

PEAR

PEAR er et prosjekt under PHP som gir utviklere tilgang til å laste ned og bruke ferdige klasser og tilhørende funksjoner. Alle klasser skal være kodet etter en bestemt oppskrift og således være lik i bruk. Du finner retningslinjene på:

```
http://pear.php.net/manual/en/standards.php
```

Installasjon

Grunn-filene for PEAR er allerede på plass om du kjører PHP4, hvis du da ikke har installert PHP med kommandoen:

```
./configure --without-pear
```

Har du i tillegg en versjon etter 4.3 så finnes pakkeadministratoren til PEAR på systemet ditt også. For versjon 4.2.* og tidligere må du manuelt installere pakkeadministratoren.

Denne administratoren kan lage nye pakker, holde orden på installerte pakker og sjekke avhengigheter ved installasjon av nye pakker.

Windows

For versjoner etter 4.3.2 har du installasjonsskriptet allerede. Fyr opp:

```
stasjon:hvor\du\har\php\go-pear.bat
```

Svar på spørsmålene og legg til stien til PEAR i Path på din maskin. Veien dit er:

Start->Innstillinger->Kontrollpanel->System->Avansert->Miljøvariabler.

Om du sitter med en eldre versjon og trenger å laste ned skriptet går du til:

```
http://go-pear.net
```

Lagre dette som *.bat.

Linux

Brukere med versjon før 4.3 skriver i et kommandovindu:

```
$ lynx -source http://go-pear.org | php
```

Følg retningslinjene på skjermen. Ferdig!

Installere pakker

PEAR kommer med en CLI (command line installer) som gjør installasjon av nye pakke vanvittig enkelt. Du må bare vite hva pakken heter... Start et kommandovindu og skriv:

```
$ pear install <pakkenavn>
```

For å liste opp hvilke pakker som finnes:

```
$ pear list-remote-packages
```

Finne ut hvilke pakker som er oppdatert av dem du har installert:

```
$ pear lu
```

For å oppgradere:

```
$ pear upgrade <pakkenavn>
```

Full liste av kommandoer får du ved å skrive:

```
$ pear
```

Om pakker

Du får enkelt listet opp hvilke pakker som finnes, men hva gjør de enkelte? Om du trenger mer informasjon går du til:

```
http://pear.php.net/packages.php
```

Her finner du flere kategorier. Under "PHP" finnes du for eksempel "PHPDoc" som har forklaringen: "PHPDoc is an attempt to adopt Javadoc to the PHP world". Tilsvarende forklaringer finnes for de andre pakkene også.

Hjelp

Hjelpefilene er HTML-filer som kan lastes ned eller leses på nett. Dette er den beste alternativet til selvhjelp. PEAR er forholdsvis nytt og dette merkes på hjelpefilene. Utviklingen går nok forttere enn utviklingen av støttelitteratur. Opplevd dette før?

Det finnes også epostlister og nyhetsgruppe for PEAR.

Pakken DB

PEAR har nå støtte for følgende databaser:

```
mysql -> MySQL
pgsql -> PostgreSQL
ibase  -> InterBase
msql   -> Mini SQL
mssql  -> Microsoft SQL Server
oci8   -> Oracle 7/8/8i
ODBC   -> (Open Database Connectivity)
sybase -> SyBase
ifx    -> Informix
fbsql  -> FrontBase
```

Få kontakt med din database

Jeg bruker selv MySQL, og kontakter min tjener/database slik:

```
// må laste inn fila DB.php
require_once("DB.php");

$dsn = "mysql://brukernavn:passord@localhost/databasenavn";
$db = DB::connect($dsn);
```

Dette er helst informasjon du vil holde skjult for andre. Om du er litt uheldig med koden og får som resultat at PHP-skriptet vises i klartekst for brukerne vil de kunne lese all informasjon slik du har skrevet den. En fornuftig løsning er å legge all påloggingsinformasjon i en egen fil. For eksempel `database.php`. Fila kan se slik ut:

```
<?php
```

```
// sleng brukeren til et annet sted om filen blir forsøk åpnet direkte
if (ereg("database.php", $_SERVER['PHP_SELF'])) {
    Header("Location: index.php");
    die();
}
// må laste inn fila DB.php
require_once("DB.php");
$dsn = "mysql://brukeravn:passord@localhost/databasenavn";
$db = DB::connect($dsn);
// ved feil
if (DB::isError($db)) {
    die ($db ->getMessage());
}
?>
```

