

Programowanie obiektowe

Wykład 15

Marcin Młotkowski

18 czerwca 2020

Plan wykładu

- 1 Simula 67
 - Koncepcje Tony'ego Hoare, 1966
 - SIMULA 67
- 2 Smalltalk
- 3 Języki prototypowe
- 4 Zakończenie

Plan wykładu

- 1 **Simula 67**
 - Koncepcje Tony'ego Hoare, 1966
 - SIMULA 67
- 2 Smalltalk
- 3 Języki prototypowe
- 4 Zakończenie

C. A. R. Hoare, 1966

Koncepcja obiektu

Wprowadzenie koncepcji obiektu (rzeczywistego)
modelowanego w systemach komputerowych jako rekord.

C. A. R. Hoare, 1966

Koncepcja obiektu

Wprowadzenie koncepcji obiektu (rzeczywistego) modelowanego w systemach komputerowych jako rekord.

Klasy

Klasy odzwierciedlają klasyfikację obiektów.

Inne koncepcje

Referencja

"Adres" rekordu w pamięci. Referencje mają określone typy, które są sprawdzane podczas kompilacji.

Inne koncepcje

Referencja

"Adres" rekordu w pamięci. Referencje mają określone typy, które są sprawdzane podczas kompilacji.

Tworzenie rekordów

Rekordy są tworzone dynamicznie i usuwane za pomocą odśmiecania.

Inne koncepcje

Referencja

"Adres" rekordu w pamięci. Referencje mają określone typy, które są sprawdzane podczas kompilacji.

Tworzenie rekordów

Rekordy są tworzone dynamicznie i usuwane za pomocą odśmiecania.

Podklasy

Deklarownie klas rekordów poprzez rozszerzanie zadeklarowanych wcześniej klas.

Inne pomysły Hoare

null

Wprowadzenie pojęcia pustej referencji.

Inne pomysły Hoare

null

Wprowadzenie pojęcia pustej referencji.

QCon, London, 2009

I call it my billion-dollar mistake. It was the invention of the null reference in 1965.

Powstanie języka SIMULA

Zespół

Kristen Nygaard i Ole-Johan Dahl (Univac, Norwegian Computing Center)

Główne cele projektu

Stworzenie języka symulującego dyskretne zjawiska.

Przykład deklaracji klasy

```
Class Figura(X, Y); Real X, Y;  
  Begin  
    Real Pole;  
    Procedure Drukuj  
    Begin  
      OutText ("figura");  
      OutFix (X, 3, 10); OutFix (Y, 3, 10);  
      OutImage;  
    End;  
    OutText ("Konstrukcja obiektu");  
  Figura;  
End;
```

Przykład deklaracji klasy

```
Class Figura(X, Y); Real X, Y;  
  Begin  
    Real Pole;  
    Procedure Drukuj  
    Begin  
      OutText ("figura");  
      OutFix (X, 3, 10); OutFix (Y, 3, 10);  
      OutImage;  
    End;  
    OutText ("Konstrukcja obiektu");  
  Figura;  
End;  
  
Ref (Figura) fig;  
fig :- New Figura(1.0, 1.0);
```

Podklasy i metody wirtualne

```
Class Figura;  
virtual: Real Procedure Pole;  
Begin  
    ...  
End;  
  
Figura Class Prostokąt(W, S); Real W, S;  
Begin  
    Real Procedure Pole;  
        Pole := W * S;  
    End  
End
```

Ukrywanie implementacji

Pola i metody mogą być oznaczone jako **Hidden** lub **Protected**.

Załączki refleksji

`xb is B`

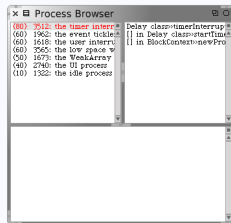
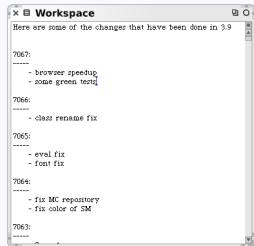
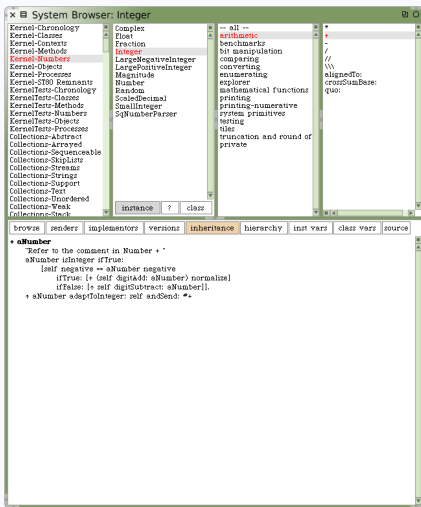
Obiekt jest klasy B.

`xb in B`

Obiekt jest klasy B lub podklasy B.

Plan wykładu

- 1 Simula 67
 - Koncepcje Tony'ego Hoare, 1966
 - SIMULA 67
- 2 Smalltalk
- 3 Języki prototypowe
- 4 Zakończenie



Cele projektu

Zaprojektowanie od podstaw czystego języka obiektowego:
wszystko ma być obiektem.

