**C# Interface**

**Інтерфейси**

Ще одним способом досягнення абстракції в C# є інтерфейси.

***Інтерфейс*** - це повністю "абстрактний клас", який може містити лише абстрактні методи та властивості (з порожніми тілами):

// interface

interface Animal

{

void animalSound(); // interface method (does not have a body)

void run(); // interface method (does not have a body)

}

Вважається гарною практикою починати інтерфейс з літери ***"I",*** оскільки це полегшує запам'ятовування того, що це інтерфейс, а не клас, як для вас, так і для інших.

За замовчуванням члени інтерфейсу є абстрактними та загальнодоступними.

*Примітка: Інтерфейси можуть містити властивості та методи, але не поля.*

Щоб отримати доступ до методів інтерфейсу, інтерфейс повинен бути "реалізований" (успадкований) іншим класом. Щоб реалізувати інтерфейс, використовуйте символ ***:*** (так само, як і при успадкуванні). Тіло методу інтерфейсу надається класом, що його реалізує.

Зверніть увагу, що вам не потрібно використовувати ключове слово ***override*** при реалізації інтерфейсу:

// Interface

interface IAnimal

{

void animalSound(); // interface method (does not have a body)

}

// Pig "implements" the IAnimal interface

class Pig : IAnimal

{

public void animalSound()

{

// The body of animalSound() is provided here

Console.WriteLine("The pig says: wee wee");

}

}

class Program

{

static void Main(string[] args)

{

Pig myPig = new Pig(); // Create a Pig object

myPig.animalSound();

}

}

**Зауваження про інтерфейси:**

* Як і абстрактні класи, інтерфейси не можна використовувати для створення об'єктів (у наведеному вище прикладі неможливо створити об'єкт "***IAnimal***" у класі Program) методи інтерфейсу не мають тіла - тіло надається класом-реалізатором
* При реалізації інтерфейсу потрібно перевизначити усі його методи
* Інтерфейси можуть містити властивості та методи, але не поля/змінні
* Члени інтерфейсу за замовчуванням є абстрактними та загальнодоступними
* Інтерфейс не може містити конструктор (оскільки з його допомогою не можна створювати об'єкти)

**Навіщо і коли використовувати інтерфейси?**

1) Для досягнення безпеки - приховати певні деталі і показати лише важливі деталі об'єкта (інтерфейсу).

2) C# не підтримує "множинне успадкування" (клас може успадковувати тільки від одного базового класу). Однак це можна зробити за допомогою інтерфейсів, оскільки клас може реалізовувати декілька інтерфейсів.