Formhåntering i PHP

Hvordan du håntere <form>'s med POST og GET data.

Skrevet den **06. Feb 2009** af **tdafoobar** I kategorien **Programmering / PHP** | ★★☆☆☆

En hurtig kigger på ekspertens php spørgsmål viser at der stadig er store problemer med folks håntering af forms, så jeg valgte lige at skrive en lille hurtig artikel for at bringe jer alle up-to-date.

Først vil jeg påpege at der er alt for mange forkerte tutorials, og gammelt skrammel på nettet, så hvis du har lært noget af dette, glem det igen.

Simpel <form> håntering
Først skal vi lige bygge en simpel html form op, her bruger jeg html 4.01 men du kan også vælge at bruge xhtml, der er ingen forskel for php kodens vedkommne. Jeg tager her udgangspunkt i felterne til en simpel gæstebog, for så at bruge det til en afsluttende kommentar omkring SQL sikkerhed.

```
<form action="" method="POST">
  <fieldset>
    <legend>Skriv i min gæstebog</legend>
    <p>
      <label for="navn">Navn</label><br>
      <input type="text" name="navn">
    </p>
    <p>
      <label for="besked">Besked</label><br>
      <textarea name="besked" cols="0" rows="0" style="width: 300px; height:
200px;"></textarea>
    </p>
    <p>
      <input type="submit" value="Send!" name="mysubmit">
    </p>
  </fieldset>
</form>
```

En hurtig forklaring:
- action="" reffere til den side vores PHP kode er på til at fange vores POST data, når den står til "" betyder det samme side, dog er det mest politisk korrekt at skrive filnavnet her, men jeg er personligt for doven, da der også skulle alle url argumenter med.
- method="POST" , metoden her kan være POST eller GET afhæning af hvilken måde vi vil sende data'en på.

POST og GET
POST og GET er 2 forskellige måder at sende data på.

Når du har sendt POST data, og trykker på reload (F5) vil du få en advarsels popup som siger at "this page you're trying to view contains POSTDATA", og hvis du trykker OK, vil du sende data'en igen. Når data'en kommer fra en <form> vil $_POST['navn'] f.eks. reffere til værdien af <input name="navn" ..>

GET er querystring argumenter, som f.eks. www.hej.dk/?side=hejsa
hvor $_GET['side'] så vil give værdien hejsa.

Korrekt håntering ved brug af isset()
Det er her de fleste mennesker fejler. Du har allerede set du SKAL bruge $_POST eller $_GET til at få data'en med, selvom du måske har brugt $navn istedet for $_POST['navn'] før (dette reffere til register_globals som er noget gammelt usikkert møg du skal glemme alt om).
For at checke om POST eller GET data er sat skal vi bruge funktionen isset() (www.php.net/isset) da $_POST og $_GET er arrays (www.php.net/array) , og hvis man forsøger at tilgå en værdi i et array som ikke findes vil man få en warning.

Derfor bør du altid gøre sådan her:

```
<?php
  if(isset($_POST['key']))
  {
    $key = $_POST['key'];
  }
  if(isset($_GET['key'])) // www.hej.dk/?key=foo
  {
    $key = $_GET['key']; // værdi: "foo"
  }
?>
```

Hvor du måske før har gjort sådan her

```
<?php
  if($_POST['key']) // FORKERT !!
  {
    $key = $_POST['key'];
  }
?>
```

Hvis du ikke får fejl på overstående kode, er det fordi at din PHP opsætning ikke er striks nok, og du risikere så at få en masse problemer/bøvl i fremtiden. Gør det rigtigt første gang og det løsser de fleste problemer.

Tilbage til vores gæstebog
Nu mangler vi så lige at få koden til at hente data'en fra vores gæstebog. Denne stump kode placere vi så FØR noget som helst HTML, overhovedet! Da vi ellers ikke kan lave et redirect som man typisk vil gøre for at undgå dobbeltposting. Derudover er det en god tommelfinger regel at have så meget som muligt php før noget som helst output/html.

```
<?php

if(isset($_POST['mysubmit']))
{
  $navn = addslashes($_POST['navn']);
  $text = addslashes($_POST['besked']);
}
```

```
?>
```

Denne kode giver dig så 2 pæne variabler med værdierne I.
Men hov, hvad er addslashes() ? Jo, det er for at sikre mod SQL injections (og specielle tegn som ' og \ der vil drille).