Kilka faktów

- Laboratorium Xerox Palo Alto Research Center
- okres powstania: lata 70;
- wersja referencyjna: Smalltalk'80;
- twórcy: Alan Kay, Dan Ingalls

Wszystko jest obiektem

- klasy są obiektami;
- liczby i wartości boolowskie są obiektami;
- tablice są obiektami;
- metody są obiektami;

Bardzo prosta składnia

3 * 5 -2

3 neg

tablica at: 3 put: 'trzy'

tablica at: 3

Tworzenie nowych obiektów

`r1 := Rectangle new.`

`r2 := Rectangle width: 3 height: 4.`

Środowisko programistyczne

Prawdziwe GUI!

- Przeglądarka klas;
- Uruchamianie programów;
- Zaimplementowane w Smalltalku (MVC).

Inne cechy

- implementacja maszyny wirtualnej;
- przenośność;
- wbudowany debugger;
- implementacja Smalltalka w Smalltalku;
- inspektor klas i obiektów.

Wpływ języka

- Większość współczesnych języków obiektowych, w tym: Objective-C
- popularny język w programowaniu zwinnym i szybkim prototypowaniu;

Objective-C

Objective-C: język C ze Smalltalkiem

```
[myColor changeColorToRed:5.0 green:2.0 blue:6.0];
```

Swift

Swift: Objective-C bez C

- wywołanie metod z kropką;
- wiele z programowania funkcyjnego (inferencja typów, typy opcjonalne).

Plan wykładu

- 1 Simula 67
 - Koncepcje Tony'ego Hoare, 1966
 - SIMULA 67
- 2 Smalltalk
- 3 Języki prototypowe
- 4 Zakończenie

Założenia

Brak klas

Obiekty są tworzone *ex nihilo* lub jako klony innych obiektów, a następnie modyfikowane.

Założenia

Brak klas

Obiekty są tworzone *ex nihilo* lub jako klony innych obiektów, a następnie modyfikowane.

Dziedziczenie

Klasa ma referencję do tzw. prototypu („nadklasy”) i może korzystać z jego metod.

Realizacje koncepcji

- Self
- JavaScript
- ActionScript

JavaScript

- Nie mylić z językiem JAVA!
- obecnie JavaScript to implementacja standardu ECMAScript

Tworzenie obiektów ex nihilo

Konstruktor

```
function ObiektOsoba(imie, nazwisko) {  
  this.imie = imie;  
  this.nazwisko = nazwisko;  
  function info() {  
    alert("imie " + this.imie);  
  }  
  this.info = info;  
}  
  
var informatyk = new ObiektOsoba("Alan", "Turing");
```

Tworzenie obiektów w locie

```
var informatyk = {  
  imie: 'Alan',  
  nazwisko: 'Turing',  
  info: function() { alert('info'); }  
}
```

Prototypowanie (dziedziczenie)

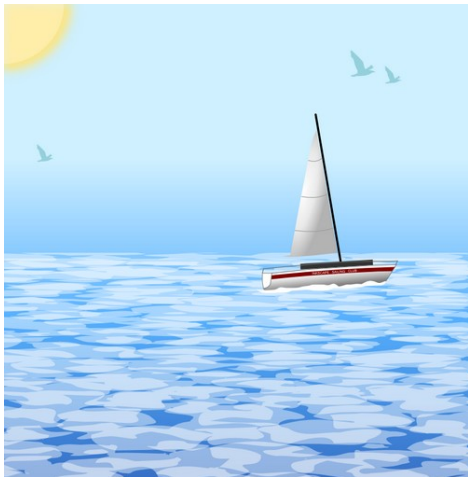
```
function Student(kierunek) {  
  this.info = function() {  
    alert("Kierunek " + this.kierunek);  
  }  
}
```

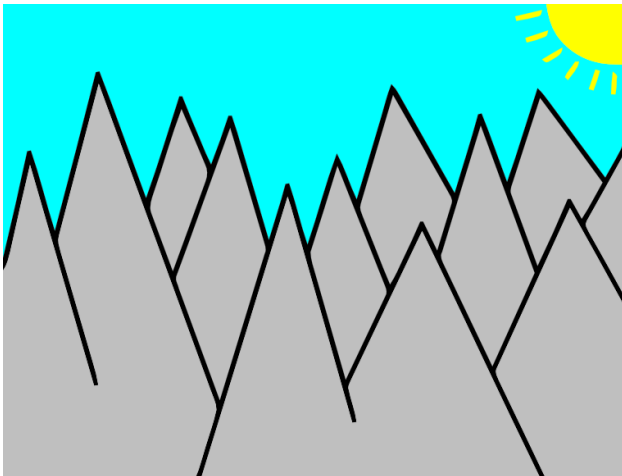
```
Student.prototype = Osoba;
```

Plan wykładu

- 1 Simula 67
 - Koncepcje Tony'ego Hoare, 1966
 - SIMULA 67
- 2 Smalltalk
- 3 Języki prototypowe
- 4 Zakończenie

PYTANIA?





KONIEC