Programowanie i projektowanie obiektowe

Wstęp

Paweł Daniluk

Wydział Fizyki

Jesień 2016

Plan wykładu

1. Wstęp – podstawowe pojęcia, modelowanie dziedziny
2. Projektowanie obiektowe
3. Obiekty i klasy w Pythonie
4. Typy sekwencyjne, iteratory
5. Wyjątki, własności, dekoratory
6. Zaawansowane aspekty obiektowości w Pythonie
7. Zastosowania
8. XML i przetwarzanie tekstów
9. SQLAlchemy
10. CherryPy
11. Jak działa Python?
12. Inne języki obiektowe

Programowanie "nieobiektowe"

Dostępne typy danych

- Typy proste – liczby, znaki, wartości logiczne, napisy
- Tablice – numerowane ciągi wartości innych typów, np. wektor liczb całkowitych.
- Struktury (rekordy) – Zestaw nazwanych pól określonych typów.

Programowanie "nieobiektowe"

Dostępne typy danych

- Typy proste – liczby, znaki, wartości logiczne, napisy
- Tablice – numerowane ciągi wartości innych typów, np. wektor liczb całkowitych.
- Struktury (rekordy) – Zestaw nazwanych pól określonych typów.

Struktury

```
o1.nazwisko="Kowalski"
o1.imie="Jan"
o1.adres="Banacha 5"

o2=o1

print o2.nazwisko
```

Osoba

- nazwisko
- imie
- adres

Programowanie "nieobiektowe" c.d.

Funkcje

Funkcja przyjmuje argumenty, oblicza i zwraca wynik. W programie może występować co najwyżej jedna funkcja o danej nazwie.

Problem

Mamy struktury opisujące figury geometryczne: `Kolo`, `Kwadrat`, `Prostokąt`, `Trójkąt`. Jak zrealizować funkcję obliczającą pole?

Programowanie "nieobiektowe" c.d.

Funkcje

Funkcja przyjmuje argumenty, oblicza i zwraca wynik. W programie może występować co najwyżej jedna funkcja o danej nazwie.

Problem

Mamy struktury opisujące figury geometryczne: `Kolo`, `Kwadrat`, `Prostokąt`, `Trójkąt`. Jak zrealizować funkcję obliczającą pole?

Wariant 1

- `poleKola(kolo)`
- `poleKwadratu(kwadrat)`
- `poleTrojkata(trojkat)`
- `...`

Programowanie "nieobiektowe" c.d.

Funkcje

Funkcja przyjmuje argumenty, oblicza i zwraca wynik. W programie może występować co najwyżej jedna funkcja o danej nazwie.

Problem

Mamy struktury opisujące figury geometryczne: `Kolo`, `Kwadrat`, `Prostokąt`, `Trójkąt`. Jak zrealizować funkcję obliczającą pole?

Wariant 2

```
def pole(figura):
    if(figura is Kolo):
        ...
    else if(figure is Kwadrat):
        ...
```

Programowanie obiektowe

Cel

- Typy pierwotne (i tablice) są niewygodne do opisywania rzeczywistości.
- Obiekty odpowiadają rzeczywistym bytom.
- Podobne obiekty grupowane są w klasy.
- Forma i materia – klasa i obiekty.

Historia

- Simula 67 – opracowana do modelowania statków (Norsk Regnesentral)
- Smalltalk – pierwszy nowoczesny język obiektowy (dziedziczenie) (Palo Alto Research Center)
- C++, Java, Objective-C, Python

Programowanie obiektowe c.d.

Najważniejsze cechy

- Obiekty oprócz atrybutów zawierają metody.
- Dziedziczenie
- Polimorfizm

Polimorfizm

Zbiór technik pozwalających stosować funkcje do różnych typów.

Statyczny vs. dynamiczny

Czy na etapie kompilacji można określić, która funkcja ma zostać wywołana?

Polimorfizm

Zbiór technik pozwalających stosować funkcje do różnych typów.

Statyczny vs. dynamiczny

Czy na etapie kompilacji można określić, która funkcja ma zostać wywołana?

Uniwersalny

Funkcja może działać na różnych typach danych. Np. iloczyn skalarny.

Polimorfizm

Zbiór technik pozwalających stosować funkcje do różnych typów.

Statyczny vs. dynamiczny

Czy na etapie kompilacji można określić, która funkcja ma zostać wywołana?

Uniwersalny

Funkcja może działać na różnych typach danych. Np. iloczyn skalarny.

