Denne guide er oprindeligt
udgivet på Eksperten.dk

COMPUTERWORLD

Kontrol-strukturer i PHP

Denne artikel gennemgår kontrolstrukturer i PHP. 'if', 'switch', 'while' og 'for' bliver gennemgået. Den forudsætter lidt grundlæggende kendskab til PHP. *Opdateret 7-10-04*

Skrevet den **04. Feb 2009** af **taskmgr** I kategorien **Programmering / PHP** | ★★☆☆☆

Update 7-10-04: URL'en i bunden er rettet.
Update 30-08-04: Artiklen er nu gratis. Det samme gælder min artikel om arrays.
Update 26-04-04: Tilføjet info omkring 'default' i switch()

Kontrol-strukturer (blandt andet også kaldt "Control-statements") er metoder til at styre eksekveringen af en applikations forskellige dele. Syntaksen for kontrol-strukturer kan sammenlignes med det regulære engelske sprog, og forståelsen kræver praktisk talt kun en smule logisk sans. Jeg vil her komme ind på nogle af de kontrol-strukturer der kan benyttes i PHP.

'if'-konstruktioner

De hyppigst benyttede konstruktioner i PHP og programmering generelt, er baseret på 'if'-statements. Navnet afspejler præcist formålet.

Vi starter med syntaksen:

```
//eksempel 1
if (expr) {
    statement
}

//eksempel 2
if (expr) {
    statement
} else {
    statement
}

//eksempel 3
if (expr1) {
    statement
} elseif (expr2) {
    statement
} else {
    statement
}
```

De tre ovenstående eksempler er herunder beskrevet med en abstraktion på godt dansk:

```
//eksempel 1
hvis (dette udtryk er sandt) {
    ... så udfør dette
}

//eksempel 2
hvis (dette udtryk er sandt) {
    ... så udfør dette
} ellers {
    ... så udfør dette
}

//eksempel 3
hvis (dette udtryk er sandt) {
    ... så udfør dette
} ellers hvis (dette udtryk er sandt) {
    ... så udfør dette
} ellers {
    ... så udfør dette
}
```

Vi kan nu bruge et eksempel fra hverdagen, hvor jeg vil ud og købe en bil:

```
if (min_kapital > bilens_pris) {
    så køber jeg bilen med det samme
} elseif (banken_vil_laane_mig_pengene) {
    så låner jeg pengene
    og køber bilen
} else {
    så finder jeg et job
}

Denne struktur kan i PHP skrives således:

<?php
$min_kapital = 30000;
$bilens_pris = 40000;
$banken_vil_laane_mig_pengene = FALSE;

if ($min_kapital > $bilens_pris) {
    echo "Jeg køber bilen";
} elseif ($banken_vil_laane_mig_pengene) {
    echo "Jeg låner pengene";
    echo "<br />\n";
    echo "Jeg køber bilen";
} else {
    echo "Jeg finder et job";
}
?>
```

Ovenstående kontrol-struktur kan næppe bruges til noget i praksis, men det skulle give dig en

fornemmelse af hvad if-sætninger bruges til.

'switch'-konstruktioner

Brugen af en switch-sætning kan sammenlignes med en serie af if-statements der bruger det samme udtryk. Du vil måske sammenligne den samme variabel med mange forskellige værdier og eksekvere et forskelligt stykke kode, alt efter hvilken værdi variablen har. Dette kan naturligvis gøres med en serie af if og elseif, hvilket jeg efterfølgende vil vise, men du kan med fordel bruge switch i stedet. Det er både nemmere og mere overskueligt.

Her er to eksempler som gør det samme - det ene med en serie af if og elseif og det andet med en switch.

```
<?php

// ved brug af if og elseif:
if($i == 1){
    echo "i er lig med 1";
}elseif($i == 2){
    echo "i er lig med 2";
}elseif($i == 3){
    echo "i er lig med 3";
}

// ved brug af switch:
switch($i){
case 1:
    echo "i er lig med 1";
    break;
case 2:
    echo "i er lig med 2";
    break;
case 3:
    echo "i er lig med 3";
    break;
}

?>
```

Det er vigtigt at forstå hvordan en switch bliver eksekveret hvis man vil undgå fejl. Statements i en switch-sætning bliver kun udført hvis der findes en 'case' der passer til det angivne udtryk ($i her). Altså - hvis du i ovenstående sætter $i til f.eks. 4, vil ingen af dem udføres. Hvis der derimod findes en case som passer til $i, så fortsætter PHP med at eksekvere statements indtil slutningen af switch-sætningen, eller indtil den møder et 'break' statement. Hvis man derfor ikke tilføjer et break statement i slutningen af en case, så bliver de eventuelle efterfølgende cases også udført.

Samme eksempel uden 'break':

```
<?php

switch($i){
case 1:
    echo "i er lig med 1";
case 2:
```

```
    echo "i er lig med 2";
case 3:
    echo "i er lig med 3";
}

?>
```

Her vil PHP eksekvere alle tre statements hvis $i er lig med 1. Hvis $i er lig med 2, vil kun 'case 2' og 'case 3' blive udført.

