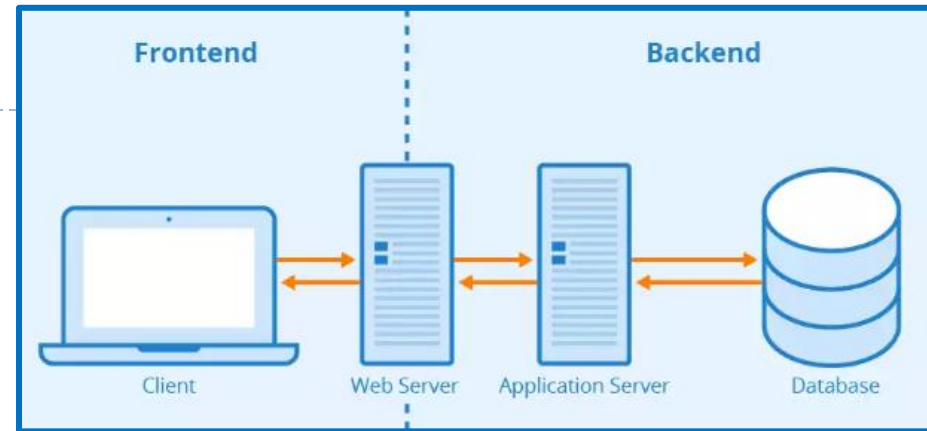


## I. PHP - podstawy

dr Artur Bartoszewski  
UTH Radom

# Czym jest backend?



Źródło: <https://teamquest.p>

Istnieje wiele technologii tworzenia mechaniki konkretnego rozwiązania informatycznego czyli backendu. Backend bowiem to ten fragment kodu, który odpowiada za działanie podstawowych funkcji aplikacji albo strony internetowej. W dziedzinie programowania (development istnieją dziś dwa główne rodzaje technologii: frontend development oraz backend development.

Przy tworzeniu frontendu koncentrujemy się na wyglądzie i działaniu aplikacji, która ma być zaprojektowana. Z kolei technologie backendowe pomagają w zarządzaniu podstawową logiką biznesową. Obie te dziedziny razem łączą więc interfejs użytkownika i mechanikę oraz bazę danych, tworząc jednolity produkt.



źródło: <https://nofluffjobs.com/>

Jakie frameworki są najczęściej wymagane przez pracodawców?

1	Hibernate	5	Laravel
2	Ruby on Rails	6	Django
3	Spring	7	Magento
4	Spark	8	Symfony

źródło: <https://nofluffjobs.com/>



## PHP

PHP pomaga nie tylko tworzyć frontend, ale także backend aplikacji. PHP jest używane w ponad 80% witryn bo pozwala na konstruowanie aplikacji internetowych w dość łatwy i skuteczny sposób. Umiejętności i koncepcje potrzebne do pracy mogą być stosunkowo bezboleśnie przyswojone przez wielu programistów.

PHP jest darmowe, istnieje też wiele frameworków i narzędzi innych firm opracowanych przy użyciu PHP. Także różne systemy zarządzania treścią, takie jak WordPress, Magento czy Drupal są tworzone przy użyciu PHP.

### Python

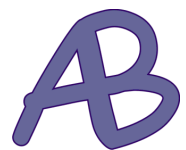
Python istnieje od 20 lat i dał się poznać we wszystkich aspektach programowania i tworzenia stron internetowych. Python jest także używany w aplikacjach backendowych, umożliwiając bezproblemowe połączenie z bazą danych i zarządzanie wszystkimi aspektami administracyjnymi aplikacji internetowej. Jest dość łatwy do zrozumienia i stanowi odpowiedni język dla procesu tworzenia zaplecza bo pozwala zapewnić wysoką przepustowość. Dostępne są też różne gotowe narzędzia wspomagające używanie Pythona w backendzie.

### Ruby On Rails

Ruby On Rails jest oparty na otwartych standardach dostarczonych przez Pythona. Wysokowydajny charakter Ruby on Rails sprawia, że jest to doskonały wybór do zarządzania i rozwijania backendu. To, że działa w oparciu o Pythona, ułatwi migrację na to rozwiązanie osobom znającym już Pythona.