Ad-hoc

Właściwa funkcja jest dobierana na podstawie podanych argumentów.

Polimorfizm

Zbiór technik pozwalających stosować funkcje do różnych typów.

Statyczny vs. dynamiczny

Czy na etapie kompilacji można określić, która funkcja ma zostać wywołana?

Uniwersalny

Funkcja może działać na różnych typach danych. Np. iloczyn skalarny.

Ad-hoc

Właściwa funkcja jest dobierana na podstawie podanych argumentów.

Rzutowania i konwersje typów.

Analiza, a projektowanie

Analiza

Badanie problemu i wymagań, ale nie rozwiązania. Analiza obiektowa (ang. object-oriented analysis) zajmuje się badaniem i klasyfikacją obiektów pojęciowych.

Obiekty pojęciowe nie mają nic wspólnego z programowaniem - reprezentują pojęcia i koncepcje ze świata rzeczywistego.

Przykład

W wypadku systemu informacyjnego dla Zakładu Transportu Miejskiego obiektami pojęciowymi mogą być: Autobus, Linia i Kierowca.

Analiza, a projektowanie c.d.

Projektowanie

Wymyślanie koncepcyjnego rozwiązania (programistycznego i sprzętowego), które realizuje wymagania.

Projektem może być opis schematu bazy danych lub klas programowych.

Podczas projektowania zazwyczaj pomija się niskopoziomowe lub oczywiste (dla zamierzonych odbiorców projektu) szczegóły i koncentruje się na wysokopoziomowych pomysłach oraz ideach.

Analiza i projektowanie dają się krótko streścić jako "zrób co należy (analiza) oraz zrób to jak należy (projektowanie)".

Proces wytwórczy

Podejście kaskadowe

1. Analiza
2. Projekt
3. Implementacja

Proces wytwórczy

Podejście kaskadowe

1. Analiza
2. Projekt
3. Implementacja

Tak się nie da.

Proces wytwórczy

Podejście kaskadowe

1. Analiza
2. Projekt
3. Implementacja

Tak się nie da.

W praktyce działa się przyrostowo przeplatając fazy.

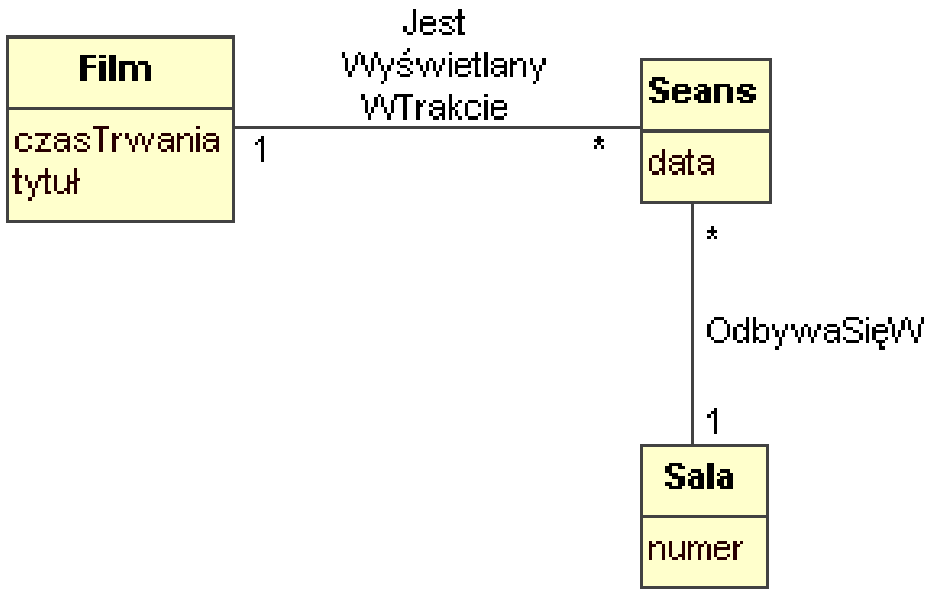
Obiektowe modelowanie dziedziny

Model dziedziny ma odzwierciedlać pojęcia z modelowanej części świata rzeczywistego oraz ich zależności.

W modelu dziedziny nie zajmujemy się klasami programowymi (ang. software class). Może on posłużyć jako źródło inspiracji przy ich projektowaniu, ale nie w drugą stronę.

