Laboratorium 7

1. Podstawy programowania obiektowego w PHP

1.1 Klasy i obiekty

Klasa i obiekt to dwa kluczowe pojęcia towarzyszące programowaniu obiektowemu. Pisanie kodu obiektowego polega na definiowaniu klas, związków i relacji między nimi oraz manipulowaniu obiektami. Klasa jest typem danych zaś obiekt — zmienną.

W języku PHP na ogół nie deklarujemy typu zmiennej. Jeśli zmienna przechowuje liczbę czy napis to w skrypcie nie występuje jawne ustalenie typu. Typ zmiennej wynika z wykonanych na niej operacji.

Do sprawdzenia typu zmiennej stosujemy funkcję *var_dump()*. Po wykonaniu poniższego kodu zmienna *$z1* będzie typu *int*, o czym przekona nas uzyskany wydruk:

```
//listing 7-1
$z1 = 123;
var_dump($z1);
```

Sprawdź: http://pgmys.edu.pl/paiklasa4/lekcje/listing7-1.php

W przeglądarce pojawi się tekst *int(123)* informujący o typie (*int*) i wartości (*123*) zmiennej *$z1*. W powyższym przykładzie występuje zatem typ *int* oraz zmienna *$z1*, przy czym typ *int* pojawia się w sposób nie jawny: wynika on z wartości przypisanej do zmiennej.

W przypadku zmiennych będących obiektami sprawa wygląda odmiennie. Tworząc obiekt musimy podać nazwę klasy (czyli typ zmiennej). Kod obiektowy:

```
//listing 7-2
//definicja typu
class ALAMAKOTA
{
}

//utworzenie zmiennej
$z2 = new ALAMAKOTA;

//sprawdzenie typu zmiennej
var_dump($z2);
```

sprawdź: http://pgmys.edu.pl/paiklasa4/lekcje/listing7-2.php

zawiera definicję typu *ALAMAKOTA* oraz zmienną *$z2*. Tym razem w przeglądarce pojawi się tekst *object(alamakota)* mówiący o tym, że obiekt *$z2* jest typu obiektowego *ALAMAKOTA*.

Pomiędzy powyższymi dwoma przykładami należy odnotować dwie różnice. Typ *int* jest predefiniowany i nie musimy pisać jego definicji, podczas gdy klasa *ALAMAKOTA* musi być

zawarta w skrypcie. Druga różnica polega na tym, że typ zmiennej *$z1* zostaje ustalony na podstawie wartości 123 przypisanej do zmiennej, zaś w odniesieniu do obiektu *$z2* nazwa typu *ALAMAKOTA* zostaje jawnie podana.

W dwóch przedstawionych przykładach występują: typ *int* i zmienna *$z1*oraz typ *ALAMAKOTA* i zmienna *$z2*.

W stosunku do typów i zmiennych obiektowych stosujemy terminologię klasa i obiekt. Mówimy „klasa *ALAMAKOTA*" oraz „obiekt *$z2*".

1.2 Definicja klasy

Definicja klasy w języku PHP jest następująca:

```
class A
{
    //składowe klasy
}
```

Po słowie kluczowym ***class*** umieszczamy nazwę klasy, zaś pomiędzy nawiasami klamrowymi — składowe klasy, którymi w PHP są *pola* oraz *metody*. Pola to zmienne, w których przechowujemy dane, natomiast metody to funkcje przetwarzające obiekt. Kolejność pól i metod w definicji klasy może być dowolna, warto jednak stosować ogólnie przyjętą konwencję i umieszczać na początku klasy wszystkie pola, po nich zaś — metody.

W części języków programowania składową klasy może być także *właściwość* — specjalne pole, do którego dostęp powoduje wywołanie dodatkowych metod. W PHP4 właściwości nie występują w ogóle, zaś w PHP5 do definiowania właściwości możemy użyć funkcji *__get()* oraz*__set()*. Termin *właściwość* jest stosowany także w odniesieniu do zwykłych pól klasy.

