**Klasy i obiekty w PHP**

Podstawowym elementem programowania obiektowego jest klasa. Klasa zawiera zmienne (*właściwości*) i funkcje (*metody*). Obiekt jest przedstawicielem (*instancją*) klasy. Jest tworzony na podstawie klasy.

W języku PHP można programować obiektowo ale nie jest to wymuszone. Do zalet podejścia obiektowego należy:

* przejrzystość kodu programu,
* elastyczność kodu,
* łatwość rozbudowy aplikacji,
* możliwość powtórnego wykorzystania kodu.

**Definiowanie klasy i obiektu**

Do definiowania klasy służy: Tworzenie obiektu:

$nazwa\_zmiennej = **new** nazwa\_klasy();

class nazwa\_klasy

{

//składniki klasy

}

***Przykład***

<?php

class Osoba

{

public $imie,$nazwisko,$sr\_ocen;

function wpisz($im, $naz, $sr)

{

$this->imie=$im;

$this->nazwisko=$naz;

$this->sr\_ocen=$sr;

}

function wypisz()

{

echo '<p>IMIĘ:'.$this->imie.' NAZWISKO: '.$this->nazwisko.' ŚREDNIA OCEN: '.$this->sr\_ocen.'</p>';

}

}//koniec klasy

$os1=new Osoba(); //stworzenie obiektu $os1

$os1->wpisz('Jan','Kowalski',2.56);

$os1->wypisz();

?>

***Uwaga do hermetyzacji:*** Dostępem do składników klasy regulujemy za pomocą: *private*, *public* i *protected*. Przez domniemanie przyjmuje się dostęp *public*.

***Ćwiczenie 1*** Zapisz klasę *Pracownik*. Wstaw do niej wybrane cechy i metody. Utwórz obiekty tej klasy i pokaż na stronie, że funkcjonują.

**Konstruktor i destruktor**

**Konstruktor** to specjalna metoda klasy wywoływana automatycznie podczas tworzenia obiektu tej klasy. Głównym jej przeznaczeniem jest inicjalizacja właściwości obiektu.

**Destruktor** to specjalna metoda klasy wywoływana automatycznie, gdy obiekt jest usuwany z pamięci (niszczony). Głównie jest używany wtedy, gdy obiekt używa zasobów, które trzeba zwolnić przed usunięciem obiektu (np. zamykanie plików, zamykanie połączeń z bazą danych).

function \_\_destruct()

{

//ciało destruktora

}

function \_\_construct()

{

//ciało konstruktora

}

***UWAGA!*** Przed nazwą jest \_\_ czyli podwójna „podłoga”.

***UWAGA!*** Konstruktor może mieć parametry lub nie ale w klasie może być **tylko jeden**.

***Przykłady***

Konstruktor bez parametrów: Konstruktor z parametrami:

function \_\_construct($im, $naz, $sr)

{

$this->imie=$im;

$this->nazwisko=$naz;

$this->sr\_ocen=$sr;

}

function \_\_construct()

{

$this->imie='Jerzy';

$this->nazwisko='Nowak';

$this->sr\_ocen=3.5;

}

Destruktor:

function \_\_destruct()

{

echo 'No... już po Tobie...'.$this->nazwisko;

}

***Ćwiczenie 2*** Dodaj do klasy Pracownik konstruktor i destruktor. Przetestuj obiekty tej klasy.

**Dziedziczenie**

**Dziedziczenie** to mechanizm w programowaniu obiektowym umożliwiający tworzenie hierarchii klas. Dzięki niemu można stworzyć nową klasę (tzw. klasę potomną) na podstawie klasy już istniejącej (tzw. klasy bazowej).

Poniżej zapisano klasę o nazwie ***Corka*** (jest to klasa potomna) jako stworzoną na podstawie klasy ***Matka*** (jest to klasa bazowa). Mechanizm dziedziczenia realizuje słowo: ***extends***.

class Corka extends Matka

{

//składniki klasy Corka

}

W wyniku dziedziczenia obiekty klasy ***Corka*** mają dostęp do składników klasy ***Matka***. Za wyjątkiem tych, które są składnikami prywatnymi.

Podsumowanie WSZYSTKIEGO:

<https://www.youtube.com/watch?v=Af8xRrWGYu0&list=PLE974A9BEF34A967A&index=18>