Introduction to PHP (Part-II)

PHP in HTML

- To embed PHP within HTML, write a PHP tag by beginning with <?php and ending with ?>. Then, you can enter PHP code between these tags.
- If you want to do some processing before deploying the HTML, you embed PHP before the start of the HTML document.
- If you want to use PHP to display dynamic data within an HTML document, you embed PHP within the HTML.
- PHP has a syntax that is similar to the syntax of Java or JavaScript.
    - A statement controls the operations of a program, and ended with a semicolon.
    - To make your code easier to read, you should use indentation and extra spaces to align statements and parts of statement.

PHP data types

Type	Description
Integer	Whole numbers that range from -2,147,483,648 to 2,147,483,647.
double	Numbers with decimal places that is up to 16 significant digits.
boolean	A TRUE or FALSE value.
string	Text that consists of any characters.
array	A container that holds multiple values of one or more data types.
object	A container that contains data (properties) and functions (methods).

Declaring Variables and Constants

- To declare a variable, write the dollar sign ($) followed by the variable name. Then, to assign a value to the variable.
- Variable names
    - are case-sensitive.
    - can contain letters, numbers, and underscores.
    - can't contain special characters.
    - can't begin with digit or two underscores.
- PHP assigns a data type to the variable depending on the value that's assigned to the variable.
- A literal is a value that doesn't change. To code a numeric literal, code the number without quotation marks. To code a string literal, code the value within quotation marks. To code a Boolean literal, code TRUE or FALSE. These keywords aren't case-sensitive.
- To declare a constant, you can use the define function to specify the name and value of the constant. Remember, you don't code a dollar sign before its name even when you use it.

Using HTTP GET & HTTP POST

When to use the HTTP GET?

- A request for getting data from a server (especially from a database server)
- A request that can be executed multiple times without causing any problems.

When to use the HTTP POST?

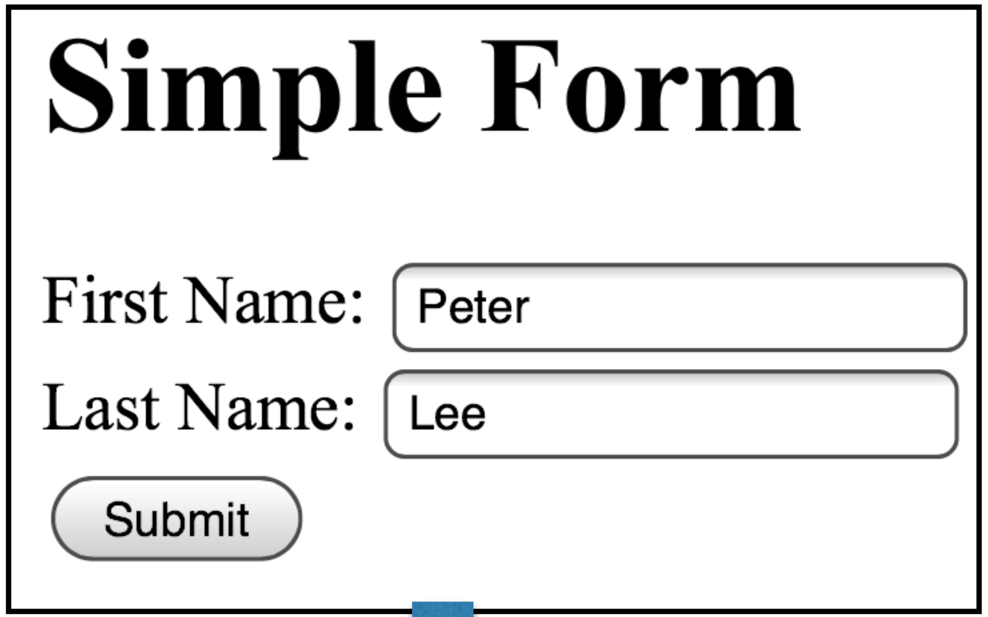
- A request for saving data to a server (especially from a database server)
- Don't want to include the parameters in the URL for securing reasons.
- Need to transfer more than 4 KB of data.

