Tworzenie Aplikacji Internetowych

Laboratorium 10

Technologie po stronie serwera – PHP c.d.

Operacje na plikach:

- Pobranie wskaźnika do pliku: $plik = fopen(nazwa, tryb); tryby:
  - r – odczyt,
  - r+ - odczyt i zapis,
  - w – zapis,
  - w+ - odczyt i zapis,
  - a – do dopisu
- Zakończenie pracy z plikiem: fclose($plik)
- Przesunięcie pozycji w pliku: fseek($plik, oile);
- Dokładny odczyt danych z pliku: fread($plik, ileznaków);
- Odczyt jednej linii z pliku: fgets($plik);
- Zapis danych do pliku: fputs($plik, dane);
- Sprawdzenie czy osiągnięto koniec pliku: feof($plik);
- Sprawdzenie czy plik istnieje: file_exists($plik);
- Rozmiar pliku: filesize($plik);
- Wczytanie całego pliku do tablicy: file(nazwa);
- Wczytanie całego pliku jako tekst: file_get_contents(nazwa);
- Usunięcie pliku: unlink(nazwa);
- Sprawdzenie czy można zapisać do pliku (prawa dostępu): is_writeable(nazwa);
- Sprawdzenie czy można odczytać plik (prawa dostępu): is_readable(nazwa);
- Pobranie listy plików i katalogów z danego katalogu: scandir(katalog);
- Zablokowanie pliku do zapisu (inne procesy nie będą mogły nic zrobić): flock($plik, LOCK_EX);
- Zablokowanie pliku do odczytu (inne procesy będą mogły odczytywać): flock($plik, LOCK_SH);
- Odblokowanie pliku: flock($plik, LOCK_UN);

Przykład wczytania i wyświetlenia pliku:

```php
$plik = fopen("dane.txt", "r");
if ($plik) { // czy udało się otworzyć plik?
    while(!feof($plik)) {
        $linia = fgets($plik);
        echo("Linia: ".$linia."<br>");
    }
    fclose($plik);
}
```

Przykład zapisu danych z tablicy do pliku:

```php
$tab = [ "raz", "dwa", "trzy" ];
$plik = fopen("dane.txt", "w");
foreach ($tab as $val) {
    fputs($plik, $val."\r\n"); // znak końca linii to także: PHP_EOL
}
fclose($plik);
```

Przykład dopisania nowej linii do pliku:

```php
$plik = fopen("dane.txt", "a");
fputs($plik, "tekst\r\n");
fclose($plik);
```

Przykład zablokowania i odblokowania pliku:

```php
$plik = fopen("dane.txt", "w");
if (flock($plik, LOCK_EX)) {
    fputs($plik, "tekst\r\n");
    flock($plik, LOCK_UN);
}
fclose($plik);
```

Przekazywanie danych przez formularze:

Dane z pól umieszczonych wewnątrz znacznika <form> (np. pól input, select czy textarea) można przesłać do serwera wywołując zdarzenie **submit** (domyślnie zdarzenie to jest wywoływane np. po wciśnięciu klawisza enter wypełniając pola input). Zdarzenie to można również wywołać przez metodę submit() przy kliknięciu w przycisk typu button: (<button onclick="submit();">Wyślij dane</button>. Formularze mogą być przesyłane do innych stron. Aby to zrobić należy dodać atrybut action="adresstrony".

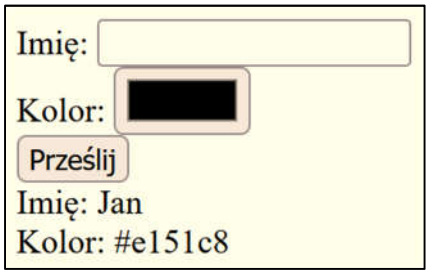
Domyślnie dane z formularza przesyłane są metodą GET (i są dostępne na stronie do której prowadził formularz w tablicy $_GET). Dodając atrybut method="post" do znacznika form sposób przesłania danych będzie zmieniony na POST, co ma znaczenie m.in. przy dużej liczbie przesyłanych danych (dane dostępne będą wtedy w tablicy $_POST).

