```
<?php
  //create short variable names
  $tireqty = $_POST["tireqty"];
  $oilqty = $_POST["oilqty"];
  $sparkqty = $_POST["sparkqty"];
  $address = $_POST["address"];
  $DOCUMENT_ROOT = $_SERVER["DOCUMENT_ROOT"];
?>
<html>
<head>
  <title>Bob's Auto Parts - Order R
</head>
<body>
<h1>Bob's Auto Parts
<h2>Order Results
<?php
  $totalqt
  $tota
  $to += qty;
  $tota rkqty;
  $tota 0.00;
  define PRICE', 100);
```

NO need to do this if "register_globals" is t ON! (php.ini)

PHP Hypertext Preprocessor

07 PHP IV

PHP IV

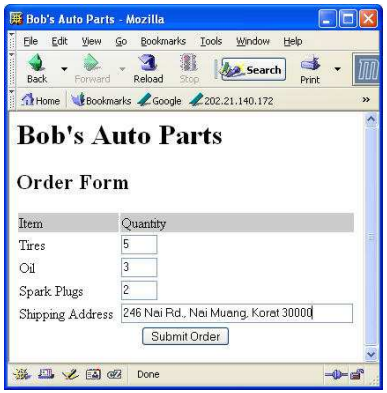
- Storing and Retrieving Data
- Array & File
- Workshop

Storing and Retrieving Data

- Saving Data for Later
- Overview of File Processing
- Opening a File
- Writing to a File
- Closing a File
- Reading from a File
- Other Useful File Functions
- File Locking
- Problem With Using Flat Files

Storing and Retrieving Data

- Saving Data for Later
  - There are basically two ways you can store data: in
    - a flat files
    - a database.
  - A flat file can have many formats but, in general, when we refer to a *flat file* ,we mean simple text file. In this example, we'll write customer orders to a text file, one order per line.



Overview of File Processing

- There are three steps to *writing data* to a file:
  - 1. Open the file. If the file doesn't already exist, it will need to be created.
  - 2. Write the data to the file.
  - 3.Close the file.
- Similarly, there are three steps to *reading data* from a file:
  - 1. Open the file. If the file can 't be opened (for example, if it doesn't exist), we need to recognize this and exit gracefully.
  - 2. Read data from the file.
  - 3. Close the file.
- When you want to read data from a file, you have choices about how much of the file to read at a time.

Opening a File

- To open a file in PHP, we use **fopen()** function.
- Specify how we intend to use it. This is known as the *file mode*.
- File Modes
  - File modes give the operating system a mechanism to determine how to handle access requests from other people or scripts and a method to check that you have access and permission to this particular file.
  - There are three choices you need to make when opening a file:
    - 1. You might want to open a file for reading only, for writing only, or for both reading and writing.
    - 2. If writing to a file, you might want to overwrite any existing contents of a file or to append new data to the end of the file.
    - 3. If you are trying to write to a file on a system that differentiates between binary and text files, you might want to specify this.
  - The **fopen()** function supports combinations of these three options.

Opening a File (cont.)

- Using fopen() to open a file
- When fopen() is called, it expects two or three parameters. (Usually you'll use two, as shown in this code line.)

```
$fp = fopen("$DOCUMENT_ROOT/../orders/orders.txt", "w");
```

- The first parameter should be the file you want to open.
- The second parameter is the file mode (see next slide for details).

Summary of File Mode for fopen()

Mode	Meaning
r	**Read mode** –Open the file for reading, beginning from the start of the file.
r+	**Read mode** –Open the file for reading and writing, beginning from the start of the file.
w	**Write mode** –Open the file for writing, beginning from the start of the file. If the file already exists, delete the existing contents. If it does not exist, try and create it.
w+	**Write mode** –Open the file for writing and reading, beginning from the start of the file. If the file already exists, delete the existing contents. If it does not exist, try and create it.
a	**Append mode** –Open the file for appending writing only, starting from the end of the existing contents, if any. If it does not exist, try and create it.
a+	**Append mode** –Open the file for appending writing and reading, starting from the end of the existing contents, if any. If it does not exist, try and create it.

