mysqli, Table relationships, Uploading files and PHP graphics

INFO/CS 2300:
Intermediate Web Design and Programming

P3 – M1 – M2

We expect to grade all P3M1 (that are turned in on time) on Tuesday evening.

Don't change your files by starting work on M2 before you are graded for M1. Alternatively, make a copy to work on until you are graded.

Click In!

Click In!

How do we first connect to a MySQL database in PHP?

A. new mysqli(host, user, password, db);
B. mysqli(host, user, password, db);
C. mysql_connect(host, user, password, db);
D. mysqli_connect(host, user, password, db);
E. A or D

Click In!

How do we first connect to a MySQL database in PHP?

A. new mysqli(host, user, password, db);
B. mysqli(host, user, password, db);
C. mysql_connect(host, user, password, db);
D. mysqli_connect(host, user, password, db);
E. A or D

Given a mysqli connection, $connect, what do we need to do next to get the first row of the movies table?

A. $connect->fetch_row();
B. $connect->fetch_assoc();
C. $connect->get_row(movies);
D. $connect->query(“SELECT * FROM movies”);
E. A or B

Given a mysqli connection, $connect, what do we need to do next to get the first row of the movies table?

A. $connect->fetch_row();
B. $connect->fetch_assoc();
C. $connect->get_row(movies);
D. $connect->query(“SELECT * FROM movies”);
E. A or B

If $result is the result of $connect->query(“SELECT * FROM movies”), what can be the value returned by $result->fetch_row()?

A. array(‘Argo’, 2012, 120);
B. array(‘title’ => ‘Argo’, ‘year’ => 2012, ‘length’ => 120);
C. Both of the above
D. Neither of the above

If $result is the result of $connect->query(“SELECT * FROM movies”), what can be the value returned by $result->fetch_row()?

A. array(‘Argo’, 2012, 120);
B. array(‘title’ => ‘Argo’, ‘year’ => 2012, ‘length’ => 120);
C. Both of the above
D. Neither of the above

PHP commands for MySQL

MySQL commands

Recall:

```
new mysqli( DB_HOST, DB_USER,
   DB_PASSWORD, DB_NAME );
```

Returns an instance of a mysqli object connecting to the MySQL DB.

config.php

```
<?php  // ** MySQL connection settings ** //
    // database host
    define('DB_HOST', 'localhost');

    // database name
    define('DB_NAME', 'info230_SP15_netidsp15');

    // Your MySQL username
    define('DB_USER', 'netidsp15');

    // ...and password
    define('DB_PASSWORD', ‘your_password');
?>
```

Mistake in earlier lecture. Should be username

Your course server credentials

movies.php

```
require_once ‘config.php’;
$mysqli = new mysqli( DB_HOST, DB_USER, DB_PASSWORD, DB_NAME );
```

Where to put config.php?

```
www
   p3
      css
            style.css
      includes
            functions.php
            settings.php
      add-edit.php
      index.php
```

Here if your credentials are different on your local and the server

Here if your credentials are the same on your local and the server

Our course server doesn't let you put them above www

MySQL commands

$result = *mysqli*->query(*'SELECT * FROM …'*);

Issues *sqlquery* to MySQL DB given by *mysqli* instance.

- For INSERT, UPDATE, DELETE, returns true if successful, false if not
- For SELECT, returns a mysqli *result object* if successful, false if not.

$row = *$result*->fetch_row();

$row = *result*->fetch_assoc();

Returns array containing the next record from the result set given by *result*, or false if no more records.

Mysqli – object vs procedural

```
<?php //procedure style
$mysqli = mysqli_connect("host", "user", "password", "database");
if ( mysqli_connect_errno( $mysqli ) ) {
    echo "Failed to connect to MySQL: " . mysqli_connect_error();
}

$result = mysqli_query($mysqli, "SELECT…");
$row = mysqli_fetch_assoc($result);
echo $row['_msg'];

//Object style
$mysqli = new mysqli("host", "user", "password", "database");
if ( $mysqli->connect_errno ) {
    echo "Failed to connect to MySQL: " . $mysqli->connect_error;
}

$result = $mysqli->query("SELECT …");
$row = $result>fetch_assoc();
echo $row['_msg'];
?>
```

