Ntec
NEW ZEALAND TERTIARY GROUP

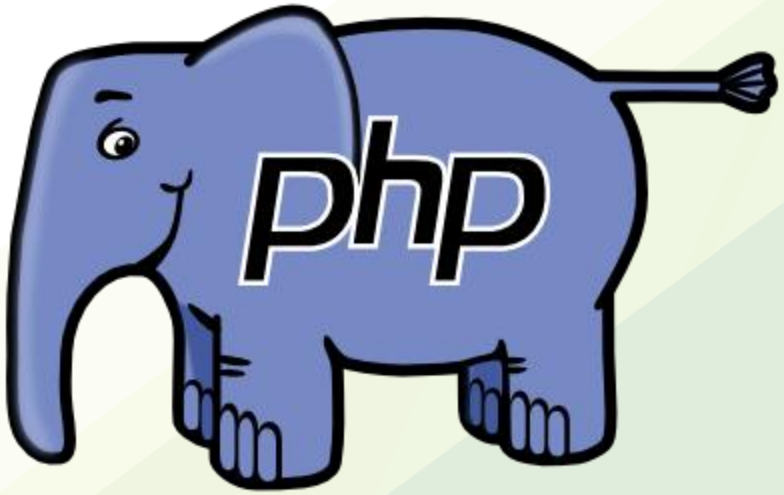
Concordia Institute of Business
National Institute of Education
National Technology Institute
The College of Future Learning NZ

AUCKLAND
CHRISTCHURCH
HASTINGS
TAURANGA

INT621 Internet Programming

LECTURE 06
PHP Part 2

PHP

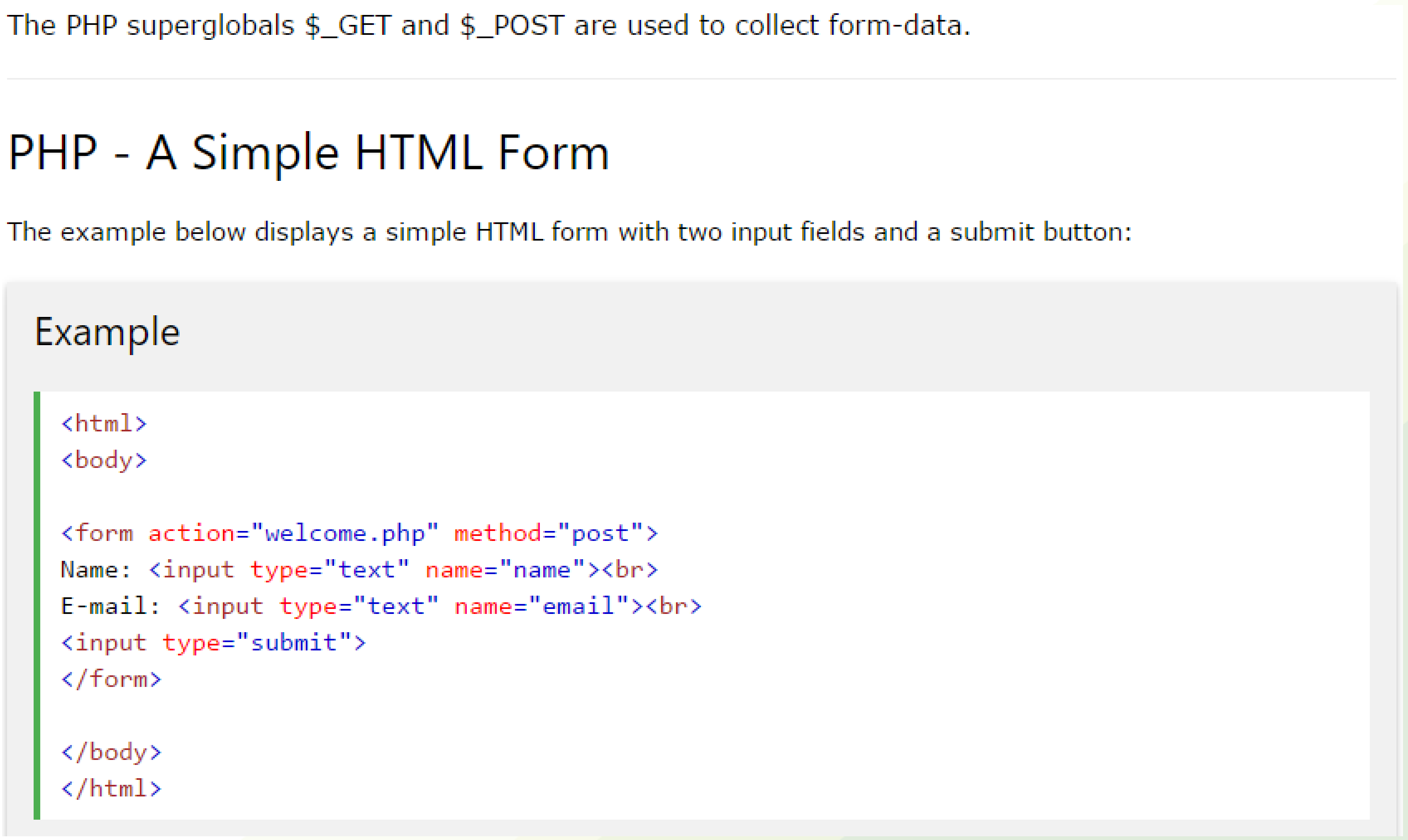


PHP Forms

- HTML forms have two main functions with PHP:
    - Form Handling
    - Form Validation

Complete list of reference here.

PHP Form Handling



The PHP superglobals $_GET and $_POST are used to collect form-data.

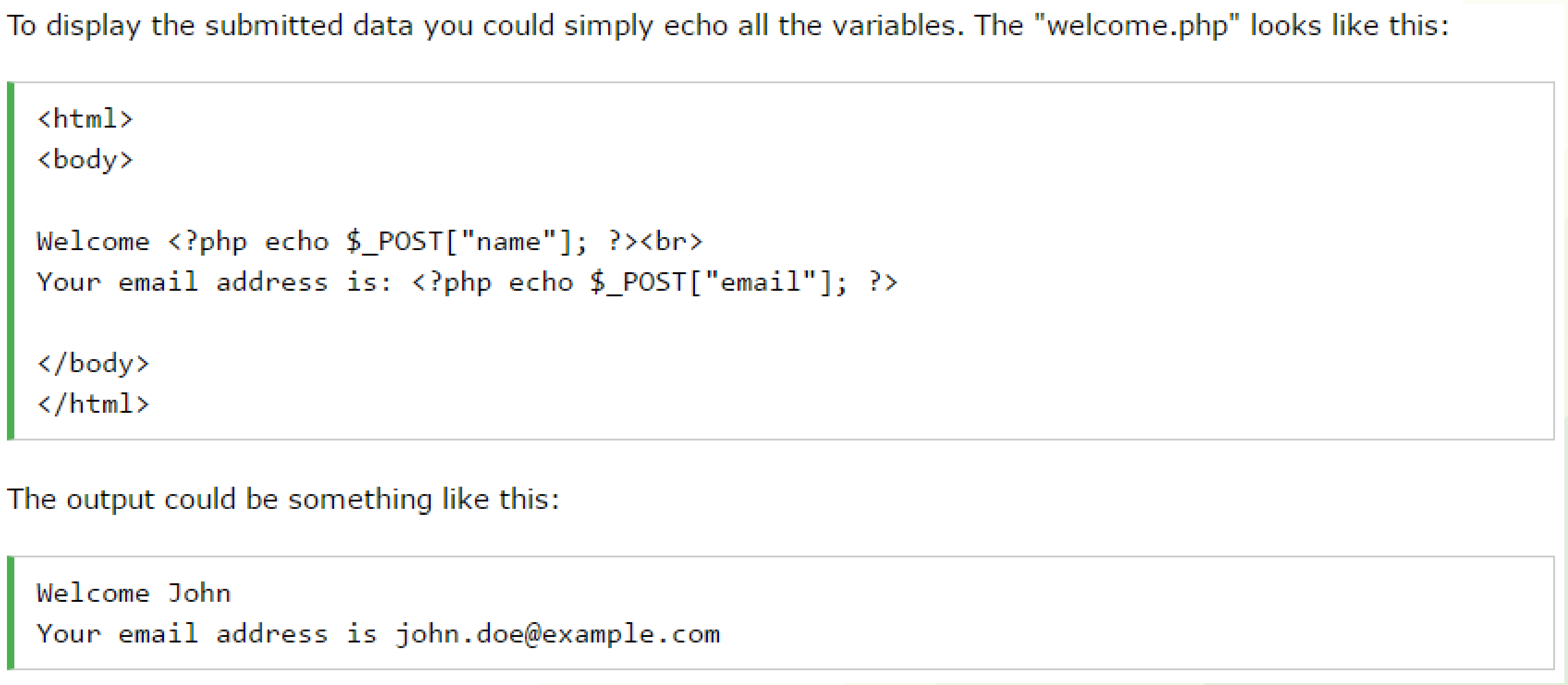
PHP - A Simple HTML Form

The example below displays a simple HTML form with two input fields and a submit button:

Example

```
<html>
<body>

<form action="welcome.php" method="post">
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>

</body>
</html>
```

PHP Form Handling



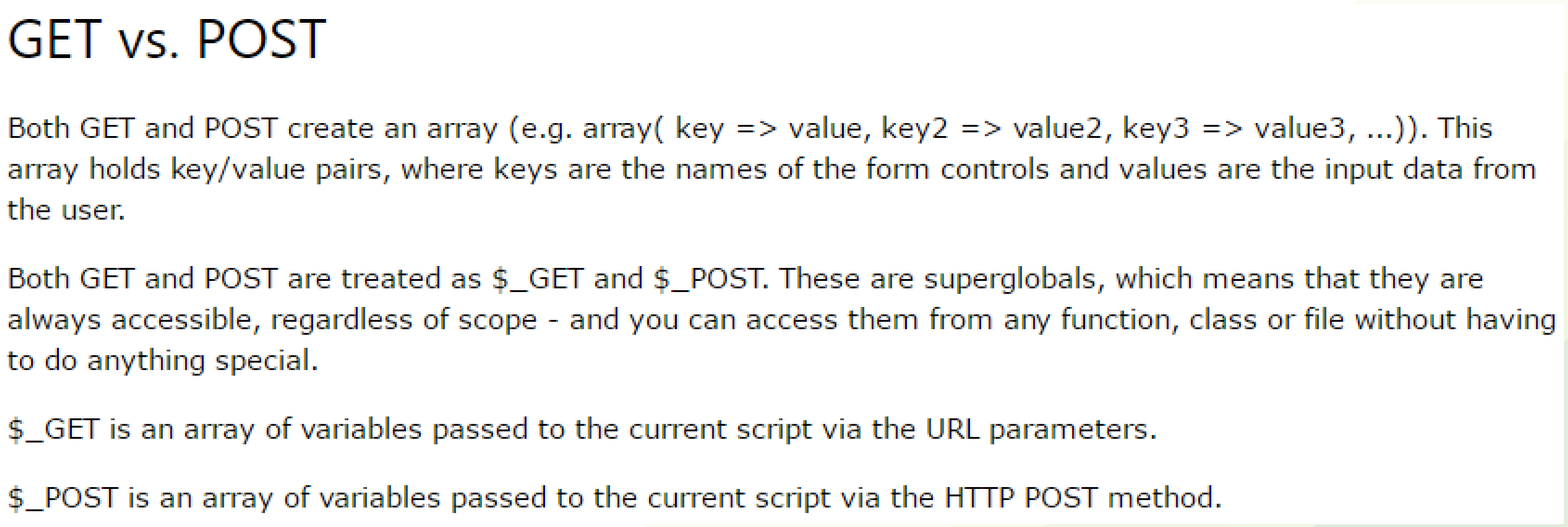
To display the submitted data you could simply echo all the variables. The "welcome.php" looks like this:

```
<html>
<body>

Welcome <?php echo $_POST["name"]; ?><br>
Your email address is: <?php echo $_POST["email"]; ?>

</body>
</html>
```

The output could be something like this:

```
Welcome John
Your email address is john.doe@example.com
```

PHP Form Handling



GET vs. POST

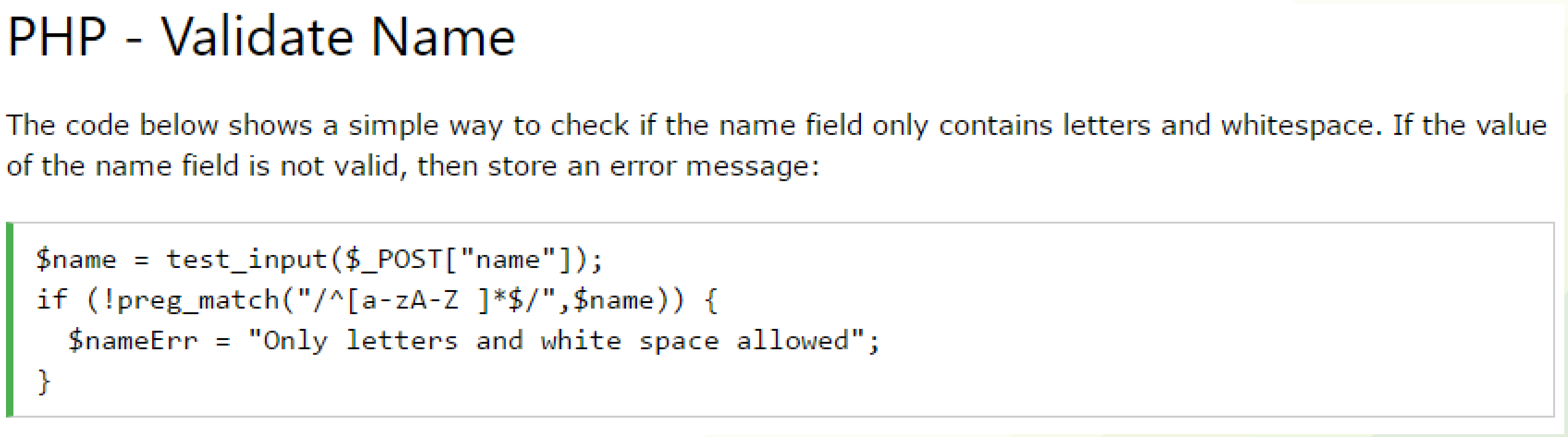
Both GET and POST create an array (e.g. array(key => value, key2 => value2, key3 => value3, ...)). This array holds key/value pairs, where keys are the names of the form controls and values are the input data from the user.

Both GET and POST are treated as $_GET and $_POST. These are superglobals, which means that they are always accessible, regardless of scope - and you can access them from any function, class or file without having to do anything special.

$_GET is an array of variables passed to the current script via the URL parameters.

$_POST is an array of variables passed to the current script via the HTTP POST method.

