Struktura języka PHP

Struktura skryptu

Skrypt PHP to nic innego, jak tylko normalna strona WWW, z tą jednak różnicą, że zawierająca kod PHP. Kod ten umieszczony jest w specjalnych sekcjach ograniczonych znacznikami <?php i ?> — pierwszy z nich otwiera sekcję kodu PHP, drugi zamyka sekcję. Istnieje też możliwość użycia znaczników krótkich — wtedy znacznik otwierający sekcję kodu nie zawiera ciągu php. To, czy można używać znaczników krótkich, czy też nawet znaczników rodem z ASP lub JSP, zależy od konfiguracji PHP. Zaleca się jednak używanie znaczników normalnych.

przykładowy skrypt PHP.

Table 1 Przykład 1

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html lang="pl">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>Powitanie!!!</title>
</head>
<body>
  <h1>Powitanie</h1>
  <p>
  <?php
    echo "Witaj w świecie PHP!";
  ?>
  </p>
</body>
</html>
```

Powyżej kod strony WWW zawierający tylko jedną sekcję PHP. Większa część kodu to jednak standardowe znaczniki HTML tworzące strukturę najprostszej (poprawnej) strony zgodnej ze standardem HTML. Jak już zostało to wspomniane wcześniej, nie będziemy omawiać typowych konstrukcji HTML-a. Przypomnijmy tylko, że składa się on z deklaracji typu dokumentu (<!DOCTYPE>), nagłówka (<head>) oraz treści (<body>). W sekcji <head> określany jest sposób kodowania znaków oraz tytuł strony (znacznik <title>).

Ilość sekcji PHP nie jest ograniczona, mogą one być umieszczane w dowolnym miejscu strony. Sekcja kodu PHP z przykładu 1 składa się tylko z jednej instrukcji echo. Instrukcja ta ma za zadanie wyświetlenie ciągu znaków umieszczonego w cudzysłowie (lub w apostrofach), a następującego po tej instrukcji. W języku PHP każdą instrukcję kończymy średnikiem. Zamiast instrukcji echo możemy użyć też print — to kwestia przyzwyczajenia i nawyków programisty, ponieważ w podstawowym zastosowaniu instrukcje te niczym się od siebie nie różnią. Zauważmy, że napis umieszczony po instrukcji echo wyświetlony zostanie dokładnie w tym miejscu strony, w którym umieszczony został w kodzie — pokazuje to rysunek 1.

Rysunek 1. Strona WWW wygenerowana na podstawie kodu z przykładu 1

W jaki sposób uzyskać efekt jak na rysunku 4.1? Otóż kod z przykładu 1 należy zapisać w pliku z rozszerzeniem *php* (pliki z tym rozszerzeniem serwer WWW przekazywał będzie interpreterowi PHP i umieścić w folderze, którego zawartość udostępniona jest przez serwer WWW.
Kod z listingu 4.1 wyświetla zwykły, niesformatowany tekst. W PHP istnieje możliwość wyświetlania również tekstu sformatowanego, czyli zawierającego znaczniki HTML. Przykład z listingu 4.3 jest modyfikacją przykładu z listingu 4.1 polegającą na wprowadzeniu do wyświetlanego tekstu definicji koloru.

Przykład 2 Zastosowanie formatowania tekstu w wyświetlanym napisie.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html lang="pl">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>Powitanie!!!</title>
</head>
<body>
  <h1>Powitanie</h1>
  <p>
  <?php
   // Teraz formatowanie
   echo '<span style="color:red">Witaj w świecie PHP!</span>';
  ?>
  </p>
</body>
</html>
```

Jako że definicja koloru w atrybucie style znacznika span musi być ujęta w cudzysłów, tekst do wyświetlenia przez instrukcję echo umieszczony został w apostrofach — efekt końcowy jest taki sam, jak przy zastosowaniu cudzysłowu.

Stronę WWW wygenerowaną w oparciu o kod z przykładu 2 przedstawia rysunek 2.

