Webbaserade informationssystem med PHP och databaser

DD1051 Databasteknik och datorkommunikation

Dagens föreläsning

- Syfte
  - Ge de praktiska kunskaper och färdigheter som krävs för att bygga webbaserade informationssystem med databaser.
- Mål
  - Lära er den viktigaste syntaxen i PHP.
    - Variabler och tilldelning, villkor, loopar, funktioner, modularisering
  - Lära er hur databasteori kan appliceras praktiskt i ett informationssystem.
  - Lära er hur man kan göra sessioner, dvs hantera ett tillstånd såsom en shoppingvagn eller en inloggning.
    - URL-rewriting, Hidden fields, Cookies
  - Mailhantering via web. ”Gör din egen hotmail”

Dagens innehåll

- Hur skapa dynamiskt innehåll med PHP
- PHP-syntax
- Web och databaser
- Sessionshantering
- Mailhantering via web

Observationer gällande cgi-script

- I de flesta fall ska output ha mime-typen text/html
- I dynamiskt genererade html-sidor är ofta största delen av sidan ”statisk”, dvs statiska print-satser utan något dynamiskt innehåll.
- De flesta programmeringsspråk är utvecklade ”före” webben, och är inte specialanpassade just för web-bruk.

Server-Pages-språk

- Snart efter webbens uppkomst utvecklades flera så kallade ”server pages-språk”.
- Specialanpassade för web-bruk
- Hanterar problemen på föregående sida på ett ”bra sätt”.
- Exempel: asp, jsp, php

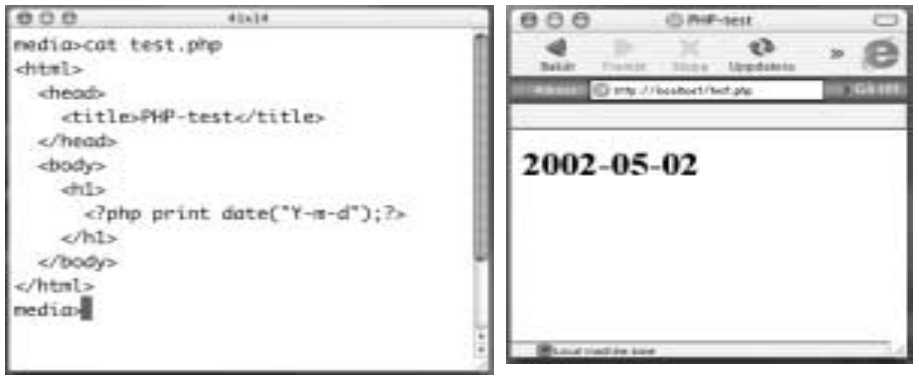
PHP

- I denna kurs använder vi i år språket PHP som vunnit mycket i populäritet de senaste åren.
- Enkelt för nybörjare
- Stor spridning (5 miljoner servers för några år sedan)
- Snabbt

Grundtanken i server-pages-språk

- Är nästan som vanliga html-dokument, men på de platser man vill ha dynamiskt innehåll kan man placera programkod.
- Webservern noterar att sidan är en PHP-sida och låter en php-tolk förbehandla sidan, dvs exekvera programsnuttarna.
- Resultatet är ren html-kod (eller xml eller gif eller vad man vill) och tolkas som vanligt av webläsaren.
- Om inget annat anges sätts MIME-typen automatiskt till text/html

Enkelt exempel i PHP



Dagens innehåll

- Hur skapa dynamiskt innehåll med PHP
- PHP-syntax
- Web och databaser
- Sessionshantering
- Mailhantering via web

Språket PHP

- PHP är uppbyggt från grunden för att passa webben.
- Är ett så kallat ”server pages-språk”, dvs html-kod blandas med programkod.
- Fördelar
  - Stor spridning
  - Snabbt
  - Enkelt
- Nackdelar
  - Ganska ostrukturerat, om en funktion är användbar läggs den in.
  - Lite okompatibelt mellan olika versioner (exempel på detta kommer längre fram)

PHP grunder

- För att kunna använda PHP på en webserver krävs att webservern är konfigurerad för detta
  - t.ex. nestor.nada.kth.se och www.nada.kth.se, men troligen inte spray, geocities etc.
- Filnamnen ska sluta på .php
- Programkod omsluts av processinstruktionen

```
<?php
   diverse programinstruktioner
?>
```


PHP grunder (2)

- En websida kan bestå av flera block html-kod och php-kod

```
<html>
  <head>
    <title><?php print date(”Y-m-d”);?></title>
  </head>
  <body>
    <?php
        $a=17;
        print $a;
    ?>
  </body>
</html>
```