MySQL object vs procedural

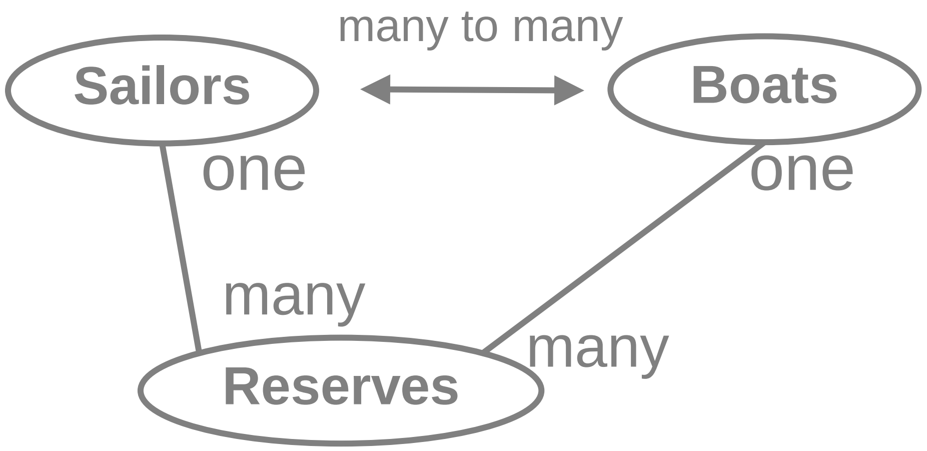
```
$row_count = $result->num_rows;
$row_count = mysqli_num_rows($result);

$mysqli->close();
mysqli_close( $mysqli );
```

Closes connection to DB given by $*mysqli*.

SQL table relationships

SQL: one to many, many to many

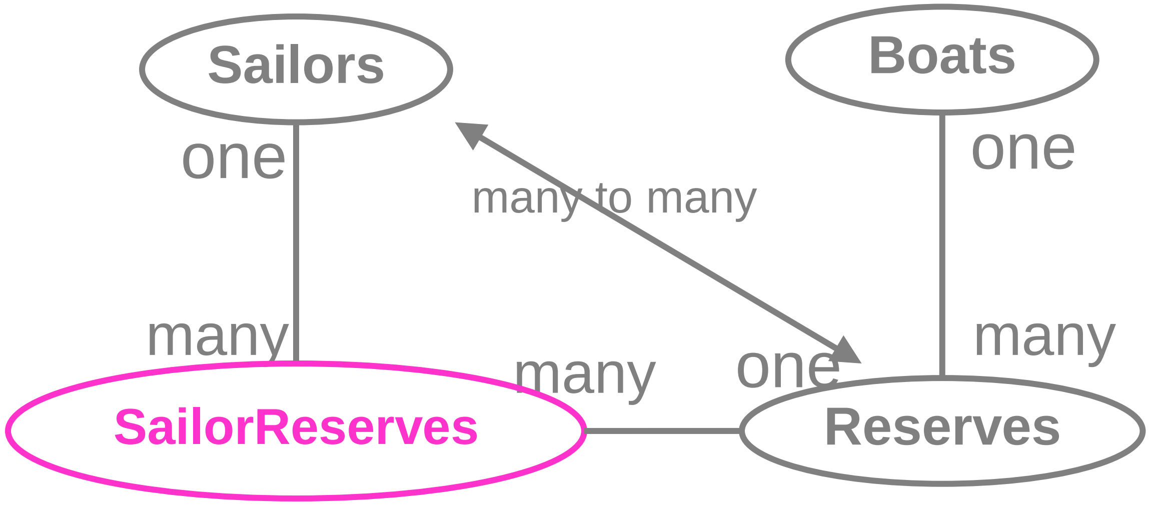
- Sailors
  - sailorId: integer
  - sailorName: string
  - rating: integer
  - age: integer
- Boats
  - boatId: integer
  - boatName: string
  - color: string
- Reserves
  - reserveId: integer
  - sailorId: integer
  - boatId: integer
  - day: date



If the business rules changed so that multiple sailors could make one reservation together, how would the schema change?