Rysunek 2. Wyświetlanie sformatowanego tekstu

W kodzie z listingu 2 powyżej instrukcji echo umieszczony został wiersz rozpoczynający się podwójnym znakiem ukośnika (ang. *slash*) — jest to komentarz. Komentarze stanowią nieodłączną część każdego kodu źródłowego, zawierają ważne informacje, przydatne np. podczas poprawiania i rozwoju oprogramowania, i mają na celu uratowanie twórcy programu przed zagubieniem w stworzonym jakiś czas temu i prawie już zapomnianym kodzie. W języku PHP istnieje kilka możliwości umieszczania komentarzy. Ten pokazany w przykładzie pochodzi z języka C++ i rozpoczyna się podwójnym znakiem ukośnika — wszystkie elementy po tych znakach są ignorowane przez interpreter — a jego zasięg ograniczony jest do końca wiersza.

Oprócz tego można również stosować komentarze blokowe oraz jednowierszowe uniksowe. Komentarz blokowy zaczyna się od sekwencji znaków /*, a kończy sekwencją */. Wszystko to, co znajduje się pomiędzy tymi znakami, zostanie zignorowane przez analizator składniowy PHP. Przykład poprawnego zastosowania wyglądać może następująco:

Przykład 3 Zastosowanie komentarzy.

```
<?php
/*W tym miejscu wyświetlamy napis powitalny*/
echo ("<h2> Witaj w świecie PHP!</h2>");
/*Koniec kodu PHP*/
?>
```

Po przetworzeniu przez PHP takiego kodu do przeglądarki zostanie wysłana jedynie linia zawierająca ciąg znaków:

<h2> Witaj w świecie PHP!</h2>

Komentarzy tego typu nie wolno zagnieżdżać, czyli we wnętrzu jednego komentarza blokowego nie wolno umieszczać kolejnego.

Komentarz jednowierszowy typu uniksowego ma takie samo działanie, jak komentarz jednowierszowy omówiony wyżej, jedynie jego wygląd jest inny. Zaczyna się on od znaku # i obowiązuje do końca danej linii skryptu. Przykład wykorzystania tego typu komentarza wygląda następująco:

Przykład 4 Zastosowanie komentarzy typu UNIX

```
<?php
# W tym miejscu wyświetlamy napis powitalny
echo ("<h2> Witaj w świecie PHP!</h2>");
# Koniec kodu PHP
?>
```

Komentarz jednowierszowy uniksowy może być również zagnieżdżony wewnątrz blokowego.

Każdy plik z rozszerzeniem *php* w momencie odwołania się do niego przez przeglądarkę WWW przekazywany jest przez serwer WWW interpreterowi PHP, dlatego też nie jest wskazane, aby rozszerzenie *php*dodawać do plików nie zawierających kodu PHP, ponieważ może to spowodować spowolnienie w serwowaniu strony — interpreter będzie niepotrzebnie angażowany.

Zwykle sekcje kodu PHP są bardziej rozbudowane — składają się z kilku, kilkudziesięciu lub nawet kilkuset linii kodu. Jeżeli chcemy użyć krótkiej sekcji (a dokładnie sekcji jednowierszowej — jak w opisywanych przykładach), możemy posłużyć się tzw. sekcją wyrażenia. Sekcja wyrażenia, znana np. z JSP, służy do wstawiania w kod (X)HTML ciągów znaków wygenerowanych za pomocą np. funkcji lub wyrażenia. Listing 4.4 zawiera przykład zastosowania sekcji wyrażenia.