### JavaScript

JavaScript jest używany zarówno w aplikacjach typu frontend, jak i backend. Deweloperzy backendu używają głównie Node.js, którego struktura umożliwia programistom obsługę danych z poziomu front-endu i tworzenie skalowalnych aplikacji sieciowych, które mogą między innymi przetwarzać wiele jednoczesnych żądań użytkowników.





# Osadzanie skryptów PHP

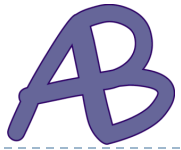
---

Osadzanie w kodzie HTML:

Najprostszym sposobem osadzania skryptów PHP w stronach internetowych jest umieszczenie kodu PHP bezpośrednio w kodzie HTML. Możesz to zrobić, używając tagu `<?php ... ?>`

```
<?php  
$zmienna = "Witaj, świecie!";  
echo $zmienna;  
?>
```





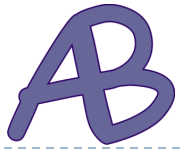
# Osadzanie skryptów PHP

---

Separacja logiki od widoku:

Dobra praktyka programistyczna sugeruje oddzielanie logiki od widoku. Możesz używać szablonów HTML i osadzać kod PHP wewnątrz tych szablonów. Przykład:

```
<html>
<head>
    <title>Strona internetowa</title>
</head>
<body>
    <h1><?php echo $tytul; ?></h1>
    <p><?php echo $tresc; ?></p>
</body>
</html>
```



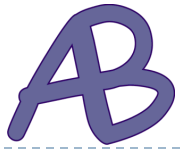
# Osadzanie skryptów PHP

---

Używanie funkcji include lub require:

Możesz osadzać fragmenty kodu PHP z innych plików, używając funkcji include lub require. W ten sposób możesz podzielić kod na mniejsze części i uniknąć powtarzania się kodu

```
<?php
    include 'header.php';
    echo "Treść strony";
    include 'footer.php';
?>
```



# Osadzanie skryptów PHP

---

Wykorzystywanie frameworków PHP:

Jeśli pracujesz nad większym projektem, zwykle używa się frameworków PHP, takich jak Laravel, Symfony, czy Zend Framework. Frameworki te mają swoje własne mechanizmy osadzania kodu PHP w widoku, często używając specjalnych szablonów i kontrolerów.

Używanie frameworków front-end:

Jeśli chodzi o front-end, można używać frameworków JavaScript, takich jak React, Angular lub Vue.js, aby zarządzać widokiem i osadzać dynamiczny kod PHP za pomocą odpowiednich endpointów API.

Wybór odpowiedniej metody zależy od konkretnej sytuacji i potrzeb projektu. Ważne jest również zapewnienie bezpieczeństwa w osadzonym kodzie PHP, zwłaszcza jeśli jest on podatny na ataki XSS (Cross-Site Scripting). Warto wówczas korzystać z funkcji takich jak `htmlspecialchars` do zabezpieczania danych przed wstrzykiwaniem niebezpiecznego kodu HTML.



Jeśli zwykłemu plikowi HTML nadamy rozszerzenie .php, to zostanie on prawidłowo wyświetlony, mimo że nie jest to skrypt PHP. Dzieje się tak dlatego, że parser PHP przetwarzając stronę ma 2 tryby pracy:

- HTML, gdzie cała treść jest wyświetlana, bez przetwarzania,
- PHP, gdzie treść jest traktowana jako skrypt do przetworzenia. Do określenia w pliku co jest kodem HTML a co PHP służą specjalne znaczniki. Początkowo parser jest w trybie HTML. Aby przejść do trybu PHP można użyć jednego z czterech znaczników:

```
<? echo („Przykładowy tekst”); ?>  
  
<?php echo(" Przykładowy tekst "); ?>  
  
<script language="php">  
    echo ("Przykładowytekst ");  
</script>  
  
<% echo ("Przykładowy tekst"); %>
```


```
<?php  
    echo "To jest test";  
?>  
  
<?php echo "To jest test" ?>
```

Można pominąć średnik gdy następuje przejście do trybu HTML, a więc po danej linii następuje symbol przejścia do trybu HTML.

