Building an advertising system with PHP, Part 3

By Dennis Pallett

Introduction

Welcome to part 3 of the "Building an advertising system with PHP" series. In the previous parts (part 1 and part 2) I have shown you how to build your own advertising system using PHP and JavaScript. We've also added two extra features to our ad system and in part 2 we built a page to manage the ads as well. If you haven't read either part yet, I highly recommend doing so before reading this part.

In this part, the final part of the series, I will show you how to first track all kinds of statistics on each ad, and after that display neat graphs using PHP/SWF Charts. Let's get started.

Gathering Ad Statistics

Gathering ad statistics is actually quite easy, and all we need to add is some simple database code to the ad.php file. But first, let's add a new table to the advertising database, which looks like this:

```
CREATE TABLE `traffic` (
  `trafficid` mediumint(10) NOT NULL AUTO_INCREMENT,
  `datetimestamp` int(10) NOT NULL DEFAULT '0',
  `type` enum('click','view') NOT NULL DEFAULT 'view',
  `ipaddress` varchar(255) NOT NULL DEFAULT '',
  `ad` mediumint(10) NOT NULL DEFAULT '0',
  PRIMARY KEY  (`trafficid`)
) TYPE=MyISAM AUTO_INCREMENT=1 ;
```

We only store a few basic things, like the IP address of the visitor and the current date/time. This table will be used for both views and clicks, hence the need for the type field. All we need to do now is to add some simple code that will log a view. Copy and paste the code below somewhere in the ad.php file (if you don't know what the ad.php file is or don't have it yet, download the demo source of the previous part).

```
// Insert new view into traffic table
 $datetime = time();
 $ipaddress = ss($_SERVER['REMOTE_ADDR']);

$db->query ("INSERT INTO traffic (ad, datetimestamp, type, ipaddress) VALUES ({$ad['adid']}, $datetime, 'view', '$ipaddress')");
```

Now all the impressions of each ad are being logged, but the clicks aren't logged yet. To be able to log those, we have to use another script that will log the click, and then redirect the visitor to the URL of the ad. The

Copyright (c) 2006 PHPit.net

script looks something like this:

```
<?php
 include 'mysql.php';

// Get ad id
 if (isset($_GET['id'])) {
        $id = intval($_GET['id']);
 } else {
        die('No ad specified');
 }

// Get ad
 $ad = $db->query_first ("SELECT adid, link FROM ad WHERE adid = $id");

if ($ad == false) {
        die('No ad specified');
 }

// Log click
 $datetime = time();
 $ipaddress = ss($_SERVER['REMOTE_ADDR']);

$db->query ("INSERT INTO traffic (ad, datetimestamp, type, ipaddress) VALUES ({$ad['adid']}, $datetime,
'click', '$ipaddress')");

// Redirect
 header ("Location: " . $ad['link']);
 ?>
```

All this script does is first check whether the ad exists, and if it does, logs the clicks and redirects the visitor to the ad link.

We also have to customize the ad system to use the redirect script instead of using the direct link, like so:

```
// Display ad:
 header ("Content-Type: text/javascript");
 echo "document.write ('<a href=\"redir.php?id={$ad['adid']}\"><img src=\"$image\" alt=\"Ad Banner\" style=\"
border:0;\" /></a>');";
```

The ad is now linked to our redirect script (redir.php) instead of the direct link. Now that all the traffic is being gathered, we have to write the next part of the statistics package: the analyzing part.

Analyzing the traffic

At the moment the traffic isn't really helpful, because we aren't getting any real statistics. Things like CTR (Click-Through-Rate) and impressions are important. We need to write another script that will analyze the raw traffic, and change it into something we can use.

This script has to do the following things:
 - Collect the number of impressions of each day, week, and month
 - Collect the number of clicks of each day, week, and month

Copyright (c) 2006 PHPit.net

- Calculate CTR for each day, week, and month
- Calculate total number of impressions, clicks and average CTR so far

Before we start writing any code, let's first create the table that will hold our stats:

```
CREATE TABLE `stats` (
   `statsid` mediumint(10) NOT NULL AUTO_INCREMENT,
   `type` enum('day','week','month','total') NOT NULL DEFAULT 'day',
   `day` int(5) NOT NULL DEFAULT '0',
   `week` int(5) NOT NULL DEFAULT '0',
   `month` int(5) NOT NULL DEFAULT '0',
   `year` int(5) NOT NULL DEFAULT '0',
   `views` int(5) NOT NULL DEFAULT '0',
   `clicks` int(5) NOT NULL DEFAULT '0',
   `ctr` decimal(5,) NOT NULL DEFAULT '0',
   `ad` mediumint(10) NOT NULL DEFAULT '0',
   PRIMARY KEY  (`statsid`),
   KEY `ad` (`ad`)
 ) TYPE=MyISAM AUTO_INCREMENT=1 ;
```

Let's start with the actual script now. Obviously, the first thing it must do is get all the traffic:

```
// Get traffic
 $traffic = $db->sql_query ("SELECT trafficid, datetimestamp, type, ipaddress, ad FROM traffic");
 if (count($traffic) == ) {     die('No traffic logged'); }
```

