Лабораторна робота № 3

Тема: PHP як засіб написання сценаріїв

Мета: Засвоїти елементи програмування серверної частини застосувань мовою PHP.

Основні елементи PHP

PHP (Препроцесор Гіпертексту) є мовою сценаріїв загального призначення з відкритим вихідним кодом. PHP використовується для ведення web-розробок і може записуватись безпосередньо в HTML-коді.

Синтаксис мови бере початок з C, Java, Perl і є легким для вивчення. Переважним призначенням PHP є надання web-розробникам можливості швидкого створення динамічно генерованих web-сторінок, однак сфера застосування PHP лише цим не обмежується.

В офіційній документації мова PHP подається як HTML – вбудована мова програмування (скриптів) на стороні сервера. Це означає, що:

- оброблення PHP-коду відбувається на сервері ще до того, як web-сторінка буде передана браузеру;
- PHP-код може бути вбудований безпосередньо в HTML-код сторінки. Цим він відрізняється від Perl;
- PHP не є ні компілятором, ні інтерпретатором, а чимось середнім між ними. PHP обробляє сценарії, які подаються на вхід. Він транслює його в спеціальний байт-код (внутрішнє представлення). Після цього PHP виконує байт-код. У такому разі виконавчий файл не створюється. Байт-код значно компактніший, ніж звичайний код програми. Тому він швидше інтерпретується та виконується. Тому PHP більше інтерпретатор, ніж компілятор.

Використання інтерпретатора (а, значить, і PHP) має переваги:

- не потрібно дбати про звільнення виділеної пам'яті, закривати файл після закінчення роботи з ним – це здійснює браузер, оскільки програма виконується під його контролем;
- не потрібно визначати типи змінних та оголошувати змінну до її першого використання;
- відлагоджування програми та виявлення помилок істотно полегшуються – інтерпретатор повідомить про це;
- існує можливість створення програми, яка напише іншу програму, а потім введе в себе код щойно написаної програми і виконає його.

Виконання PHP-програм відбувається за схемою (рис. 4.1).

Рис. 4.1. Виконання програм PHP

Перед надсиланням сервером браузеру файла його переглядає препроцес-інтерпретатор. Файли, що піддаються обробленню препроцесором, повинні мати розширення .phtml або .php і містити код для препроцесора. Перед надсиланням

сторінки PHP-код виконується сервером Apache і браузеру видається результат у вигляді HTML-сторінки (вона відрізняється від тієї, що зберігається на сервері). Сторінки з розширенням .html/.htm web-сервер відправляє браузеру без будь-якого опрацювання.

Основна відмінність від CGI-скриптів, написаних мовами Perl або C, полягає в тому, що в CGI-програмах треба писати виведений HTML-код, а PHP – дає змогу вносити програму – скрипт у код HTML-сторінки (обмежену тегами <? і ?>).

Переваги PHP такі:

продуктивність – сервер може обслуговувати мільйони звернень на день; *інтеграція з базами даних* – PHP володіє вбудованим зв'язком з багатьма системами баз даних (MySQL, PostgreSQL, mSQL, Oracle, dbm, Hyperware, Informix, InterBase, і Sybase та інших на основі ODBC-драйверів);

простота для вивчення – синтаксис PHP оснований на мовах програмування C та Perl;

можливість перенесення – пакет PHP можна використовувати під керуванням різних операційних систем (Linux, FreeBSD, Unix, Windows);

наявність вбудованих бібліотек – можна генерувати Gif-зображення, під'єднатись до багатьох мережевих служб, відправляти повідомлення електронною поштою, працювати з cookie-наборами і генерувати PDF-документи;

вартість – пакет PHP є безкоштовний.

Приклади використання PHP

Розглянемо приклади використання PHP-коду в тексті HTML-сторінки та самостійного PHP коду:

- в тексті сторінки код роздруковує повідомлення:

```
<!DOCTYPE html>
<html>
<head>
<title>Hello world</title>
</head>
<body>
<?php
print("Hello, World");
?>
</body>
</html>
```

- самостійний код – підрахунок кількості відвідувачів сторінки:

```
<?
if(file_exists("count.dat"))
{
  $exist_file = fopen("count.dat", "r");
  $new_count = fgets($exist_file, 255);
  $new_count++;
  fclose($exist_file);
  //
  print("$new_count people have visited this page");
  $exist_count = fopen("count.dat", "w");
  fputs($exist_count, $new_count);
```

```
  fclose($exist_count);
}
else
{
$new_file = fopen("count.dat", "w");
fputs($new_file, "1");
fclose($new_file);
}
?>
```

Для ввімкнення коду в текст сторінки використовуємо оператор `require`:

```
<? require  ("requiredfile.php"); ?>
```

Цю саму функцію виконує оператор `include()`:

```
include("filename");
include("header.php");
require(. . . . .  );
```

Для внесення тексту в код сторінки використовуються команди `print()` або `echo()`:

```
print("text");
print("Hello, World!");
print("Escape \"chars\" are the SAME as in Java!\n");
```

Приклади відповідей серверної частини у json форматі:

Приклад повернення помилки

```
{status: false, error:{code: 100, message: "Not found student"}}
```

Приклад успішного додавання студента

```
{status: true, error:null, id: 1}
```

Приклад отримання інформації про студента

```
{status: true, error:null, user: {
id: 1,
name_first: "Test1",
name_last: "Test2",
status: true
}}
```

Завдання до лабораторної роботи № 3:

Розробити web-сторінку згідно макета (wireframe)
https://cacoo.com/diagrams/ZvVhYS3UpG5PdbBy/EDE3A

Обираємо для розробки шаблон Students Add/Edit
Використовуємо Gitlab (https://gitlab.com)

Розробити 2 складові частини проекту :

- Сайт з формою, що надсилає на сервер дані студента
- Серверна частина перевіряє чи наявні та коректні всі поля і повертає інформацію у json форматі
- Сайт реагує на відповіді. Якщо прийшла помилка відображає, якщо немає помилки додає у таблицю студента.