PHP obsługuje 3 metody oznaczania komentarzy:

```
<?php  
  
    echo "Witaj świecie!<br>"; // To jest jednowierszowy komentarz w stylu C++  
  
/* To jest wielowierszowy  
   blok komentarza */  
  
    echo "Witamy ponownie.<br>"; # To jest komentarz w stylu skryptów Uniksa  
?>
```

```
<?php  
    /*  
        echo "A tutaj mamy problem."; /* Komentarz  
    ten jest  
                                   nieprawidłowy */  
    */  
?>
```



**echo** nie jest funkcją, ale konstruktem języka. Jego argumenty są listą wyrażeń następujących po słowie kluczowym, oddzielonych przecinkami i nierozdzielonych nawiasami. W przeciwieństwie do niektórych innych konstrukcji językowych, nie ma żadnej wartości zwracanej,

Podobną rolę jak echo pełnić może funkcja **print()**;

```
<?php
echo "tekst do wypisania.";

echo "hello";
echo "world";

// to samo co powyżej - wiele parametrów
echo "hello", "world";

echo "Tekst może być wypisany
w wielu
liniach";

echo "Tekst może być wypisany\n w wielu\n liniach";
// UWAGA - HTML i tak to zignoruje

// Zmienna zostanie przekonwertowana na string
$zmienna = 101;
echo „Wartość zmiennej = $zmienna”;
?>
```



# Print()



Funkcja `print()`;

```
print("Hello, world!"); // Z nawiasami  
print "Hello, world!"; // Bez nawiasów (dozwolone, ale mniej popularne)
```

Funkcja `print()` automatycznie dodaje znak nowej linii ("`\n`") na końcu wyjścia, więc każde wywołanie `print()` rozpoczyna nową linię.

PHP posiada następujące typy:

1. liczby zmiennoprzecinkowe,
2. liczby całkowite,
3. ciągi,
4. Tablice,
5. obiekty.

Typ zmiennej ustalany jest w oparciu o kontekst w jakim jest użyta (nie jest ustalany jawnie).

Należy o tym pamiętać ponieważ niejawna konwersja typów może spowodować trudne do odnalezienia błędy.

Np.: poniższa instrukcja jest prawidłowa i spowoduje wyświetlenie liczby 15:

```
print( 3* „5 lat”);
```

PHP posiada funkcje `gettype()` i `settype()` oraz kilka funkcji przeznaczonych dla określonych typów, na przykład: `is_integer()` lub `is_array()`.

## Liczby całkowite i zmiennoprzecinkowe

```
<?php
    $int1 = 450;           // liczba dziesiętna
    $int2 = -8;            // dziesiętna ujemna
    $int3 = 01016;         // ósemkowa reprezentacja liczby
    $int4 = 0x10A;         // szesnastkowa reprezentacja liczby
    $float1 = 523.197;      // liczba zmiennoprzecinkowej
    $float2 = 5.23197e2;    // notacja naukowa liczby zmiennoprzecinkowa
?>
```

## Ciągi znaków (łańcuchy)

Łańcuchy w PHP są ograniczane apostrofami (') lub cudzysłowami (").  
Różnią się sposobem interpretacji ciągu.

- Cudzysłowy - zmienne zapisane w łańcuchu zostają zamienione na ich wartości.
- Apostrofy – łańcuch wypisany „dosłownie” – za wyjątkiem: lewy ukośnik (\\) i apostrof (\').

Aby zapisać znaki specjalne w łańcuchu otoczonych cudzysłowami, należy użyć lewego ukośnika

Znak serujący	Znaczenie
\n	nowa linia
\r	powrót karetki (CR)
\t	tabulacja
\\	lewy ukośnik
\"	cudzysłów
\\$	znak dolara

## Konwersja łańcuchów na liczby

Jeżeli PHP próbuje skonwertować ciąg na liczbę, korzysta z następujących zasad:

- Jeżeli ciąg zaczyna się od danych numerycznych, zostaną one skonwertowane na liczbę.
- Jeżeli ciąg nie zaczyna się prawidłowymi danymi liczbowymi, wartością ciągu będzie zero (0).
- Jeżeli dane numeryczne zawierają jeden ze znaków .,e lub E, wartość będzie liczbą zmiennoprzecinkową a w przeciwnym przypadku liczbą całkowitą.