Getting data from request

- When using the HTTP GET method in a form, the parameters are appended to the URL when the form is submitted. These parameters are stored in the $_GET array.
- The $_GET array that contains the keys and values for the data is passed with the HTTP request.
  - `$first_name = $_GET['first_name'];`
  - `$last_name = $_GET['last_name'];`
- The <a> tag always uses the HTTP GET method when it passes parameters to a page.
  - `<a href="display_simple_form.php?first_name=Lily&last_name=Wu">Display Lily</a>`
- The $_GET variable is a **superglobal** variable, which means it's always available to the PHP code for a page.

Example of $_GET array

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="UTF-8">
      <title>Simple Form</title>
   </head>
   <body>
      <?php
      ?>
      <h1>Simple Form</h1>
      <form method="get" action="display_simple_form.php">
         <label>First Name: </label>
         <input type="text" name="first_name" /><br />
         <label>Last Name: </label>
         <input type="text" name="last_name" /><br />
         <input type="submit" name="Submit" value="Submit" />
      </form>
   </body>
</html>
```

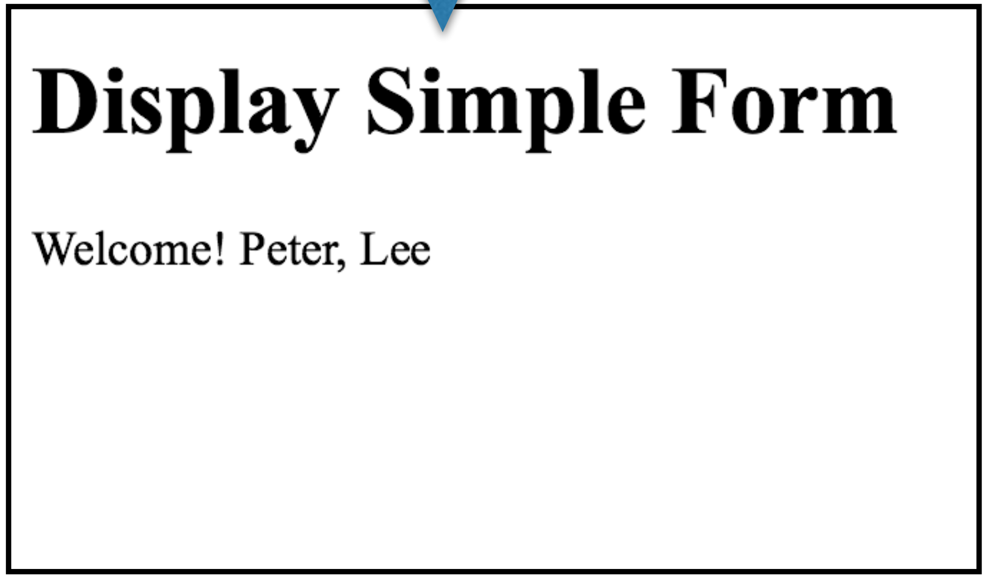


Simple Form

First Name: Peter

Last Name: Lee

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="UTF-8">
      <title>Display Simple Form</title>
   </head>
   <body>
      <h1>Display Simple Form</h1>
      <?php
         $first_name = $_GET['first_name'];
         $last_name = $_GET['last_name'];
         echo "Welcome! $first_name, $last_name";
      ?>

   </body>
</html>
```