Det du nå gjør er å legge til en peker til fila `database.php` i de filene som skal ha kontakt med databasen. Eks.:

```
<?php
require_once("database.php");
    // litt mere kode
?>
```

Nå er det ikke så farlig om du gjør noen feil ved kodingen. Brukernavn og passord kan ikke nå leses direkte fra skjermen. Legg merke til toppen av fila `database.php`. Der er det lagt inn en test om brukeren prøver å laste fila direkte i nettleseren. Om dette skjer vil brukeren i dette tilfellet bli kastet direkte til `index.php`.

Spørring mot databasen

Når vi har fått kontakt kan vi begynne å hente ut informasjon fra basen også.

```
$sql = "SELECT * FROM filmstjerner WHERE oscarvinner = 'ja'";
$sql_result = $db->query($sql);
if (DB::isError($sql_result)) {
    die ($sql_result->getMessage());
}
```

Nå har jeg fått den uvanen at jeg skriver elementer som hører hjemme i SQL-språket med store bokstaver. Du gjør selvsagt som du selv vil!

Variabelen `$db` kommer fra fila `database.php`. Feilmeldingene som eventuelt kommer gir beskjed om at tabellen eller feltet ikke finnes.

For å utføre UPDATE, DELETE eller INSERT så bytter du bare ut SELECT-setningen med det som passer til din oppgave.

Resultater

Vi har spurt databasen om all informasjon fra tabellen "filmstjerner" der feltet "oscarvinner" er satt til "ja". Nå vil vi gjerne se resultatet også. Det finnes 2 måter å gjøre dette på, `fetchRow` og `fetchInto`. Begge måtene returnerer en rad, NULL eller en feil. `fetchInto` krever i tillegg en variabel.

```
while ($row = $res->fetchRow()) {
    $id = $row[0];
}
// eller:
while ($res->fetchInto($row)) {
    $id = $row[0];
}
```

Her finner vi navn og inntekt på filmstjernene våre:

```
$row = $sql_result->fetchRow(DB_FETCHMODE_ASSOC);
$navn = $row["navn"];
$inntekt = $row["inntekt"];
```

Dette vil vise en person da vi ikke går gjennom en løkke for å hente alle som passer til spørringen. En bedre løsning er:

```
while ($row = $sql_result->fetchRow(DB_FETCHMODE_ASSOC)) {
    $navn = $row["navn"];
    $inntekt = $row["inntekt"];
    // her må det være noe fornuftig
}
```

Nå vil listingen gå frem til resultatet er tomt.

Her har vi i tillegg `DB_FETCHMODE_ASSOC` i parentesen til `fetchRow`. Valgene er `DB_FETCHMODE_ORDERED`, `DB_FETCHMODE_ASSOC` og `DB_FETCHMODE_OBJECT`.

Dette er i samme rekkefølge, vanlig matrise, assosiativ matrise og objekt.

Om det ikke er noen som passer til spørringen, ingen resultat, så kan vi skrive en beskjed til brukeren om dette også. Vi må først teste resultatet:

```
if ($sql_result->numRows() == 0) {
    // god kode kommer her
}
```

Vi rydder opp

På slutten av fila kan du gjerne frigjøre resultatet av spørringen og koble fra databasen:

```
$sql_result->free();
$db->disconnect();
```

Regneark kan også lages

Pakken "Spreadsheet_Excel_Writer" lager regnebøker med flere regneark slik du sikkert er vant med fra Excel eller tilsvarende programmer.

Forklaringen til pakken er denne: "Spreadsheet_Excel_Writer was born as a porting of the Spreadsheet::WriteExcel Perl module to PHP.

It allows writing of Excel spreadsheets without the need for COM objects.

It supports formulas, images (BMP) and all kinds of formatting for text and cells.

It currently supports the BIFF5 format (Excel 5.0), so functionality appeared in the latest Excel versions is not yet available."