PHP Validation Examples

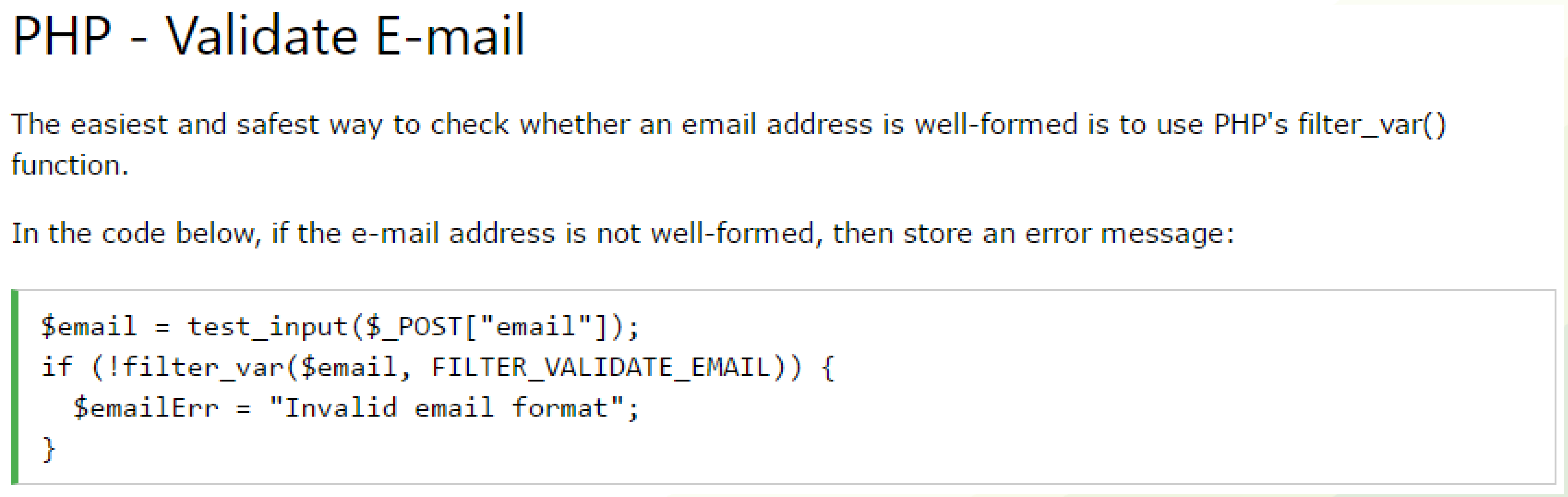


PHP - Validate Name

The code below shows a simple way to check if the name field only contains letters and whitespace. If the value of the name field is not valid, then store an error message:

```
$name = test_input($_POST["name"]);
if (!preg_match("/^[a-zA-Z ]*$/",$name)) {
  $nameErr = "Only letters and white space allowed";
}
```

PHP Validation Examples



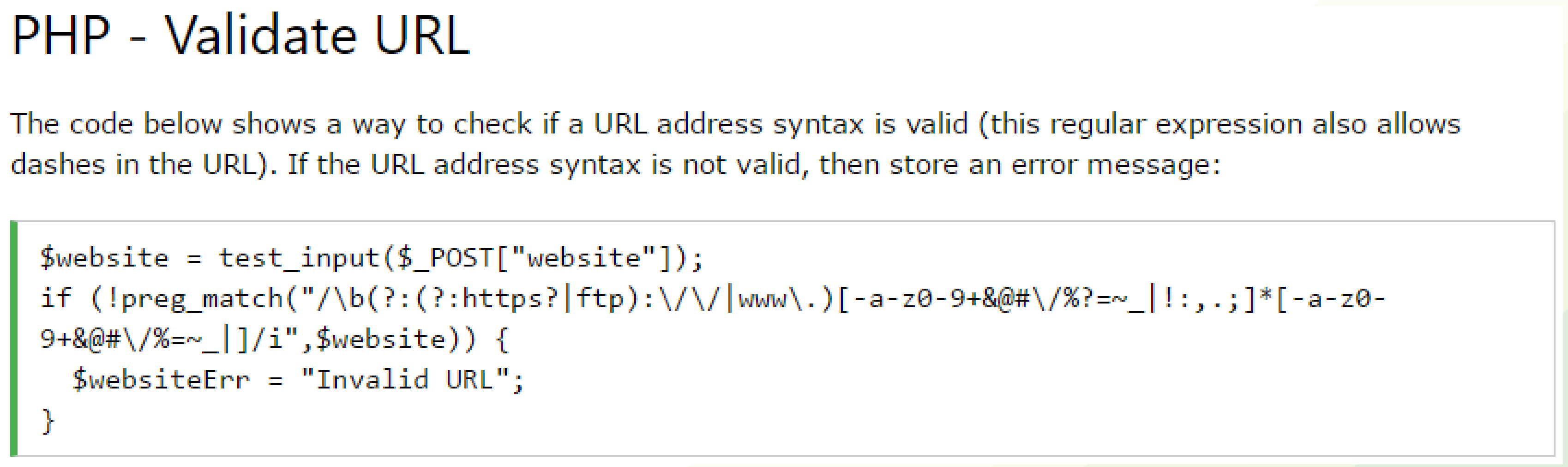
PHP - Validate E-mail

The easiest and safest way to check whether an email address is well-formed is to use PHP's filter_var() function.

In the code below, if the e-mail address is not well-formed, then store an error message:

```
$email = test_input($_POST["email"]);
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
  $emailErr = "Invalid email format";
}
```

PHP Validation Examples

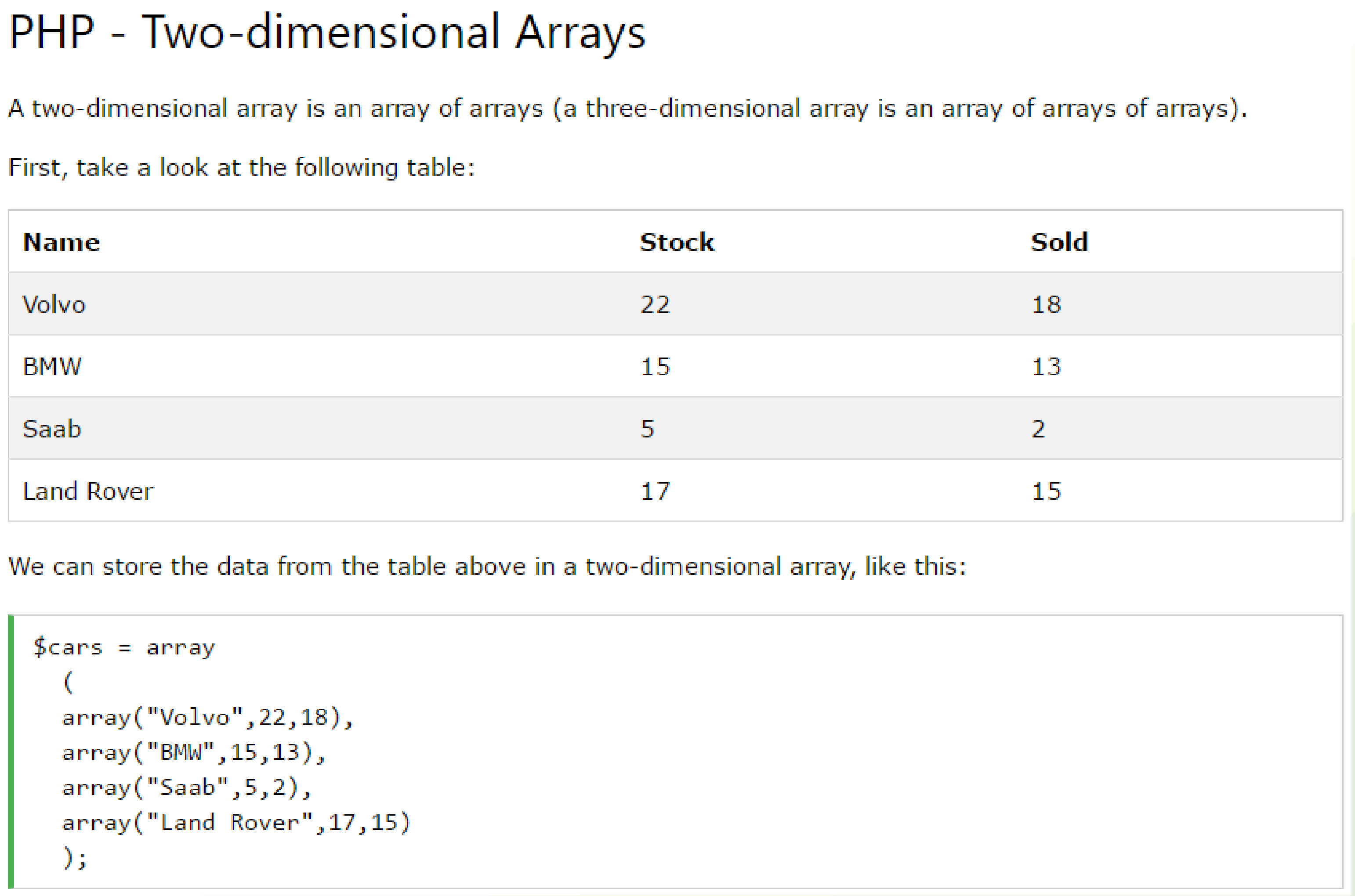


PHP - Validate URL

The code below shows a way to check if a URL address syntax is valid (this regular expression also allows dashes in the URL). If the URL address syntax is not valid, then store an error message:

```
$website = test_input($_POST["website"]);
if (!preg_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~_|!:,.;]*[-a-z0-
9+&@#\/%=~_|]/i",$website)) {
  $websiteErr = "Invalid URL";
}
```

PHP Multidimensional Arrays



PHP - Two-dimensional Arrays

A two-dimensional array is an array of arrays (a three-dimensional array is an array of arrays of arrays).