Hertil kan man tilføje en særlig 'default' case. Statements i denne case vil blive udført hvis ingen af de andre cases passer:

```
<?php
switch($i){
case 1:
    echo "i er lig med 1";
    break;
case 2:
    echo "i er lig med 2";
    break;
case 3:
    echo "i er lig med 3";
    break;
default:
    echo "i er forskellig fra 1, 2 og 3";
}
?>
```

I ovenstående vil der udskrives: "i er lig med 1/2/3" hvis $i er lig med 1, 2 eller 3. Hvis $i ikke matcher en af de tre cases, vil statements under 'default' blive udført.

'while'-loops

Et loop er en sekvens af instruktioner som gentages indtil en bestemt betingelse er opfyldt. Loops er, ligesom if-konstruktioner, en fundamental del i PHP og programmering generelt. I et typisk loop bliver en proces udført, så som at hente data og modificere disse, indtil en betingelse ikke længere er opfyldt. Loops bliver ofte kaldt for 'løkker' på dansk.

While-loops er forholdsvist simple. De beder PHP om at eksekvere de(t) statement(s) som er bygget ind i konstruktionen gentagelsesvist, så længe at while udtrykket er sandt. Udtrykkets værdi (om det er sandt eller falsk) tjekkes hver gang i begyndelsen af loopet, for at bestemme om det skal eksekveres endnu en gang.

Hertil har jeg ligeledes lavet et par eksempler:

```
// Den grundlæggende syntax:
while (expr){
    statement
}

// En abstraktion på dansk:
```

```
så længe at (dette udtryk er sandt) {
    ... så udfør dette
}
```

```
// Det første eksempel:
<?php

$i = 1;
while($i <= 5){
    echo $i;
    $i++;
}

?>
```

Dette vil udskrive: "12345". Første gang løkken bliver kørt er udtrykket sandt, fordi $i er lig med 1 og derfor mindre eller lig med 5. Løkken fortsætter indtil $i er lig med 6 og derfor hverken er mindre eller lig med 5.

```
// Eksempel 2:
<?php

$array = array('a', 'b', 'c');
$i = 0;
while($i < count($array)){
    echo $array[$i];
    $i++;
}

?>
```

I eksempel nr. 2 vil "abc" blive udskrevet. Her tjekkes der om $i er mindre end summen af elementer i et array ($array). I samme forbindelse bruger vi $i inde i løkken til at definere hvilken nøgles værdi der skal udskrives.

'for'-loops

For-loops er en del mere komplekse, men kan sådan set betragtes som en sammenskrivning af mit første eksempel på en while-løkke.

```
// Syntax:
for (expr1; expr2; expr3){
    statement
}
```

Mit første eksempel på en while-løkke kan 'oversættes' til en for-løkke således:

```
<?php

for($i = 1; $i <= 5; $i++){
    echo $i;
}

// Igen bliver der udskrevet: "12345"
?>
```

Det første udtryk ($i = 1) bliver uden betingelser eksekveret én gang i starten af løkken.

Før hver gentagelse bliver udtryk nr. 2 ($i <= 5) tjekket. Løkken bliver kørt igen hvis dette udtryk er sandt.

I slutningen af hver gentagelse bliver det sidste udtryk ($i++) eksekveret.

Lidt at slutte på

Dette var så mit bud på en grundlæggende gennemgang af nogle vigtige kontrol-strukturer i PHP. Som med så meget andet, gælder det også her at man skal kaste sig ud i tingene hvis man vil lære noget. Derfor foreslår jeg (igen) at du prøver dig frem, evt. ud fra mine eksempler.

PHP manualens afsnit om emnet kan findes på
http://dk2.php.net/manual/en/language.control-structures.php

Kommentar af otis d. 05. Oct 2005 | 1

God og beskrivende artikel

Kommentar af minau d. 09. Oct 2004 | 2

Kommentar af webcreator d. 01. Jun 2004 | 3

En rigtig fin artikel. Bør som nævnt læses af enhver begynder. Udemærket at "Defaults" nu er beskrevet i afsnittet om switches. Faktisk har jeg selv siddet fast i en switch engang, fordi jeg ikke kendte til nødvendigheden af Break;. Så jeg har kun ros til overs for dit arbejde.

Kommentar af qtax87 (nedlagt brugerprofil) d. 19. Apr 2004 | 4

Synes det er godt beskrevet, men som phpwiz siger ville det være smart at nævne om default, ellers "keep going!"

Kommentar af alleykat d. 18. May 2004 | 5

God, basic gennemgang af de grundliggende php kontrolstrukturer, forklaret på ordentligt dansk. Overflødig for erfarne, men ABSOLUT en must-read for begyndere!

Kommentar af jbob d. 10. May 2005 | 6

Glinmerende og overskueligt forklaret! At det er basic har da ikke noget med kvaliteten af artiklen at gøre. Det vil være det samme som at kritisere en børnebog for at være for simpel.

Kommentar af phpwiz d. 16. Apr 2004 | 7

der er lidt mangler, bl.a. kunne du godt fortælle om muligheden for "default" i switch'es

Kommentar af xalinx d. 25. Apr 2004 | 8

Meget basic