

# PHP Sample Application

Simple graphics and database

## Sample PHP Application

- PHP is commonly used to:
  - Draw graphics on the fly
  - Access a database
- Let's put these together to make an application that:
  - Lets a user make a query on a web form
  - Looks up some data from an Access database (similar if using SQL database)
  - Displays a graph of the data from the database

# Simple Graphics

- First, let's see how to draw some simple graphics
- PHP 5 includes the "GD" graphics library, created by Boutell.com
- PHP 4 requires an extension be added to access this library
- PNG format used
  - Similar but superior to GIF
  - GIF patented by Unisys, expired 2003

## Base image functions

- `imagecreate(width,height)`
  - Creates and returns an image of specified height
- `colorimagecolorallocate(image, R,G,B)`
  - Creates a color for image with the given Red, Green, and Blue colors
- `boolean imagefill(image, x, y, color)`
  - Flood fills at the given point with the given color
- `header("Content-type: image/png");`
  - Displays the MIME header for a PNG image
- `imagepng(image)`
  - Displays the data for the PNG image

## Example: Green image

```
<?php
$image = imagecreate(300,200);
$colorGreen = imagecolorallocate($image, 0, 255, 0);
imagefill($image, 0, 0, $colorGreen);
header("Content-type: image/png");
imagepng($image);
?>
```



## More image functions

- `imagestring (image, int font, int x, int y, string s, int col)`
  - Draws the string `s` in the image at coordinates `x`, `y` in color `col`. If font is 1, 2, 3, 4 or 5, a built-in font is used.
- `imagefilledrectangle(image, x1,y1, x2,y2,color)`
  - Draws a filled rectangle at upper left corner `x1,y1` bottom right corner `x2,y2`
- `imagefilledellipse(image, x1,y1, width,height,color)`
  - Draws a filled ellipse in the bounding rectangle specified by `x1,y1,x2,y2`
- `imagerectangle(image,x1,y1,x2,y2,color)`
  - Rectangle with no fill
- `imageellipse(image, x1,y1, width,height,color)`
  - Ellipse with no fill

# Image example

```
<?php
$image = imagecreate(300,200);
$colorWhite = imagecolorallocate($image, 255, 255, 255);
imagefill($image, 0, 0, $colorWhite);

imagefilledrectangle($image, 50,50, 100, 75,
                    imagecolorallocate($image,255,0,0));
imageellipse($image, 150, 50, 100, 50,
             imagecolorallocate($image,0,0,255));
imagestring($image, 0, 10, 10, "Hello!",
           imagecolorallocate($image,0,0,0));

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

# ImagePolygon and ImageLine

```
<?php
$image = imagecreate(300,200);
$colorWhite = imagecolorallocate($image, 255, 255, 255);
imagefill($image, 0, 0, $colorWhite);

$colorBlack = imagecolorallocate($image, 0, 0, 0);

imageLine($image, 50, 0, 200, 150, $colorBlack);

$pointsTriangle = array(50, 10, 10, 90, 90, 90);
imagePolygon($image, $pointsTriangle, count($pointsTriangle)/2, $colorBlack);

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

How could you display this image multiple times in a web page?

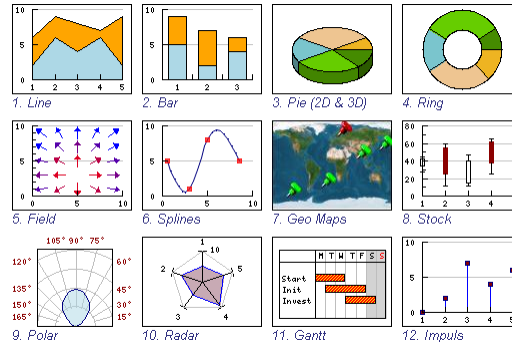
## Many more image functions

- Read jpg, gif, etc.
- Rotate, blending
- Resize
- Draw arcs
- Gamma correct
- Tiling
- ...
- PHP Reference or textbook is a great way to learn about these functions, with code samples

## Drawing Graphs

- What if you wanted to draw charts and graphs?
  - Pie chart, bar chart, line chart, scatter plot, etc.
  - You could do it using the graphics primitives we have covered
- Fortunately, someone has already done this for you
  - JPGraph Library
  - <http://jpgraph.net>

# JPGraph Samples



17. Antispam

See test suite with code samples....