Variabler

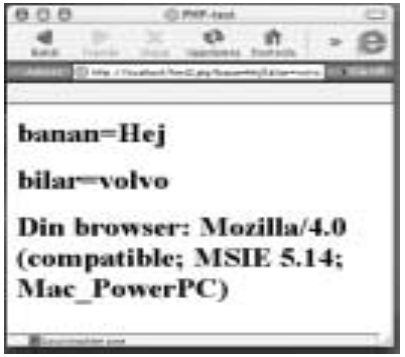
- PHP har ingen ”strikt” datatypning. Variabler måste inte deklareras innan de används.
- Variabler föregås av $-tecken

```
<?php
  $banan = 17;
  $gurka = ”Hej hopp i lingonskogen";
  $myFloat = 14.1;
  print $banan;
  print $gurka;
  print $myfloat;
?>
```


Om vi gör ett formulär i html ...

...så kommer vi åt variablerna via vektorn $_REQUEST

```
media>cat test2.php
<html>
  <head>
    <title>PHP-test</title>
  </head>
  <body>
    <h1>banan=<?php  print $_REQUEST[“banan”]; ?></h1>
    <h1>bilar=<?php  print $_REQUEST[“bilar”]; ?></h1>
    <h1>Din  browser:
    <?php  print getenv('HTTP_USER_AGENT'); ?>
    </h1>
  </body>
</html>
```



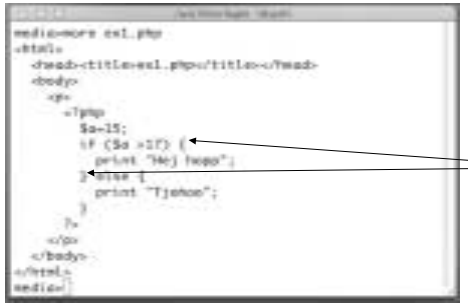
$_REQUEST används för GET, POST (och cookie) variabler.

Formulär-variabler i PHP

- Det är mycket enkelt att komma åt variabler som skickats i en GET eller POST-request.
- De kan refereras genom $_REQUEST[”variabelnamn”] ibland även med $variabelnamn beroende på hur PHP installerats.
- Övriga variabler (t.ex. HTTP_USER_AGENT) kan hämtas med funktionen GETENV.

Villkor med if-else

Block



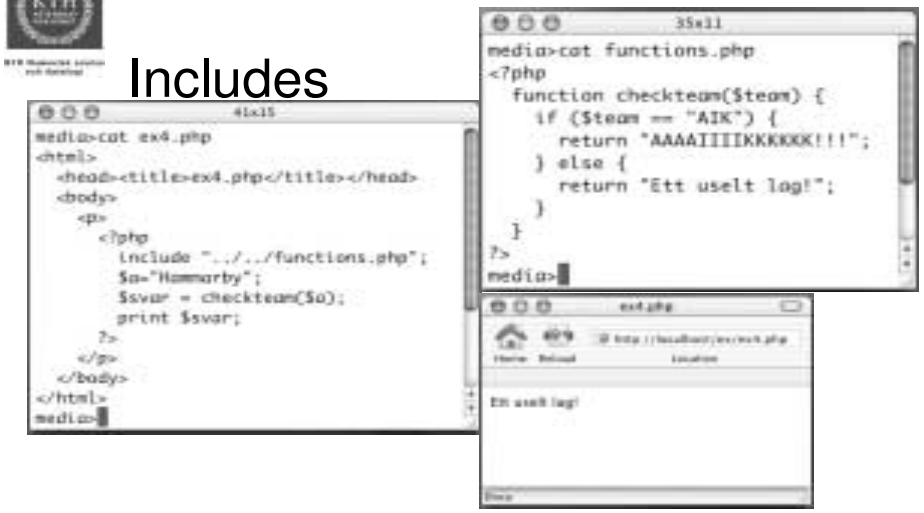
För att knyta samman flera satser till ett block används ”måsvingar” {} på samma sätt som i Java och C, inte indentering som i Python. Dessa fås vanligen genom att samtidigt trycka ner altgraph-7 respektive altgraph-0

Loopar med while

Obs! Vanlig html-kod som genereras

Funktioner

Includes



Includes (2)