Umieszczenie powyższej klasy wewnątrz skryptu nie spowoduje żadnych widocznych efektów. Definicja klasy jest definicją typu. Ustala ona budowę obiektu: określa jego pola i metody. Jednakże żaden konkretny obiekt podanego typu nie jest tworzony, a metody nie są wywoływane.

Klasy w języku PHP podlegają dwóm następującym ograniczeniom składniowym:

- klasa musi być umieszczona w jednym bloku kodu PHP,
- definicje metod muszą znajdować się wewnątrz definicji klasy.

1.3 Obiekty — instancje klasy

Jak już powiedzieliśmy obiekt jest zmienną, której typ jest klasą. Obiekty tworzymy nieco inaczej niż zmienne skalarne. W przypadku zwykłych zmiennych wystarczy instrukcja przypisania:

```
$imie = 'Jan';
```

natomiast w przypadku obiektu stosujemy operator *new*, po którym podajemy nazwę klasy. Jeśli w skrypcie pojawia się definicja klasy *A*, wówczas utworzenie obiektu podanego typu przyjmie postać:

```
$ob = new A;
```

Mówimy „obiekt *$ob* klasy *A*" lub nazywamy zmienną *$ob* „instancją klasy *A*" czy też „egzemplarzem klasy *A*".

1.4 Dostęp do pól i metod

Do składowych klasy odwołujemy się operatorem ->. Jeśli klasa *A* posiada pola *$pole1*, *$pole2* oraz metody *fun1()* i *fun2()*, wówczas po utworzeniu obiektu *$tmp* możemy przypisywać wartości do pól i wywoływać metody:

```
$tmp = new A;

$tmp->pole1 = "Ala ma kota";

$tmp->fun1();
```

Zwróćmy uwagę na fakt, że znak dolara pojawia się zawsze jeden raz przed zmienną. W stosunku do pola *$pole1* poprawnym odwołaniem jest:

```
$tmp->pole1 = "Ala ma kota";
```

podczas, gdy instrukcja:

```
PRZYKŁAD NIEPOPRAWNY

$tmp->$pole1 = "Ala ma kota";
```

jest błędna z racji na wystąpienie znaku dolara przed nazwą pola *pole1*(pomimo, że dolar występuje przed nazwą pola w definicji klasy *var $pole1;*).

1.5 Referencja this

Również wewnątrz definicji metod klasy możemy uzyskiwać dostęp do jej składowych. W tym celu stosujemy referencję ***$this***. Referencja *$this* jest nazywana w dokumentacji języka PHP *pseudo zmienną*. Odwołania takie mają postać:

```
function fun1()
{
    $this->pole1 = 10;
    $this->fun2();
}
```

Pamiętajmy, że odwołania pozbawione referencji *$this* (takie, które byłyby poprawne w językach C++) w PHP są niepoprawne. Jeśli wewnątrz metody *fun1()* napiszemy:

```
function fun1()
{
    $pole1 = 33;
}
```

to instrukcja taka spowoduje utworzenie lokalnej zmiennej o nazwie *$pole1* wewnątrz funkcji *fun1()*. Wartość 33 nie będzie zapisana do pola obiektu, a do zmiennej lokalnej i zostanie utracona po zakończeniu działania funkcji. Jest to dość powszechny błąd nękający programistów rozpoczynających poznawanie obiektowego PHP. Błąd jest o tyle kłopotliwy, że nie powoduje (w odróżnieniu do dwóch dolarów *$obiekt->$pole*) komunikatów ze strony parsera PHP.

Warto odnotować, że w niektórych specyficznych przypadkach referencja *$this* nie wskazuje obiektu, w którym wystąpiła w definicji. Dzieje się tak w przypadku statycznego wywoływania metod klasy przy użyciu operatora ::. W poniższym kodzie referencja *$this* wskazuje obiekt *b* klasy *B*, a nie, jak by się wydawało obiekt klasy *A*:

```
class A
{
    var $pole1 = 'black';
    function fun1()
    {
        echo $this->pole1;
    }
}
class B
{
    var $pole1 = 'white';
    function fun2()
    {
        A::fun1();
    }
}
$b = new B;
$b->fun2();
```

Powyższa dwuznaczność została w PHP rozwiązana poprzez zakaz stosowania referencji *$this* wewnątrz metod statycznych.

