Denne guide er oprindeligt
udgivet på Eksperten.dk

COMPUTERWORLD

Caching af sider med PHP

Denne artikel kommer ind på teknikker til at mindske belastningen af ens database ved hjælp af caching.

Skrevet den **10. Feb 2009** af **philip** I kategorien **Programmering / PHP** | ★★☆☆☆

Målgruppe
Denne artikel er skrevet til ejeren af det halvstore website. Der er ikke den store ide i at benytte caching på sine websider, hvis siderne ikke bliver vist ofte nok, til at der er stor gevinst, ved at benytte caching. Der forventes en grundlæggende viden omkring PHP og dets grundlæggende funktioner.

Caching
Ved at søge på Google ("define:caching") får man følgende definition:
" The process of storing frequently used results from a request to the Web server locally for quick retrieval, until it is time to refresh the information. " (http://plutonium.cs.umanitoba.ca/DB2Docs/db2t0/db2t0.htm).
Således benyttes caching, til at mindske antallet af udtræk fra en database, ved at gemme udtrækkene lokalt på serverens harddisk. Dette vil give betydelige hastighedsforbedringer, da en fil fra serverens harddisk, er langt hurtigere at hente, end ting fra databasen.
Caching kan bruges til mange forskellige ting, men i denne artikel fokuserer jeg på de forskellige typer af caching man kan benytte, fremfor at komme med kodeeksempler som kan illustrere hastighedsforskellen.

Forskellige caching typer
I denne artikel vil jeg komme ind på to forskellige caching typer: Løbende opdatering og opdatering ved nyt indhold.

Løbende opdatering
Den caching type, som jeg har valgt at benævne 'løbende opdatering' dækker over en opdatering som ikke automatisk foretages når der lægges nyt indhold ind. Istedet bestemmes det om cacheet skal opdateres ud fra hvor lang tid det er siden det sidst blev opdateret, fremfor hvornår der kom nyt indhold.
Derfor kan brugere af siden komme ud for, at blive præsenteret for gammelt indhold, hvis cacheet ikke er opdateret.

Nedenfor ses koden til at lave denne form for caching:

```
<?php
#indstiller hvor gammelt cacheet må blive før det refreshes
$levetid = 3600 #En time
#sæt filen
$fil = "cache.htm";


# Hvis filen ikke eksisterer ELLER filen er over $levetid sekunder gammel, laves der et nyt cache
if (!is_file($fil) || filemtime($fil)+$levetid<time()) {


  #start output bufferen
  ob_start();


  ## Her placeres den PHP kode hvis output skal caches!
```

```
  # Åbner filen til at skrive
  $f = fopen($fil, 'w');
  # Skriver
  fwrite($f, ob_get_contents());
  # Lukker filen
  fclose($f);
  # Udskriver outputtet til browseren
  ob_end_flush();

#Hvis de førnævnte betingelser ikke er opfyldt inkluderer vi bare filen fra før.
} else {
  include($fil);
}
?>
```

Imidlertid er den førnævnte metode ofte ikke den mest hensigtsmæssige, i og med at de færreste webmastere kan acceptere at deres indhold er forsinket. Derfor kan man omstrukturere koden lidt (til en funktion), som så skal kaldes når der er nyt indhold. Dette gøres således:

```
<?php
#sæt filen
$fil = "cache.htm";


# Hvis filen ikke eksisterer ELLER filen er over $levetid sekunder gammel, laves der et nyt cache
function refreshcache () {
  #start output bufferen
  ob_start();


  ## Her placeres den PHP kode hvis output skal caches!


  # Åbner filen til at skrive
  $f = fopen($fil, 'w');
  # Skriver
  fwrite($f, ob_get_contents());
    # Lukker filen
    fclose($f);
    # Udskriver outputtet til browseren
    ob_end_flush();

#Hvis de førnævnte betingelser ikke er opfyldt inkluderer vi bare filen fra før.
} else {
    include($fil);
}

}
?>
```

Koden er nu opbygget som en funktion, 'refreshcache'. Når denne funktion bliver kaldt vil cacheet blive

opdateret, og det nyeste indhold vil blive præsenteret for brugeren som ser på siden derefter. Denne funktion bør blive kaldt ummilbart efter indholdet er indsat i databasen, eksempel nedenfor:

```
<?php
mysql_query("INSERT into tabel (overskrift, tekst) values ('$overskrift','$tekst')");
cacherefresh();
?>
```

Hvad caching ikke kan
Caching har mange muligheder. Dog kan det gøre koden svær at gennemskue. Desuden kræver det langt mere avanceret caching, hvis man eksempelvis skal have forskelligt indhold, alt efter om brugeren er logget ind eller ej.

Slutbemærkninger
Det er vigtigt at huske, at man skal chmodde mappen som cache filen ligger i, til 777, for at give scriptet skriverettigheder.

Kommentarer og forslag til udvidelser eller forbedringer er yderst velkomne.

Nyttige links
http://dk.php.net/manual/en/ref.outcontrol.php

Kommentar af jensgram d. 22. Feb 2005 | 1

Alt i alt OK, men jeg synes det løber en anelse ud i sandet til sidst.

Kommentar af dustie d. 24. Aug 2005 | 2

Kommentar af imago-dei d. 24. Feb 2005 | 3

Det kunne være sjovt at se nogle tidsmålinger for at se hvor meget mere effektiv koden er med caching i forhold til uden caching.

Jeg har selv engang i tidernes morgen lavet et projekt ang. caching (udført på en anden måde end ovenstående) hvor jeg opnåede mellem 2 og 30 gange mere effektiv kodeafvikling. Tip: man kan komprimere koden med gzip eller lignende, som browsere kan læse. Dermed kan man optimere visningstiden for en bruger specielt hvis vedkommende er på en langsom netforbindelse.

Jeg vil dog sige til artiklen at det er tvivlsomt om det er hurtigere at hente fra en fil på harddisken end fra databasen. Det kommer naturligvis an på hvor mange records du har i databasen og hvor komplicerede dine query's er.

Er det forresten ikke muligt at cache til en variabel istedet for til disk? Det ville være væsentligt hurtigere at cache i RAM frem for en fil.

Kommentar af simonhans73 d. 11. Feb 2006 | 4

Kommentar af alister_crowley d. 22. Feb 2005 | 5

/me like :D