- klasy pojęciowe
- powiązania między klasami pojęciowymi
- atrybuty klas pojęciowych

Przykład


Film
czasTrwania
tytuł
Jest
Wyświetlany
WTrakcie
1
*
Seans
data
*
OdbywwaSięW
1
Sala
numer

Odnajdywanie klas pojęciowych

Przy znajdowaniu klas pojęciowych należy:

- używać istniejących nazw
- nie zajmować się niczym, co nie dotyczy modelowanej części rzeczywistości
- nie dodawać rzeczy, których nie ma.

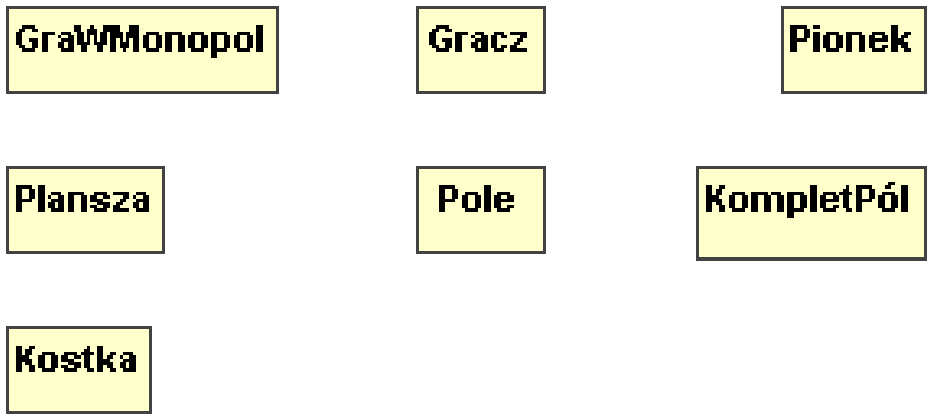
Analiza fraz rzeczownikowych

Wyszukiwanie fraz rzeczownikowych w tekstowym opisie dziedziny lub wymagań (jeżeli takimi dysponujemy).

Często spotykane kategorie klas pojęciowych

Kategoria	Przykłady	Klasy z dziedziny gry w Monopol
transakcje	SprzedażBiletu, RezerwacjaMiejsc	
pozycje transakcji	PozycjaRezerwacji	
produkty bądź usługi związane z transakcjami i kontraktami lub ich pozycjami	Bilet	
gdzie transakcje są odnotowywane	Kasa, WykazDostępnychMiejsc	
role ludzi i organizacji związanych z transakcją	Kasjer	Gracz
miejsce zajścia transakcji/obsługi transakcji	Kasa, SalaKinowa	
zdarzenia (często trzeba pamiętać czas ich zajścia)	Seans	GraWMonopol
obiekty fizyczne	Bilet, Kino, Kasa, Miejsce	Plansza, Pionek, Kostka
opisy	OpisSeansu	
katalogi	KatalogSeansów, KatalogFilmów	
kontenery rzeczy fizycznych lub informacji	Kino, SalaKinowa	Plansza
rzeczy w kontenerach	SalaKinowa, Miejsce	Pole
inne współpracujące systemy	SystemAutoryzującyPłatnościElektroniczne	
potwierdzenia, rejestry, kontrakty, zagadnienia prawne	Pokwitowanie, PotwierdzenieRezerwacji	
instrumenty finansowe	Czek, Gotówka	
harmonogramy, instrukcje, dokumenty regularnie używane podczas wykonywania prac	DziennaListaPromocji	

Klasy pojęciowe dla gry w Monopol



Odnajdywanie powiązań

Powiązanie (ang. association) między klasami wskazuje, że między ich egzemplarzami może występować jakaś zależność.

- szukamy powiązań, które są niezbędne do wypełnienia wymagań informacyjnych i pomagają zrozumieć dziedzinę
- warto pokazywać powiązania między klasami, jeżeli przez jakiś czas "trzeba pamiętać" o zależności między ich egzemplarzami

