$password')");
if(!$insert)
echo "Der skete en fejl. Prøv igen. <a href=\"java script:history.back(-1)\">Tilbage</a><br>";
else
echo "Brugeren blev oprettet. <a href=\"index.php\">Forside</a><br>";
}
```

Det første vi er nødt til for at arbejde med vores database er at oprette en forbindelse til den. Dette sker ved de første to linjer kode. Den første linje skal rettes til, så det passer med de oplysninger dit webhotel/server/lokale installation bruger. Du vil oftest finde disse oplysninger et sted i dit kontrolpanel på dit webhotel eller local server som du har på din computer :)
Anden linje fortæller hvilken database vi gerne vil arbejde med. På nogle webhoteller har man kun en database til rådighed, og så vil navnet på den nok stå oplyst samme sted som de resterende oplysninger. I andre tilfælde kan man selv oprette databaser, og så vælger du blot et passende navn.

```
<html>
<head>
<title>Login</title>
</head>
<body>
<h1>Login</h1>
<form action="login.php" method="post">
Brugernavn: <input type="text" name="brugernavn">
<br>
Password: <input type="password" name="password">
<br>
```

```
<input type="submit" value="Opret">
</form>
</body>
</html>
```

Til at logge ind bruges en simpel html-form der minder meget om oprettelsesformen.

den her fil kan du kalde hvad du har lyst til..

PHP-delen
PHP-delen af login-scriptet skal tjekke om brugernavn og password passer sammen, og om det indtastede brugernavn overhovedet eksisterer i vores brugertabel. I fald brugernavn og password matcher og eksisterer i vores tabel, skal brugeren registreres som logget ind.

Vi tager det en del af gangen - første tjekker vi med databasen:

login.php

```
<?php
session_start();
mysql_connect("host","brugernavn","password");
mysql_select_db("database");
$bruger = $_POST["brugernavn"];
$pass = $_POST["password"];
$error = "";
$userQuery = mysql_query("SELECT id,brugernavn,password FROM brugere WHERE
brugernavn='$bruger'");
if(mysql_num_rows($userQuery) != 1)
{
$error .= "Brugeren eksisterer ikke.<br>";
}
else
{
$userArray = mysql_fetch_array($userQuery);
if($userArray["password"] != MD5($pass))
{
$error .= "Password og brugernavn passer ikke sammen.<br>";
}
}
?>
```

Hvis jeg var dig så havde jeg virkelig valgt at bruge SHA1 da det er klart bedre og mere sikkert ;)

Den sidste del af login-koden tilføjes:

```
if($error != "")
{
echo $error . "<a href=\"java script:history.back(-1);\">Tilbage</a>";
}
```