Hjelp

Du finner hjelpefila her:

```
http://pear.php.net/manual/en/package.fileformats.spreadsheet-excel-writer.php
```

Et enkelt regneark

Vi lager et enkelt regneark, så får du prøve deg frem med endringer på denne. Koden er kommentert i selve fila.

```
// vi må laste inn fila som inneholder funksjonene
require_once("Spreadsheet/Excel/Writer.php");

// bestemmer filnavnet
$filename = "/en/plass/du/har/skriverettigheter/test.xls";

// lager selve regnebogen
$workbook = new Spreadsheet_Excel_Writer($filename);

// legger til et ark og bestemmer navnet
$worksheet =& $workbook->addWorksheet('Mitt første ark');

// lar arket være liggende, Portrait blir stående
$worksheet->setLandscape();

// marger
$worksheet->setMargins(0.5);

// kolonnebredde (første kolonne, siste kolonne, bredde);
// bredde for kolonne 1 (teller starter på 0)
$worksheet->setColumn(0, 0, 18);

// bredde for kolonne 2
$worksheet->setColumn(1, 1, 34);

// bredde for kolonne 3 til og med 8
$worksheet->setColumn(2, 7, 10);
```

```
// setter inn bilde
// øverste bildekant på linje 1 (som er nummer 0)
$row = 0;
// venstre bildekant på kolonne 3
$col = 2;
// hvilket bilde som skal brukes
$bitmap = 'en/plass/med/bilder/bilde.bmp';
$worksheet->insertBitmap ($row, $col, $bitmap, $x=0, $y=0, $scale_x=1,
scale_y=1);
// lager en type formatering, $format_title blir navnet
$format_title =& $workbook->addFormat();
$format_title->setPattern(1);
// se hjelpefil for "setFgColor" og "setBgColor"
// "setFgColor" påvirker bakgrunnsfargen - "setColor" for tekst/tall
$format_title->setFgColor(43);
// fet skrift
$format_title->setBold();
// skriftfarge
$format_title->setColor('blue');
// en formattering til...
$format_grey =& $workbook->addFormat();
// trenger egentlig ikke denne da 1 er standard
$format_grey->setPattern(1);
// ikke bare tallverdier her - kan også bruke ('black'), ('red')
$format_grey->setFgColor(43);
// justering i cellen
$format_grey->setAlign('center');
// tallene skal ha tusenskilte
$format_grey->setNumFormat(3);
// nå skriver vi til regnearket, 0, 0 blir første celle (A1)
$worksheet->write(0, 0, 'En lang tittel', $format_title);
// på neste rad skriver vi
$worksheet->write(1, 0, 'Enkel undertittel', $format_title);
// den lange tittelen vår skal gå over 3 celler
```

```
// merge (første rad, første kolonne, siste rad, siste kolonne)
$worksheet->mergeCells (0, 0, 0, 2);
$worksheet->mergeCells (1, 0, 1, 2);
// skriver noen tall
$worksheet->write(2, 0, 9000, $format_grey);
$worksheet->write(3, 0, 550, $format_grey);
// fila avsluttes
$workbook->close();
```

Bra! Ble litt svett nå?

Se litt gjennom dokumentasjonen på nett. Du har store muligheter innen formatering av celler. Topp- og bunntekst kan skrives. Repetering av rader og kolonner er mulig. Vi har sett på `$worksheet->write` for å skrive til regnearket, men spesielle funksjoner finnes for å skrive tall, formler, URL, tekst og notat.

En liten godbit

Jeg tar med en liten godbit til slutt som egentlig ikke har noe med PEAR å gjøre, men kan i mange tilfeller være kjekk å ha når vi lager dokumenter direkte.

```
// Last ned filen til lokal bruker
// navnet på fila som nettopp ble laget (vanligvis $variabelnavnet)
$filename = "der/du/har/lagret/fila.xls"
// størrelsen på fila
$size = filesize($filename);
// navnet fil fila og ikke hele banen
$file = basename ($filename);
// du trenger dette for å kunne få en "lagre som"
header("Content-Type: application/save");
header("Content-Length: $size");
header("Content-Disposition: attachment; filename=$filename");
header("Content-Transfer-Encoding: binary");
ReadFile($filename);
// fila skal nå kunne lagres lokalt til bruker
```