- Includes är lämpliga för att lägga funktioner som förekommer flera gånger.
- Det går även att lägga t.ex. Variabler i ”config-filer” som återkommer i flera filer, t.ex. IP-adresser till andra datorer, databaslösenord etc.
- Includefilerna kan, till skillnad från html och php-filer, ligga var som helst i filstrukturen vilket kan vara både bra och dåligt ur säkerhetssynvinkel (`include ”/etc/passwd”`)

Databaser och web

- PHP-syntax
- Web och databaser
- Sessionshantering
- Mailhantering via web

MySQL

- Gratis (open source) databashanterare till Unix och Windows (www.mysql.com) utvecklat sv svenskt företag
- Fördelar
  - Gratis
  - Brett ”community support”
  - Mycket snabb då man gör enkla saker

Postgresql

- I år använder vi Postgresql som ni redan använt i laboration 2.

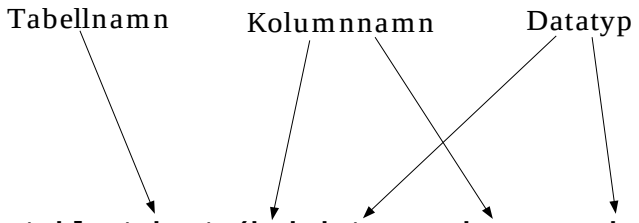
Postgresql använda psql

```
nada11:~>rxtelnet nestor
inge@nestor's password: nada-lösenordet
Last login: Fri May 02 2003 14:14:57 from nada11.nada.kth.
Sun Microsystems Inc.   SunOS 5.6       Generic August 1997
No mail.
[1] 3756
nestor>psql
Välkommen till psql 8.2.3, den interaktiva PostgreSQL-terminalen.

Skriv:  \copyright för upphovsrättsinformation
        \h för hjälp om SQL-kommandon
        \? för hjälp om psql-kommandon
        \g eller avsluta med semikolon för att köra en fråga
        \q för att avsluta

inge=#
```

Ledtext är databasens namn =#.

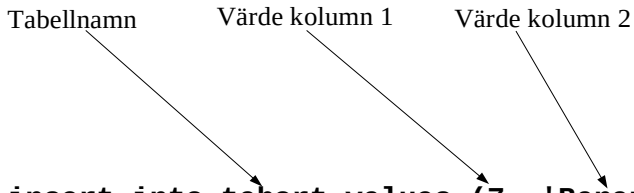
Ni får var och en en databas som heter som ert användarnamn

Postgresql – skapa en tabell




```
inge=# create table tabort (hej integer, hopp varchar(30));

CREATE TABLE
```

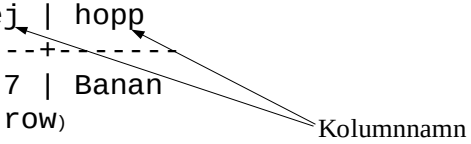

Postgresql – sätta in värden i en tabell




```
inge=# insert into tabort values (7, 'Banan');

INSERT 0 1
```

Postgresql – selektera värden från en tabell

```
inge=# select * from tabort;

 hej | hopp
-----+-------
   7 | Banan
(1 row)
```



Postgresql – uppdatera rader i en tabell

```
inge=# update tabort set hopp='Tomat' where hej=7;

UPDATE 1
```


Postgresql – ta bort rader från en tabell

```
inge=# delete from tabort where hej=7;

DELETE 1
```


Postgresql – radera en tabell

```
inge=# drop table tabort;

DROP TABLE
```

Postgresql från php

- Obs!!! Från och med nu skippar jag html-innehållet som ska vara före och efter php-blocken för att spara skärmutrymme!!!

```
<?php
    $conn_id = pg_connect ("host=localhost user=XXX dbname=sangbok")
               or die("Kunde inte kontakta databasen sangbok");
    $query = "SELECT firstname, lastname, personnr FROM personer";
    $result = pg_query ($conn_id, $query)
              or die("Kan inte hämta personer");
    while ($row = pg_fetch_row($result)) {
        print("<p>$row[0] $row[1]</p>\n");
    }
    pg_close($conn_id);
 ?>
```

SQL-sats

Exekvera SQL-sats

Kolumn-nummer

INSERT och UPDATE

- Insert och update fungerar som select, fråntaget att satsen inte returnerar något ”result-set”, endast en eventuell statuskod.

```
<?php
    $conn_id = pg_connect ("host=localhost user=XXX dbname=sangbok")
               or die("Kunde inte kontakta databasen sangbok");
    $query = "INSERT INTO personer VALUES ('Jan', 'Banan', '77-77')";
    pg_query ($conn_id, $query)
       or die("Kan inte sätta in i personer");
    pg_close($conn_id);
?>
```


Sessionshantering

- PHP-syntax
- Web och databaser
- Sessionshantering
- Mailhantering via web

Vad är en session

- Webben är i princip tillståndslös. Varje hämtning av en sida är oberoende av andra hämtningar.
- Ofta vill man dock ha en ”kontext”, där resultat av tidigare sidladdningar kan påverka framtida.
- En session skapar just en sådan kontext

Hur sessioner?

