Creating a CAPTCHA with PHP

This article will learn how to creat a CAPTCHA utilizing the graphics capabilities of PHP.

You may be thinking just exactly what is a captcha? Well you are likely to have already seen them across the web. They are those little images with a code on the front that you type into a box in order to submit something. This kind of system helps to prevent automatic submitting of an operation by some kind of program or robot. In this tutorial I will show you how to make a CAPTCHA just like the one below. Its not the most advanced captcha available because it uses a simple system font and nothing more.

The lines that you see above are to make any robots job of trying to work out that code a little harder. The dots in the background also help with this. I will now show you how to create one of these that uses a background file that you can easily change.

How does a CAPTCHA work?

To put it simply a captcha works by generating a random string, writing it to an image, then storing the string inside of a session or cookie or by some other method. This is then checked when the form or operation is performed. Below is a step by step layout of how it works. 1. Random text generated 2. Text written to image 3. Text stored in session/cookie/database 4. Image displayed to user 5. User enters the code 6. User entered code is checked against the stored key 7. If they match then something is done

On the next page I will show you how to generate the random text to be used later on in the script. Creating the random text

Right now we are up to generating the random text. To do this I will use the php functions, microtime() and mktime() to generate a number. This number will then be encrypted using md5(). With this 32 character long encrypted string we will then use substr() to cut it down to a 5 letter long string. This is our random text.

Note: You may notice session_start() at the top of this script, this is to start the session which will be used later....<?php
//Start the session so we can store what the code actually is.
session_start();

//Now lets use md5 to generate a totally random string
$md5 = md5(microtime() * mktime());

/*
We dont need a 32 character long string so we trim it down to 5
*/
$string = substr($md5,0,5);
?>

Next we will write this string to the image and output it to the user

Writing the text to the image

Now that we have the text to write we actually need to write it to the image and display it to the user. This is made fairly easy with GD.<?php
/*
Now for the GD stuff, for ease of use lets create
the image from a background image.
*/

$captcha = imagecreatefrompng("./captcha.png");

/*
Lets set the colours, the colour $line is used to generate lines.
Using a blue misty colours. The colour codes are in RGB

```
*/

$black = imagecolorallocate($captcha, 0, 0, 0);
$line = imagecolorallocate($captcha,233,239,239);

/*
Now to make it a little bit harder for any bots to break,
assuming they can break it so far. Lets add some lines
in (static lines) to attempt to make the bots life a little harder
*/
imageline($captcha,0,0,39,29,$line);
imageline($captcha,40,0,64,29,$line);
?>
```

As you can see from the code above we are loading the basic image from CAPTCHA.png instead of building the image itself which could be a little complex for this basic tutorial. When we use colour in GD we need to allocate the colour to a variable, we do this with imagecolorallocate(). Once we have the colours stored inside of the respected variables we then use them to draw the lines through the image. This is to make the robots job of cracking the captcha just that little bit harder, because we are nice to the robots like that :)

Finally we have to write the text to the image which is made easy with imagestring() . The only thing left to do on this image is to output it which is done by setting the content type of the page to image/png with header() and outputting the image to the browser with imagepng(). It is also worth mentioning that the string is encrypted and stored in the session variable $_SESSION['key']<?php

```
/*
Now for the all important writing of the randomly generated string to the image.
*/
imagestring($captcha, 5, 20, 10, $string, $black);


/*
Encrypt and store the key inside of a session
*/

$_SESSION['key'] = md5($string);

/*
Output the image
*/
header("Content-type: image/png");
imagepng($captcha);
?>
```

Check if the user entered the code correctly

To check if the user entered the code correctly you must first allow the user to do this. You can do this with a simple text form that requires a code to be entered, a simple text field called code or something similar should do nicely. Then you just display the image to the user with a simple <img xsrc="captcha.php" border="0"> tag. It is really too low a level to show you how to make a form like this, if you don't know how to make a form like I described above then this tutorial is probably not for you.

Now assuming that this form has been submitted we need to check if the code matches what was on the image, after all this is the whole point of a captcha system. You can do this in any php file as long as the form described above submits to it. For basic checking we will use the code below.<?php

```
session_start();

//Encrypt the posted code field and then compare with the stored key

if(md5($_POST['code']) != $_SESSION['key'])
{
  die("Error: You must enter the code correctly");
}else{
  echo 'You entered the code correctly';
}
```

```
?>
```

The session_start() you see here simply continues the session from the previous page, easy enough. Then its just a case of simple text matching which you can see is done by the if statement.

Improvements

Well that is all there is to CAPTCHA images just a simple writing of text to an image and storing of the text (key). However the captcha I just described how to build is not the best in the world by a long shot. If you're feeling adventurous you could try the following things:

- Use a TTF font
- Move the lines randomly
- Randomly position the text on the image
- Rotate the text randomly
- Use words instead of that string (ie: have a randomly picked word out of say a file of about 1000)