Advanced

Content

- PHP Date and Time
- Embedding External Files
- PHP File Handling
- PHP Cookies
- PHP Session
- PHP Error Handling

PHP Date and Time

- The PHP **date()** function formats a timestamp to a more readable date and time.
- Syntax: `date(format,timestamp)`
- Example

```
echo date("Y/m/d") . "<br />";
echo date("Y.m.d") . "<br />";
echo date("Y-m-d");
```

- Result

```
2009/05/11
2009.05.11
2009-05-11
```

PHP Date and Time

- The **mktime()** function returns the Unix timestamp for a date.
- Syntax:

```
mktime(hour,minute,second,month,day,year,is_dst)
```

- Example: to go one day in the future:

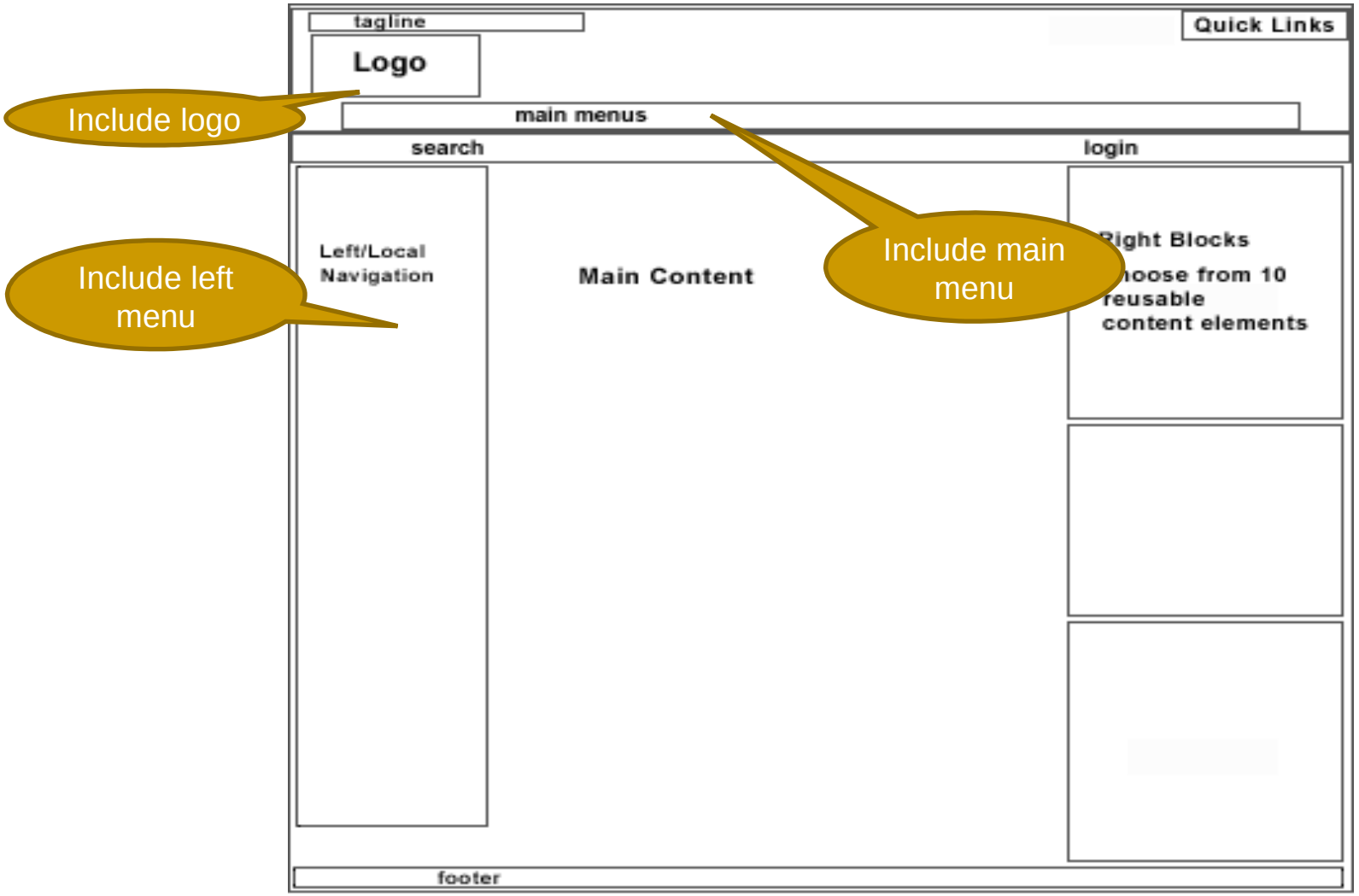
```
<?php
$tomorrow = mktime(0,0,0, date("m"), date("d")+1,
date("Y"));
echo "Tomorrow is ".date("Y/m/d", $tomorrow);
?>
```

- The output: `Tomorrow is 2009/05/12`

PHP Date and Time

- Check Date in PHP
  - **Syntax**
    `checkdate(month,day,year)`
  - Validates a Gregorian date
  - A Gregorian date is valid if
    - the month is specified between 1 to 12
    - the day is within the allowed number of days for the particular month
    - the year is between 1 and 32767 inclusive
  - Example
    - `checkdate(2,29,2012)` => true
    - `checkdate(2,29,2013)` => false