t = text mode, b = binary mode

Writing to a File

```
<?php
  //create short variable names
  $tireqty = $_POST['tireqty'];
  $oilqty = $_POST['oilqty'];
  $sparkqty = $_POST['sparkqty'];
  $address = $_POST['address'];
  $DOCUMENT_ROOT = $_SERVER['DOCUMENT_ROOT'];
?>
<html>
<head>
  <title>Bob's Auto Parts - Order Results</title>
</head>
<body>
<h1>Bob's Auto Parts</h1>
<h2>Order Results</h2>
<?php
  $totalqty = 0;
  $totalqty += $tireqty;
  $totalqty += $oilqty;
  $totalqty += $sparkqty;
  $totalamount = 0.00;
  define('TIREPRICE', 100);
```

1 of 3

NO need to do this if "register_globals" is turned ON! (php.ini)

Writing to a File (cont.)

```
define('OILPRICE', 10);
define('SPARKPRICE', 4);
$date = date('H:i, jS F');
echo '<p>Order processed at ';
echo $date;
echo '<br />';
echo '<p>Your order is as follows:';
echo '<br />';
if( $totalqty == 0 ){
  echo 'You did not order anything on the previous page!<br />';
}
else{
  if ($tireqty>0)
    echo $tireqty.' tires<br />';
  if ($oilqty>0)
    echo $oilqty.' bottles of oil<br />';
  if ($sparkqty>0)
    echo $sparkqty.' spark plugs<br />';
}
$total = $tireqty * TIREPRICE + $oilqty * OILPRICE +
         $sparkqty * SPARKPRICE;
$total=number_format($total, 2, '.', ' ');
```

2 of 3

Note: number_format(*number,decimals,decimalpoint,separator*)

Writing to a File (cont.)

```
  echo '<P>Total of order is '.$total.'</p>';
  echo '<P>Address to ship to is '.$address.'<br />';
  $outputstring = $date."\t".$tireqty." tires \t".$oilqty." oil\t"
                  .$sparkqty." spark plugs\t\$".$total
                  ."\t". $address."\n";

  // open file for appending
  $fp = fopen("$DOCUMENT_ROOT/../orders/orders.txt", 'a');
  flock($fp, LOCK_EX);
  if (!$fp){
    echo '<p><strong> Your order could not be processed at this time.  '
         .'Please try again later.</strong></p></body></html>';
    exit;
  }
  fwrite($fp, $outputstring);
  flock($fp, LOCK_UN);
  fclose($fp);
  echo '<p>Order written.</p>';
?>
</body>
</html>
```

3 of 3

Closing a File

- When you've finished using a file, you need to close it.
- You should do this with the fclose() function as follows:

  fclose($fp);
- This function will return true if the file was successfully closed, or false if it wasn't.

Reading from a File (cont.)

- Reading a Line at a Time: **fgets(), fgetss(), and fgetcsv()**
  - fgets()
    - Prototype: **string fgets (resource handle [, *int length*]);**
    - This function is used to read one line at a time from a file. In this case, it will read until it encounters a newline character (\n) , encounters an EOF or has read **length - 1** bytes from the file.

```
<?php
  $handle = fopen("test.txt", "r");
  while(!feof($handle)){
    $buffer = fgets($handle, 4096);
    echo $buffer;
  }
  fclose($handle);
?>
```


Reading from a File (cont.)

- Reading a Line at a Time: **fgets(), fgetss(), and fgetcsv()**
  - fgetss()
    - Prototype: **string fgetss (resource handle, int length [, *string* [*allowable_tags*]);**
    - This function will strip out any PHP and HTML tags found in the string.
    - If you want to leave any particular tags in, you can include them in the allowable_tags string.

```
<?php
  $handle = fopen("test.txt", "r");
  while(!feof($handle)){
    $buffer = fgetss($handle, 4096, "<br>\n");
    echo $buffer;
  }
  fclose($handle);
?>
```


Reading from a File (cont.)