Display Simple Form

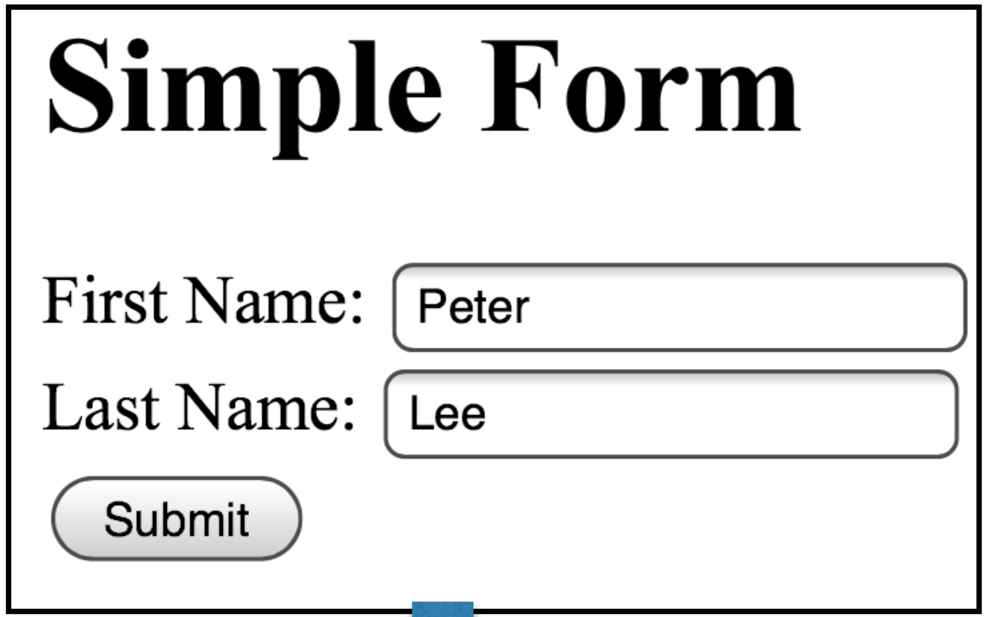
Welcome! Peter, Lee

Posting data to server

- When using the HTTP POST method in a form, the parameters are NOT appended to the URL when the form is submitted. The data is passed to the server by the way of the HTTP message data fields. These parameters are stored in the $_POST array.
- The $_POST array that contains the keys and values for the data is passed with the HTTP request.
  - `$first_name = $_POST['first_name'];`
  - `$last_name = $_POST['last_name'];`
- The $_POST variable is a **superglobal** variable, which means it's always available to the PHP code for a page.

Example of $_POST array

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="UTF-8">
      <title>Simple Form</title>
   </head>
   <body>
      <?php
      ?>
      <h1>Simple Form</h1>
      <form method="post" action="display_simple_form.php">
         <label>First Name: </label>
         <input type="text" name="first_name" /><br />
         <label>Last Name: </label>
         <input type="text" name="last_name" /><br />
         <input type="submit" name="Submit" value="Submit" />
      </form>
   </body>
</html>
```

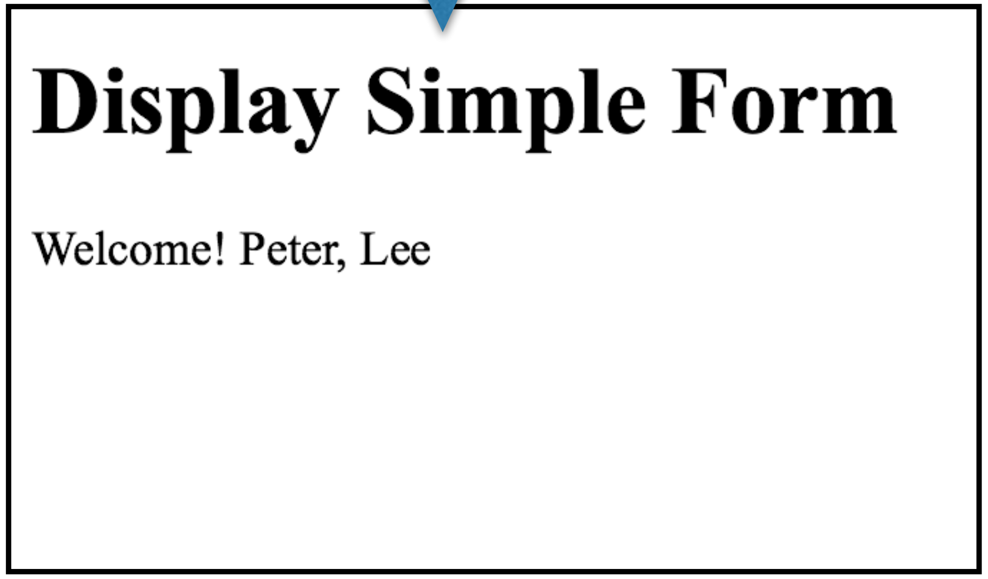


Simple Form

First Name: Peter

Last Name: Lee

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="UTF-8">
      <title>Display Simple Form</title>
   </head>
   <body>
      <h1>Display Simple Form</h1>
      <?php
         $first_name = $_POST['first_name'];
         $last_name = $_POST['last_name'];
         echo "Welcome! $first_name, $last_name";
      ?>

   </body>
</html>
```



Display Simple Form

Welcome! Peter, Lee

Manipulating the data

- You can use single or double quatrain marks to code a sting, but PHP processes single quotation marks more efficiently.
- A null value indicates that the value is unknown. To assign a null value to a string, you use the NULL keyword.
- If you code a variable name within double quotes, it will be converted to a string.
- To send data to the browser, you can use the **echo** statement. And it can include HTML tags.

