Базовые понятия ООП (Объектно-ориентированное программирование)

Мальгина Н.Г., 2019

Введение

- В окончательном виде любая программа представляет собой набор инструкций процессора. Все, что написано на любом языке программирования – более удобная, упрощенная запись этого набора инструкций, облегчающая написание, отладку и последующую модификацию программы. Чем выше уровень языка, тем в более простой форме записываются одни и те же действия. С ростом объема программы становится невозможным удерживать в памяти все детали, и становится необходимым структурировать информацию, выделять главное и отбрасывать несущественное. Этот процесс называется повышением степени абстракции программы.

Введение понятия класса является естественным развитием идей модульности. В классе структуры данных и функции их обработки объединяются. Класс используется только через его интерфейс – детали реализации для пользователя класса не существенны. Идея классов отражает строение объектов реального мира – ведь каждый предмет или процесс обладает набором характеристик или отличительных черт, иными словами, свойствами и поведением. Класс является типом данных, определяемым пользователем. В классе задаются свойства и поведение какого-либо предмета или процесса в виде полей данных (аналогично структуре) и функций для работы с ними.

Определение

Объектно-ориентированное программирование – это методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования.

Объект – это нечто, имеющее четко определенные границы. Однако, этого недостаточно, чтобы отделить один объект от другого или дать оценку качества абстракции. Объект обладает состоянием, поведением и идентичностью; структура и поведение схожих объектов определяет общий для них класс; термины «экземпляр класса» и «объект» взаимозаменяемы.

Классы и *объекты*

Класс - это тип данных, а *объект* - экземпляр типа класс.

Например, кнопка вообще – это класс Button. А конкретная кнопка – это объект Button1 или Button1.

Текстовое поле вообще – это класс TextBox. А конкретное поле – это объект, например, TextBox1.

Класс — это просто набор методов, работающих с определёнными локальными данными.

Пять основных черт объектно-ориентированного языка

- **Все является объектом** (форма, текстовые поля, списки, меню, кнопки и т. д.). Объект как хранит информацию, так и способен ее преобразовывать.
- **Программа — совокупность объектов, указывающих друг другу что делать.**
- **Объект может включать другие объекты.**
- **У каждого объекта есть тип.** Тип называют классом.
- **Все объекты одного типа могут получать одинаковые сообщения.**

Свойства объекта

- Состояние объекта характеризуется перечнем всех свойств данного объекта и текущими значениями каждого из этих свойств.
- Индивидуальность объекта – это такое свойство объекта, которое отличает его от всех других объектов. В большинстве языков программирования при создании объект именуется.

Методы

- В C# метод служит в качестве аналога подпрограммы. (К числу других функций-членов относятся свойства, события и конструкторы.) Таким образом, методы класса содержат код, воздействующий на поля, определяемые этим классом.
- **Ме́тод** в объектно-ориентированном программировании — это функция или процедура, принадлежащая какому-то классу или объекту.
- Как и процедура в процедурном программировании, метод состоит из некоторого количества операторов для выполнения какого-то действия и имеет набор входных аргументов.

Основные принципы ООП

Объектно-ориентированное программирование строится на трех основополагающих принципах:

- Инкапсуляция,
- Полиморфизм
- Наследование

Инкапсуляция

- ***Инкапсуляция*** — это механизм программирования, объединяющий вместе код и данные, которыми он манипулирует, исключая как вмешательство извне, так и неправильное использование данных. В объектно-ориентированном языке данные и код могут быть объединены в совершенно автономный черный ящик. Внутри такого ящика находятся все необходимые данные и код. Когда код и данные связываются вместе подобным образом, создается объект. Иными словами, **объект** — это элемент, поддерживающий инкапсуляцию.

Наследование

- Наследование представляет собой способность производить новый класс из существующего базового класса. Производный класс – это новый класс, а базовый класс – существующий класс. Когда вы порождаете один класс из другого (базового класса), производный класс наследует элементы базового класса.

Пример

- Пусть у нас есть следующий класс Person, который описывает отдельного человека:

```
class Person
{
```

- Но вдруг нам потребовался класс, описывающий сотрудника предприятия - класс Employee. Поскольку этот класс будет реализовывать тот же функционал, что и класс Person, так как сотрудник - это также и человек, то было бы рационально сделать класс Employee производным (или наследником, или подклассом) от класса Person, который, в свою очередь, называется базовым классом или родителем (или суперклассом):

```
class Employee : Person
{
```

Полиморфизм

- В языках программирования **полиморфизмом** называется способность функции обрабатывать данные разных типов.

Итак,…

Класс – это программная единица, которая задает общий шаблон для конкретных объектов. Класс содержит все необходимые описания переменных, свойств и методов, которые относятся к объекту.

Объект – это экземпляр класса. Свойства объекта содержат конкретные данные, характерные для данного экземпляра.

Динамическое создание объектов

Чаще всего для размещения на форме кнопки, поля ввода или других управляющих элементов используется дизайнер среды Visual Studio: нужный элемент выделяется в панели элементов и размещается на форме. Однако иногда создавать элементы нужно уже в процессе выполнения программы. Поскольку каждый элемент управления представляет собой отдельный класс, его помещение на форму программным способом включает несколько шагов:

1. Создание экземпляра класса.
2. Привязка его к форме.
3. Настройка местоположения, размеров, текста и т. п.

На самом деле, каждый раз, когда на форму помещается новый элемент управления или вносятся какие-то изменения в свойства элементов управления, Visual Studio генерирует специальный служебный код, который проделывает приведенные выше операции по созданию и настройке элементов управления. Попробуйте поместить на форму кнопку, изменить у нее какие-нибудь свойства, а затем найдите в обозревателе решений ветку формы `Form1`, разверните ее и сделайте двойной щелчок по ветке `Form1.Designer.cs`. Откроется файл с текстом программы на языке C#, которую среда создала автоматически. Менять этот код вручную крайне не рекомендуется! Однако можно его изучить, чтобы понять принципы создания элементов управления в ходе выполнения программы.