- Reading a Line at a Time: **fgets(), fgetss(), and fgetcsv()**
  - fgetcsv()
    - Prototype: **array fgetcsv (resource handle, int length [, *string delimiter* [, *string enclosure*]]);**
    - Used for breaking up lines of files when you have used a delimiting character such as the tab character or a comma.

	A	B	C
1	Widget1	blue	£10
2	Widget2	red	£12
3	Widget3	green	£14
4	Widget4	black	£16
5	Widget5	white	£18
6			

```
<?php
  $row = 1;
  $handle = fopen("test.txt", "r");
 while(($data = fgetcsv($handle, 1000, ",")) !== FALSE){
        $num = count($data);
        echo "<p> $num fields in line $row: <br>";
        for($c=0; $c<$num; $c++){
                echo $data[$c]."<br>";
        }
        $row++;
  }
  fclose($handle);
?>
```

widgets.csv - Notepad

File Edit Format View Help

```
Widget1, blue, £10
Widget2, red, £12
Widget3, green, £14
Widget4, black, £16
Widget5, white, £18
```


Reading from a File (cont.)

- Reading the Whole File: **readfile(), fpassthru(), and file()**
  - readfile()
    - Prototype: **int readfile (string *filename* , int [*use_nclude_path*]);**
    - This function will returns the number of bytes read on success, or FALSE and an error on failure. You can hide the error output by adding an '@' in front of the function name.opens the file, echoes the content to standard output (the browser), and then closes the file.

```
<?php
  $total = readfile("orders.txt");
  echo "<br>Total = $total bytes. ";
?>
```

Reading from a File (cont.)

- Reading the Whole File: **readfile(), fpassthru(), and file()**
  - fpassthru()
    - need to open the file using fopen() first.
    - For example:
      - » $fp = fopen("orders.txt", 'r');
      - » fpassthru($fp);
    - Dump the contents of the file from the pointer 's position onward to standard output.
    - Close the file when it is finished.

```
<?php
   $fp = fopen("orders.txt", "r");
   fpassthru($fp);
?>
```


Reading from a File (cont.)

- Reading the Whole File: **readfile(), fpassthru(), and file()**
  - file()
    - Identical to readfile(), except that instead of echoing the file to standard output, it turns it into an array.
    - For example:
      - » $filearray = file($fp);

Return Value: Returns the file in an array. Each element of the array corresponds to a line in the file, with the newline still attached. Upon failure, **file()** returns **FALSE**

```
<?php
  $data = file("orders.txt");
  foreach($data as $line_num => $line){
    echo "Line #<b>$line_num</b>: $line<br>";
  }
?>
```


Reading from a File (cont.)

- Reading a Character: **fgetc()**
  - Read a single character at a time from a file.

```
<?php
  $fp = fopen("orders.txt", "r");
  if(!$fp){
    exit("Could not open file \"orders.txt\"");
  }
  while(($char = fgetc($fp)) !== false){
    if($char !== "\n")
       echo "$char";
    else
       echo "<br>";
  }
?>
```

- Reading an Arbitrary Length: **fread()**
  - Prototype: **string fread(int *fp*, int *length*);**
  - Read up to *length* bytes or to the end of file, whichever comes first.

Reading from a File (cont.)

```
<html>
<head>
  <title>Bob's Auto Parts - Customer Orders</title>
</head>
<body>
  <h1>Bob's Auto Parts</h1>
  <h2>Customer Orders</h2>
  <?php
    @ $fp = fopen("$DOCUMENT_ROOT/../orders/orders.txt", 'rt');
   if (!$fp){
     echo '<p><strong>No orders pending.'.'Please try again later.</strong></p>';
     exit;
   }
   while (!feof($fp)){
     $order= fgets($fp, 999);
     echo $order.'<br>';
   }
   fclose($fp);
  ?>
</body>
</html>
```

Other Useful File Functions

- Checking Whether a File Is There: **file_exists()**
  - If you want to check if a file exists without actually opening it, you can use file_exists(), as follows:

```
if (file_exists("$DOCUMENT_ROOT/../orders/orders.txt")
   echo 'There are orders waiting to be processed.';
else
   echo 'There are currently no orders.';
```

- Knowing How Big a File Is: **filesize()**
  - Check the size of a file with the filesize() function.
  - Returns the size of file in bytes
  - For example:
    - echo filesize("$DOCUMENT_ROOT/../orders/orders.txt");

Other Useful File Functions (cont.)