Examples - String expressions

Statements	Description
$first_name = 'Carl'; $last_name = 'Buck';	Using single quotes to improve PHP efficiency.

$phone = ''; // an empty string
$phone2 = NULL; // a NULL
	Assign NULL values and empty string

$fname = "My name is $first_name";	Use double quotes to insert a variable into a sting
$last_name = "O'McVander"; $line = 'He said, "Hi,"';	Use single and double quotes for special purposes
$first_name = 'Carl'; $last_name = 'Buck'; $name = $first_name . ' ' . $last_name; echo $name; $price = 56.7;	Use the concatenation operator (.) to join strings and a number.

PHP library Functions

number_format($number, [, format])

- Returns a number that's formatted with a thousand-comma. If the second optional parameter is specified, the numbers will be rounded to the specified number of decimal places.

date(format)

- Returns the current date with the specified format string.

isset($val)

- Returns a true value if the variable has been set and is not a null value.

empty($val)

- Returns true if the variable hasn't been set, contains a null value, contains an empty string.

is_numeric($val)

- Returns a true value if the variable is a number or a string that can be converted to a number.

Example of build-in functions

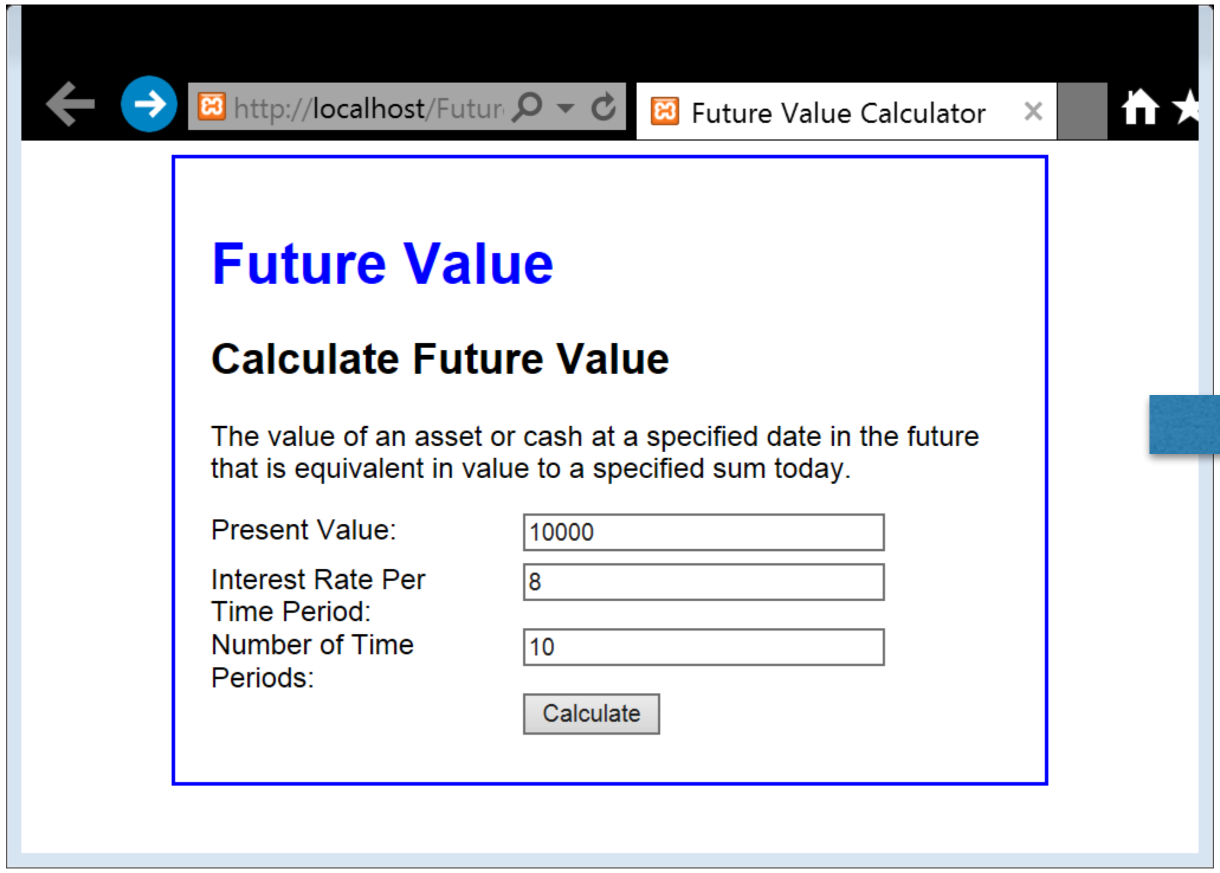
```
$af = number_format( 1234 );  // 1,234
$af = number_format( 1234, 2 );  // 1,234.00
$af = number_format( 1234.56, 1);  // 1,234.6

$d = date( 'Y-m-d' );     // 2014-01-16
$d = date( 'm/d/y' );     // 01/16/14
$d = date( 'm.d.y );     // 01.16.14

isset( $name )     // true if $name has been set and is  not null
empty( $name ).  // true if $name is empty
is_numeric($price). // true if $price is a number
```