1.6 Konstruktor w PHP4

Pośród wszystkich metod danej klasy jedna pełni specjalną rolę. Metodą tą jest konstruktor. W PHP4 konstruktorem jest metoda o nazwie identycznej jak klasa, w której występuje. Konstruktor klasy *A* nazywa się *A* i jest zdefiniowany w klasie *A*:

```
class A
{
    function A()
    {
        echo '...xxx...';
    }
}
```

Konstruktor jest wywoływany automatycznie podczas tworzenia obiektu. Zatem umieszczenie w kodzie skryptu instrukcji:

```
$tmp = new A;
```

spowoduje wywołanie konstruktora klasy *A*, skutkiem czego w przeglądarce pojawi się tekst *"...xxx..."*.

Konstruktor, podobnie jak i inne metody, może posiadać parametry. W takim przypadku utworzenie instancji wymaga podania po nazwie klasy parametrów konstruktora:

```
$ob = new Ksiazka($tytul, $id);
```

Przykład.

Utworzona zostanie przykładowa klasa **Slon** przykład

```
<?
class Slon
{
    public $imie;
    public $dlugoscTraby;
    public $nieTegoChce = ':(';

    public function machnijTraba() {
        echo 'macham traba';
    }
};
$dumbo = new Slon;          //1
$dumbo->imie = 'dumbo';     //2
```

```
$dumbo->dlugoscTraby = 50;
$dumbo->machnijTraba();      //3
echo $dumbo->dlugoscTraby;   //4
?>
```

Sprawdź: http://pgmys.edu.pl/paiklasa4/lekcje/listing7-3.php

- //1 // Tworzymy obiekt klasy Slon, referencję czyli jego adres przechowujemy w zmiennej $dumbo.
- //2 // Ustawiamy właściwość $imie. Do właściwości lub metody odwołujemy się operatorem ->. Nazwy właściwości podajemy bez znaku dolara $
- //3 // Wywołujemy metodę machnijTraba()
- //4 // Wypisujemy na ekran wartosc właściwości $dlugoscTraby

Należy pamiętać że nazwy właściwości do której się odwołujemy podajemy bez znaku dolara $

Ćwiczenie

Utwórz klasę w PHP na podstawie poniższego opisu

Janek, Ania, Zosia to obiekty klasy Człowiek. Każde z nich może spać, jeść, poruszać się i to są właśnie metody zdefiniowane w klasie Człowiek. Oprócz tego każdy człowiek posiada imię oraz datę urodzenia, jednak są one indywidualne dla każdego obiektu, czyli nie są bezpośrednio powiązane z klasą, a z jej instancją (obiektem).

Rozwiązanie:

```
class Human
{
    private $_name;
    private $_birthDate;

    public function __construct($name, $birthDate) {/*...*/}
    public function eat() {/*...*/}
    public function sleep() {/*...*/}
    public function move() {/*...*/}
}

$janek = new Human('Janek', '1982-01-09');
$anna = new Human('Anna', '1950-02-19');
$zosia = new Human('Zosia', '1999-11-17');
```

Zadania:

1. **Napisz program, realizujący prosty licznik tekstowy, którego wskazanie jest pamiętane w pliku tekstowym. Pamiętaj o blokowaniu pliku.**

2. **Napisz program, który będzie wyświetlał przysłowie na dany dzień – jedno wylosowane z wielu, zapamiętanych w pliku tekstowym.**

3. **Napisz program, który będzie zbierał informacje o użytkowniku. Dane niech będą zbierane w pliku czasowym, którego nazwa będzie przekazywana w ukrytym polu.**
4. **Utwórz stronę www, która będzie umożliwiała:**
   a) tworzenie pliku
   b) dopisywanie do pliku (na końcu pliku)
   c) odczyt pliku

   Zastosuj odpowiednio przygotowane formularze