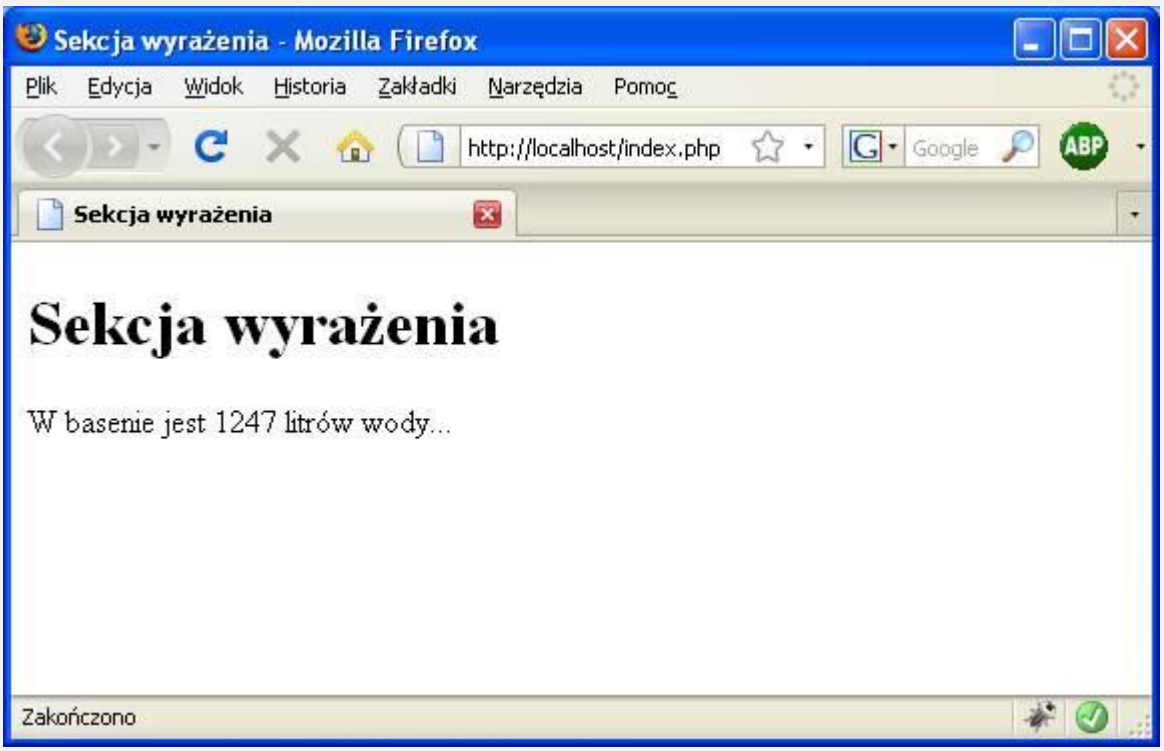
Listing 4.4. Przykład zastosowania sekcji wyrażenia

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
<html lang="pl">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>Powitanie!!!</title>
</head>
<body>
  <h1>Powitanie</h1>
  <p>
    W basenie jest <?= (125 * 10) - 3; ?> litrów wody...
  </p>
</body>
</html>
```

Aby wyświetlić wynik wyrażenia, nie trzeba używać instrukcji echo ani print. Należy jednak pamiętać o składni.

Efektem, który uzyskamy, będzie wstawienie wartości obliczonej na podstawie wyrażenia arytmetycznego (125 * 10) – 3, czyli 1247, w miejsce określone w kodzie HTML. Zastosowanie sekcji wyrażeń może w niektórych przypadkach znacznie uprościć kod strony. Pamiętać należy, że sekcje wyrażeń można stosować, gdy konfiguracja PHP dopuszcza użycie krótkich znaczników ograniczających kod PHP (opcja ta jest zwykle domyślnie wyłączona.

Rysunek 3 przedstawia efekt zastosowania sekcji wyrażenia.



Rysunek 3. Wynik zastosowania sekcji wyrażenia

Kod generujący pojedynczą stronę WWW nie musi być w całości umieszczany w jednym pliku (skrypcie) — czasem wręcz wskazane jest, aby zastosować rozbicie skryptu na kilka plików, tym bardziej że często różne strony wchodzące w skład jednej aplikacji wykonują te same operacje. Gdyby nie stosować w takim przypadku modularyzacji kodu aplikacji (podziału na moduły — tutaj pliki z tzw. wspólnym kodem), trzeba by było w każdym skrypcie generującym pojedynczą stronę przepisać ten sam kod. Pomijając fakt, że pliki niepotrzebnie zwiększyłyby swoją objętość, nie byłoby dobrze, gdyby się nagle okazało, że kod, który tak misternie i z takim mozołem wstawiany był do 25 skryptów, zawiera błąd… W przypadku zastosowania modularyzacji kod ten byłby umieszczony w jednym pliku, który byłby włączany do pozostałych 25 plików i poprawienie ewentualnego błędu w tym skrypcie spowodowałoby poprawne działanie owych 25 przykładowych skryptów.

Istnieje kilka sposobów dołączania plików z kodem PHP (metody będą omawiane podczas kolejnych lekcji).

Język PHP jest bardzo elastyczny, również pod względem składni — dopuszcza on tzw. składnię alternatywną w przypadku niektórych instrukcji. W tym kursie postawiono na stosowanie składni

najbardziej popularnej, zbliżonej do języków takich jak C, C++ czy Java. Więcej informacji na temat składni alternatywnej można znaleźć w dokumentacji PHP na stronie *http://www.php.net*.

Znaczniki PHP

Wiemy już, że treść skryptu PHP musi być oddzielona od kodu HTML specjalnymi znacznikami. Dokładniej rzecz ujmując, każdy fragment kodu PHP (nawet jeśli nie ma wokół niego kodu HTML) musi być ujęty w znaczniki, tak aby aparat wykonawczy PHP wiedział, że dana sekcja to instrukcje PHP, a nie inne dane. Podstawowe znaczniki już poznaliśmy, tak naprawdę jest ich jednak nieco więcej. W sumie możemy wyróżnić cztery różne typy:
znaczniki kanoniczne,
znaczniki typu SGML,
znaczniki typu ASP,
znaczniki skryptów HTML.

ZNACZNIKI KANONICZNE

Znaczniki kanoniczne to standardowe znaczniki PHP, które poznaliśmy już wcześniej. Znacznik otwierający to <?php, natomiast zamykający to ?>. Schematyczna konstrukcja wykorzystująca ten typ ma zatem następującą postać:

```
<?php
  //tutaj kod skryptu
?>
```

Znaczniki kanoniczne są rozpoznawane zawsze, niezależnie od tego, jakie opcje są włączone w pliku konfiguracyjnym *php.ini*. Jest to również zalecany sposób umieszczania skryptów w kodzie HTML i właśnie ten sposób będzie wykorzystywany w przykładach w dalszej części.

ZNACZNIKI TYPU SGML

Znaczniki typu SGML to znaczniki w postaci skróconej. Znacznik otwierający to <?, zamykający — ?>. Schematyczna konstrukcja wykorzystująca ten typ będzie miała postać:

```
<?
  //tutaj kod skryptu
?>
```

Jest to najkrótsza forma znaczników bloku PHP, jakie można zastosować. Aby jednak móc korzystać z tego sposobu, należy włączyć rozpoznawanie tych znaczników, co można zrobić, wykorzystując jeden z przedstawionych niżej sposobów:
W przypadku PHP w wersji 3 można wywołać funkcję short_tags().
Można włączyć opcję enable-short-tags przed kompilacją pakietu (./configure --enable-short-tags).
W pliku konfiguracyjnym *php.ini* można umieścić linię short_open_tag = On.
Wykorzystywanie tego typu znaczników, choć możliwe, nie jest jednak zalecane.

ZNACZNIKI TYPU ASP

Znaczniki typu ASP są znane użytkownikom technologii ASP. Znacznik otwierający to <%, zamykający — %>. Schematyczna konstrukcja wykorzystująca ten typ będzie więc miała postać:

```
<%
  //tutaj kod skryptu
%>
```

Aby móc korzystać z tego typu wyróżnienia bloków PHP, należy w pliku konfiguracyjnym włączyć opcję asp_tags = On.

ZNACZNIKI SKRYPTÓW HTML

Ta postać jest dobrze znana użytkownikom (X)HTML. Jest to typowy znacznik <script> z parametrem language ustawionym na wartość php oraz (o ile kod ma być zgodny ze standardem HTML 4) parametrem type ustawionym na text/php. Znacznik otwierający będzie miał zatem postać <script language="php" type="text/php">, natomiast zamykający — </script>. W związku z tym schematyczna konstrukcja wykorzystująca ten typ znaczników to:

```
<script language="php" type="text/php">
  //tutaj kod skryptu
</script>
```

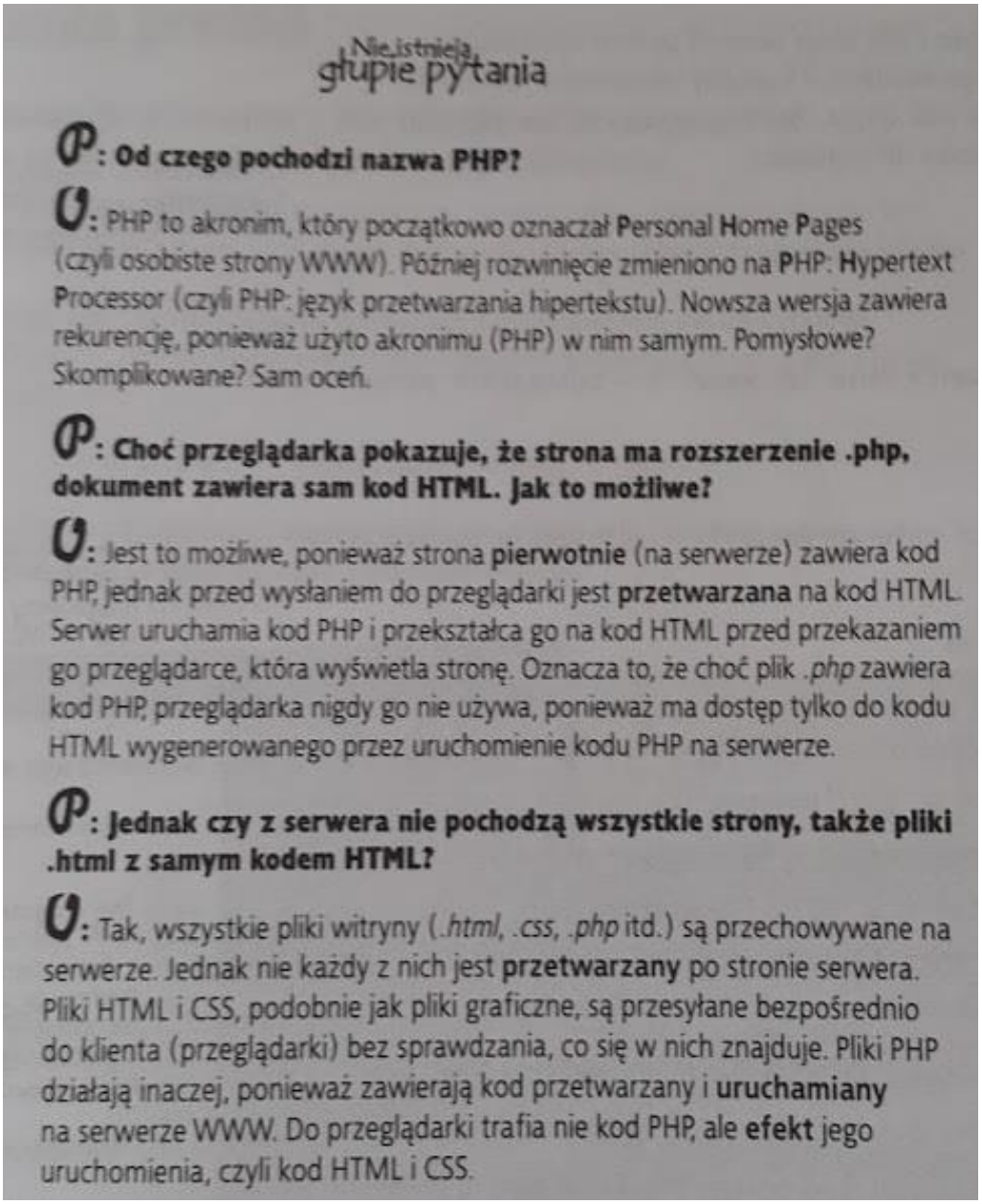
Postać ta, podobnie jak kanoniczna, jest rozpoznawana standardowo i nie wymaga włączania żadnych dodatkowych opcji konfiguracyjnych. Należy pamiętać, że atrybut language jest niepoprawny dla HTML w wersji 4.1 oraz XHTML.

Zadania

1. Utwórz skrypt PHP drukujący na ekranie komunikat *Witaj*.
2. Utwórz skrypt PHP drukujący na ekranie Twoje inicjały. Program Leszka Iwańskiego powinien wydrukować na ekranie:

```
3. LL              II
4. LL              II
5. LL              II
6. LL              II
7. LL              II
8. LLLLLL          II
```

3. Utwórz skrypt PHP drukujący na ekranie Pytania i odpowiedzi zawarte w obrazie zamieszczonym poniżej.



Nie istnieją głupie pytania

P: Od czego pochodzi nazwa PHP?

O: PHP to akronim, który początkowo oznaczał Personal Home Pages (czyli osobiste strony WWW). Później rozwinięcie zmieniono na PHP: Hypertext Processor (czyli PHP: język przetwarzania hipertekstu). Nowsza wersja zawiera rekurencję, ponieważ użyto akronimu (PHP) w nim samym. Pomysłowe? Skomplikowane? Sam oceń.

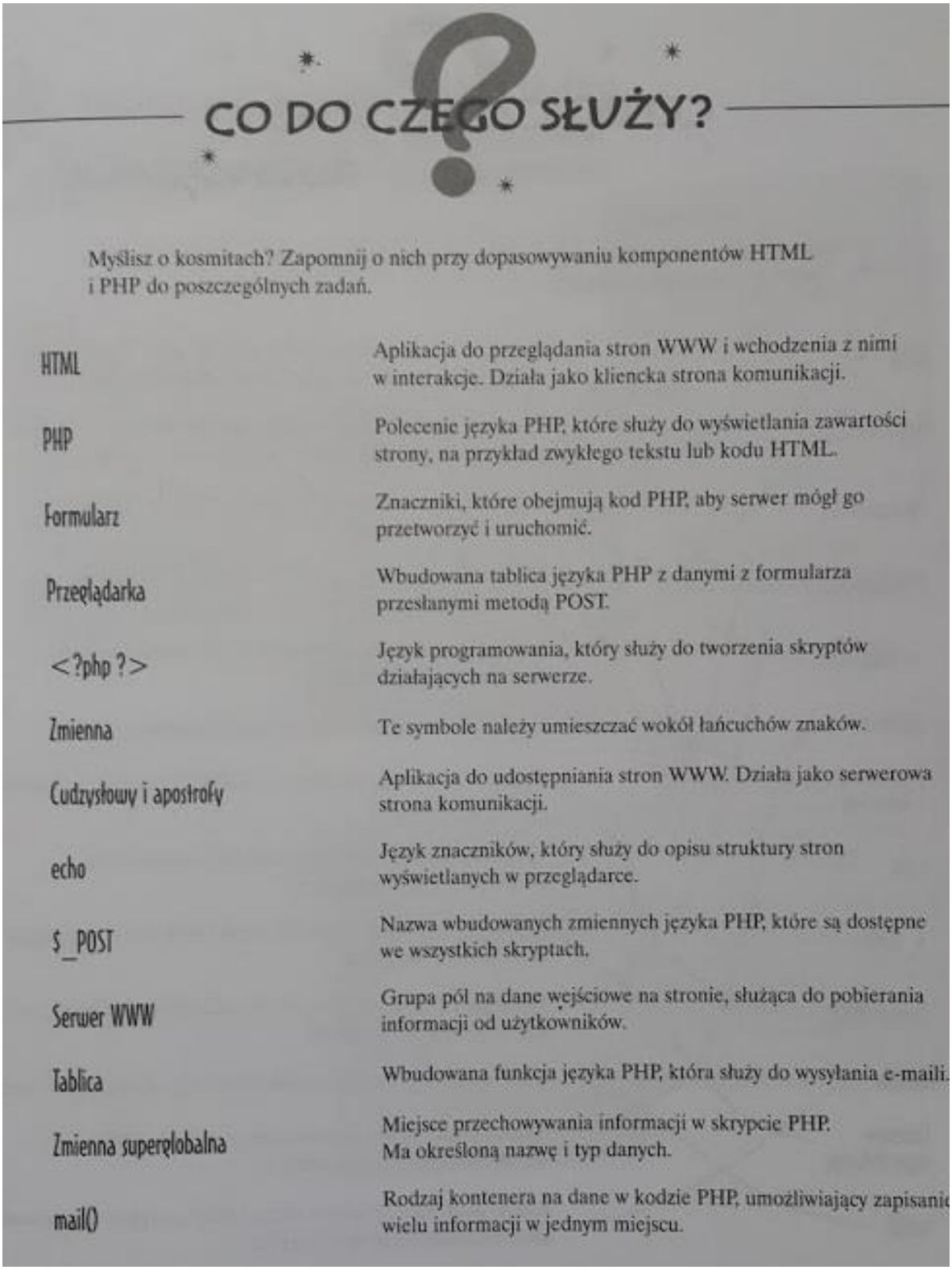
P: Choć przeglądarka pokazuje, że strona ma rozszerzenie .php, dokument zawiera sam kod HTML. Jak to możliwe?

O: Jest to możliwe, ponieważ strona **pierwotnie** (na serwerze) zawiera kod PHP, jednak przed wysłaniem do przeglądarki jest **przetwarzana** na kod HTML. Serwer uruchamia kod PHP i przekształca go na kod HTML przed przekazaniem go przeglądarce, która wyświetla stronę. Oznacza to, że choć plik *.php* zawiera kod PHP, przeglądarka nigdy go nie używa, ponieważ ma dostęp tylko do kodu HTML wygenerowanego przez uruchomienie kodu PHP na serwerze.

P: Jednak czy z serwera nie pochodzą wszystkie strony, także pliki .html z samym kodem HTML?

O: Tak, wszystkie pliki witryny (*.html*, *.css*, *.php* itd.) są przechowywane na serwerze. Jednak nie każdy z nich jest **przetwarzany** po stronie serwera. Pliki HTML i CSS, podobnie jak pliki graficzne, są przesyłane bezpośrednio do klienta (przeglądarki) bez sprawdzania, co się w nich znajduje. Pliki PHP działają inaczej, ponieważ zawierają kod przetwarzany i **uruchamiany** na serwerze WWW. Do przeglądarki trafia nie kod PHP, ale **efekt** jego uruchomienia, czyli kod HTML i CSS.

