Въведение в обектно-ориентираното програмиране

Любомир Чорбаджиев
Технологическо училище "Електронни системи"
Технически университет, София
lchorbadjiev@elsys-bg.org
Revision : 1.8

4 октомври 2004 г.

Обектно-ориентирано програмиране

- Първият обектно-ориентиран език за програмиране е SIMULA I, разработен в периода 1962–1965 от Ole-Johan Dahl и Kristen Nygaard. Малко по-късно се появява и SIMULA 67 (1967).
- В езикът SIMULA 67 са въведени повечето от ключовите концепции на обектно-ориентираното програмиране — обекти, класове, наследяване, полиморфизъм.
- През 90 години на XX век обектно-ориентираното програмиране се превръща в доминираща софтуерна технология.

Обектно-ориентирана терминология

- **Обект** — обединение между данните и функциите, предназначени за обработването на тези данни.
- **Метод** — предоставя услуга, характерна за даден обект.
- **Съобщение** — заявка за изпълнение на даден метод.
- **Клас** — шаблон, по който се създават обекти (инстанции на даден клас).
- **Инстанция** — обект, който принадлежи на даден клас.

Обектно-ориентирана терминология

- **Капсулация** — скриване на информацията (данните), която се държи от обектите.
- **Наследяване** — механизъм, който позволява повторно използване на спецификациите на класовете.
- **Йерархия от класове** — дървовидна структура, която отразява наследствените взаимоотношения между класовете.
- **Полиморфизъм** — възможността на различни класове да отговарят на едни и същи съобщения по различен начин.

Обектно-ориентиран подход

- Програмната система е организирана около обекти.
- Обектите изпращат съобщения един на друг.
- Данните и функциите, които ги обработват, са обединени в обекти.
- Предметната област на програмната система се моделира в термините на обекти.

Обекти

- Обектите притежават състояние, поведение и идентичност. Структурата и поведението на подобни обекти се дефинира от техния общ клас. Термините обект и инстанция са взаимозаменяеми.
- Обектите представят реални или абстрактни неща.
- Имат добре дефинирани отговорности.

Обекти

- Проявяват добре дефинирано поведение.
- Имат добре дефиниран интерфейс, който е колкото се може по-прост.
- Обикновено не са много сложни или големи.
- Имат представа за малък брой от другите обекти.
- Трябва да се колкото се може по-малко обвързани с интерфейсите на другите обекти.

Обекти

- Обектът е инстанция на даден клас. Обектът притежава идентичност (уникален екземпляр).
- Класът дефинира както интерфейсът, така и реализацията на множество обекти. Класът определя поведението на всички негови обекти.
- Абстрактен клас се нарича клас, който няма инстанции.
- След като обектът се създаде, той не може да променя класът към който принадлежи.

Характеристики на обектите

- Притежават идентичност.
- Групират се в различни категории или класове.
- Имат добре дефинирано поведение и отговорности.
- Скриват вътрешната си структура.
- Имат състояние.
- Предоставят различни услуги.
- Изпращат съобщения на другите обекти.
- Получават съобщения от други обекти и реагират подходящо.
- Делегират отговорности на други обекти.

Класове

- Класът е множество от обекти, които имат общи атрибути и поведение.
- Класификацията на дадено множество от обекти зависи от гледната точка.
- "Клас" е общ термин, който се използва при различни видове класификация.
- "Група еднородни предмети, които по своите особености заемат определено място след други подобни предмети" [Български Тълковен Речник, Издателство на БАН, 1993г].

Класове

- Група, множество или вид, притежаващи общи атрибути.
- В контекста на обектно-ориентираното програмиране, класът е множество от обекти, които притежават обща структура и общо поведение.
- Класът представлява спецификация на структурата, поведението (методите) и наследствеността на обектите.

Капсулация

- Капсулация се нарича скриването на всички детайли на обекта, които нямат принос към неговите съществени характеристики.
- Капсулацията е принцип, който се използва при създаването на общата структура на програмата, съгласно който всеки компонент на програмата трябва да капсулира (или скрива) вътрешното си устройство.
- Интерфейсът на всеки модул трябва да се дефинира така, че да разкрива колкото се може по-малко от вътрешната структура на модула.

Капсулация

- Да се разкриват само детайлите, които са съществени за взаимодействието на обектите с външния свят. Ударението се поставя върху въпроса "Какво?" прави обекта, а не "Как?" го прави.
  - ◇ Скриват се детайлите от реализацията на поведението на класа.
  - ◇ Запазва обектите от външно вмешателство в тяхното вътрешно състояние.
  - ◇ Намалява зависимостите между различни части на програмата.
  - ◇ Обектите взаимодействат един с друг само посредством техните методи.

Наследяване

- Наследяването е начин да се опише как един клас се различава от друг клас.
- Наследяването предоставя естествен начин за класификация на видовете обекти и позволява сходството между обектите да бъде използвано в моделирането и конструирането на обектни системи.