First, take a look at the following table:

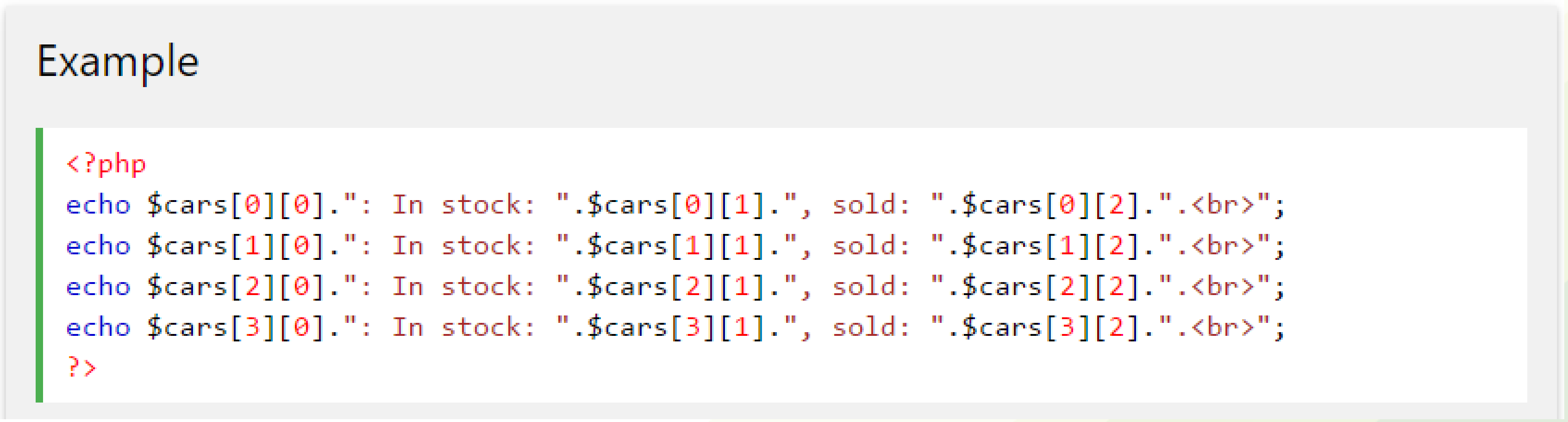
Name	Stock	Sold
Volvo	22	18
BMW	15	13
Saab	5	2
Land Rover	17	15

We can store the data from the table above in a two-dimensional array, like this:

```
$cars = array
  (
  array("Volvo",22,18),
  array("BMW",15,13),
  array("Saab",5,2),
  array("Land Rover",17,15)
  );
```

Complete list of reference here.

PHP Multidimensional Arrays

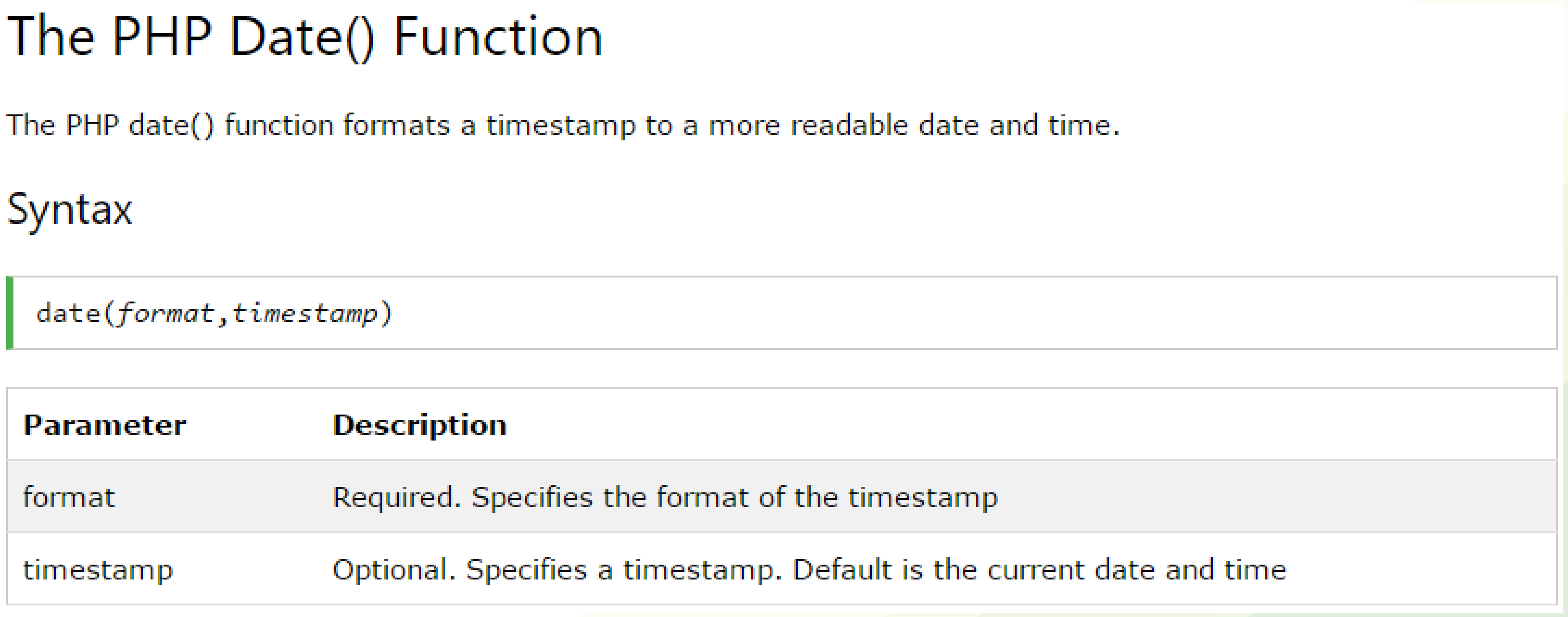


Example

```php
<?php
echo $cars[0][0].": In stock: ".$cars[0][1].", sold: ".$cars[0][2].".<br>";
echo $cars[1][0].": In stock: ".$cars[1][1].", sold: ".$cars[1][2].".<br>";
echo $cars[2][0].": In stock: ".$cars[2][1].", sold: ".$cars[2][2].".<br>";
echo $cars[3][0].": In stock: ".$cars[3][1].", sold: ".$cars[3][2].".<br>";
?>
```

Complete list of reference here.

PHP Date( )



The PHP Date() Function

The PHP date() function formats a timestamp to a more readable date and time.