Każde pole formularza musi posiadać atrybut **name** o przypisanej wartości (np. imie dla pola w którym użytkownik podaje imię). Następnie podając tą wartość jako klucz tablicy $_POST będzie można odwołać się do przesłanych danych (np. $_POST["imie"]).

Przykład formularza przesyłającego dane do strony na której się znajduje:

```php
<!DOCTYPE html>
<html>
    <head>
    </head>
    <body>
        <form method="post">
            Imię: <input name="imie"><br>
            Kolor: <input name="kolor" type="color"><br>
            <button onclick="submit();">Prześlij</button>
        </form>
        <?php
            if ($_SERVER["REQUEST_METHOD"] == "POST") { // czy dane są przesłane
                echo("Imię: ".$_POST["imie"]."<br>");
                echo("Kolor: ".$_POST["kolor"]);
            }
        ?>
    </body>
</html>
```

Wynik działania powyższego skryptu:



Warto zauważyć, że w poprzednim przykładzie po przesłaniu formularza strona otwiera się od nowa i nie ma wpisanych **danych** w formularzu. Aby je wyświetlić należałoby sprawdzić czy dane zostały przesłane, sprawdzić ich poprawność i umieścić wartości w atrybucie **value** znaczników input:

```
<?php
if (isset($_POST["imie"])) { // czy przesłano imie
    $imie = strip_tags($_POST["imie"]); // usunięcie znaczników z imienia
} else {
    $imie = ""; // gdy nie przesłano wartość będzie pusta
}
if (isset($_POST["kolor"])) { // czy przesłano kolor
    $kolor = strip_tags($_POST["kolor"]);
} else {
    $kolor = "#0000ff"; // gdy nie przesłano wartość będzie niebieska
}
?>

<form method="post">
    Imię: <input name="imie" value="<?php echo($imie); ?>"><br>
    Kolor: <input name="kolor" value="<?php echo($kolor); ?>" type="color"><br>
    <button onclick="submit();">Prześlij</button>
</form>

<?php
    if ($_SERVER["REQUEST_METHOD"] == "POST") { // czy dane są przesłane
        echo("Imię: ".$_POST["imie"]."<br>");
        echo("Kolor: ".$_POST["kolor"]);
    }
?>
```

W powyższym przykładzie dane z formularza są przetwarzane przez metodę strip_tags, która usuwa znaczniki HTML z przesyłanych danych. Jest to podstawowe zabezpieczenie danych, poprawność danych powinna być jeszcze m.in. sprawdzona przez wyrażenia regularne itp. (np. czy dane są liczbą dla danych liczbowych).

Sesje

PHP zapewnia prosty mechanizm sesji, pozwalający zapamiętywać dane w ramach sesji użytkownika (takie dane są dostępne domyślnie dla wszystkich stron z serwera). Aby uruchomić mechanizm sesji wystarczy wywołać metodę session_start(); na samym początku strony (przed przesłaniem jakichkolwiek danych) lub skonfigurować serwer w taki sposób aby sesje były zawsze włączone. Dane użytkownika zapamiętywane są w oparciu o identyfikator zapisywany w ciasteczkach przeglądarki, a zatem będą utracone po wyczyszczeniu ciasteczek lub upływie czasu działania mechanizmu sesji. Aby korzystać z danych powiązanych z sesją użytkownika wystarczy odwoływać się do tablicy $_SESSION podając wybrane przez siebie klucze. Przykład:

```
<?php
    session_start();
    if (isset($_SESSION["licznik"])) {
        $_SESSION["licznik"]++; // jeżeli $_SESSION["licznik"] istnieje to inkrementujemy
    } else {
        $_SESSION["licznik"] = 0; // jeżeli $_SESSION["licznik"] nie istniała zaczynamy od 0
    }
?>
<!DOCTYPE html><html>
    <head><meta charset="utf-8"></head>
    <body>
        <?php echo("Liczba odsłon: ".$_SESSION["licznik"]); ?>
    </body>
</html>
```

Zadanie 1 **W przypadku pracy zdalnej należy samodzielnie pobrać i zainstalować środowisko XAMPP**

Celem zadania jest stworzenie prostej ankiety działającej na plikach.

