Temat : Język programowania php.

PHP to język skryptowy, wykonywany po stronie serwera. Kod PHP osadzony na stronach HTML, nie jest widoczny dla użytkowników, w przeciwieństwie np. do skryptów JavaScript, jest natomiast interpretowany przez serwer WWW, który na podstawie kodu, tworzy stronę HTML, lub inne wyniki.

1. Zalety PHP:

- Bardzo niski koszt, -php jest bezpłatny (http://www.php.net)
- Wysoka wydajność. PHP jest w stanie obsłużyć na jednym serwerze bardzo wiele odwiedzin dziennie.
- Integracja z bazami danych. PHP posiada interfejs do wielu różnych systemów baz danych: najpopularniejszy to MySQL.
- Wbudowane biblioteki. PHP posiada wiele wbudowanych funkcji do rozwiązywania różnych popularnych zadań w sieci WWW.
- Przenośność. PHP jest dostępne dla wielu systemów operacyjnych

2. ***Xampp*** - darmowy program do tworzenia zaawansowanych stron internetowych opartych na php i bazach danych MySQL. Program umożliwia uruchomienie na komputerze lokalnym serwera www z wszystkimi najważniejszymi składnikami i tworzenie strony bez konieczności ciągłego łączenia się z naszym serwerem w internecie. Plik index.php znajduje się C:/XAMPP / HTDOCS / INDEX.PHP
Wpisujemy w przeglądarce lokalhost, lub wpisujemy adres 127.0.0.1

3. Wygląd skryptu PHP

Dokumenty, które zawierają skrypty PHP składają się zazwyczaj z kodu HTML z wplecionym tekstem programu. Oznacza to, że tekst skryptu PHP umieszczamy bezpośrednio w kodzie HTML oddzielając go specjalnymi znacznikami. Początek i koniec skryptu mogą tu być oznaczone na różne sposoby:

<?php ?>

4. Instrukcje echo, print – instrukcje wyjściowe (pomiędzy echo, a print nie ma żadnej różnicy)

```
echo ”ale” ;
echo( ‘fajny’ );
print ‘ten’;
print(‘skrypt’ );
```

Wszystkie instrukcje są poprawne

UWAGA!
W php po każdej instrukcji wymagany jest średnik, w przeciwnym razie na ekranie wystąpi błąd
Napis otoczony znakami cudzysłowu może zawierać znaki specjalne oraz zmienne. Nazwy zmiennych zostaną zastąpione ich wartościami. W napisie otoczonym apostrofami wszystkie znaki są interpretowane dosłownie.

6. **Znacznik META** stanowi tzw. *metainformację*, pozwalającą określić pewne ogólne wiadomości, dotyczące dokumentu, m.in. sposób kodowania znaków, opis zawartości strony, jej autora czy język, w którym została napisana. Metainformacje nie wpływają bezpośrednio na wygląd dokumentu, lecz cechy które podają, są równie ważne. Chociaż nie są one wymagane, warto je stosować, ponieważ może to np. pomóc w odszukaniu strony przez wyszukiwarki sieciowe.)

Przykład

```
<HTML>
<HEAD>
<META HTTP-EQUIV="Content-Type" Content="tekst/html"; charset=iso-8859-2">//lub /<META CHARSET=UTF-8>
<TITLE>Mój pierwszy skrypt PHP
</TITLE>
</HEAD>
<BODY>

<?
// Skrypt ten wypisze tekst, używając funkcji "print"
print("Ale fajny ten skrypt");
?>

</BODY>
</HTML>
```