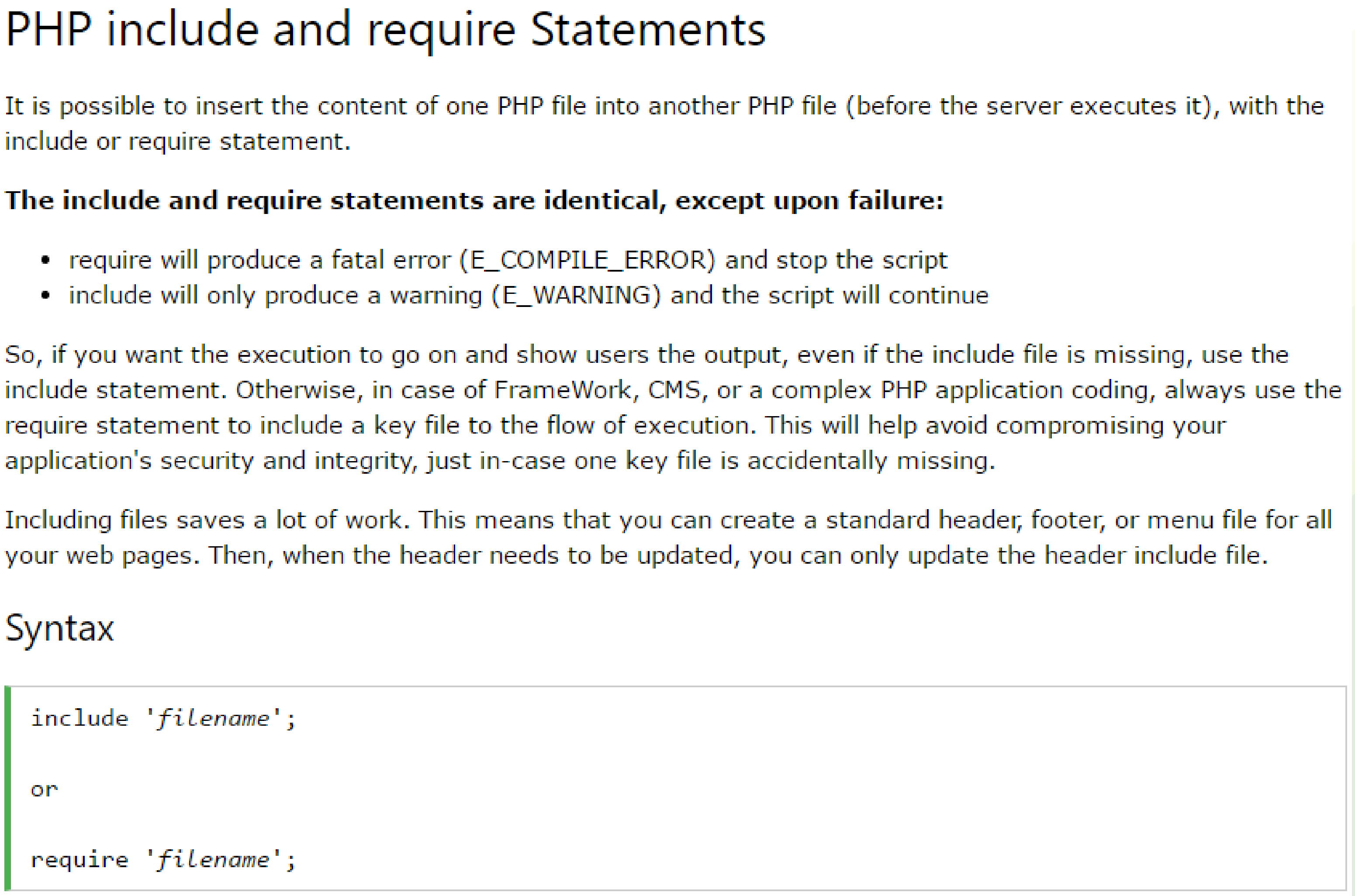
Syntax

```
date(format,timestamp)
```

Parameter	Description
format	Required. Specifies the format of the timestamp
timestamp	Optional. Specifies a timestamp. Default is the current date and time

Complete list of reference here.

PHP Include & Require



PHP include and require Statements

It is possible to insert the content of one PHP file into another PHP file (before the server executes it), with the include or require statement.

The include and require statements are identical, except upon failure:

- require will produce a fatal error (E_COMPILE_ERROR) and stop the script
- include will only produce a warning (E_WARNING) and the script will continue

So, if you want the execution to go on and show users the output, even if the include file is missing, use the include statement. Otherwise, in case of FrameWork, CMS, or a complex PHP application coding, always use the require statement to include a key file to the flow of execution. This will help avoid compromising your application's security and integrity, just in-case one key file is accidentally missing.

Including files saves a lot of work. This means that you can create a standard header, footer, or menu file for all your web pages. Then, when the header needs to be updated, you can only update the header include file.

Syntax

```
include 'filename';

or

require 'filename';
```

Complete list of reference here.

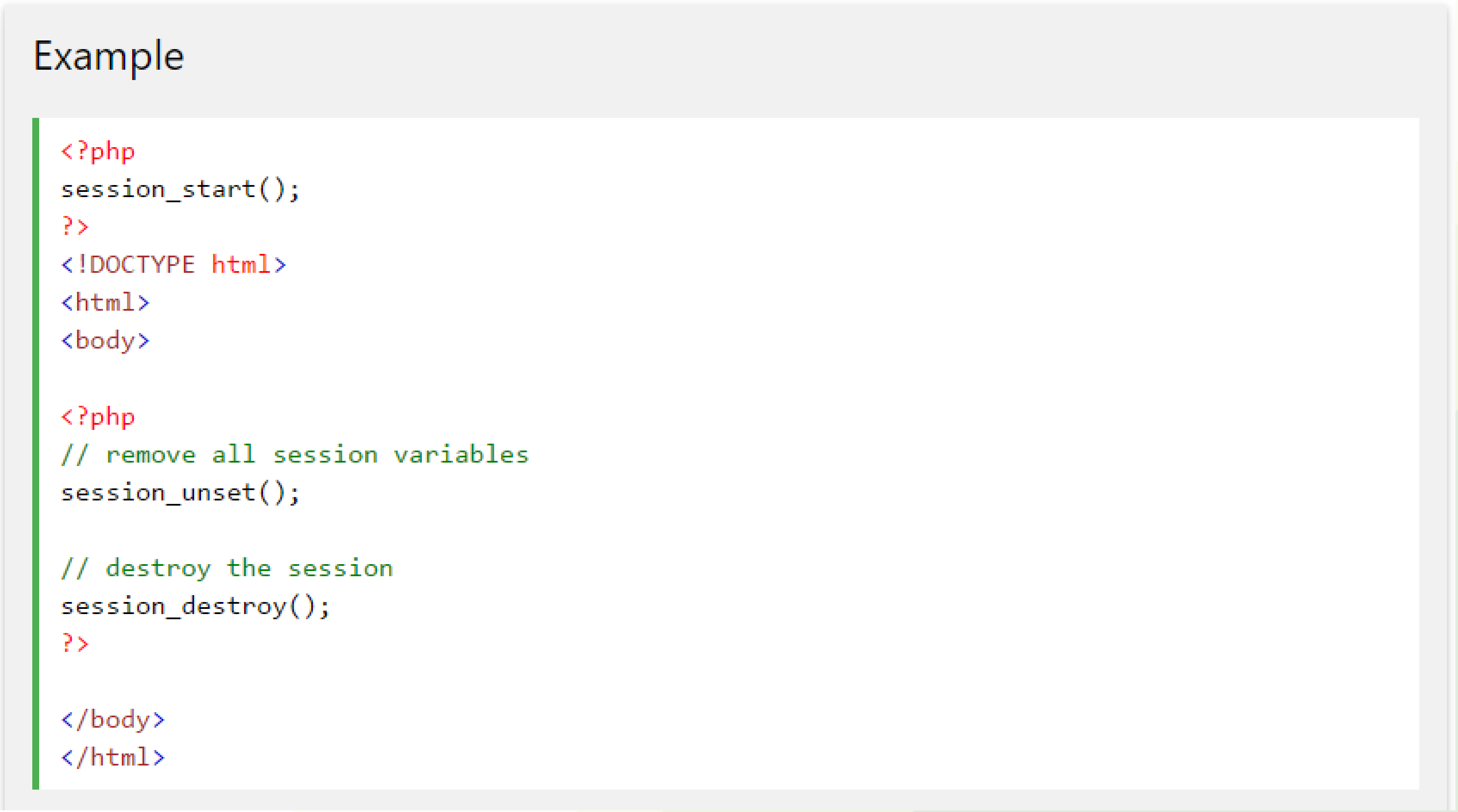
PHP File Handling

File handling is an important part of any web application. You often need to open and process a file for different tasks.

PHP has several functions for creating, reading, uploading, and editing files.

Complete list of reference here.