- Uruchomić środowisko XAMPP i serwer Apache.
- W katalogu C:\xampp\htdocs utworzyć plik **ankieta.php** wewnątrz którego umieścić standardową strukturę dokumentu HTML.
- Utworzyć formularz działający w oparciu o metodę POST, który pozwoli użytkownikowi wpisać:
  - imię,
  - wiek,
  - kwotę oszczędności,
  - wybrać jeden z czterech preferowanych sposobów inwestycji: nieruchomości, akcje, złoto, obligacje (wybór jednej z czterech opcji zrealizować w dowolny sposób)
- Napisać skrypt PHP, który będzie **dopisywał** do pliku **wyniki.txt** dane podane przez użytkownika w następującym formacie (każde dopisanie zakończone nową linią, oddzielanie danych znakiem |):
  imie|wiek|kwota|wybór|data|IP
  - datę pobrać funkcją: date("Y-m-d H:i:s")
  - IP pobrać z wartości: $_SERVER["REMOTE_ADDR"]
  - resztę wartości pobrać z formularza sprawdzając poprawność danych.
- Uzupełnić kilkukrotnie ankietę i sprawdzić działanie skryptu (plik wynikowy podejrzeć w notatniku).
- Dodać mechanizm, który po poprawnym uzupełnieniu ankiety zapamięta za pomocą sesji, że użytkownik wypełnij już ankietę. Jeżeli ankieta była już uzupełniona zablokować w dowolny sposób możliwość ponownego przesyłania danych w ankiecie.

Zadanie 2

Celem zadania jest stworzenie skryptu wyświetlającego wyniki z ankiety.

- W katalogu C:\xampp\htdocs utworzyć plik **wyniki.php** wewnątrz którego umieścić standardową strukturę dokumentu HTML.
- Wczytać i wyświetlić wszystkie dane z pliku **wyniki.txt** z poprzedniego zadania.
- Napisać skrypt, który przeanalizuje wyniki z ankiety. Skrypt powinien czytać linijka po linijce wszystkie dane z pliku **wyniki.txt**. Każdą wczytaną linię należy rozbić na tablicę danych, wykorzystując metodę explode. Przykład:
  ```
  $linia = fgets($plik);
  $dane = explode("|", $linia);
  $imie = $dane[0];
  $wiek = $dane[1];
  $kwota = $dane[2];
  $wybor = $dane[3];
  …
  ```
- Wyświetlić średnią kwotę oszczędności użytkowników.
- Wyświetlić średni wiek użytkowników.
- Wyświetlić liczbę użytkowników o wieku poniżej 30 lat, od 31 do 50 lat i powyżej 50 lat.
- Zastanowić się co się stanie gdy użytkownik przy wpisywaniu imienia użyje znaku |, który został wykorzystany do rozdzielania danych.
- **(opcjonalnie)** Wyświetlić ile osób wybrało preferowany sposób typu nieruchomości, ile osób wybrało akcje, ile złoto i ile obligacje.
- **(opcjonalnie)** Wyświetlić dane odnośnie wieku użytkowników za pomocą biblioteki Chart.js. Aby to zrobić należy część skryptu JavaScript związanego z wykresem wygenerować za pomocą PHP (część związaną ze strukturą danych).

Informacja: obydwa zadania pokazują tylko prosty przykład realizacji ankiety. W rozbudowanych przypadkach wyniki należałoby traktować jako logi ankiety, a dane należałoby zapisywać w innym formacie, zawierającym tylko dane liczbowe do wyświetlania (cache wyników). Warto także rozważyć wykorzystanie bazy danych.