## Let's look at a simple bar chart

```
<?php
include ("jpgraph/jpgraph.php");
include ("jpgraph/jpgraph_bar.php");

$datay=array(12,8,19,3,10,50);
$datax=array(10,20,30,40,50,60);

// Create the graph. These two calls are always required
$graph = new Graph(300,200,"auto");
$graph->SetScale("textlin"); // Natural nums

// Add a drop shadow
$graph->SetShadow();

// Adjust the margin a bit to make more room for titles
$graph->img->SetMargin(40,30,20,40);

// Create a bar plot
$bplot = new BarPlot($datay);
$graph->Add($bplot);
```

## Simple Bar chart, cont.

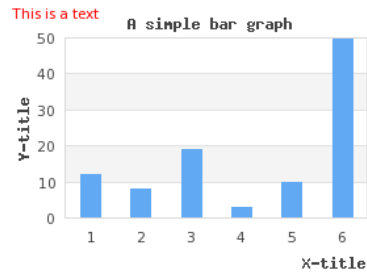
```
// Add data to X coordinate  
// $graph->xaxis->SetTickLabels($datax);
```

```
// Create and add a new text  
$txt=new Text("This is a text");  
$txt->SetPos(0,0);  
$txt->SetColor("red");  
$graph->AddText($txt);
```

```
// Setup the titles  
$graph->title->Set("A simple bar graph");  
$graph->xaxis->title->Set("X-title");  
$graph->yaxis->title->Set("Y-title");
```

```
$graph->title->SetFont(FF_FONT1,FS_BOLD);  
$graph->yaxis->title->SetFont(FF_FONT1,FS_BOLD);  
$graph->xaxis->title->SetFont(FF_FONT1,FS_BOLD);
```

```
// Display the graph  
$graph->Stroke();  
?>
```



## Database Access

- Changing directions, reading from a database and displaying data in a graph
  - In our example, let's read from a MySQL "Test" database with name popularity over the decades
    - name
    - yr1900
    - yr1910
    - ...
    - yr2000
  - PHP app: let the user enter a name then graph the popularity

# Database Schema

Field	Type	Null	Key	Default	Extra
name	varchar(50)	NO	PRI	NULL	
yr1900	int(11)	YES		NULL	
yr1910	int(11)	YES		NULL	
yr1920	int(11)	YES		NULL	
yr1930	int(11)	YES		NULL	
yr1940	int(11)	YES		NULL	
yr1950	int(11)	YES		NULL	
yr1960	int(11)	YES		NULL	
yr1970	int(11)	YES		NULL	
yr1980	int(11)	YES		NULL	
yr1990	int(11)	YES		NULL	
yr2000	int(11)	YES		NULL	

```
mysql> select name, yr1960, yr1970, yr1980, yr1990, yr2000 from names where name
='kyler';
+-----+-----+-----+-----+-----+
| name | yr1960 | yr1970 | yr1980 | yr1990 | yr2000 |
+-----+-----+-----+-----+-----+
| kyler | 1001 | 1001 | 978 | 378 | 281 |
+-----+-----+-----+-----+-----+
```

## Building our App

- First, make a form that asks for a username and then retrieves and prints all the yearly data that matches the name in the table



```

<?php
header("Content-Type: text/html");
print("<HTML><HEAD><TITLE>Name Popularity Surfer</TITLE>");
print("</HEAD>");
print("<BODY>");

if($_SERVER['REQUEST_METHOD'] != "POST")
{
    print("<FORM method=post action='names.php'>");
    print("Enter name.<p>");
    print("<INPUT type=text name='name'>");
    print("<INPUT type=submit>");
    print("</FORM>");
}

```

```

else
{
    $name = $_REQUEST['name'];
    // Database parameters
    $db_location = "localhost";
    $db_user_name = "test";
    $db_password = "test";
    $database_name = "test";
    // Connect to the DB
    $dbcnx = mysql_connect($db_location,$db_user_name,$db_password);
    mysql_select_db($database_name);

    $sql="SELECT * FROM names where name=" . $name . " ";
    $rs=mysql_query($sql);

