



ТЕМА 5

Буфер виводу, JSON, XML, cURL і файли

*Вихідний буфер · обмін даними JSON та XML · читання й запис файлів ·
HTTP-запити через cURL*

Бекенд-розробка, сервіси та мікросервіси

Комп'ютерна інженерія · 3 курс · Лекція 5 з 18

7 питань сьогоднішньої теми

01

Вихідний буфер (output buffering)

02

Формат обміну даними JSON

03

Формат обміну даними XML

04

Робота з файлами: читання і запис

05

Робота з файлами: практичні операції

06

HTTP-запити через cURL

07

cURL на практиці: JSON API

ob_start() та керування буфером

Функція	Призначення
ob_start()	Починає буферизацію виводу
ob_get_clean()	Повертає вміст і закриває буфер
ob_end_flush()	Надсилає вміст клієнту і закриває буфер
ob_end_clean()	Видаляє вміст без надсилання

```
ob_start();
echo "<h1>Привіт!</h1>";

$html = ob_get_clean(); // нічого не надіслано браузеру
echo strtoupper($html); // обробили перед фактичним виводом
```

Навіщо потрібна буферизація

1

Зміна заголовків після виводу

header() працює, лише поки жоден байт вмісту не надіслано

2

Кешування сторінок

Згенерований HTML зберігається у файл і віддається повторно без регенерації

3

Захоплення виводу сторонніх бібліотек

Отримання результату echo як рядка — для шаблонів чи PDF

"headers already sent" виникає, коли PHP уже почав надсилати вивід до виклику header(). ob_start() на початку скрипта — типовий спосіб уникнути цього.

Кодування: json_encode()

```
$user = [  
    "name" => "Оксана", "age" => 25,  
    "skills" => ["PHP", "MySQL"],  
];  
  
echo json_encode($user);  
// {"name":"Оксана","age":25,"skills":["PHP","MySQL"]}
```

Прапорець	Ефект
JSON_PRETTY_PRINT	Форматує з відступами для читабельності
JSON_UNESCAPED_UNICODE	Не екранує кирилицю як \uXXXX
JSON_UNESCAPED_SLASHES	Не екранує символ / як \/

Декодування: json_decode()

```
$json = '{"name":"Іван","age":30}';

$obj = json_decode($json);
echo $obj->name;          // Іван (об'єкт stdClass)

$arr = json_decode($json, true);
echo $arr['name'];        // Іван (асоціативний масив)
```

json_last_error() варто перевіряти завжди, коли дані надходять із зовнішнього джерела — некоректний JSON поверне null.

SimpleXML (читання) vs DOMDocument (створення)

SimpleXML

Найпростіший спосіб прочитати XML

```
$p = simplexml_load_string(
    $xmlString
);

echo $p->name;    // Ноутбук
echo $p->price;   // 25000
```

DOMDocument

Повний контроль: створення й редагування

```
$dom = new DOMDocument('1.0','UTF-8');
$p = $dom->createElement('product');
$p->appendChild(
    $dom->createElement('name','Ноутбук')
);
$dom->appendChild($p);
```

file_get_contents і file_put_contents

```
// Читання всього файлу в рядок
$content = file_get_contents('data.txt');
echo $content;

// Запис рядка у файл (перезаписує вміст)
file_put_contents('data.txt', "Новий вміст файлу");

// Додавання в кінець файлу, не перезаписуючи
file_put_contents('log.txt', "Новий запис\n", FILE_APPEND);
```

Найзручніший вибір, коли потрібно прочитати чи записати файл цілком, без покрокового контролю.

fopen(), fread(), fwrite(), fclose()

Режим	Значення
'r'	Лише читання, вказівник на початку
'w'	Лише запис; існуючий вміст видаляється
'a'	Лише запис; дописування в кінець
'r+'	Читання і запис, вказівник на початку

```
$h = fopen('data.txt', 'r');  
while (!feof($h)) {  
    echo fgets($h);  
}  
fclose($h);  
  
$out = fopen('output.txt', 'w');  
fwrite($out, "Рядок 1\n");  
fclose($out);
```

Керування файлами та CSV

Функція	Призначення
<code>file_exists(\$path)</code>	Перевіряє існування
<code>unlink(\$path)</code>	Видаляє файл
<code>mkdir(\$path)</code>	Створює теку
<code>copy() / rename()</code>	Копіює / перейменовує

```
$h = fopen('users.csv', 'r');
while (($row = fgetcsv($h)) !== false) {
    echo $row[0] . ' - ' . $row[1];
}
fclose($h);

$out = fopen('export.csv', 'w');
fputcsv($out, ['Ім'я', 'Бік']);
fclose($out);
```

Зберігання даних у JSON-файлі

```
$settings = ['theme' => 'dark', 'language' => 'uk'];  
file_put_contents(  
    'settings.json',  
    json_encode($settings, JSON_PRETTY_PRINT)  
);  
  
// Пізніше, при наступному запуску:  
$settings = json_decode(file_get_contents('settings.json'), true);  
echo $settings['theme'];    // dark
```

Простий спосіб зберігати невеликі структуровані дані (налаштування, кеш) без бази даних.

Базовий GET-запит

```
$ch = curl_init();

curl_setopt($ch, CURLOPT_URL, 'https://api.example.com/users/1');
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$response = curl_exec($ch);

if (curl_errno($ch)) {
    echo 'Помилка cURL: ' . curl_error($ch);
} else {
    $data = json_decode($response, true);
    echo $data['name'];
}

curl_close($ch);
```

Опції CURLOPT_*

Опція	Призначення
URL	Адреса запиту
RETURNTRANSFER	Повернути відповідь як рядок
POST	Виконати запит методом POST
POSTFIELDS	Дані в тілі запиту
HTTPHEADER	Масив користувацьких заголовків
TIMEOUT	Максимальний час очікування

POST-запит з JSON-тілом

```
$payload = json_encode(['title' => 'Нова задача']);

$ch = curl_init('https://api.example.com/tasks');
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS, $payload);
curl_setopt($ch, CURLOPT_HTTPHEADER, [
    'Content-Type: application/json',
    'Authorization: Bearer ' . $apiToken,
]);

$response = curl_exec($ch);
$statuscode = curl_getinfo($ch, CURLINFO_HTTP_CODE);
curl_close($ch);
```

Підсумок теми 5

- `ob_start()` дозволяє накопичувати вивід і змінювати заголовки пізніше
- `json_encode/decode` — стандарт обміну даними для сучасних API
- SimpleXML для читання XML, DOMDocument для створення й редагування
- `file_get_contents/file_put_contents` — швидкі операції з файлами цілком
- `fopen/fread/fwrite/fclose` — покроковий контроль для великих файлів
- cURL: `curl_init` → `curl_setopt` → `curl_exec` → `curl_close`, з перевіркою `errno` і HTTP-статусу