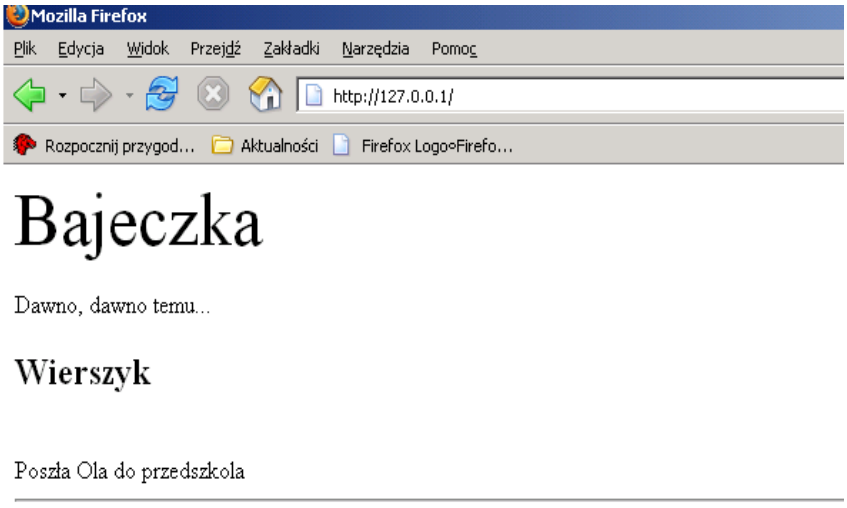
7. **Komentarze -** możemy je zawrzeć wewnątrz znaczników:

/* */ Komentarz zawarty pomiędzy tymi znacznikami może zajmować dowolną ilość wierszy
// Komentarz za tym znacznikiem może być zawarty tylko do końca danego wiersza
Jak wyżej. Wszystko do końca wiersza jest komentarzem

8. Aby skrypt został wykonany, serwer WWW musi stwierdzić, że jest to właśnie skrypt PHP. W tym celu plik musimy zapisać z odpowiednim rozszerzeniem. .php.php3.php4.phtml

przykład

```
// Skrypt ten wypisze tekst, używając funkcji "print"
```



Temat : PHP – zmienne.

1. Zmienna przechowuje dane w trakcie wykonywania skryptu. Każda zmienna posiada swoją nazwę i wartość. . Nazwy zmiennych mogą się składać z liter, cyfr i znaków podkreślenia. Nie mogą jednak rozpoczynać się od cyfry. Każdą nazwę zmiennej poprzedza się znakiem dolara "$" np. $liczba1, $MojeImie
Należy pamiętać, że w nazwach zmiennych jest uwzględniana wielkość liter.

2. Typy zmiennych
Zmienne w PHP dzielą się na typy, np.:

- liczby całkowite (integer)
- liczby rzeczywiste (double)
- ciągi (string)
- tablice (array)
- obiekty (object)

$imie="Krzysztof" - zmienna typu string
$liczba1=30 - zmienna typu integer
$liczba2=30.4 -zmienna typu double

operatory matematyczne

operator	przeprowadzana operacja
+	dodawanie $a=$b+3;
-	odejmowanie $a=$b-3;
*	mnożenie $a=$b*3;
/	dzielenie $a=$b/3;
%	dzielenie modulo (zwraca liczbę całkowitą stanowiącą resztę z dzielenia) Reszta z dzielenia $a przez $b.
++	inkrementacja $a++ (dodaje do zmiennej wartość 1)
--	dekrementacja $a-(odejmuje od zmiennej wartość 1)
pow(podstawa, wykładnik)	potęgowanie

Operatory logiczne oraz operatory porównania

Operatory porównania jak sama nazwa wskazuje porównują wartości (np. zmiennych) i zwracają wartość TRUE (prawda) lub FALSE (fałsz).

operator	przeprowadzana operacja
<	mniejsze $a<$b (zwraca wartość true gdy $a jest mniejsze od $b)
>	większe $a>$b (zwraca wartość true gdy $a jest większe od $b)
<=	mniejsze lub równe $a<=$b (zwraca wartość true gdy $a jest mniejsze lub równe $b)
>=	większe lub równe $a<=$b (zwraca wartość true gdy $a jest większe lub równe $b)
==	Operator porównania $a==$b (zwraca wartość true gdy $a jest równe $b)
!=	różne $a!=$b (zwraca wartość true gdy $a jest różne od $b)
&& (AND)	koniunkcja (logiczne i) $a==1 && $b==2 (zwraca wartość true jeśli jest spełniony warunek dla $a i $b)
|| (OR)	alternatywa (logiczne lub) $a==2 || $b==1 (zwraca wartość true jeśli jest spełniony warunek dla $a lub $b)
! (NOT)	negacja !$a==1 (negacja powoduje zmianę prawda na fałsz)

Należy zwrócić tu uwagę na operator przypisania (=) oraz operator równości (==). Pierwszy z nich sprawia, że obie strony działania są sobie równe np.: $a=45 (W tym wypadku zmiennej $a została przypisana wartość 45). Operator równości pyta zaś, czy obie strony działania są sobie równe (jeśli tak, to zwraca wartość true, jeśli nie, to zwraca wartość false).