SQL: one to many, many to many

- Sailors
  - sailorId: integer
  - sailorName: string
  - rating: integer
  - age: integer
- Boats
  - boatId: integer
  - boatName: string
  - color: string
- Reserves
  - reserveId: integer
  - ~~sailorId: integer~~
  - boatId: integer
  - day: date



- SailorReserves
  - sailorReserveId: integer
  - sailorId: integer
  - reserveID: integer

Click In!

Given the schema

Hotel (hotelId, hotelName, city)
Room (roomId, hotelId, type, price)
Booking (roomId, guestId, dateFrom, dateTo)
Guest (guestId, guestName, guestAddress)

Which tables are in a many – many relationship?

A. Hotel – Room
B. Room – Booking
C. Hotel – Guest
D. Room – Guest
E. Guest - Booking

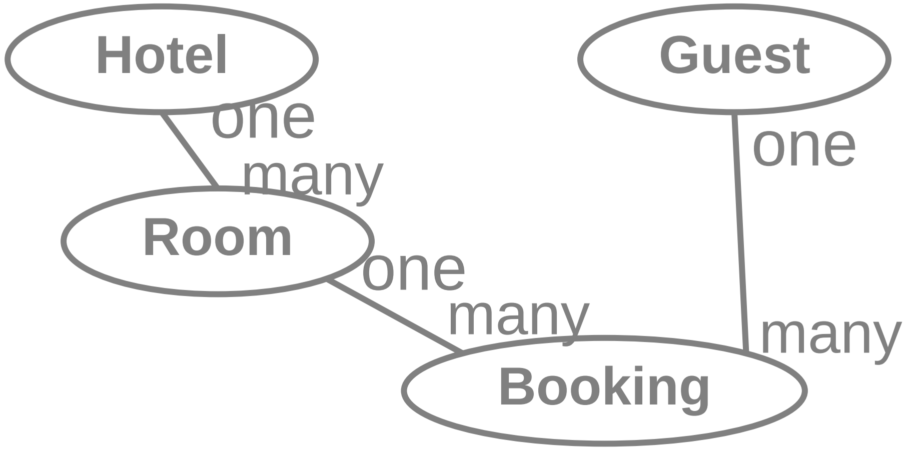
Click In!

Given the schema

Hotel (hotelId, hotelName, city)
Room (roomId, hotelId, type, price)
Booking (roomId, guestId, dateFrom, dateTo)
Guest (guestId, guestName, guestAddress)

Which tables are in a many – many relationship?