Addslashes() og mysql_real_escape_string() har det formål at søge for at databasen kun for det resultat der er ønsket. For at fjerne de ekstra \ (slashes) igen bruger du stripslashes().

Mini-eksempel med SQL kode i.

```
<?php

if(isset($_POST['mysubmit']))
{
  $navn = addslashes($_POST['navn']);
  $text = addslashes($_POST['besked']);
  $sql = "INSERT INTO guestbook(name,msg) VALUES('$navn','$text')";
  mysql_query($sql) or die(mysql_error())
}

?>
```

Dette burde have givet dig viden nok omkring hvordan man bruger POST,GET og <form>'s. Hvis dette strider hvordan du allerede har lært det, har du næsten helt sikkert lært det forkert! Men skriv endelig hvis du er uening.

Little hurtig 15minutters guide, men jeg fandt det rimelig nødvendigt.

php.net links
http://www.php.net/isset
http://www.php.net/addslashes
http://www.php.net/stripslashes
http://www.php.net/mysql_real_escape_string
http://dk2.php.net/variables.external (GET og POST)

Post Scriptum
Stavefejl er gratis :-)

Changelog
22. Maj 2006 - Udgivet
22. Maj 2006 (aften)

- Olebole har ret I sin kritik, så har navngivet submit knappen "mysubmit" istedet, da dette vel vil være det samme som at lave et hidden felt ekstra.

- terrak (og evt. andre) hvis I ønsker dybere forklaring på bestemte punkter uddybe endeligt, har oprettet diskussionstråd til artikel her: http://www.eksperten.dk/spm/711161

-mclemens mysql_real_escape_string kræver en åben mysql connection, og derfor er det f.eks. ikke brugbart hvis du arbejde med andre databaser ;-) Jeg vil lade det være op til jer selv at vælge, og

eventuelt skrive en artikel mere omkring escaping senere.
magic_quotes_gpc() bliver forresten fjernet I PHP6.

30. Maj 2006

- Rettede tip fra coderdk, mht. 'for' aribtutten i <label>

-tjanum Jeg snakker til daglig med flere hundrede php kodere, nye som gamle, og netop det denne artikel drejer sig om er at fremvise syntaksen, ikke så meget brugen af POST data. Alt for mange "øvede" php programmører dropper escaping, bruger php3 syntax, eller alle mulige underlig hacks for at opnå hvad man ellers kan gøre som vist her i Artikelen. Og jeg skulle mene at den bør være let nok foreståelig for nybegyndere, da dette er det ultimative basic i PHP, den kode som enhver php koder vil skrive igen, igen og igen.

Pratisk talt er denne artikel bare hvad jeg, groft sagt, siger til nye&gamle programmører et par gange om dagen.
Igen vil jeg påpege, at artiklen bare er en hurtig DANSK forklaring af hvad I iforvejen burde have læst på php.net, selvom jeg godt ved mange ikke gør sig den ulejlighed omkring emner de tror de kender i forvejen.

Kommentar af olebole d. 22. May 2006 | 1

Udmærket, men det er en _meget_ slem fejl at kalde noget somhelst 'submit', 'form', 'iframe', el.lign. Det kan få katastrofale følger, hvis man ønsker at bruge JavaScript i dokumentet. I stedet opretter man et hidden field med f.eks. navnet 'context' og værdien 'opret'. Så spørger man i stedet på dette felt på serveren.

Kommentar af coderdk d. 27. May 2006 | 2

hvis du laver <label for="navn"> og i din input <input type="text" name="navn" accesskey="n"> så kan du både trykke på "Navn" label'en eller trykke ALT+N for at aktivere inputfeltet

Kommentar af thorjakobsen d. 31. Jul 2006 | 3

En god artikel som kan bruges til et godt opslagsværk.
Men hvorfor ikke bruge $_REQUEST["var"] istedet for $_POST["var"]?

