

Implementarea unui sistem de “e-mail queueing” pentru PHP si sendmail

Autor: Vlad Roman vlad@javascript.ro
Ultima modificare: Tuesday, April 24, 2007

Descrierea problemei	2
Solutia	2
<i>Procesarea listei de asteptare (crontab)</i>	2
<i>Baza de date (MySQL)</i>	3
<i>Scriptul PHP:</i>	3
<i>Introducerea sistemului in siteul existent</i>	6

Descrierea problemei

O data cu luptele acerbe pe care le duc companiile care ofera servicii de hosting impotriva spam-ului, din ce in ce mai multe servere incep sa impuna limitari in privinta numarului de mesaje e-mail care pot fi trimise intr-un anumit interval de timp.

Desi exista nenumarate solutii moderne de MTA (mail transport agent), precum [Postfix](#), [Exim](#), [Ipswitch, Inc.](#), IMail sau [gmail](#), multe servere folosesc in continuare popularul [sendmail](#), care a fost dezvoltat la inceputul anilor '80.

Un neajuns al acestei aplicatii este lipsa unui sistem de "queueing", unui sistem care sa pastreze mesajele email intr-o memorie temporara, trimitandu-le in functie de capacitatea/incarcarea server-ului.

Astfel, fara un sistem de genul asta, un site gazduit de un server limitat va pierde toate mesajele trimise peste cota admisa.

Solutia

Cum majoritatea site-urilor nu trimit email-urile distribuite uniform de-a lungul unui anumit interval de timp, mesajele care depasesc limita impusa de server pot fi salvate pentru a fi trimise in urmatorul interval orar mai liber. Voi descrie in continuare cum poate fi realizat cu usurinta acest lucru.

Un obiectiv important al implementarii pe care urmeaza sa o descriu este posibilitatea aplicarii solutiei in cadrul unui site existent, cu minimum de efort. Astfel codul existent va suferi modificari numai in locurile in care este folosita functia PHP [mail\(\)](#).

Pentru a putea realiza acest lucru se va crea o functie "wrapper" care va inlocui in mod transparent functia nativa PHP mail, pastrandu-i intocmai parametrii.

Aceasta functie va face urmatoarele lucruri:

1. va introduce in lista de asteptare (baza de date) datele mesajului (parametrii primiti in momentul apelarii functiei "mail").
2. va verifica daca a fost atins numarul maxim de mesaje trimise in ultimul interval de timp.
3. daca numarul maxim nu a fost atins inca, va trimite mesajul in momentul respectiv.

Procesarea listei de asteptare (crontab)

Pentru reincercarea trimiterii mesajelor salvate in lista de asteptare se va folosi urmatoarea metoda: rularea frecventa a un script PHP care va face urmatoarele lucruri:

1. va extrage din baza de date mesajele care nu au fost inca procesate
2. va trimite mesajele pana cand limita va fi atinsa

Pe sistemele de tip Unix se va putea folosi comanda "[crontab](#)" pentru programarea executiei scriptului acesta, la un anumit interval de timp.

Exemplu de setare "crontab" care executa scriptul la un interval de o ora:

Baza de date (MySQL)

Codul SQL pentru crearea tabelului MySQL in care se va tine lista de asteptare cu mesaje:

```
CREATE TABLE `email_queue` (  
  `id` mediumint(9) NOT NULL auto_increment,  
  `mail_to` varchar(255) NOT NULL,  
  `subject` varchar(255) NOT NULL,  
  `message` blob NOT NULL,  
  `additional_headers` varchar(255) NOT NULL,  
  `additional_parameters` varchar(255) NOT NULL,  
  `date` datetime NOT NULL,  
  `processed` smallint(1) NOT NULL default '0',  
  `retries` smallint(1) NOT NULL default '0',  
  PRIMARY KEY (`id`)  
);
```

Descrierea campurilor:

date – data introducerii mesajului in lista de asteptare sau data trimiterii in cazul in care campul processed are valoarea “1”

processed – prin valoarea “1” indica faptul ca mesajul a fost trimis

retries – numarul de incercari esuate de a trimite mesajul

Scriptul PHP:

```
<?php  
class queuedMail {
```

Setarea informatiilor despre limita impusa de server:

```
function queuedMail($max, $interval, $maxRetries = 3) {  
    $this->quota['count'] = $max; //emails limit  
    $this->quota['interval'] = $interval; //mysql time interval  
    $this->maxRetries = $maxRetries;  
    $this->deleteAfterRetries = false;  
}
```