## Operatory porównania

Operator	Nazwa	Wynik
==	Równy	True, jeżeli \$a jest równe \$b
===	Identyczny	True, jeżeli \$a jest równe \$b i są one tych samych typów
!=	Różny	True, jeżeli \$a jest różne od \$b
<	Mniejszy	True, jeżeli \$a jest mniejsze od \$b
>	Większy	True, jeżeli \$a jest większe od \$b
<=	Mniejszy lub równy	True, jeżeli \$a jest mniejsze lub równe \$b
>=	Większy lub równy	True, jeżeli \$a jest większe lub równe \$b

## Operatory inkrementacji i dekrementacji

Operator	Nazwa	Wynik
\$a++	Postinkrementacja	Zwraca \$a, a następnie zwiększa \$a o jeden
++\$a	Preinkrementacja	Zwiększa \$a o jeden i zwraca \$a
\$a--	Postdekrementacja	Zwraca \$a, a następnie zmniejsza \$a o jeden
--\$a	Predekrementacja	Zmniejsza \$a o jeden i zwraca \$a

## Operatory przypisania

Operator	Wynik
=	Przypisuje wartość \$b do \$a.
+=	Przypisuje wartość (\$a+\$b) do \$a. (\$a=\$a+\$b)
-=	Przypisuje wartość (\$a-\$b) do \$a. (\$a=\$a-\$b)
*=	Przypisuje wartość (\$a*\$b) do \$a. (\$a=\$a*\$b)
/=	Przypisuje wartość (\$a/\$b) do \$a. (\$a=\$a/\$b)
.=	Przypisuje wartość (\$a.\$b) do \$a. (\$a=\$a.\$b)
%=	Przypisuje wartość (\$a%\$b) do \$a. (\$a=\$a%\$b)
=	Przypisuje wartość (\$a \$b) do \$a. (\$a=\$a \$b)
&=	Przypisuje wartość (\$a&\$b) do \$a. (\$a=\$a&\$b)
^=	Przypisuje wartość (\$a^\$b) do \$a. (\$a=\$a^\$b)
<<=	Przypisuje wartość (\$a<<\$b) do \$a. (\$a=\$a<<\$b)
>>=	Przypisuje wartość (\$a>>\$b) do \$a. (\$a=\$a>>\$b)

Aby zdefiniować nową stałą używa się funkcji **define()**

```
<?php
    define( „Napis”, „Stały łańcuch znaków” );
    define( „Liczba”, 100 );
?>
```



## Tablice (wstęp)



Tablicę można utworzyć za pomocą konstrukcji języka `array()`. Przyjmuje dowolną liczbę par rozdzielonych przecinkami jako argumenty. `key => value`

```
<?php
    $array = array(
        "foo" => "bar",
        "bar" => "foo",
    );
// krótszy zapis
    $array = [
        "foo" => "bar",
        "bar" => "foo",
    ];
?>
```

Tablica indeksowana, bez klucza

```
<?php
    $array = array("foo", "bar", "hello", "world");
?>
```

## Jawne tworzenie tablic

```
<?php
// Jawne tworzenie prostej tablicy
$tab01[0] = "tekst 01";
$tab01[1] = "tekst 02";
$tab01[] = "tekst 03";
// przypisanie do pierwszego wolnego indeksu (w typ przypadku 2)
$tab01[] = "tekst 04";
echo "$tab01[0], $tab01[1], $tab01[2], $tab01[3]<br>";

// Tworzenie tablicy asocjacyjnej
$color["niebieski"] = "#0000FF";
$color["zielony"] = "#00FF00";
$color["czerwony"] = "#FF0000";
echo "Wartość szesnastkowa koloru czerwonego wynosi
{$color['czerwony']}<br>";

?>
```