Embedding External Files



Embedding External Files

- Use the `include()` and `include_once()` functions for files that will not prevent the application from running
- Use the `require()` or `require_once()` functions for files that will prevent the app from running if not present
- You can use any file extension you want for include files

Embedding External Files

- Assume we have a standard menu file, called "leftmenu.php", that should be used on all pages.

```
<a href="/default.php">Home</a>
<a href="/about.php">About Us</a>
<a href="/contact.php">Contact Us</a>
```

- All pages in the Web site should **include** this menu file. Here is how it can be done.

```
<div class="leftmenu">
<?php include("leftmenu.php"); ?>
</div>
<h1>Welcome to my home page.</h1>
<p>Some text.</p>
```

PHP File Handling

- `fopen()` function is used to open files
- `fclose()`function is used to close an open file
- `fgets()`function is used to read a single line from a file
- `fgetc()`function is used to read a single character from a file
- `feof()`function checks if the "end-of-file" (EOF) has been reached.

PHP File Handling

- Reading a file by line or by character

```
<?php
$file = fopen("welcome.txt", "r") or exit("Unable to
open file!");
//Output a line of the file until the end is reached
while(!feof($file))
{
  // read line by line
  echo fgets($file). "<br />";
  // read character by character
  echo fgets($file). "<br />";
}
fclose($file);
?>
```

PHP Cookies

- What is a Cookie?
  - A cookie is often used to identify a user.
  - A cookie is a small file that the server embeds on the user's computer.
  - Each time the same computer requests a page with a browser, it will send the cookie too
- How to Create a Cookie?

```
setcookie(name, value, expire);
```

PHP Cookies

- Create a Cookie

```
<?php
$value = ”iPhone is my favorite phone”;
setcookie("myCookie", $value);
// expire in 1 hour
setcookie("myCookie", $value, time() + 3600);
// expire one hour ago
setcookie ("myCookie", $value, time() - 3600);
?>
```

PHP Cookies

- Retrieve a Cookie value

```
<?php
if (isset($_COOKIE[“myCookie"]))
  echo "Welcome " . $_COOKIE["myCookie"];
?>
```

- Delete a Cookie

```
<?php
// set the expiration date to one hour ago
setcookie("myCookie", "", time()-3600);
?>
```

PHP Cookies

- Store information about one single user, and available to all pages in one application.
- Session is temporary and will be deleted after the user has left the website.
- Before storing user information in session, you must first start up the session.

```
<?php session_start(); ?>
```

- This function must appear BEFORE the `<html>` tag.

PHP Sessions

- Storing & retrieving a session variable

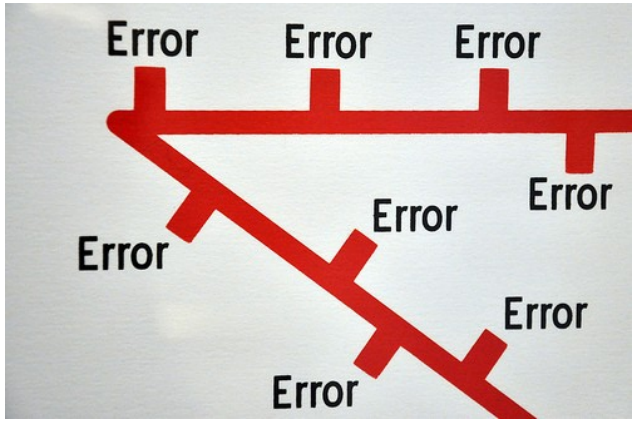
```
<?php
if(isset($_SESSION['views']))
$_SESSION['views']=$_SESSION['views']+1;
else
$_SESSION['views']=1;
echo "Views=". $_SESSION['views'];
?>
```

- Destroying a Session

```
<?php
unset($_SESSION['views']);
// or we can use
session_destroy();
?>
```

PHP Error Handling

- When creating scripts and web applications, error handling is an important part.



- Lacking error checking code, your program may be open to security risks.

PHP Error Handling

- Using the die() function

```
if(!file_exists("welcome.txt")){
  die("File not found");
}
else {
  $file=fopen("welcome.txt","r");
}
```

- If the file does not exist, an error like this:

```
Warning: fopen(welcome.txt) [function.fopen]:
failed to open stream:
No such file or directory
in C:\webfolder\test.php on line 2
```

PHP Error Handling

```
try {
// do something
} catch (error)
// about errors
}
```

- Try, throw and catch
  - **Try** - A function using an exception should be in a "try" block. If the exception does not trigger, the code will continue as normal. However if the exception triggers, an exception is "thrown"
  - **Throw** - This is how you trigger an exception. Each "throw" must have at least one "catch"
  - **Catch** - A "catch" block retrieves an exception and creates an object containing the exception information

PHP Error Handling

```
try {
// do something
} catch (error)
// about errors
}
```

- Example

```
<?php
function checkNum($number)
{
  if($number>1){
    throw new Exception("Value must be 1 or below");
  }
  return true;
}
//trigger exception in a "try" block
try {
   checkNum(2);
   echo 'If you see this, the number is 1 or below';
}
catch(Exception $e){
  echo 'Message: ' .$e->getMessage();
}
?>
```


Demo