The next step is to loop through all the traffic records, and actually do the analyzing. What the script will be doing is examining each individual record, and then updating the proper records. For example, if it sees a traffic record with the following values:
- date/time: 17/12/2005
- type: view
- ad: 1

Then it will update the following records in the stats table (all for ad 1):
- December 2005 (impressions, CTR)
- week 51 in 2005 (impressions, CTR)
- day 17/12/2005 (impressions, CTR)
- total (impressions, CTR)

The analyze script looks like this:

```
$delete = '(0';
 foreach ($traffic as $t) {
        // Get values
        $day = date('d', $t['datetimestamp']);
        $week = date('W', $t['datetimestamp']);
        $month = date('n', $t['datetimestamp']);
        $year = date('Y', $t['datetimestamp']);
        $ad = $t['ad'];

        // Which type?
        if ($t['type'] == 'view') {
                $views = 'views + 1';
```


Copyright (c) 2006 PHPit.net

```
            $clicks = 'clicks';
        } else {
            $views = 'views';
            $clicks = 'clicks + 1';
        }

        records_exist($day, $week, $month, $year, $ad);

        // Update day record
         $db->query ("UPDATE stats SET views = $views, clicks = $clicks, ctr = ((clicks/views)*100) WHERE
type = 'day' AND day = $day AND month = $month AND year = $year AND ad = $ad");

        // Update week record
         $db->query ("UPDATE stats SET views = $views, clicks = $clicks, ctr = ((clicks/views)*100) WHERE
type = 'week' AND week = $week AND year = $year AND ad = $ad");

        // Update month record
         $db->query ("UPDATE stats SET views = $views, clicks = $clicks, ctr = ((clicks/views)*100) WHERE
type = 'month' AND month = $month AND year = $year AND ad = $ad");

        // Update total record
         $db->query ("UPDATE stats SET views = $views, clicks = $clicks, ctr = ((clicks/views)*100) WHERE
type = 'total' AND ad = $ad");

}
 $delete .= ')';
```

All it does is update each record with the new values (either a new view or a new click), and it lets MySQL calculate the new CTR. Another thing worth mentioning is the records_exist() function, which is used to check whether all the records that will be updated already exist, and if necessary creates them.

One last thing the analyze script must do is delete the traffic data which has been analyzed, otherwise you might get duplicate data. The following code does what we want:

```
$db->query ("DELETE FROM traffic WHERE trafficid IN $delete");
```

It uses the $delete var created in the record loop.

The best way to use this analyze script is to create a cronjob (or use some other scheduling program), and have this script run every 6 hours or so. That means that it won't have to analyze a lot of records at the same time (thus not causing your server to slow down), and your graphs will be up-to-date. If you want to have a look at my analyze script, click here to download the full analyze script.

Let's move on to the final part of the tutorial: actually creating some neat graphs. If you want to following along with this part, you might want to download some fake data by clicking here, or else you won't be getting any useful graphs.

Creating the graphs

First head over to the homepage of PHP/SWF Charts and have a look at the demo's and tutorials to get a good feeling of how it looks and how it works.

Copyright (c) 2006 PHPit.net

The first graph we'll be creating is a simple one: the impressions and clicks plotted out against the months. For this we need two scripts: one that will display the chart (charts.php), and another to actually create the monthly chart (chart-month.php).

The code for the charts.php file is quite simple, and only contains some basic PHP:

```
<?php
 include '../charts.php';

echo InsertChart ( "../charts.swf", "../charts_library", "chart-month.php?ad=1&" . time(), 500 );
 ?>
```

What this code does is include the PHP/SWF Charts library, and then uses the insertChart function to insert the monthly chart. One thing you might notice is that I use the time() function to reference the chart. This is to prevent any browsers from caching the chart.

The chart-month.php file must first select all the data from the last 6 months, and then transform it into the proper data form that PHP/SWF Charts needs. If you have a look at the documentation you'll notice that it needs to be in a really specific form.

The code that does what we want looks like this:

```
<?php
 include 'mysql.php';
 include "../charts.php";

// Get ad id
 if (!isset($_GET['ad'])) { no_chart(); }
 $ad = intval($_GET['ad']);

// Get data for last six months
 $months = $db->sql_query ("SELECT year, month, views, clicks, ctr FROM stats WHERE ad = $ad AND
type = 'month' ORDER BY month DESC LIMIT 0,6");

if (count($months) == ) {
        no_chart();
 }

$months = array_reverse($months);

// Create chart data
 $monthnames = array('');
 $clicks = array('Clicks');
 $views = array('Impressions');

foreach ($months as $month) {
        $monthnames[] = date('F', mktime(, , , $month['month'], 1, $month['year']));
        $clicks[] = $month['clicks'];
        $views[] = $month['views'];
 }

$chart['chart_data'] = array($monthnames, $clicks, $views);
 ?>
```