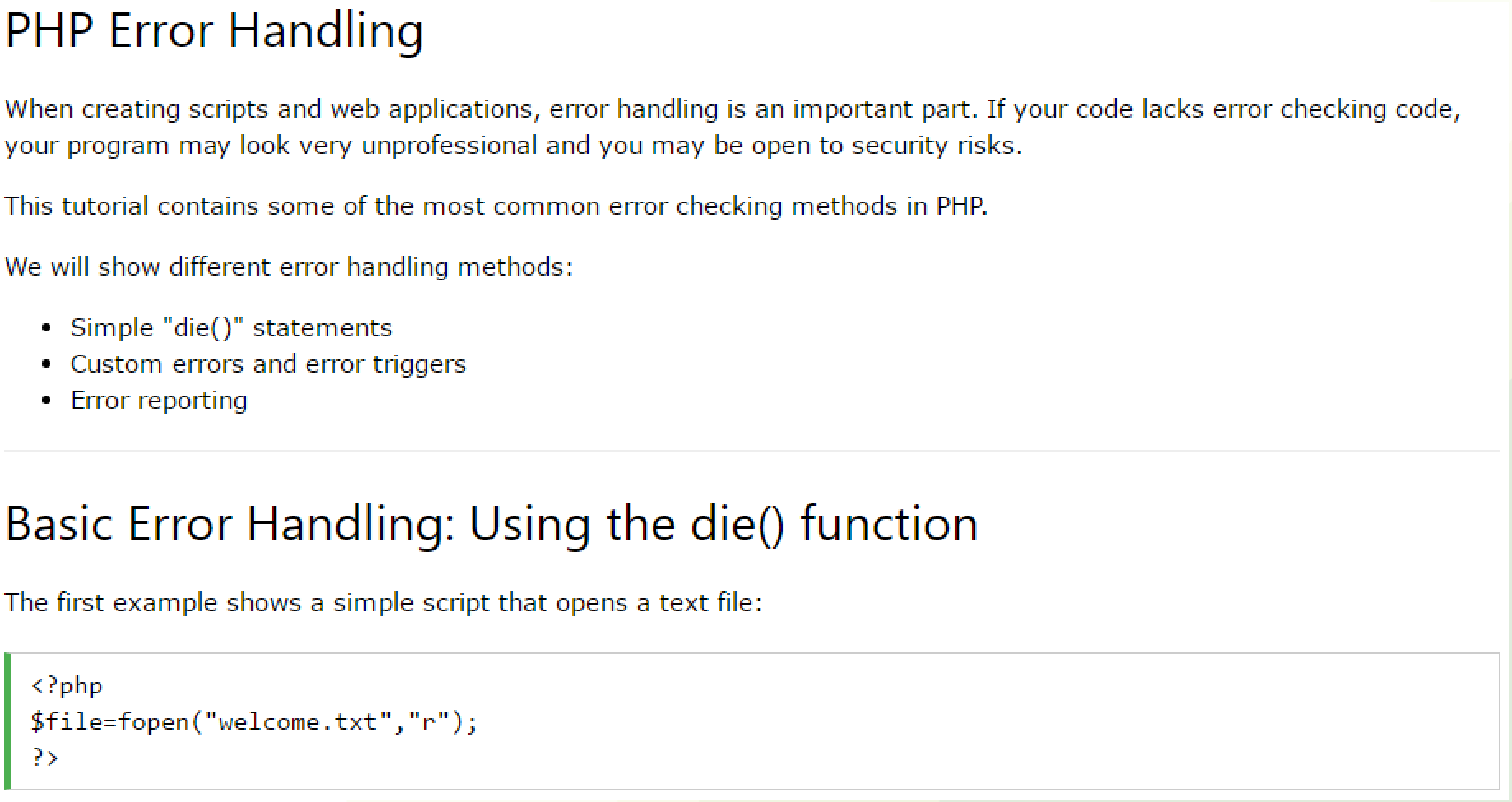
PHP Session Handling


Example

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// remove all session variables
session_unset();

// destroy the session
session_destroy();
?>

</body>
</html>
```

Complete list of reference here.

PHP Error Handling



PHP Error Handling

When creating scripts and web applications, error handling is an important part. If your code lacks error checking code, your program may look very unprofessional and you may be open to security risks.

This tutorial contains some of the most common error checking methods in PHP.

We will show different error handling methods:

- Simple "die()" statements
- Custom errors and error triggers
- Error reporting

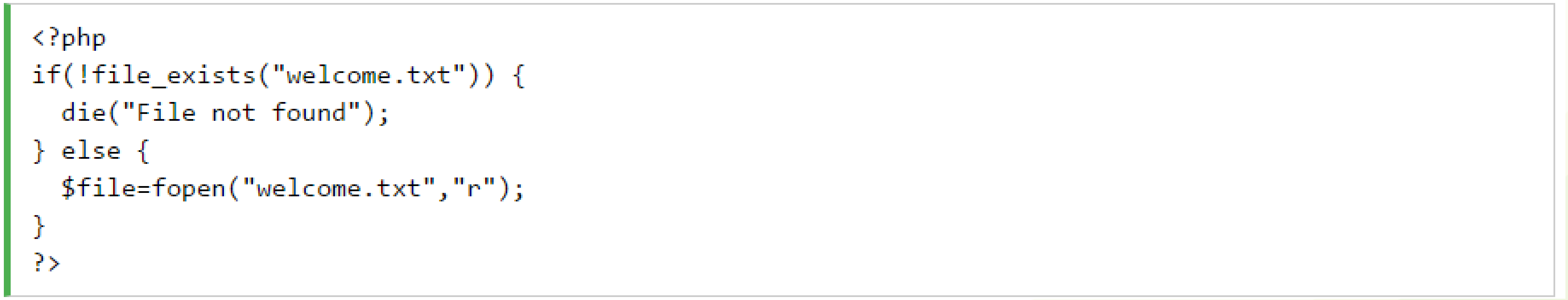
Basic Error Handling: Using the die() function

The first example shows a simple script that opens a text file:

```
<?php
$file=fopen("welcome.txt","r");
?>
```

Complete list of reference here.

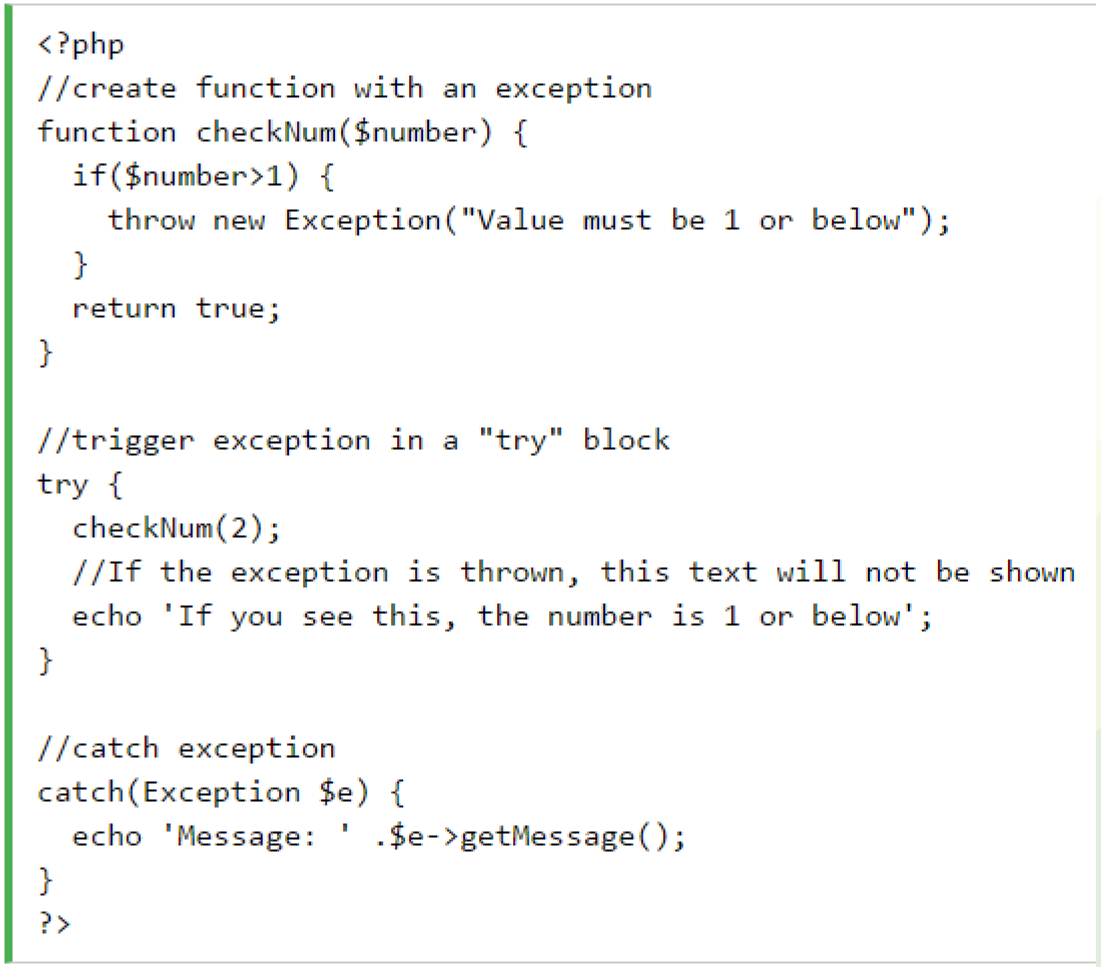
PHP Error Handling



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

Complete list of reference here.

PHP Exception Handling