A. Hotel – Room
B. Room – Booking
C. Hotel – Guest
D. Room – Guest
E. Guest - Booking



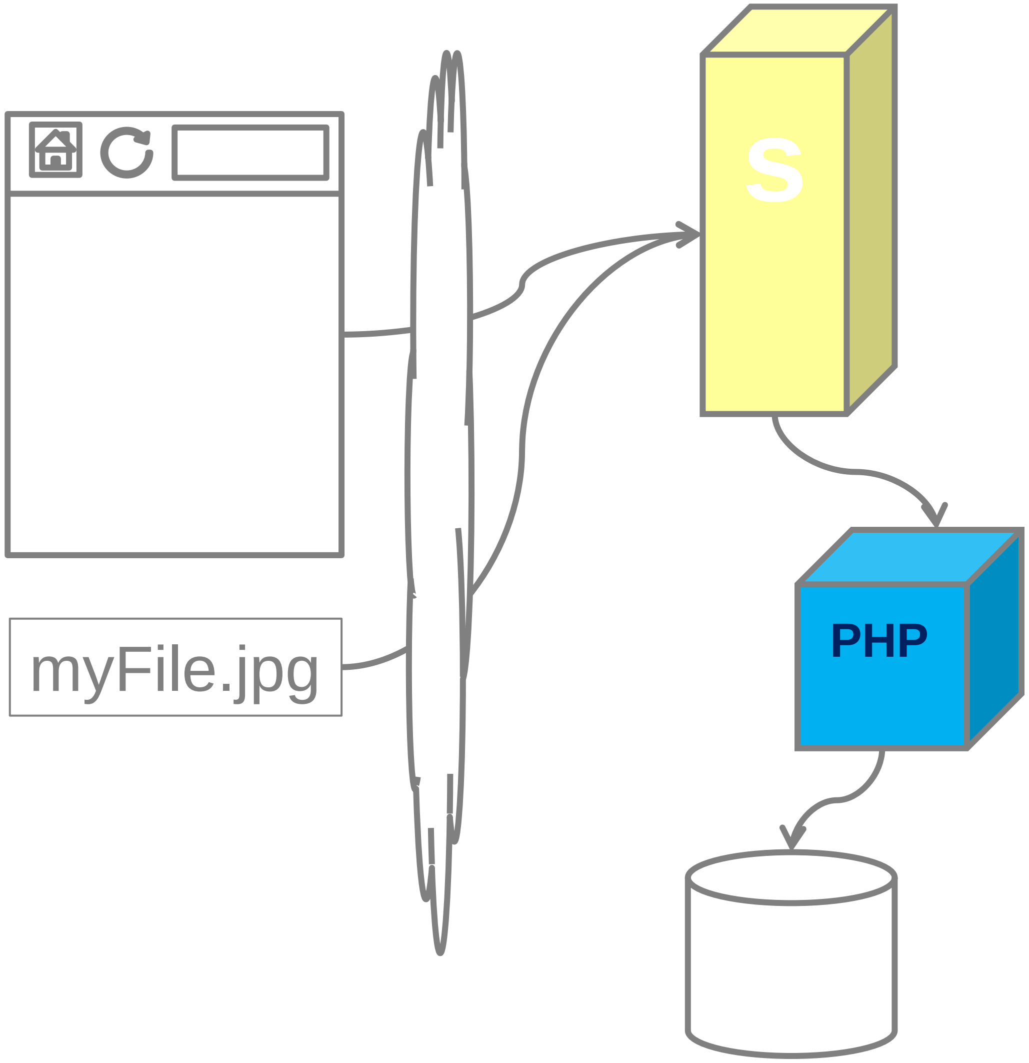
Uploading files

How to upload a file

As you might suspect, file uploading is another HTML form.

```
<form action="photos.php" method="post"
        enctype="multipart/form-data">
  <input type="file" name="newphoto">
  <input type="submit" name="Upload photo">
</form>
```

How it works



PHP global variables

```
<?php
$img = 'my_image.jpg';

function local_scope() {
    //This causes an error because $img is not defined
    $newImage = $img;
}

function global_scope() {
    global $img;

    //This works because $img was declared using 'global'
    $newImage = $img;
}
?>
```

PHP SUPERGLOBALS

$_POST and $_GET are known as superglobals.

```
function superglobal() {
    //No need to declare $_POST using ‘global’
    $newImage = $_POST[ ‘image’ ];
}
```

Information from file uploading shows up in $_FILES

```
$newFile = $_FILES[ ‘newphoto’ ];
```

The $_SESSION superglobal stores information about the current browser session – more on Wednesday

Using $_FILE information

```
$newFile = $_FILES[ ‘newphoto’ ];

$originalName = $newFile[ ‘name’ ];
$tempName = $newFile[ ‘tmp_name’ ];
$size_in_bytes = $newFile[ ‘size’ ];
$type = $newFile[ ‘type’ ];
$error = $newFile[ ‘error’ ];
```

name of file in temporary storage

“image/jpeg” or “application/pdf”

error code (0 if no error)

Moving the file

move_uploaded_file(*source*, *destination*)

Moves uploaded file out of temporary storage to wherever you want to keep it

```
$newFile = $_FILES[ ‘newphoto’ ];
$originalName = $newFile[ ‘name’ ];
$tempName = $newFile[ ‘tmp_name’ ];

move_uploaded_file($tempName, “images/$originalName” );
```

Debugging

```
//Display the $_FILES array nicely formatted
echo '<pre>' . print_r( $_FILES, true ) . '</pre>';

//Hide it from users – if debugging a live site
echo '<pre style="display:none;">' .
                              print_r( $_FILES, true ) . '</pre>';
```

photos.php - upload

```
<form action="photos.php" method="post" enctype="multipart/form-data">
    Single photo upload: <input type="file" name="newphoto" /><br />
    <input type="submit" value="Upload photo" />
</form>

<?php
    print '<pre style="display:none;">' . print_r( $_FILES, true ) . '</pre>';
    if ( ! empty( $_FILES[ 'newphoto‘ ]  )  ) {
        $newPhoto = $_FILES[ 'newphoto‘ ];
        $originalName = $newPhoto[ 'name‘ ];
        if ( $newPhoto[ ‘error’ ] == 0 ) {
            $tempName = $newPhoto[ 'tmp_name‘ ];
            move_uploaded_file( $tempName, “images/$originalName");
            $_SESSION['photos'][] = $originalName;
            print("The file $originalName was uploaded successfully.\n");
        } else {
            print(“Error: The file $originalName was not uploaded.\n");
        }
    }
?>
```