“Interfatarea” functiei native PHP “mail”:

```

function mail($to, $subject, $message, $additional_headers = false,
$additional_parameters = false) {
    $this->status = false;
    //add to queue
    $id = $this->queue(array(
        "mail_to" => $to,
        "subject" => $subject,
        "message" => $message,
        "additional_headers" => $additional_headers,
        "additional_parameters" => $additional_parameters,
        "PROCESSED" => 0
    ));
    //send it, if possible
    if ($this->canSend()) {
        return $this->send($id);
    } else {
        $this->status = "queued";
        return true;
    }
}

```

Adaugarea unui mesaj nou in lista de asteptare:

```

function queue($params) {
    mysql_query("INSERT INTO `email_queue` ( MAIL_TO, SUBJECT,
MESSAGE, ADDITIONAL_HEADERS, ADDITIONAL_PARAMETERS, DATE,
PROCESSED ) VALUES ( '". $params['mail_to']. "', '". $params['subject']. "',
'". $params['message']. "', '". $params['additional_headers']. "',
'". $params['additional_parameters']. "', NOW(), 0 )");
    $rs = mysql_query("SELECT LAST_INSERT_ID()");
    $row = mysql_fetch_row($rs);
    return $row[0];
}

```

Verificarea limitei pentru intervalul de timp current:

```

function canSend() {
    $rs = mysql_query("SELECT COUNT(id) FROM `email_queue` WHERE
((`date` >= DATE_SUB(NOW(), INTERVAL '". $this->quota['interval']. "') AND (`date`
<= NOW()) AND (PROCESSED = 1) ) LIMIT 0, 1");
    $row = mysql_fetch_row($rs);
    if ($row[0] <= $this->quota['count']) {
        return true;
    } else {
        return false;
    }
}

```

```
}
```

Procesarea listei de asteptare:

```
function processQueue() {
    $rs = mysql_query("SELECT id FROM `email_queue` WHERE
PROCESSED=0 ORDER BY ID ASC");
    while ($record = mysql_fetch_assoc($rs)) {
        if ($this->canSend()) {
            $this->send($record['id']);
        } else {
            return false;
        }
    }
}
```

Trimiterea efectiva a unui mesaj:

```
function send($id) {
    $rs = mysql_query("SELECT * FROM `email_queue` WHERE (id =
\".$id.\") LIMIT 1 ");
    $record = mysql_fetch_assoc($rs);
    $rs = mail($record['mail_to'], $record['subject'], $record['message'],
$record['additional_headers'], $record['additional_parameters']);
    if ($rs) {
        mysql_query("UPDATE `email_queue` SET PROCESSED=1
WHERE (id = \".$id.\")");
        $this->status = "sent";
        return true;
    } else {
        if ($record['retries'] < $this->maxRetries && !$this-
>deleteAfterRetries) {
            mysql_query("UPDATE `email_queue` SET
DATE=NOW(), RETRIES=retries+1 WHERE (id = \".$id.\")");
        } else {
            mysql_query("DELETE FROM `email_queue` WHERE
(id = \".$id.\")");
        }
        $this->status = "failed";
        return false;
    }
}
```

```
}
```

```
?>
```

Introducerea sistemului in siteul existent

Implementarea propriu-zisa consta in doi pasi:

1. Includerea scriptului in site si initializarea lui. Exemplu:

```
<?php  
require_once("qmail.php");  
$qmail = new queuedMail($limit = 200, $interval = "1 HOUR", $maxRetries = 3);  
?>
```

2. Modificarea functiei "mail()". Exemplu:

- se va inlocui

```
mail($to, $subject, $message, $additional_headers, $additional_parameters);
```

- cu:

```
$qmail->mail($to, $subject, $message, $additional_headers,  
$additional_parameters);
```

Modificarea reprezentand defapt simpla adaugare a unui text invariabil, aplicarea se poate face cu ajutorul unei aplicatii (editor de cod) care are functia de a cauta si inlocui in mod automat text, in toate fisierele dintr-un anumit director.