- Deleting a File: **unlink()**
  - Delete the file after the orders have been processed.
  - For example:
    - unlink("orders.txt");
- Navigating Inside a File: **rewind(), fseek(), and ftell()**
  - The **rewind()** function resets the file pointer to the beginning of the file.
  - The **ftell()** function reports how far into the file the pointer is in bytes.
  - The function **fseek()** can be used to set the file pointer to some point within the file. Its prototype is

    int fseek (int *fp* , int *offset* [, int *whence*])
    - Set the file pointer *fp* at a point starting from *whence* and moving *offset* bytes into the file

File Locking

- Using function flock()
  - Prototype: **bool flock(int *fp* , int *operation* [, int *&wouldblock*])**
  - This function should be called after a file has been opened, but before any data is read from or written to the file.
  - Note that flock() does not work with NFS or other networked file systems. It also does not work with older file systems that do not support locking such as FAT.

Value of operation	Meaning
LOCK_SH (formerly 1)	Reading lock. This means the file can be shared with other readers.
LOCK_EX (formerly 2)	Writing lock. This is exclusive. The file cannot be shared.
LOCK_UN (formerly 3)	Release existing lock.
LOCK_NB (formerly 4)	*Adding* 4 to the operation prevents blocking while trying to acquire a lock.

File Locking (cont.)

- Using function flock()
  - You can alter processorder.php as follows:

```
$fp = fopen ("$DOCUMENT_ROOT/../orders/orders.txt", 'at');
flock($fp, LOCK_EX); // lock the file for writing
fwrite($fp, $outputstring);
flock($fp, LOCK_UN); // release write lock
fclose($fp);
```

  - You should also add locks to vieworders.php

```
$fp = fopen("$DOCUMENT_ROOT/../orders/orders.txt", 'rt');
flock($fp, LOCK_SH); // lock file for reading
// read from the file
flock($fp, LOCK_UN); // release read lock
fclose($fp);
```

Storing and Retrieving Data

Problem With Using Flat Files

- When a file gets large, it can be very slow to work with.
- Searching for a particular record or group of records in a flat file is difficult.
- Dealing with concurrent access can become problematic.
- Significant overhead because of using sequential processing
- Beyond the limits offered by file permissions, there is no easy way of enforcing different levels of access to data.

Array & File

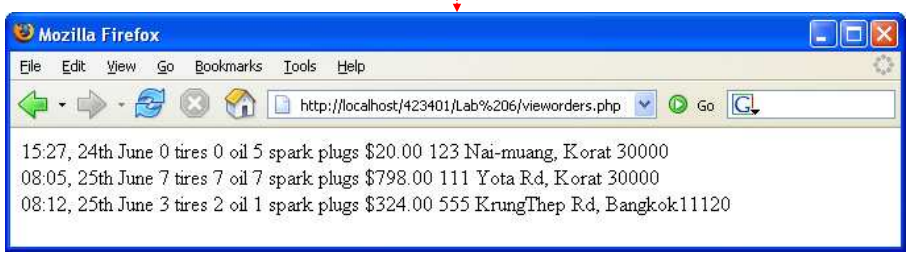
Loading Arrays from Files

- Using **file()** to load the entire file into an array
- Each line in the file becomes one element of an array.
- Using **count()** function to see how many elements are in an array.
- Function **explode()** is used to split up each line, so that we can apply some processing and formatting before printing.