Copyright (c) 2006 PHPit.net

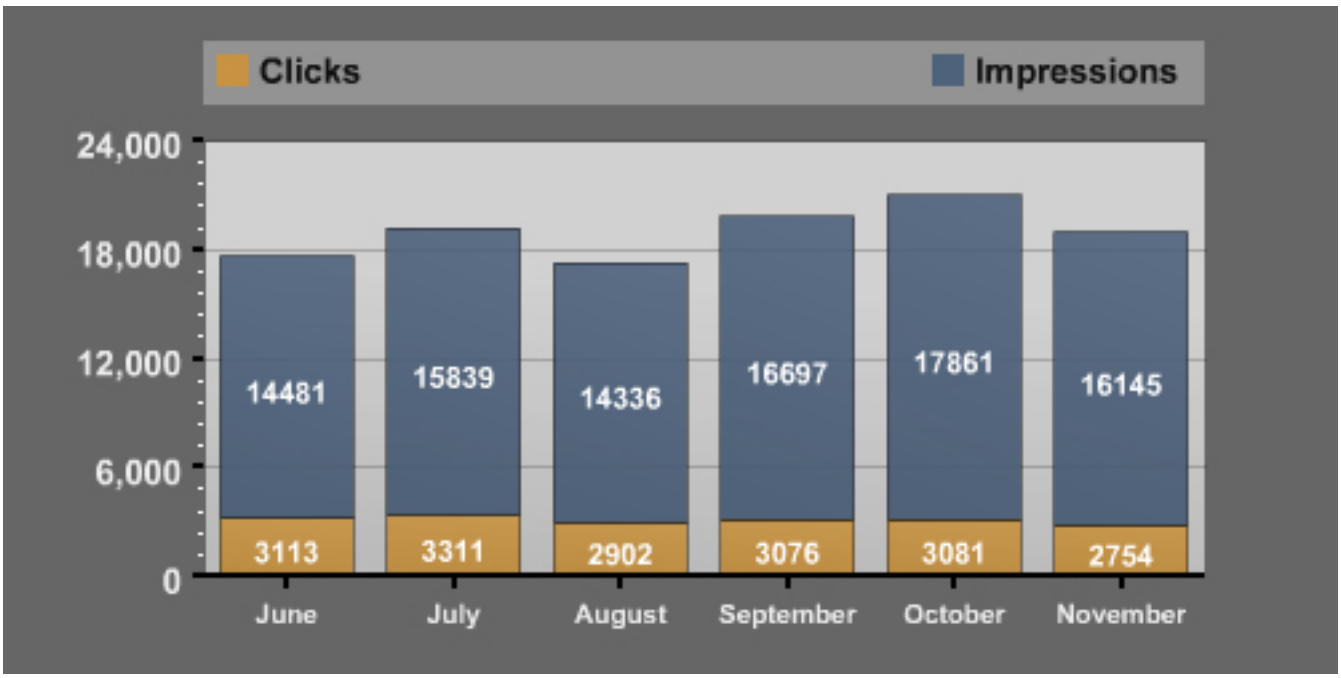
The above code first selects all the data of the last six months, and then transforms it into the right form, just like we want. Now add some formatting options:

```
$chart['axis_value'] = array('color' => 'FFFFFF');
 $chart['axis_category'] = array('color' => 'FFFFFF', 'size' => '10');
 $chart['axis_ticks'] = array('value_ticks' => true, 'minor_color' => 'FFFFFF');

$chart['chart_type'] = 'stacked column';

$chart [ 'chart_value' ] = array ('position' =>  "middle", 'bold' =>  true, 'size' =>  11, 'color' => "FFFFFF", 'alpha' => 190);
```

And finalize the chart with the SendChartData() function that comes with the PHP/SWF library:

```
SendChartData ($chart);
```

And the end result should look something like this:



If you want to know what each formatting option means, have a look at the PHP/SWF Reference for a complete explanation on each option.

Now we have a neat graph showing the impressions and clicks of the last 6 months. The same thing can be done for the last 31 days, or the last week. All the above code stays pretty much the same, except for a few minor adjustments. It's rather pointless to copy all the code from chart-week.php and chart-day.php here, so click here to download chart-week.php and here to download chart-day.php.

If you combine all three chart files, and use another page to show all the charts, the end result looks something like this: charts.php (source of charts.php). Looks pretty professional doesn't it?

Conclusion

In this part of the series I have shown you how to gather statistics of each ad, and how to create great-looking charts with the PHP/SWF Charts library. The charts we created are really simple, and if you browse through

Copyright (c) 2006 PHPit.net

the Chart Gallery you will come across much better looking charts.

This was the final part of the 'Building an advertising system, with PHP' series, and I hope I've shown you how you can create your own advertising system for your website or network. If you combine all the examples and code I've given you, together with some custom changes, you can have a very professional ad system for your website and its advertisers.

In the future I might release an open-source advertising system, based on this series, but I can't promise anything. I do have to admit though that I can't wait to use PHP/SWF Charts in one of my projects, as it's really a great library.

If you have any comments or suggestions, leave them in the comments or post them on the forums.

Click here to download the demo files of this part (Note that you need to download PHP/SWF Charts yourself)

About this PDF

You may distribute this PDF in any way you like, as long as you don't modify it in any way. You can ONLY distribute the unchanged original PDF.

For more information, contact us at support@pallettgroup.com.

Copyright (c) 2006 PHPit.net