Image information

getimagesize(*filename*)

Returns an array with image file information.

```
$imageInfo = getimagesize(“myphoto.jpg”);
$imageWidth = $array[0];
$imageHeight = $array[1];
$imageType = $array[2];
$imageTagDimensions = $array[3];
```

IMG_GIF | IMG_JPG | IMG_PNG | IMG_WBMP | IMG_XPM

‘height=“*ht*” width=“*wt*” ‘

photos.php – display

```
foreach ( $_SESSION[ 'photos‘ ] as $photo ) {
    $file = "images/$photo";
    $imagesize = getimagesize( $file );
    $size = "Actual size: {$imagesize[3]}";
    $taken = '';
    $exif_data = exif_read_data ( $file );
    if ( ! empty( $exif_data[ 'DateTimeOriginal‘ ] ) ) {
          $taken = " Taken: {$exif_data[ 'DateTimeOriginal' ]}";
    }
    print "<img src='$file' alt='$photo' title='$photo $size
                                                            $taken'><br />\n";
}
```

height="yyy"
width="xxx"

Multiple file uploads

In HTML5, there is the ability to upload multiple files at once.

Already implemented in Safari, Firefox, Chrome and IE 10.

Multiple file uploads

Change form from:

```
<input type="file" name="newphoto" >
```

to:

```
<input type="file" name="newphotos[]"
   multiple>
```

The “s” is helpful but not required

Multiple file uploads in PHP

Now information is in

```
$_FILES[‘newphotos’][‘name’][0],
$_FILES[‘newphotos’][‘name’][1], etc…
```

With less repeating

```
$photos = $_FILES[‘newphotos’][‘name’];

$firstPhoto = $photos[0];
$secondPhoto = $photos[1];
```

photos.php – upload multiple

```
if ( isset( $_FILES['newphotos'] ) ) {
    $newPhotos = $_FILES['newphotos'];
    for ( $i = 0; $i < count( $newPhotos['name'] ); $i++) {
        $originalName = $newPhotos['name'][$i];
        if ($newPhotos[ 'error' ][$i] == 0) {
                $tempName = $newPhotos[ 'tmp_name' ][$i];
                move_uploaded_file( $tempName,
                                        "images/$originalName" );
                $_SESSION['photos'][ ] = $originalName;
                print("$originalName was uploaded successfully. ");
        } else {
                print(“The file $originalName was not uploaded.");
        }
    }
}
```

Using PHP Graphics to make thumbnails

We can use PHP graphics to save a smaller, resized version of our photo (a “thumbnail”), which we can show instead.

imagesx(*image*), imagesy(*image)*

Gives width, height of *image*.

imageCopyResized(*dst_img, src_img, dst_x, dst_y, src_x, src_y, dst_width, dst_height, src_width,src_height*)

(also imageCopyResampled)

Copy a rectangle from *src_img* to *dst_img* with resizing

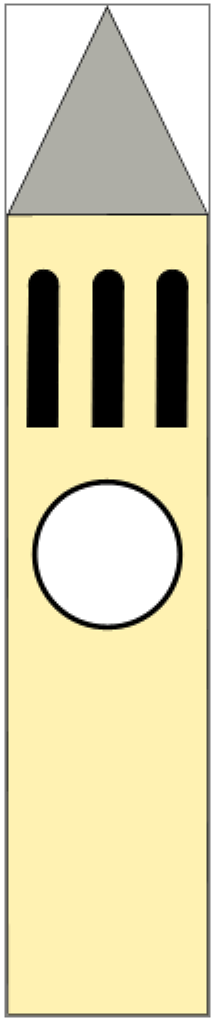
(Resampled also tries to interpolate colors to keep image clarity).

imageJPEG(*image*, *filename*)