```php
<?php
//create function with an exception
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);
  //If the exception is thrown, this text will not be shown
  echo 'If you see this, the number is 1 or below';
}

//catch exception
catch(Exception $e) {
  echo 'Message: ' .$e->getMessage();
}
?>
```

Complete list of reference here.

Connecting to a MySQL DBMS

- **In order for our PHP script to access a database we need to form a connection from the script to the database management system.**

```
resourceId = mysql_connect(server, username, password);
```

- Server is the DBMS server
- username is your username
- password is your password

Connecting to a MySQL

- **In order for our PHP script to access a database we need to form a connection from the script to the database management system.**

```
resourceId = mysql_connect(server, username, password);
```

- The function returns a resource-identifier type.
- a PHP script can connect to a **DBMS** **anywhere in the world**, so long as it is connected to the internet.
- we can also connect to multiple DBMS at the same time.

Selecting a database

- **Once connected to a DBMS, we can select a database.**

```
mysql_select_db(databasename, resourceId);
```

• the resourceId is the one returned by mysql_connect()
• the function returns true if the selection succeeded; false, otherwise.

Example: Connect to a DBMS and access database

```
<?php
$dbLocalhost = mysql_connect("localhost", "root", "")
    or die("Could not connect: " . mysql_error());
mysql_select_db("glassesrus", $dbLocalhost)
    or die("Could not find database: " . mysql_error());
echo "<h1>Connected To Database</h1>";
?>
```

- **die()** stops execution of script if the database connection attempt failed.
- **mysql_error()** returns an error message from the previous MYSQL operation.

Reading from a database

- We can now send an SQL query to the database to retrieve some data records.

```
resourceRecords = mysql_query(query, resourceId);
```

- the resourceId is the one returned by mysql_connect()
- the function returns a resource identifier to the returned data.

Example: Connect to a DBMS, access database, send query

```php
<?php
$dbLocalhost = mysql_connect("localhost", "root", "")
     or die("Could not connect: " . mysql_error());
mysql_select_db("glassesrus", $dbLocalhost)
     or die("Could not find database: " . mysql_error());
$dbRecords = mysql_query("SELECT * FROM customers", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());
echo "<h1>Connected To Database</h1>";
?>
```

- **the function will return a resource pointer (not the actual data) to all the records that match the query.**
- **If all goes well, this script will output nothing on screen.**

Extract contents of one

- **We can now extract the actual data from the resource pointer returned by mysql_query().**

```
fieldData= mysql_result(resourceRecords, row, field);
```

- the resourceRecords is the one returned by mysql_query()
- field – database field to return
- the function returns the **data stored in the field**.

Example: Connect to a DBMS, access database, send query

```
<?php
$dbLocalhost = mysql_connect("localhost", "root", "")
     or die("Could not connect: " . mysql_error());
mysql_select_db("glassesrus", $dbLocalhost)
     or die("Could not find database: " . mysql_error());
$dbRecords = mysql_query("SELECT * FROM customers", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());
$strSurname = mysql_result($dbRecords, 0, "Surname");
echo "<p>$strSurname</p>";
?>
```

- **the function will return a resource pointer (not the actual data) to all the records that match the query.**
- **If all goes well, this script will output a surname on screen.**

SQL statement

```
SELECT * FROM customers
```

- Go and **obtain** from the database
- **every** field
- **FROM** the
- **customers** table

Separating the database connection

It is worth separating the database connectivity from our scripts and placing it in a separate file.

- It provides a convenient means of moving your scripts from one database platform to another.

Example: Separating the database connection

```
<?php
// File: database2.php
$strLocation = "Home";
//$strLocation = "Work";
if ($strLocation == "Home") {
    $dbLocalhost = mysql_connect("localhost", "root", "")
                or die("Could not connect: " . mysql_error());
            mysql_select_db("glassesrus", $dbLocalhost)
            or die("Could not find database: " . mysql_error());
} else {
    $dbLocalhost = mysql_connect("localhost", "username",
"password")
                or die("Could not connect: " . mysql_error());
            or die("Could not find database: " . mysql_error());
```

- **$strLocation could be easily switched between 'Home' or 'Work'**

Viewing a whole record

To view the whole record returned from mysql_query(), we need another function...

```
array = mysql_fetch_row(resourceRecords)
```

• resourceRecords – resource identifier returned from mysql_query().

• it returns an array containing the database record.

Example: Displaying all customer records

```
<?php

require_once("database2.php");

$dbRecords = mysql_query("SELECT * FROM customers", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());

while ($arrRecord = mysql_fetch_row($dbRecords)) {
    echo "<p>" . $arrRecord[0] . " ";
    echo         $arrRecord[1] . " ";
    echo         $arrRecord[2] . " ";
    echo         $arrRecord[3] . "</p>";
}
?>
```

- **The function returns false when the last record is returned; thus, stopping the loop.**
- **Note, however, that the fields are referred to by using numbers – not very easy to read and mistakes can be**

Limiting the records returned

```
SELECT Surname FROM customers
```

•Retrieves only the Surname field from the table customers

Limiting the records returned

```
SELECT * FROM customers LIMIT 3,4
```

- Select a certain number of records form a table
- 3 is the starting row
- 4 is the number of records to be selected after the starting row

Searching for matching records

```
SELECT * FROM customers WHERE Title=‘Mr’
```

•The WHERE attribute specifies what to search for within the database records.

• in this example, only records which have a title of ‘Mr’ will be returned.

Searching for matching records

SELECT * FROM customers WHERE Title=‘Mr’ OR Title=‘Mrs’

- The WHERE attribute specifies what to search for within the database records.
- in this example, only records which have a title of ‘Mr’ or ‘Mrs’ will be returned.
- we can also use **AND** and **OR** to formulate more sophisticated conditions.
-

Searching for matching records

```
SELECT * FROM customers WHERE Title='Mr'
   AND Surname='Smith' OR Title='Mrs'
```

- The WHERE attribute specifies what to search for within the database records.
- in this example, only records which have a surname of 'Smith and title of 'Mr' or the title of 'Mrs' will be returned.
- we can also use **AND** and **OR** to formulate more sophisticated conditions.
-

Sorting records

The ORDER BY attribute can be used to sort the order in which records are obtained.

```
SELECT * FROM cutomers ORDER BY Surname DESC
```

- the ORDER BY attribute is followed by the data field on which to sort the record

- DESC or ASC – from high to low, or from low to high

Accessing Multiple Tables

```
<?php
// File: example15-13.php

require_once("database2.php");

$dbRecords = mysql_query("SELECT * FROM customers WHERE Title = 'Mrs'", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());

echo "<p>Customers:</p>";
while ($arrRecords = mysql_fetch_array($dbRecords)) {
    echo "<p>" . $arrRecords["Id"] . " ";
    echo $arrRecords["Title"] . " ";
    echo $arrRecords["Surname"] . " ";
    echo $arrRecords["Firstname"] . "</p>";
}

//...continued...
```

Accessing Multiple Tables

```
//continuation...