Często spotykane kategorie powiązań

Kategoria	Przykłady	Klasy z dziedziny gry w Monopol
A jest transakcją związaną z inną transakcją B	Płatność–RezerwacjaMiejsc	
A jest pozycją transakcji B	PozycjaRezerwacji–RezerwacjaMiejsc	
A jest produktem lub usługą z transakcji lub pozycji transakcji B	Bilet–SprzedażBiletu	
A jest rolą związaną z transakcją B	Klient–Płatność	
A jest fizyczną lub logiczną częścią B	Miejsce–SalaKinowa, SalaKinowa–Kino	Pole–Plansza, Pole–KompletPól, GraWMonopol–Plansza, GraWMonopol–Kostka, GraWMonopol–Pionek
A fizycznie lub logicznie przechowywane w/na B	Kasa–Kino	Pionek–Pole, Pole–Plansza
A jest opisem B	OpisSeansu–Seans	
A jest rejestrowane, zgłaszane, utrwalane, pamiętane w/na B	SprzedażBiletu–Kasa	Pionek–Pole, GraWMonopol–Plansza
A jest uczestnikiem/pracownikiem/członkiem B	Kasjer–Kino	Gracz–GraWMonopol
A jest organizacyjną podjednostką B	Kino–SiećKin	
A używa, zarządza lub posiada B	Kasjer–Kasa	Gracz–Pionek
A jest obok B	PozycjaRezerwacji–PozycjaRezerwacji	

Powiązania

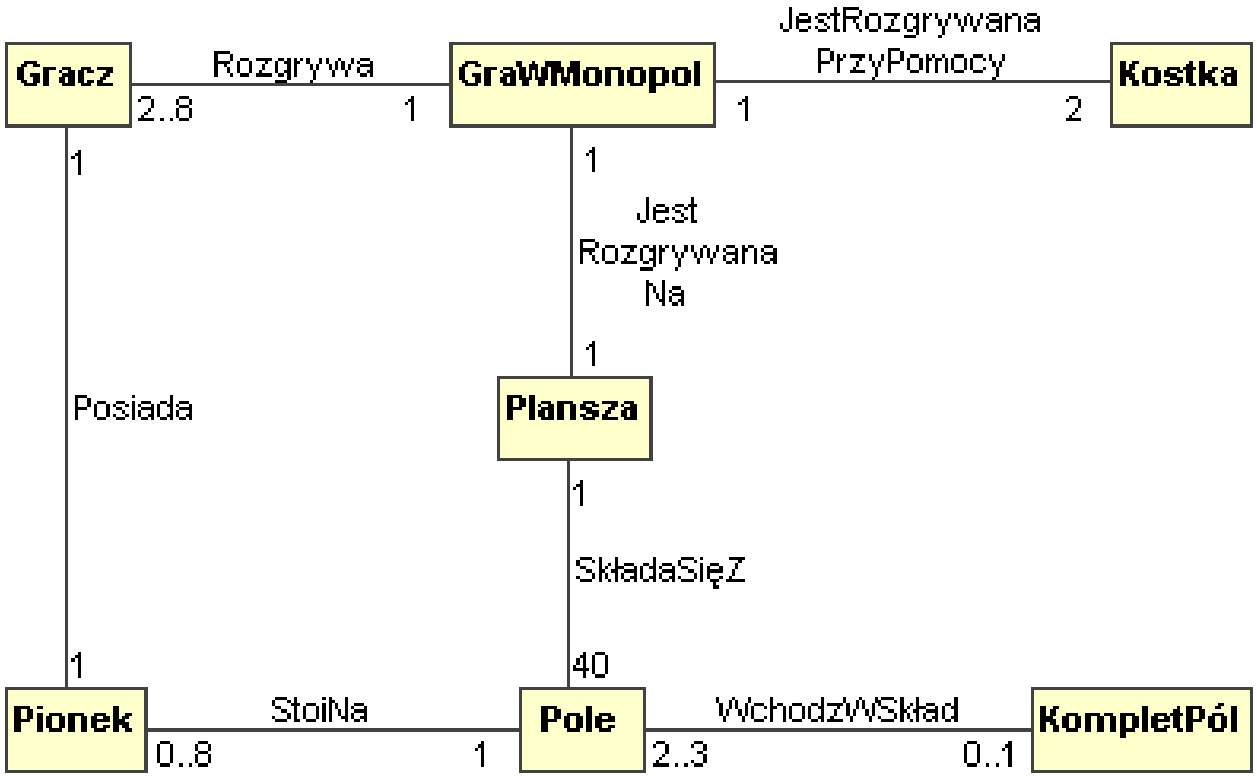
Nazywanie powiązań

Nie każda nazwa, która pasuje, pozwoli innym zorientować się, o co nam chodziło. Dla wielu powiązań będzie na przykład pasować nazwa Dotyczy, ale nie przenosi ona wielu informacji.

