# **Static Classes and Static Class Members**

Статичний клас в основному такий самий, як і нестатичний, але є одна відмінність: статичний клас не може бути екземпляром. Іншими словами, ви не можете використовувати оператор new для створення змінної типу класу. Оскільки немає змінної екземпляра, ви отримуєте доступ до членів статичного класу, використовуючи саме ім'я класу. Наприклад, якщо у вас є статичний клас з іменем UtilityClass, який має загальнодоступний статичний метод з іменем MethodA, ви викликаєте цей метод, як показано в наступному прикладі:

UtilityClass.MethodA();

Статичний клас можна використовувати як зручний контейнер для набору методів, які просто оперують вхідними параметрами і не повинні отримувати або встановлювати жодних внутрішніх полів екземпляра. Наприклад, у бібліотеці класів .NET статичний клас System.Math містить методи, які виконують математичні операції, без необхідності зберігати або отримувати дані, які є унікальними для конкретного екземпляру класу Math. Тобто, ви застосовуєте члени класу, вказуючи ім'я класу та ім'я методу, як показано в наступному прикладі.

double dub = -3.14;

Console.WriteLine(Math.Abs(dub));

Console.WriteLine(Math.Floor(dub));

Console.WriteLine(Math.Round(Math.Abs(dub)));

// Output:

// 3.14

// -4

// 3

Як і у випадку з усіма типами класів, інформація про тип статичного класу завантажується середовищем виконання .NET під час завантаження програми, яка посилається на клас. Програма не може точно вказати, коли саме завантажується клас. Однак, гарантовано, що він буде завантажений, його поля будуть ініціалізовані, а статичний конструктор викликаний до того, як на клас буде зроблено перше посилання у вашій програмі. Статичний конструктор викликається лише один раз, а статичний клас залишається в пам'яті протягом усього життя домену програми, в якому знаходиться ваша програма.

У наступному списку наведено основні характеристики статичного класу:

* Містить тільки статичні члени.
* Не може бути екземпляром.
* Є запечатаним.
* Не може містити конструкторів екземплярів.

Отже, створення статичного класу - це те саме, що створення класу, який містить лише статичні члени та закритий конструктор. Закритий конструктор запобігає створенню екземплярів класу. Перевагою використання статичного класу є те, що компілятор може перевірити, чи не було випадково додано жодного члена екземпляру. Компілятор гарантує, що екземпляри цього класу не можуть бути створені.

Статичні класи є закритими і тому не можуть бути успадковані. Вони не можуть бути успадковані від будь-якого класу або інтерфейсу, окрім Object. Статичні класи не можуть містити конструктор екземпляру. Однак, вони можуть містити статичний конструктор. Нестатичні класи також повинні визначати статичний конструктор, якщо клас містить статичні члени, які потребують нетривіальної ініціалізації.

**Статичні члени(Static Members)**

Нестатичний клас може містити статичні методи, поля, властивості або події. Статичний член класу можна викликати навіть тоді, коли не було створено жодного екземпляру класу. Доступ до статичного члена завжди здійснюється за іменем класу, а не за іменем екземпляру. Існує тільки один екземпляр статичного члена, незалежно від того, скільки екземплярів класу створено. Статичні методи та властивості не можуть отримати доступ до нестатичних полів та подій у своєму вміщуючому типі, а також не можуть отримати доступ до змінної екземпляру будь-якого об'єкту, якщо тільки вона не передана явно в параметрі методу.

Статичні методи можна перевантажувати, але не перевизначати, оскільки вони належать класу, а не будь-якому екземпляру класу. Два найпоширеніші способи використання статичних полів - це підрахунок кількості екземплярів об'єктів або зберігання значення, яке має бути спільним для всіх екземплярів.

Хоча поле не може бути оголошене як статичне **const**, поле const по суті є статичним за своєю поведінкою. Воно належить типу, а не екземплярам типу. Тому доступ до полів const можна отримати, використовуючи той самий запис **ClassName.MemberName**, який використовується для статичних полів. Екземпляр об'єкта не потрібен.

Ви оголошуєте статичні члени класу, використовуючи ключове слово static перед типом, що повертається, як показано у наступному прикладі:

public class Automobile

{

public static int NumberOfWheels = 4;

public static int SizeOfGasTank

{

get

{

return 15;

}

}

public static void Drive() { }

public static event EventType? RunOutOfGas;

// Other non-static fields and properties…

}

C# не підтримує статичні локальні змінні (тобто змінні, оголошені в області видимості методу).