Kommentar af tuxic d. 23. May 2006 | 4

Kommentar af terrak d. 22. May 2006 | 5

Fin og gratis gennemgang, men hvis man er ny i php, er der nok flere ting man ikke helt forstår. Da den er skrevet til "øvede" php programmører er dette ikke et problem.

Det er godt med referencer og brugbare eksempler.

Kommentar af el_barto (nedlagt brugerprofil) d. 23. May 2006 | 6

Fint forsøg men de mange stavefejl trækker voldsomt ned.

Kommentar af tjanum d. 26. May 2006 | 7

Tror du har sat dig mellem to stole. Begyndere vil nok næppe kunne finde hoved eller hale i artiklen, og for

øvede er det jo halvgammel, elementær viden. Hvis du ønsker at begynde en forklaring af GET og POST, bør du gøre det ordentligt eller referere til en side, der forklarer det grundigt med det samme (istedet for et link nederst i artiklen måske). Du forsøger at beskrive implikationer ved at bruge POST (som iøvrigt ikke nødvendigvis er rigtig - det kommer an på, hvordan man implementerer sine form-handlere), men udelader implikationer ved at bruge GET. At du så også begynder at inddrage SQL i artiklen, gør den ikke mere anvendelig, hvis du ønsker at hjælpe PHP-begyndere med forms. Dog dermed ikke sagt, at din escaping forklaring er ligegyldig. Det er helt sikkert et plus, at du har den del med. Personligt er jeg ligeglad med stavefejlene, men det undrer mig nu alligevel lidt, at man ikke lige smider sin tekst gennem en stavekontrol, når man vælger at publicere artikler. Du skal naturligvis have ros for forsøget. Alt i alt synes jeg trods alt ikke, at artiklen er fuldstændig uanvendelig - og så har du jo gjort den gratis, så den gør jo nok ikke nogen skade :-).

Kommentar af ksvendsen (nedlagt brugerprofil) d. 29. May 2006 | 8

Jeg synes du skriver en nydelig artikel - det er tit, sådanne artikler man lige står og mangler, når man er i tvivl om f.eks. forms og håndtering af post/get. Godt med de referencer!

Kommentar af mclemens d. 22. May 2006 | 9

- - - Edit: Kommentar til kommentar: Ok :) - - - error_reporting(2047); ... Op i toppen af php filen og du er sikker på at få en fejlmelding når der er noget der ikke er helt optimalt ... (http://dk.php.net/manual/da/function.error-reporting.php)
... og mysql_real_escape_string vs. addslashes som du bruger (uddrag fra http://dk2.php.net/mysql_real_escape_string) - er selv ved at rette lidt p.t... [mysql_real_escape_string() kalder MySQLs biblioteks funktion mysql_escape_string, hvilket tilføjer en skråstreg, til følgende karakterer: NULL, \x00, \n, \r, \, ', " og \x1a.] og [Bemærk: Hvis magic_quotes_gpc er aktiveret, bør du først køre strengen gennem stripslashes(). Brugen af denne funktion, på data der allerede er sikret, vil sikre den dobbelt.] ... d.v.s. det anbefales at man bruger mysql_real... istedet for addslashes evt. stripslashes hvis magic_quotes_gpc er slået til (fremgår ikke helt tydeligt at det er bedre men check php's database sikkerheds anbefalinger ... sidste link i denne kommentar)... addslashes escaper kun: (http://dk2.php.net/addslashes) ', \, ", NULL - vi mangler dermed \x00, \n, \r og \x1a ... læg også mærke til den første bruger kommentar i addslashes der beder læseren om at bruge mysql_real_escape_string istedet... Fint den er gratis, men ikke godt at den ikke er helt 100% sikkerhedsmæssigt - når det er det man slår på tromme for :/ P.s.: Trals man ikke kan bruge
's i kommentarer... Denne her anbefales varmt!!! http://www.php.net/manual/da/security.database.php

Kommentar af phil-profil d. 24. May 2006 | 10

det var ikke det bedste jeg hr hørt men det var meget godt for begyndere