```

```

if ($row = mysql_fetch_assoc($rs))
{
    $arr = array();
    $arr[] = $row['yr1900'];
    for ($i = 10; $i < 100; $i+=10)
        $arr[] = $row['yr19' . $i];
    $arr[] = $row['yr2000'];
    print_r($arr);
    print("<br>");
}
if (!isset($arr))
    print("$name not found in the database.");
}
print("</BODY>");
?>

```

## Drawing Graph

- Now let's hook this up with jppgraph to draw a graph of the popularity over time
- You could run this yourself if you copy/install the jppgraph folder to your own HTML folder
  - Can access database from any account using the supplied username/password

# Tie-In Graphics

```
<?php
include("jpggraph.php");
include("jpggraph_line.php");

if($_SERVER['REQUEST_METHOD'] != "POST")
{
    print("<FORM method=post action='names.php'>");
    print("Enter name.<p>");
    print("<INPUT type=text name='name'>");
    print("<INPUT type=submit>");
    print("</FORM>");
}
else
{
    $name = $_REQUEST['name'];
    // Database parameters
    $db_location = "localhost";
    $db_user_name = "test";
    $db_password = "test";
    $database_name = "test";
    // Connect to the DB
    $dbcnx = mysql_connect($db_location,$db_user_name,$db_password);
    mysql_select_db($database_name);
```

# Tie-In Graphics

```
$sql="SELECT * FROM names where name='". $name . "'";
$rs=mysql_query($sql);
if ($row = mysql_fetch_assoc($rs))
{
    $ydata = array();
    $xdata = array();
    $ydata[] = $row['yr1900'];
    $xdata[] = 1900;
    for ($i = 10; $i < 100; $i+=10)
    {
        $ydata[] = $row['yr19' . $i];
        $xdata[] = 1900 + $i;
    }
    $ydata[] = $row['yr2000'];
    $xdata[] = 2000;

    // Create the graph. These two calls are always required
    $graph = new Graph(600,400,"auto");
    $graph->SetScale("linlin");

    // Add a drop shadow
    $graph->SetShadow();
    // Adjust the margin a bit to make more room for titles
    $graph->img->SetMargin(40,30,20,40);

    // Create a line plot
    $plot = new LinePlot($ydata,$xdata);
    $plot->SetColor("blue");
    $plot->SetWeight(2);
    $graph->Add($plot);

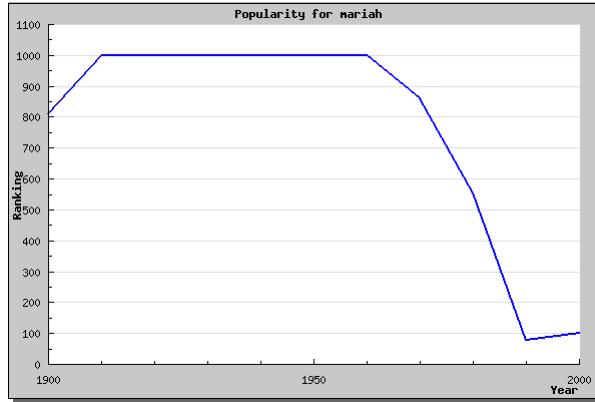
    // Add data to X coordinate
    $graph->xaxis->SetTickLabels($xdata);

    // Setup the titles
    $graph->title->Set("Popularity for " . $name);
    $graph->xaxis->title->Set("Year");
    $graph->yaxis->title->Set("Ranking");

    $graph->title->SetFont(FF_FONT1,FS_BOLD);
    $graph->yaxis->title->SetFont(FF_FONT1,FS_BOLD);
    $graph->xaxis->title->SetFont(FF_FONT1,FS_BOLD);

    // Display the graph
    $graph->Stroke();
}
if (!isset($arr))
{
    header("Content-Type: text/html");
    print("<HTML><HEAD><TITLE>Name Surfer</TITLE>");
    print("</HEAD>");
    print("<BODY>");
    print("$name not found in the database.");
    print("</BODY></HTML>");
}
?>
```

# Output



## Summary

- We discussed how to integrate business graphics with a simple database web application
- Should have error checking for no user found, SQL injection, etc.
- Similar process for updating and modifying the database
- Very easy to make simple graphics
- Libraries for more complex graphs
- Overall it is fairly easy to create sophisticated web applications
  - Other environments like .NET encapsulate much of the primitive HTML information, e.g. datagrid