Saves image in JPG file *filename*.

```
/*
 * Saves a thumbnail of the given image
 * Parameters:
 * $source: the path and file name relative to the current directory
 * $thumbPathAndFile: the path and file name of the thumbnail to be created (relative to the current
     directory)
 * $thumbWidth: the width of the thumbnail being created
 */
function save_thumbnail( $source, $thumbPathAndFile, $thumb_width = 200 ) {
	//Create a new image from the given image
	$img = imagecreatefromjpeg( $source );

	//Calculate the dimensions
	$width = imagesx($img);
	$height = imagesy($img);

	//Set the new dimensions by proportionally scaling the height to the given width
	$new_width = $thumb_width;
	$new_height = floor($height * ($thumb_width/$width));

	//Create a new, empty image of the correct size
	$new_img = imagecreatetruecolor($new_width, $new_height);

	//Copy and resize the original into the new
	imagecopyresampled($new_img, $img, 0, 0, 0, 0, $new_width, $new_height, $width, $height);

	//Save the image to the given path
	$return = imagejpeg($new_img, $thumbPathAndFile);

	//Free up memory
	imageDestroy($img);
	imageDestroy($new_img);

	//Return the success/failure status
	return $return;
}
```

Gee whiz! PHP Graphics: A clock



```
$hour = 2;
$minute = 30;
$im = imageCreateFromGIF("tower.gif");

$black = imageColorAllocate($im, 0, 0, 0);

imageSetThickness($im, 5);
$array = getHand(48, 276, $hour, 'hour' );
imageLine( $im, $array[0], $array[1], $array[2], $array[3], $black);

imageSetThickness($im, 3);
$array = getHand(48, 276, $minute, 'minute' );
imageLine( $im, $array[0], $array[1], $array[2], $array[3], $black);

header('Content-type: image/png');
imagePNG($im);
imageDestroy($im);
```

Review

- Upload files via input type=“file”; use results on PHP side via $_FILES.
- Can draw images using data from user input, databases, other sources with PHP graphics.

Appendix: PHP Graphics

Disclaimer

PHP graphics have been part of the syllabus for this course in the past. I've never had a use for PHP graphics so I'm not lecturing on it. Rather than cut the material completely, I'm leaving it here in case you are interested.

Credits

There's tons to cover on PHP graphics; we' re only scratching the surface. To learn more, see the following nice introduction about PHP graphics:

http://www.nyphp.org/content/presentations/GDintro/

Graphics

It is possible to create images “on the fly” with PHP via a graphics package gd2.

Including a PHP-generated image

No different than including any other image.

```
<img src="myimage.php" height="200" width="200" alt="A random image" />
```

The general form

1. Create an image of specified size in memory.
2. Put content in the image (shapes, text, etc.)
3. Output a header (so your browser knows it is an image, and not HTML)
4. Output the image
5. Free the memory used to create the image.

Most of the steps

imageCreate(*width, height*)

imageCreateTrueColor(*width, height*)

Returns an image resource of *width* pixels by *height* pixels.

header(‘Content-type: image/png’)

header(‘Content-type: image/jpeg’)

Outputs headers for PNG or JPEG type images respectively.

Drawing on existing images

imageCreateFromPNG(*filename*)

imageCreateFromGIF(*filename*)

imageCreateFromJPEG(*filename*)

Use as a starting image the given PNG/GIF/JPEG.

Outputting and destroying

imagePNG(*image*)

imageJPEG(*image*)

Given an image resource, outputs a PNG or JPEG image respectively.

imageDestroy(*image*)

Given an image resource, frees the memory associated with the image.

Example

```
<?php
   $im = imageCreate(200, 200);

   // Do the interesting stuff here

   header(‘Content-type: image/png’);
   imagePNG($im);
   imageDestroy($im);
?>
```

The interesting stuff

We can specify the colors we will use in RGB format.

imageColorAllocate(*image*, *red, green, blue)*

Returns a color resource

E.g.