4. Utwórz skrypt w którym wyświetlone zostaną pojęcia i przypisane im określenia. Spis nieuporządkowanych określeń i pojęć zawiera obraz poniżej.



CO DO CZEGO SŁUŻY?

Myślisz o kosmitach? Zapomnij o nich przy dopasowywaniu komponentów HTML i PHP do poszczególnych zadań.

HTML	Aplikacja do przeglądania stron WWW i wchodzenia z nimi w interakcje. Działa jako kliencka strona komunikacji.
PHP	Polecenie języka PHP, które służy do wyświetlania zawartości strony, na przykład zwykłego tekstu lub kodu HTML.
Formularz	Znaczniki, które obejmują kod PHP, aby serwer mógł go przetworzyć i uruchomić.
Przeglądarka	Wbudowana tablica języka PHP z danymi z formularza przesłanymi metodą POST.
<?php ?>	Język programowania, który służy do tworzenia skryptów działających na serwerze.
Zmienna	Te symbole należy umieszczać wokół łańcuchów znaków.
Cudzysłowy i apostrofy	Aplikacja do udostępniania stron WWW. Działa jako serwerowa strona komunikacji.
echo	Język znaczników, który służy do opisu struktury stron wyświetlanych w przeglądarce.
$_POST	Nazwa wbudowanych zmiennych języka PHP, które są dostępne we wszystkich skryptach.
Serwer WWW	Grupa pól na dane wejściowe na stronie, służąca do pobierania informacji od użytkowników.
Tablica	Wbudowana funkcja języka PHP, która służy do wysyłania e-maili.
Zmienna superglobalna	Miejsce przechowywania informacji w skrypcie PHP. Ma określoną nazwę i typ danych.
mail()	Rodzaj kontenera na dane w kodzie PHP, umożliwiający zapisanie wielu informacji w jednym miejscu.