



**AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE**

Programowanie obiektowe w Perlu

Piotr Wydrych

<http://www.kt.agh.edu.pl/~wydrych/>

Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki
Katedra Telekomunikacji

30 kwietnia 2009



- Obiekt to struktura danych wraz z zestawem metod wykonujących operacje na danej strukturze.
- Klasa definiuje zestaw metod wykonujących operacje na obiektach danej klasy i klas pochodnych (dziedziczących).
- W Perlu obiektem jest referencja przypisana do klasy.
- Klasą jest pakiet.
- Metodą jest funkcja, oczekująca jako pierwszego argumentu:
 - nazwy pakietu, jeżeli jest to funkcja statyczna;
 - referencji przypisanej do danej klasy.



- W Perlu obiektem jest referencja przypisana do klasy.
- Przypisanie to tworzy funkcja `bless`.

bless

bless REF

bless REF, CLASSNAME

- Wersja jednoargumentowa przypisuje referencję do bieżącego pakietu.
- Wersja dwuargumentowa:
 - przypisuje referencję do pakietu podanego jako drugi argument;
 - pozwala na tworzenie dziedziczonych konstruktorów.



- W Perlu klasą jest pakiet.
- W każdym pakiecie tablica @ISA odpowiedzialna jest za dziedziczenie.

Przykład

```
package Prostokat;  
  
use Czworobok;  
our @ISA = qw(Czworobok);  
  
1;
```

- Jeżeli wywoływana funkcja nie istnieje w pakiecie, szukana jest w pakietach wymienionych w tablicy @ISA.
- Perl pozwala na dziedziczenie wielokrotne.



- W Perlu metodą jest funkcja, oczekująca jako pierwszego argumentu:
 - nazwy pakietu, jeżeli jest to funkcja statyczna;
 - referencji przypisanej do danej klasy.

Przykład

```
sub liczbaKatow() { my $class = shift; ... }  
  
sub pole() { my $self = shift; ... }
```



AGH

Metody – wykonywanie

- Istnieją dwa sposoby zapisu wykonania metody.
- Prostszy, częstszy i czytelniejszy — za pomocą operatora `->`.
- Kompilator Perla zadba o dodanie na pierwszym miejscu listy argumentów odpowiednio nazwy klasy lub referencji.

Przykład

```
print Czworobok->liczbaKatow;
```

```
print $mojKwadrat->pole;
```



- Pojęcie konstruktora jest w Perlu umowne.
- Konstruktor jest metodą statyczną tworzącą nową referencję i zwracającą ją po uprzednim przypisaniu do klasy.

Przykład

```
sub new {  
    my $class = shift;  
    my $self = {};  
    $self->{parametr} = shift;  
    bless $self, $class;  
    return $self;  
}
```



AGH

Destruktory

- Kiedy nie zostanie już żadna referencja wskazująca na obiekt, jest on niszczone.
- Przed zniszczeniem wykonywana jest funkcja DESTROY.

Przykład

```
sub DESTROY {  
    my $self = shift;  
    close $self->{filehandle};  
}
```

- Klasa `Czworobok` posiada metody `opisz`, `parametry` (opisującą słownie wartości parametrów) i `pole`. `opisz` korzysta z `parametry`.
- Klasa `Rownoleglobok` dziedziczy po klasie `Czworobok`. Posiada metody `new`, `parametry` i `pole`. Konstruktor przyjmuje trzy argumenty — podstawę, wysokość i kąt. Jeżeli trzeci argument nie jest podany, konstruktor zakłada wartość 90 dla kąta.

- Klasa `Prostokat` korzysta z dziedziczonego konstruktora klasy `Rownolegلوبok`. Posiada metodę `parametry`, która korzysta z metody `parametry` klasy nadrzędnej.
- Klasa `Kwadrat` posiada własny konstruktor korzystający z konstruktora klasy `Prostokat`. Posiada także metodę `parametry`.



AGH Przykład — Czworobok.pm

```
package Czworobok;

use strict;
use warnings;

sub opisz {
    my $self = shift;
    return sprintf 'Jestem_%s_o_parametrach:_%s',
        ref($self), $self->parametry;
}

sub parametry { my $self = shift; return 'n/n'; }

sub pole { my $self = shift; return undef; }

1;
```



AGH Przykład — Rownoleglobok.pm

```
package Rownoleglobok;

use strict;
use warnings;
use Czworobok;
our @ISA = qw(Czworobok);

sub new {
    my $class = shift;
    my $self = {};
    $self->{podstawa} = shift;
    $self->{wysokosc} = shift;
    $self->{kat} = shift;
    $self->{kat} = 90 if not defined $self->{kat};
    bless $self, $class;
    return $self;
}
```



AGH Przykład — Rownoleglobok.pm c.d.

```
sub parametry {  
    my $self = shift;  
    return sprintf 'podstawa:_%0.2f, _wysokosc:_%0.2f, _kat:_%d',  
        $self->{podstawa}, $self->{wysokosc}, $self->{kat};  
}  
  
sub pole {  
    my $self = shift;  
    return $self->{podstawa} * $self->{wysokosc};  
}  
  
1;
```



AGH Przykład — Prostokat.pm

```
package Prostokat;

use strict;
use warnings;
use Rownoleglobok;
our @ISA = qw(Rownoleglobok);

sub parametry {
    my $self = shift;
    return $self->SUPER::parametry . '_(oczywiscie)';
}

1;
```



AGH Przykład — Kwadrat.pm

```
package Kwadrat;

use strict;
use warnings;
use Prostokat;
our @ISA = qw(Prostokat);

sub new {
    my $class = shift; my $bok = shift;
    my $self = Prostokat->new($bok, $bok);
    bless $self, $class;
    return $self;
}

sub parametry {
    my $self = shift; return sprintf 'bok:_%0.2f', $self->{
        podstawa};
}

1;
```



AGH

Bibliografia



Autor nieznany
perlobj - Perl objects.

<http://perldoc.perl.org/perlobj.html>.