PHP built-in functions that allows you to transfer control to another page

Name	Description
include($path)	Inserts and runs the specified file. However, if this function fails, it causes a warning that allows the script to continue. The parentheses are optional for this function. e,g, include('index.php);
include_once($path)	Same as include, but it makes sure that the file is included only once.
require($path)	Works the same as the include function. However, if this function fails, it causes a fatal error that stops the script. e,g, require('index.php);
require_once($path)	Same as require, but it makes sure that the file is only required once.
exit([$status])	Exits the current PHP script. If $status isn't supplied, the parentheses are optional. If $status is supplied, this function sends the $status string to the browser before it exits. e,g, exit('Unable to make a connection. ');
die([$status])	Works the same as the exit function.

- To pass control to another page, you can use the **include** and **require** functions.
- To exit from the PHP script that is running, you can use the exit or die function.

Example of Future Value Calculator

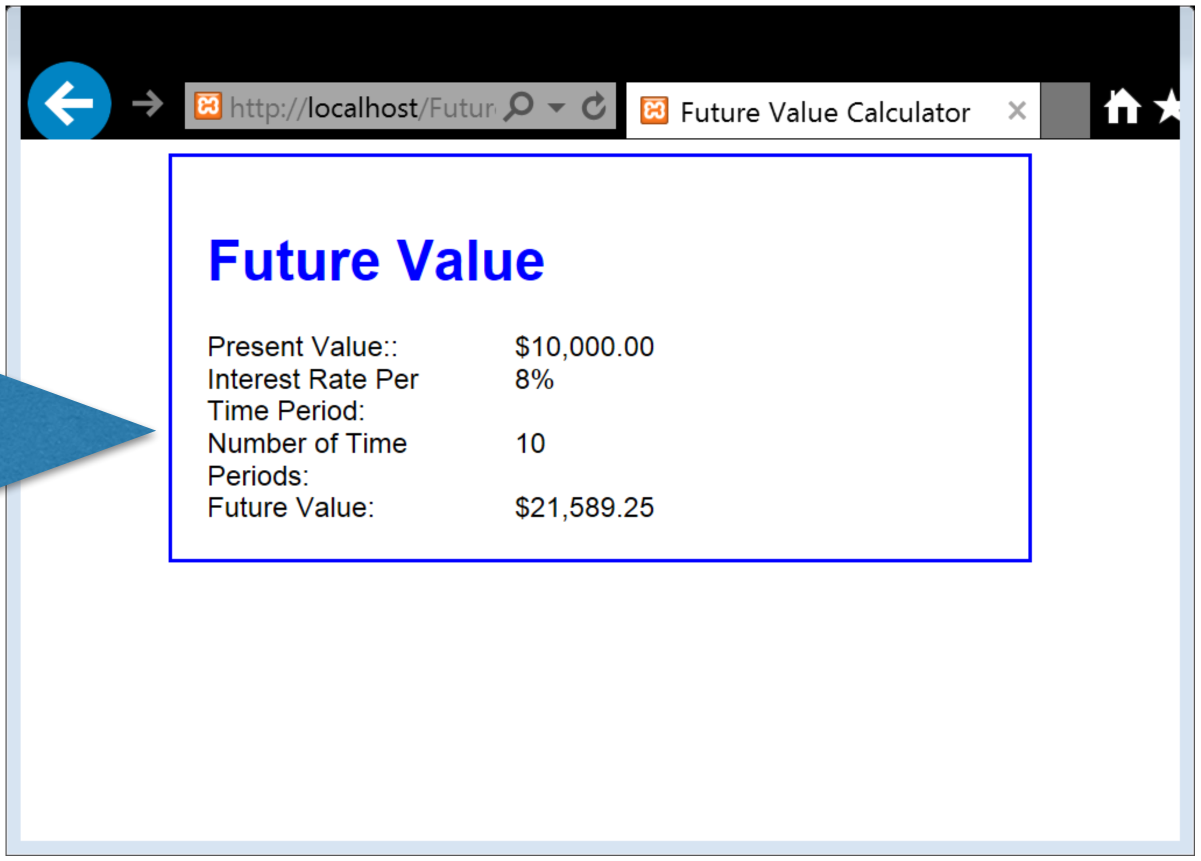


Future Value

Calculate Future Value