Operatory przypisania

Wszystkie operatory przypisania zapisują wartość w zmiennej.

$a=785.5;

Operatory utworzone przez połączenie operatora przypisania z innym operatorem, dokonują obliczeń na podstawie wartości umieszczonej z jego prawej strony jak i wartości zmiennej zapisanej z lewej strony operatora. Wynik zaś zapisywany jest w zmiennej, umieszczonej z lewej strony operatora np.:

$a=10; //zmiennej $a została przypisana wartość z prawej strony czyli 10
$a+=5; //do zmiennej $a=10 dodano 5, czyli zmienna $a=15. Ten zapis jest tym samym co: $a=$a+5;

operator	przeprowadzana operacja
=	przypisuje zmiennej zapisanej z lewej strony operatora wartość wyrażenia umieszczonego z jego prawej strony.
+=	dodaje wartość podaną z prawej strony operatora do wartości zmiennej zapisanej z jego lewej strony i zapisuje ją w tej zmiennej.
-=	odejmuje wartość podaną z prawej strony operatora od wartości zmiennej zapisanej z jego lewej strony i zapisuje ją w tej zmiennej.
*=	mnoży wartość podaną z prawej strony operatora przez wartość zmiennej zapisanej z jego lewej strony i zapisuje ją w tej zmiennej.
/=	dzieli wartość podaną z prawej strony operatora przez wartość zmiennej zapisanej z jego lewej strony i zapisuje ją w tej zmiennej.
%=	przypisuje zmiennej zapisanej z lewej strony operatora jej wartość podzieloną modulo przez wartość podaną z prawej strony operatora.
.=	przypisuje zmiennej zapisanej z lewej strony operatora łańcuch znaków będący połączeniem łańcucha zapisanego w zmiennej z łańcuchem podanym z prawej strony operatora.
Pobieranie zmiennej z formularza	$_POST[‘nazwa zmiennej’]

3. Stałe

Często potrzebujemy, aby pewne obiekty miały niezmienną, z góry określoną wartość. Takie obiekty nazywamy stałymi. Ich wartość nie ulegnie zmianie podczas wykonywania programu.
Stałe definiujemy za pomocą funkcji

define (definiowanie wartości stałej) np.:

define ("JAKAŚ_STAŁA", 247.6)	Funkcja ta posiada dwa argumenty: nazwa stałej ("JAKAŚ_STAŁA") oraz jej wartość (tutaj wynosi ona 247.6)

Nazwa zmiennej została zapisana wielkimi literami. Nie jest to konieczne, ale dla odróżnienia od zmiennych powszechnie przyjęte.

instrukacja warunkowa if

Często będziemy chcieli wykonać jakieś działanie, ale tylko wtedy, gdy zostanie spełniony pewien warunek. W tym celu posłużymy się instrukcją if. Ma ona następującą postać:

```
if (warunek_1) {
blok instrukcji 1
} elseif (warunek_2) {
blok instrukcji 2
} elseif (warunek_3) {
blok instrukcji 3
.....
} else {
blok instrukcji n
}
```

lub forma uproszczona:

```
if (warunek) {
blok instrukcji
}
```

Działanie tej instrukcji nie zbyt skomplikowane. Najpierw jest sprawdzana wartość warunku_1. Jeżeli jest on prawdziwy (czyli ma wartość TRUE) wykonywany jest blok instrukcji 1. Jeśli tak nie jest (czyli warunek_1 ma wartość FALSE) cały ten blok jest pomijany i sprawdzane są kolejne warunki i jeśli jeden z nich jest spełniony to wykonywany jest odpowiedni blok instrukcji. Gdy zostanie odnaleziona instrukcja, której warunek będzie posiadał wartość TRUE, to pozostałe instrukcje zostaną pominięte. Jeżeli żaden z warunków nie jest spełniony to wykonywany jest blok instrukcji w warunku else (o ile ten istnieje).

Jeżeli zdecydujemy się na formę uproszczoną to zostanie sprawdzony warunek. Jeśli jest spełniony (wartość TRUE) to zostanie wykonany blok instrukcji. W przeciwnym razie (warunek posiada wartość FALSE) program przechodzi do kolejnych instrukcji.