Liczebność

- 1 (dokładnie jeden)
- 11 (dokładnie jedenaście)
- 3, 5, 7 (trzy lub pięć lub siedem)
- 2..8 (od dwóch do ośmiu)
- 0..1 (zero lub jeden)
- 1..* (co najmniej jeden)
- * (dowolna ilość również zero)

Klasy pojęciowe i powiązania dla gry w Monopol



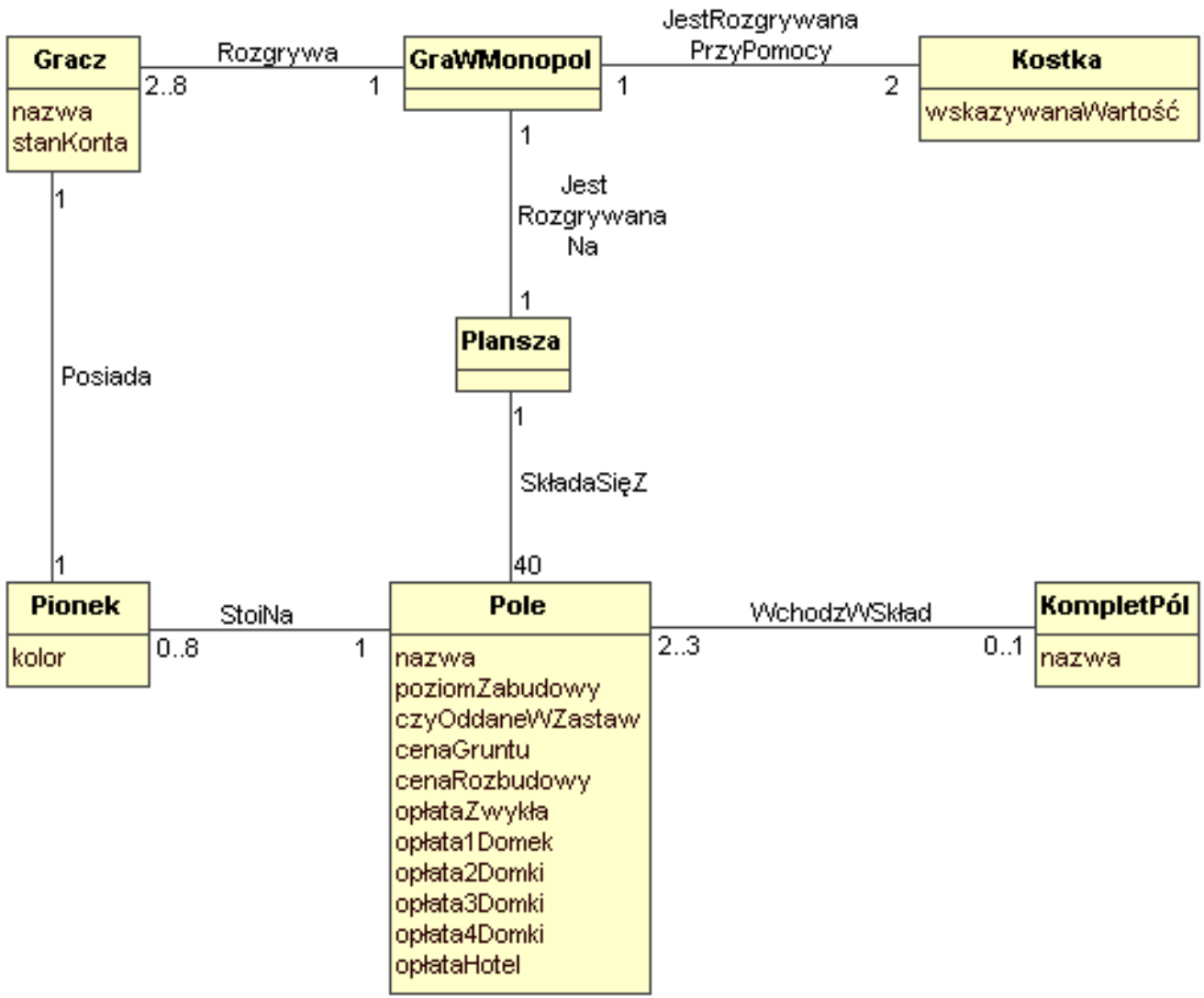
Dodawanie atrybutów

Wartości atrybutów opisują egzemplarze klas pojęciowych. Nie wszystkie klasy pojęciowe muszą mieć atrybuty.