```
else
{
$_SESSION["logged_in"] = 1;
$_SESSION["user_id"] = $userArray["id"];
header("Location: user.php");
}
```

Jeg er ordblind og jeg har måske lave nogle fejl, men jeg har fået lidt hjælp af en ven til at rette min stave fejl og så har jeg tilføj nogle tinge af og til..

Håber du kan forstår mig eller Skrive til PM :) hvis der er mere du vil vide eller lign :)

Kommentar af DeeDawg d. 26. Mar 2012 | 1

Dejligt at du tager initiativ, til at skrive en guide, men jeg kan altså desværre kun give den **2 stjerner**.

1. Du benytter dig stadigvæk af den gamle MySQL API, selvom olebole har brugt tid på at skrive en guide om MySQLi, hvilket giver dig mulighed for at komme igang med det. Vi skal gerne have folk til at bruge det, så det hjælper ikke rigtig at de guides, der er på sitet, ikke benytter det.

2. Det er ikke **OOP**. Uanset om det er en begynder guide eller ej, bør du altid fortælle og vise folk mulighederne med OOP.

3. Det virker lidt som om, at du ikke har undersøgt de ting du prøver at lære andre. Du skriver således

"*Du kan også valg at bare sige 100 af varchar men det er klart bedste at bruge alle sammen f.eks hvis en bruger ind taster et password på 101 så kommer det sidste ikke med!.. så bruge dem alle det er klart det som jeg vil gøre hvis jeg var dig..*"

Dem alle sammen? Et VARCHAR felt kan max indeholde **65.535** tegn og ikke kun **255**. Hvis man har defineret feltet til kun at kunne indeholde **100** tegn, og brugeren har indtastet **101** tegn, da de oprettede deres bruger, uden at du som udvikler ikke forhindrede og fortalte dem om det, er jo en fallit erklæring.

Kommentar af DeeDawg d. 26. Mar 2012 | 2

Rettelse: Det jeg har angivet som punkt 3, ville jeg egentlig bare have skrevet som kommentar.

Der burde have stået

3. Der intet nyt i det du viser, og derved er der ikke et behov for denne guide. Du kan finde utallige guides der viser det samme, bare ved en enkelt søgning på Google. I mængden finder du også nogle på dansk, så det er heller ikke et problem. :)

Kommentar af kjeldsted d. 26. Mar 2012 | 3

#1:
I har vel egentlig lidt begge ret ang. VARCHAR:
" [NATIONAL] VARCHAR(M) [CHARACTER SET charset_name] [COLLATE collation_name]
A variable-length string. M represents the maximum column length in characters. In MySQL 5.0, the range of M is 0 to 255 before MySQL 5.0.3, and 0 to 65,535 in MySQL 5.0.3 and later." Det afhænger af om MySQL er < 5.0.3 eller ej.

Men derudover. Hvorfor tillade brugere at skrive fx. brugernavne på 255 tegn. Dels kan det ødelægge et design ret nemt. Og så kan man, så vidt jeg lige hurtigt kan gennemskue lave et brugernavn som fx.: <script>window.location="http://exp.dk"</script>

Så hver gang ens brugernavn optræder på siden bliver man viderestillet her til Eksperten, eller hvad man nu end kan klemme ind af sjove scripts på 255 tegn.

Og som der også bliver pointeret i #1, så bør i det mindste guides være lavet i mysqli

Kommentar af kjeldsted d. 26. Mar 2012 | 4

DamnYouIPad. Fortsættelse:

Og som der også bliver pointeret i #1, så bør i det mindste guides være lavet med mysqli APIen, da det er langt mere tidssvarende i dag. Så ret den til mysqli API, og ret hvad der nu ellers er af sikkerheds fejl og det kan være den bliver ganske nyttig.

Kommentar af kjeldsted d. 26. Mar 2012 | 5

Og tak til eksperten for igen at ødelægge en kommentar med det link fis. Der skulle i #3 selvfølgelig stå som brugernavn:

<script>window.location="http:// exp.dk"</script>

Kommentar af DeeDawg d. 26. Mar 2012 | 6

Ja, det har du selvfølgelig ret i kjeldsted, men derfor står min kommentar stadigvæk ved magt. Han bør indikere at han taler om en ældre version, før han kommer med en påstand.

Jeg er vel ikke helt forkert på den, når jeg antager at en ny guide tager udgangspunkt i de nyeste teknologier og seneste versioner af dem? :)

Kommentar af tobrukDk d. 27. Mar 2012 | 7

Ja okay jeg kan godt se hvad du mener DeeDawg, ;) jeg skulle nok have opdater den eller hvad man skal sige :)

Kommentar af bare-mig d. 19. Apr 2012 | 8

Jeg kan ikke få din header(location) til at fungere. Den sender mig ikke videre til "user.php".
Hvad kan der være galt?

Kommentar af Kennerhoj d. 19. Apr 2012 | 9

Hey Folkens. Se lige alt det han har skrevet, det er ikke noget man klare på 2 sekunder. Ingen grund til at skrappe op i laver også selv fejl.

Kommentar af gsa d. 04. May 2012 | 10

#9 - Dette er jo en GUIDE til andre, så man må da forvente at det ikke kommer "frit fra leveren", men er afprøvet og tilføjet her som copy/paste?? .... Ellers burde man jo i stedet åbne et indlæg med en titel som f.eks: "forslag til .....". Hvis nogen skal lære af en guide, så skal den være så tæt på fejlfri som mulig, ellers taber man nybegynderen og vedkommende vil ikke vide hvad der er galt. Ligeledes kan jeg anbefale copy/paste metoden for også at eliminere stavefejl etc. (ikke at jeg skriver perfekt, men jeg skriver jo

heller ikke guides). Stavefejl flytter også fokuset et forkert sted hen.

Kommentar af Temp_dk d. 07. Jul 2012 | 11

Gem aldrig password i ren tekst gem en hash værdi istedet for

$password = sha1($_POST[password']);

så kan $password gemmes i databasen og fylder kun 40 byte uanset hvor lang password der er brugt.