$dbRecords = mysql_query("SELECT * FROM products WHERE Name = 'Wine Glass'", 
$dbLocalhost)
    or die("Problem reading table: " . mysql_error());

echo "<p>Products:</p>";
while ($arrRecords = mysql_fetch_array($dbRecords)) {
    echo "<p>" . $arrRecords["Id"] . " ";
    echo $arrRecords["Name"] . " ";
    echo $arrRecords["Description"] . " ";
    echo $arrRecords["Quantity"] . " ";
    echo $arrRecords["Cost"] . "</p>";
}
?>
```

Using records to read another table

Read a customer record, and then show the products purchased by that customer.

<u>Tables</u>

- Customers
- Products
- Purchases
- PurchaseProducts

BIRD'S EYEVIEW

Using records to read another table

```
...
$strSurname = "Jones";
$dbCustRecords = mysql_query("SELECT * FROM customers WHERE Surname = '$strSurname' ",...)
while ($arrCustRecords = mysql_fetch_array($dbCustRecords)) { //#1
    $intId = $arrCustRecords["Id"];
    //display customer's details
    $dbPurRecords = mysql_query("SELECT * FROM purchases WHERE customers_Id = '$intId'", ...)
while ($arrPurRecords = mysql_fetch_array($dbPurRecords)) {//#2
    $intPurId = $arrPurRecords["Id"];
    //display purchase date
$dbProRecords=mysql_query("SELECT * FROM purchaseProducts WHERE purchases_Id='$intPurId'
",..)
    while ($arrProRecords = mysql_fetch_array($dbProRecords)) { //#3
        $intProductId = $arrProRecords["products_Id"];
        //display Quantity
        $dbProductRecords = mysql_query("SELECT * FROM products WHERE Id = '$intProductId'",..)
        $arrProductRecord = mysql_fetch_array($dbProductRecords);
        //display product details
     } #3
   } #2
} //#1
```

Complete version

Using records to read another table

```
<?php
require_once("database2.php");

$strSurname = "Jones";

$dbCustRecords = mysql_query("SELECT * FROM customers WHERE Surname = '$strSurname'
", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());

while ($arrCustRecords = mysql_fetch_array($dbCustRecords)) {
    $intId = $arrCustRecords["Id"];
    echo "<p>Customer: ";
    echo $arrCustRecords["Title"] . " ";
    echo $arrCustRecords["Surname"] . " ";
    echo $arrCustRecords["Firstname"] . "</p>";

    $dbPurRecords = mysql_query("SELECT * FROM purchases WHERE customers_Id = '$intId'",
$dbLocalhost)
    or die("Problem reading table: " . mysql_error());
```

Complete version

Using records to read another table

```
while ($arrPurRecords = mysql_fetch_array($dbPurRecords)) {
        $intPurId = $arrPurRecords["Id"];
        echo "<p>Purchased On: ";
        echo $arrPurRecords["Day"] . "/";
        echo $arrPurRecords["Month"] . "/";
        echo $arrPurRecords["Year"] . "</p>";

     $dbProRecords= mysql_query("SELECT * FROM purchaseProducts WHERE purchases_Id='$intPurId'
", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());

    while ($arrProRecords = mysql_fetch_array($dbProRecords)) {
        $intProductId = $arrProRecords["products_Id"];
        echo "<p>" . $arrProRecords["Quantity"] . " ";

        $dbProductRecords = mysql_query("SELECT * FROM products WHERE Id = '$intProductId'",
$dbLocalhost)
    or die("Problem reading table: " . mysql_error());

        $arrProductRecord = mysql_fetch_array($dbProductRecords);
        echo $arrProductRecord["Name"] . " (" . $arrProductRecord["Description"] . ") at &#163;";
        echo $arrProRecords["Cost"] . " each.</p>";
        }
    }
}
?>
```

Inserting records

How to create new database records and insert them into a table?

```
INSERT INTO table (field1, field2,...) VALUES ('value1', 'value2',...)
```

•Alternatively, we have a simplified syntax:

```
INSERT INTO table VALUES ('value1', 'value2',...)
```

```
$dbProdRecords = mysql_query("INSERT INTO products
VALUES ( ' ', 'Beer Mug', '600 ml Beer Mug', '100', '5.99')",
$dbLocalhost)
```

Inserting records

```
<?php
// File: example15-15.php

require_once("database2.php");

$dbProdRecords = mysql_query("INSERT INTO products VALUES ('', 'Beer Mug', '600
ml Beer Mug', '100', '5.99')", $dbLocalhost)
   or die("Problem writing to table: " . mysql_error());

$dbProdRecords = mysql_query("SELECT * FROM products", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());

while ($arrProdRecords = mysql_fetch_array($dbProdRecords)) {
    echo "<p>" . $arrProdRecords["Id"] . " ";
    echo $arrProdRecords["Name"] . " ";
    echo $arrProdRecords["Description"] . " ";
    echo $arrProdRecords["Quantity"] . " ";
    echo $arrProdRecords["Cost"] . "</p>";
}
?>
```

Deleting records

How to delete database records from tables?

```
DELETE FROM table WHERE field=‘value’
```

e.g.

```
$dbCustRecords = mysql_query("DELETE FROM customers
WHERE Id='3'", $dbLocalhost)
```

Note: If you have a relational database, you should tidy-up the other tables
their connection with the record you’ve deleted.

Deleting records

How to delete database records from tables?

```
DELETE FROM table
```

This will delete all records from a table!

Note: back-up your database first!

Amending records

How to modify the contents of an existing database record?

```
UPDATE table SET field='value1', field='value2'...WHERE
   field='value'
```

• requires you to specify the table, the list of fields with their updated values, and a condition for selection (WHERE).

Amending records

```
<?php
// File: example15-18.php

require_once("database2.php");

$dbCustRecords = mysql_query("UPDATE products SET Description='250 ml Tall
Glass' WHERE Id='6'", $dbLocalhost)
   or die("Problem updating table: " . mysql_error());

$dbProdRecords = mysql_query("SELECT * FROM products", $dbLocalhost)
    or die("Problem reading table: " . mysql_error());

while ($arrProdRecords = mysql_fetch_array($dbProdRecords)) {
    echo "<p>" . $arrProdRecords["Id"] . " ";
    echo $arrProdRecords["Name"] . " ";
    echo $arrProdRecords["Description"] . " ";
    echo $arrProdRecords["Quantity"] . " ";
    echo $arrProdRecords["Cost"] . "</p>";
}
?>
```

Amending records

How to modify the contents of an existing database record?

UPDATE table SET field=‘value1’, field=‘value2’...WHERE field=‘value’

Another Example:

```
$dbCustRecords = mysql_query("UPDATE products SET Name='Beer and Lager Glass' WHERE Name='Beer Glass'", $dbLocalhost)
```

•A number of records will be updated in this example.

Counting the number of records

How to count the number of records after running a query?

```
$dbProdRecords = mysql_query("SELECT * FROM products",
    $dbLocalhost)
    or die("Problem reading table: " . mysql_error());


$intProductCount = mysql_num_rows($dbProdRecords);
```

- you can also use the same function to determine if a record exists.

Select a substring

How to count the number of records after running a query?

```
SELECT * FROM products WHERE substring(Name,1,4)=‘Wine’
```

•This will return all records from the products table where the first four characters in the name field equals ‘Wine’

PHP Exercise

Practice PHP

- Create a registration form using HTML in index.php page
- The form should have atleast 5 fields
- Create a new page result.php
- OnSubmit of form, retrieve all form element values on result page and display them in a table
- Create a table in the database for those 5 form fields
- Save the data obtained from the form in the database table
- Practise the following queries on the database table through php code:
  - Select
  - Insert
  - Update
  - Delete

Complete list of reference – PHP for Databases here.