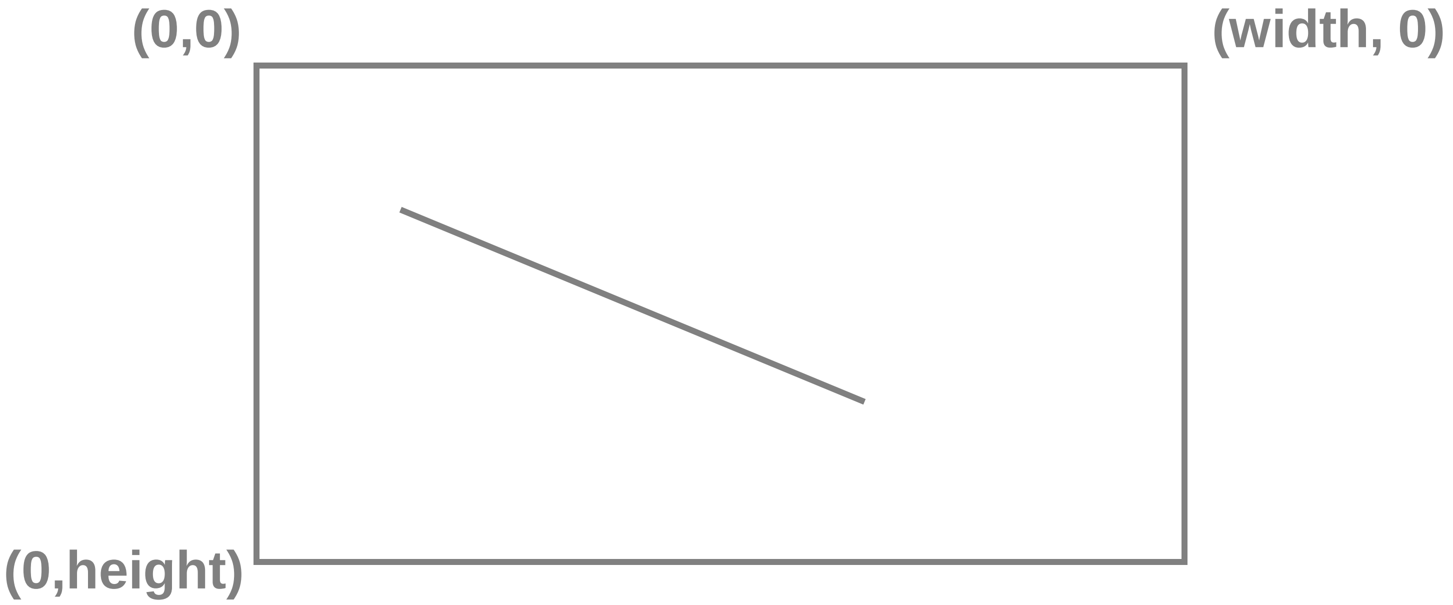
$white = imageColorAllocate($im, 0xFF, 0xFF, 0xFF);

$black = imageColorAllocate($im, 0, 0, 0);

Lines

imageLine(image, x1, y1, x2, y2, color)

Draws a line on *image* of color *color* from (x1,y1) to (x2,y2)



Rectangles

imageRectangle(*image, x1, y1, x2, y2, color*)

Draws a rectangle in *image* of color *color* with opposite corners at (x1,y1) and (x2, y2)

imageFilledRectangle(*image, x1, y1, x2, y2, color*)

Same as above, but rectangle is filled in.

Ellipses

imageEllipse(*image, x, y, w, h, color*)

Draws an ellipse of color *color* centered at (*x*,*y*) of width *w* and height *h*

imageFilledEllipse(*image, x, y, w, h, color*)

Same as above, but ellipse is filled in

```php
<?php

    $im = imageCreate(500, 500);

    $black = imageColorAllocate($im, 0, 0, 0);
    $blue = imageColorAllocate($im, 0, 0, 0x80);
    $red = imageColorAllocate($im, 0x80, 0, 0);

    imageLine($im, 20, 20, 300, 400, $black);
    imageRectangle($im, 200, 10, 100, 100, $red);
    imageFilledRectangle($im, 200, 490, 100, 400, $blue);
    imageEllipse($im, 150, 250, 100, 200, $blue);
    imageFilledEllipse($im, 150, 100, 75, 75, $red);


    header('Content-type: image/png');
    imagePNG($im);
    imageDestroy($im);

?>
```

Why…?

For fixed images, probably would rather use a drawing program.

But helpful for drawing images based on user input, database information, etc.