The value of an asset or cash at a specified date in the future that is equivalent in value to a specified sum today.

Present Value:	10000
Interest Rate Per Time Period:	8
Number of Time Periods:	10

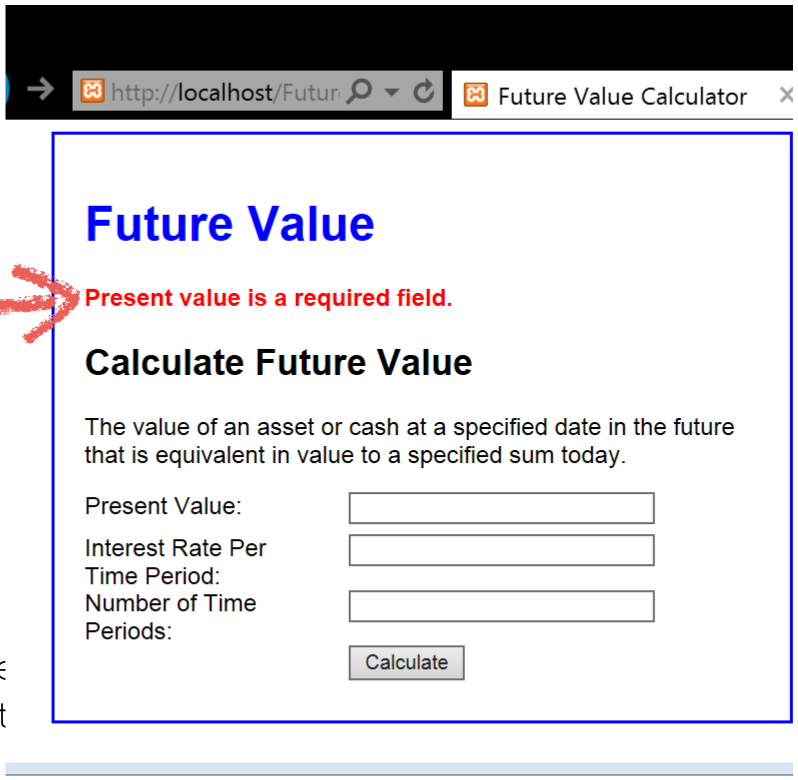


Future Value

Present Value::	$10,000.00
Interest Rate Per Time Period:	8%
Number of Time Periods:	10
Future Value:	$21,589.25

Example of Future Value Calculator

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>Future Value Calculator</title>
        <link rel="stylesheet" type="text/css" href="styles.css"/>
    </head>
    <body>
        <div id="content">
            <h1>Future Value</h1>
            <?php if (!empty($error_message)) { ?>
                <p class="error"><?php echo $error_message; ?></p>
            <?php } // end if ?>
            <form action="future_value.php" method="post">
                <h2>Calculate Future Value</h2>
                <p>The value of an asset or cash at a specified date in the
                    future that is equivalent in value to a specified sum t
                </p>
                <div id="data">
                    <label>Present Value:</label>
                    <input type="text" name="present_value"
                           value="<?php if (!empty($present_value)) {echo $present_value;} ?>"/><br />