# Tablice

Jawne tworzenie  
tablic c.d.



```
<?php
// Tworzenie tej samej co poprzedniej tablicy asocjacyjnej
// inny zapis
$color = array( "niebieski" => "#0000FF",
                "zielony"   => "#00FF00",
                "czerwony"  => "#FF0000" );
echo "Wartość szesnastkowa koloru zielonego wynosi
{$color['zielony']}<br>";

// Ręczne tworzenie tablicy wielowymiarowej
$tab02[0][0] = "Zero Zero";
$tab02[0][1] = "Zero Jeden";
echo "Wartością \$tab02[0][1] jest {$tab02[0][1]}<br>";

// Ręczne tworzenie asocjacyjnej tablicy wielowymiarowej
$tab03["Idaho"][0] = "Ada";
$tab03["Idaho"][1] = "Adams";
$tab03["Idaho"][2] = "Bannock";
$tab03["Arizona"][0] = "Apache";
$tab03["Arizona"][1] = "Cochise";
$tab03["Arizona"][2] = "Coconino";
echo "\$counties['Idaho'][0] = {$tab03['Idaho'][0]}<br>" ;
?>
```

# Instrukcje wyboru



## if else

```
<?php
    if ( 1 < 2 )
        echo "To zostanie wypisane.<br>";
    else
        echo "To nie zostanie wydrukowane.<br>";
?>
```

## elseif

```
<?php
    $dana = 2;
    if ( $dana == 1 )
    {
        print( "\$dana == 1<br>" );
    }
    elseif ( $dana == 2 )
    {
        echo "\$dana == 2<br>";
    }
    elseif ( $dana == 3 )
    {
        echo "\$dana == 3<br>";
    }
    else
    {
        echo "\$dana nie jest 1, 2 ani 3<br>";
    }
?>
```

```
<?php
    $nIndex = 1;
    switch ( $nIndex )
    {
        case 0:
            echo "zero<br>";
            break;
        case 1:
            echo "jeden<br>";
            break;
        case 2:
            echo "dwa<br>";
            break;
        default:
            echo "Nie jest to zero,
jeden ani dwa<br>";
    }
?>
```

Podobnie jak w c nie można zapomnieć o instrukcji brak

Zmienną sterującą może być też łańcuch

# Pętla while i do while



## while

```
<?php
    $i = 0;
    while($i<10)
    {
        $i++;
        echo "Punkt $i.<br>";
    }
?>
```

## do while

```
<?php
    $i = 0;
    do
    {
        $i++;
        echo "Punkt $i.<br>";
    } while($i<10);
?>
```

Obie pętle działają dokładnie tak samo jak w języku c

# Pętla for i foreach



for

```
<?php
    for ($i=0; $i<10; $i++)
    {
        echo "Punkt $i.<br>";
    };
?>
```

Pytanie: czy pokazana tu pętla  
robi dokładnie to co pętla z  
poprzedniego slajdu ?

foreach

```
<?php
    $paleta = array( "red", "green", "blue" );
    foreach( $paleta as $kolor )
    {
        echo "<p style=\"color: $kolor ;\">Bieżąca wartość to $kolor</p>";
    }
?>
```

Bieżąca wartość to red

Bieżąca wartość to green

Bieżąca wartość to blue

# Pętla for i foreach

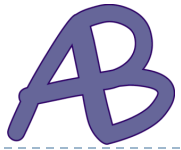


**foreach**

```
foreach ($tab as $klucz => $wartosc)
{
    echo @klucz;
    echo @wartosc;
}
```

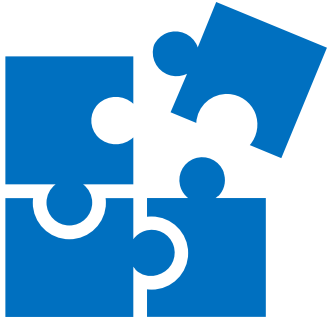
Pętla przeglądająca tablicę  
asocjacyjną (klucz wartość)





## Przykład do wykonania

---



Należy napisać skrypt który wygeneruje tabelę:

```
<table>
  <tr>
    <td>1</td> <td>2</td> <td>3</td>
  </tr>
  <tr>
    <td>4</td> <td>5</td> <td>6</td>
  </tr>
  <tr>
    <td>7</td> <td>8</td> <td>9</td>
  </tr>
</table>
```



## Literatura

---

W prezentacji użyto przykładów z książki:

- Żygłowicz Jerzy - PHP - Kompendium wiedzy, Helion

- <https://www.php.net>