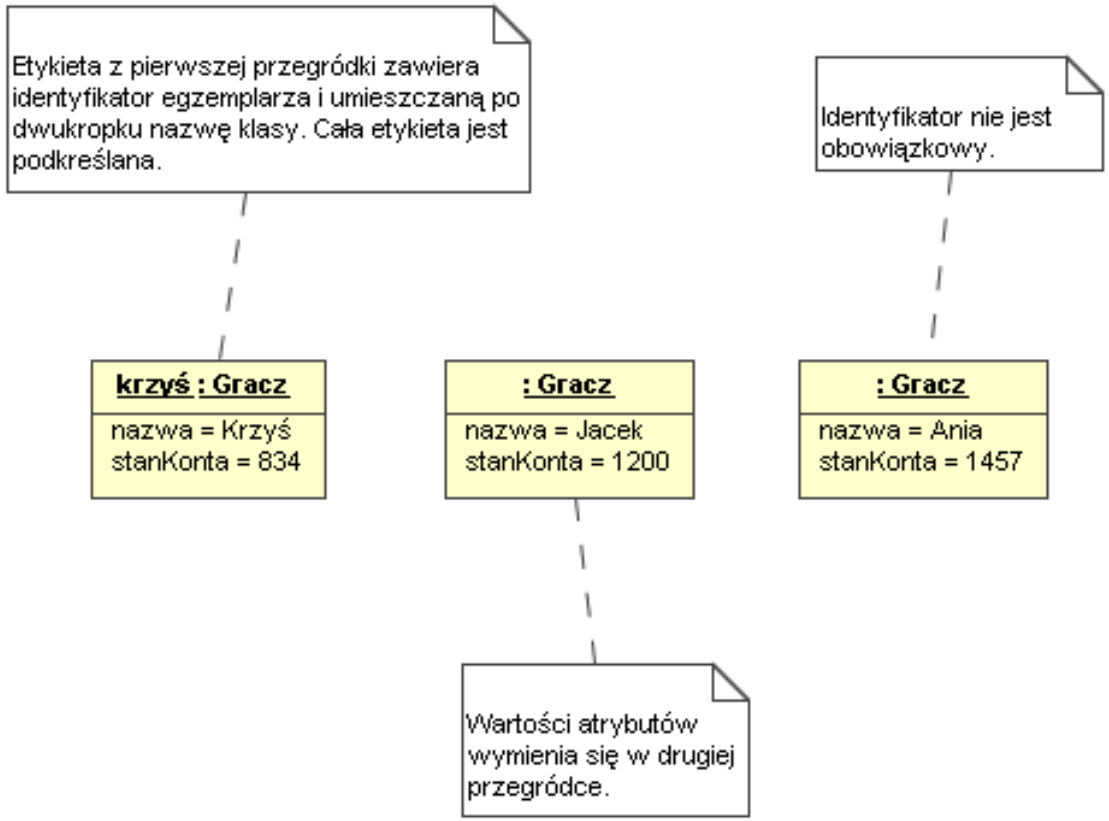
Atrybuty mają być przypisane do właściwych klas.

Atrybuty powinny być wartościami typów podstawowych.

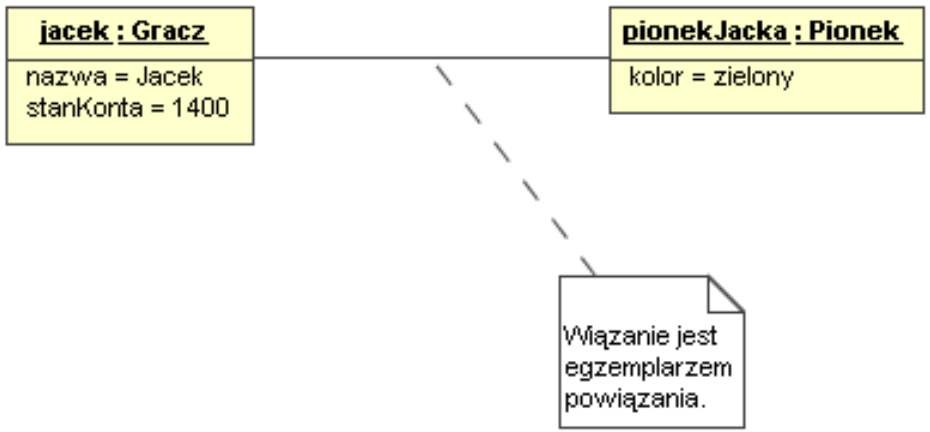
Częściowy model dziedziny dla gry w Monopol



Obiekty i ich stan



Wiązania


jacek : Gracz
nazwa = Jacek
stanKonta = 1400
pionekJacka : Pionek
kolor = zielony
Wiązanie jest egzemplarzem powiązania.