Array & File

Loading Arrays from Files (cont.)

```
<?php
   //create short variable name
   $DOCUMENT_ROOT = $_SERVER['DOCUMENT_ROOT'];
   $orders= file("$DOCUMENT_ROOT/../orders/orders.txt");
   $number_of_orders = count($orders);
   if ($number_of_orders == 0){
      echo '<p><strong>No orders pending. Please try again later.</strong></p>';
   }
   for ($i=0; $i<$number_of_orders; $i++){
      echo $orders[$i].'<br />';
   }
?>
```



Array & File

Loading Arrays from Files (cont. [Display in table])

```
<?php
   //create short variable name
   $DOCUMENT_ROOT = $_SERVER['DOCUMENT_ROOT'];
?>
<html>
   <head><title>Bob's Auto Parts - Customer Orders</title></head>
<body>
   <h1>Bob's Auto Parts</h1>
   <h2>Customer Orders</h2>
<?php
   //Read in the entire file.
   //Each order becomes an element in the array
   $orders= file("$DOCUMENT_ROOT/../orders/orders.txt");
   // count the number of orders in the array
   $number_of_orders = count($orders);
   if ($number_of_orders == 0){
      echo '<p><strong>No orders pending.Please try again later.</strong></p>';
   }
```

Loading Arrays from Files (cont.)

```
   echo "<table border=1>\n";
   echo '<tr><th bgcolor="#CCCCFF">Order Date</th>
            <th bgcolor="#CCCCFF">Tires</th>
            <th bgcolor="#CCCCFF">Oil</th>
            <th bgcolor="#CCCCFF">Spark Plugs</th>
            <th bgcolor="#CCCCFF">Total</th>
            <th bgcolor="#CCCCFF">Address</th></tr>';
   for ($i=0; $i<$number_of_orders; $i++){
      $line = explode( "\t", $orders[$i] ); //split up each line
      // keep only the number of items ordered
      $line[1] = intval( $line[1] );
      $line[2] = intval( $line[2] );
      $line[3] = intval( $line[3] );
      echo "<tr><td>$line[0]</td>// output each order
               <td align="right">$line[1]</td>
               <td align="right">$line[2]</td>
               <td align="right">$line[3]</td>
               <td align="right">$line[4]</td>
               <td>$line[5]</td></tr>";
   }
   echo "</table>";
?>
</body>
</html>
```

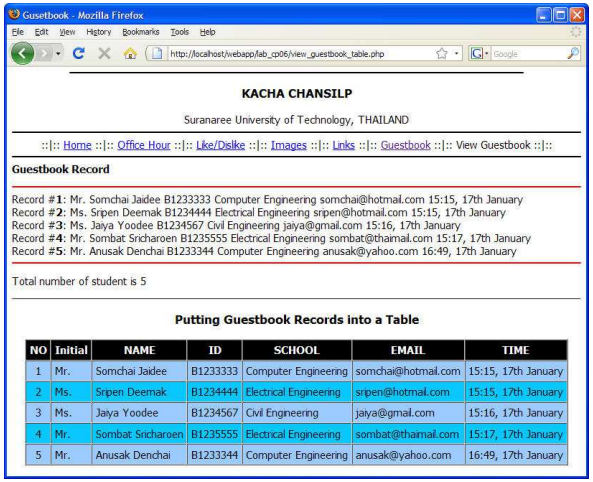

Loading Arrays from Files (cont.)

- Output from previous slide.

Order Date	Tires	Oil	Spark Plugs	Total	Address
15:27, 24th June	0	0	5	$20.00	123 Nai-muang, Korat 30000
08:05, 25th June	7	7	7	$798.00	111 Yota Rd, Korat 30000
08:12, 25th June	3	2	1	$324.00	555 KrungThep Rd, Bangkok11120

PHP IV: Using “file”

- Laboratory 7
  - Page 172 (1-4)



NO	Initial	NAME	ID	SCHOOL	EMAIL	TIME
1	Mr.	Somchai Jaidee	B1233333	Computer Engineering	somchai@hotmail.com	15:15, 17th January
2	Ms.	Sripen Deemak	B1234444	Electrical Engineering	sripen@hotmail.com	15:15, 17th January
3	Ms.	Jaiya Yoodee	B1234567	Civil Engineering	jaiya@gmail.com	15:16, 17th January
4	Mr.	Sombat Sricharoen	B1235555	Electrical Engineering	sombat@thaimail.com	15:17, 17th January
5	Mr.	Anusak Denchai	B1233344	Computer Engineering	anusak@yahoo.com	16:49, 17th January