- Grundprincipen är enkel:
  - Se till att skaffa en unik identifierare som ska identifiera sessionen, en s.k. sessionsid.
  - Se till att denna sessionsid på något sätt är tillgänglig i alla script som ska utgöra sessionen, dvs att den på något sätt skickas med från klienten vid varje uppkoppling.

Hidden fields

- Den enklaste, och mest ursprungliga metoden är att lägga in denna sessionsid i ett dolt fält (hidden variable) som fylls i i varje html-formulär som utgör sessionen.

Exempel hidden fields

URL-rewriting

Det är inte nödvändigt att använda formulär överallt i en session

Man kan istället simulera formulär genom att i alla sidor som skickas till klienten, skriva om länkar så att de ser ut som de kom från ett formulär.

Exempel:

```
<a href=”foo.php”>
```

ersätts med

```
<a href=”foo.php?mitt-session-id=4711”>
```

Nackdelar med hidden fields och URL-rewriting

- Normalt kombinerar man hidden fields och URL-rewriting.
- Formulär använder hidden fields och övriga länkar använder URL-rewriting.
- Nackdelen med denna teknik är att så fort man går till en extern site så tappas sessionen.

Sessioner med Cookies

- Cookies är ett annat, bättre sätt att hantera sessioner.
- Varje server har rätt att lagra upp till 256 tecken på klientdatorn i en speciell cookiefile. Detta är dock kontroversiellt hos somliga användare.
- Det som lagras är ett antal <variabelnamn><variabelvärde>
- Oftast är det smartast att lagra endast en sessionsid på klienten, och sedan låta all övrig intressant information som servern kan vilja lagra ligga på servern i en databas där sessionsid är ”primary key” för övrig data.

Mer om cookies

Cookies används i PHP på följande sätt:

```
setcookie(“email”, ”kalle@pp.s”);
```

sätter cookievariabeln “email” till ”kalle@pp.s”.

Värdet av en cookievariabel fås med:

```
$_REQUEST[”email”];
```

Säkerhet

- En server kan endast komma åt cookies som servern själv har satt.

Mer om cookies (2)

- Värdena på cookies för en viss webplats skickas med i http-headern varje gång en request görs.
- Från servern:
  - *Content-type: text/html*
  - *Set-Cookie: foo=bar; path=/; expires Mon, 09-Dec-2002 13:46:00 GMT*
- Från klienten
  - *Content-type: text/html*
  - *Cookie: foo=bar*
- Om inget expire-date anges raderas cookien när webläsaren avslutas.

Mail

- PHP-syntax
- Web och databaser
- Sessionshantering
- Mailhantering via web

Mail i php

- Att skicka ett mail från ett php-program är enkelt. I sin enklaste form behövs en mottagaradress, en rubrikrad och en meddelandekropp

skicka webmail

```
<html>
<head>
 <title>Mailformulär</title>
</head>
<body>
 <form action="mail.php" method="POST">
   To:<input name="receiver"/>
   Subject:<input name="subject"/>
   Message:<textarea name="message"
                     cols="20"
                     rows="5">
           </textarea>
   <input type="submit" value="Skicka"/>
 </form>
</body>
</html>
```

mail.php:

```
<html>
<head>
 <title>Skicka mail</title>
</head>
<body>
<?php
    mail($_REQUEST["receiver"],
       $_REQUEST["subject"],
       $_REQUEST["message"]);
  print "<p>Mail skickat till " +
        $_REQUEST["receiver"] +
        "</p>";
?>
</body>
</html>
```


Sammanfattning

- Databashantering är enkel i PHP. Grundprincipen är:
  - Skapa en uppkoppling mot en databas
  - Exekvera ett statement
  - Om det var ett select-statement så iterera över en resultatmängd (dvs loopa över alla rader som returneras).
- Sessioner behövs för att kunna hålla ett tillstånd bland en mängd websidor
- Hidden fields och URL-rewriting kan vara enkla ibland, men oftast är cookies bättre (alla användare accepterar dock inte cookies).
- Mailhantering är mycket enkel via webben (dock något mer komplicerad för att läsa mail).