                    <label>Interest Rate Per Time Period:</label>
                    <input type="text" name="interest_rate"
                           value="<?php if (!empty($interest_rate)) {echo $interest_rate;} ?>"/><br />

                    <label>Number of Time Periods:</label>
                    <input type="text" name="periods"
                           value="<?php if (!empty($periods)) {echo $periods;} ?>"/><br />
                </div>
                <div id="buttons">
                    <label>&nbsp;</label>
                    <input type="submit" value="Calculate"/><br />
                </div>
            </form>
        </div>
    </body>
</html>
```



Future Value

Present value is a required field.

Calculate Future Value

The value of an asset or cash at a specified date in the future that is equivalent in value to a specified sum today.

Present Value:	
Interest Rate Per Time Period:	
Number of Time Periods:	

future_value.php (part 1)

```php
<?php
    // get the data from the form
    $present_value = $_POST['present_value'];
    $interest_rate = $_POST['interest_rate'];
    $periods = $_POST['periods'];

    // validate present_value entry
    if ( empty($present_value) ) {
        $error_message = 'Present value is a required field.';
    } else if ( !is_numeric($present_value) )  {
        $error_message = 'Present value must be a valid number.';
    } else if ( $present_value <= 0 ) {
        $error_message = 'Present value must be greater than zero.';
    // validate interest rate entry
    } else if ( empty($interest_rate) ) {
        $error_message = 'Interest rate is a required field.';
    } else if ( !is_numeric($interest_rate) )  {
        $error_message = 'Interest rate must be a valid number.';
    } else if ( $interest_rate <= 0 ) {
        $error_message = 'Interest rate must be greater than zero.';
    // set error message to empty string if no invalid entries
    } else {
        $error_message = '';
    }
    // if an error message exists, go to the index page
    if ($error_message != '') {
        include('index.php');
        exit();
    }
    // calculate the future value
    $future_value = $present_value;
    for ($i = 1; $i <= $periods; $i++) {
        $future_value = ($future_value + ($future_value * $interest_rate *.01));
    }

    // apply currency and percent formatting
    $present_value_f = '$'.number_format($present_value, 2);
    $interest_rate_f = $interest_rate.'%';
    $future_value_f = '$'.number_format($future_value, 2);
?>
```

future_value.php (part 2)

```
<html>
<head>
    <title>Future Value Calculator</title>
    <link rel="stylesheet" type="text/css" href="styles.css"/>
</head>
<body>
    <div id="content">
        <h1>Future Value</h1>

        <label>Present Value::</label>
        <span><?php echo $present_value_f; ?></span><br />

        <label>Interest Rate Per Time Period:</label>
        <span><?php echo $interest_rate_f; ?></span><br />

        <label>Number of Time Periods:</label>
        <span><?php echo $periods; ?></span><br />

        <label>Future Value:</label>
        <span><?php echo $future_value_f; ?></span><br />
    </div>
</body>
</html>
```

Exercises

- **Question 1**:
  - Body Mass Index (BMI) is a measure of health based on height and weight. It can be calculated by taking your weight in kilograms and dividing it by the square of your height in meters. The interpretation of BMI for people 20 years or older is as follows:
    - BMI Interpretation
      - Below 18.5 Underweight
      - 18.5–24.9 Normal
      - 25.0–29.9 Overweight
      - Above 30.0 Obese
    - Write a web form page that has two input boxes tolet the user to enter a weight in pounds and height in inches and displays the BMI. Note that one pound is 0.45359237 kilograms and one inch is 0.0254 meters. It has a Calculate button.
    - Write a PHP script file that will take request parameters from the above web page, then process the data and display the results in HTML.

Exercises

- **Question 2**:
  - Write a web page that has one input box that allows the user to enter a four-digit integer and then submit it to the server. The server PHP script will encrypt it as follows:
    - Replace each digit by (the sum of that digit plus 7) modulus 10.
    - Then swap the first digit with the third, and swap the second digit with the fourth.
    - Then display the encrypted integer in HTML.
  - Write another web page that inputs an encrypted four-digit integer, and another PHP script to decrypt it to form the original number.