

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ “КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота № 2**

з дисципліни “Програмування”

тема “C# .Net. Розширені можливості реалізації ООП у мові C#. Події.”

Виконав студент

II курсу групи КП-02 Щербатюк Ярослав Сергійович

Перевірив “ ” “ ” 20 р

викладач Заболотня Тетяна Миколаївна

Київ 2021

# Мета роботи

Ознайомитися з такими можливостями мови програмування C# як абстрактні класи, інтерфейси, делегати. Вивчити механізми оброблення подій у C#, а також можливості, які мають методи-розширення.

# Постановка задачі

Для ієрархії класів, побудованої в лабораторній роботі №1, реалізувати:

1. Множину інтерфейсів. При чому один з класів повинен реалізовувати щонайменше 2 інтерфейси. Також продемонструвати реалізацію explicit implementation інтерфейса, обгрунтувати її використання **(1 бал)**.
2. Абстрактний клас. Забезпечити його наслідування. Наявність в цьому класі абстрактних методів - обов’язкова **(1 бал)**.
3. Механізм «делегат – подія – обробник події» **(2 бали)**.
4. Перетворити код, який забезпечує роботу з подіями та обробниками подій, на код, що використовує (\*) **(2 бали)**:
   1. анонімні методи;
   2. lambda-вирази;
   3. типи Action та Func (кожен з них).

(\*) - допускається реалізація коду однієї події різними способами, необов’язково різних подій.

5. Механізм створення та оброблення власних помилок **(2 бали)**:

* 1. створити новий клас виключної ситуації;
  2. створити новий клас аргументів для передачі їх до обробника виключної ситуації;
  3. забезпечити ініціювання створеної виключної ситуації та продемонструвати, як працює обробник даної помилки;
  4. реалізувати різні сценарії оброблення помилки.

6. Метод-розширення будь-якого класу **(1 бал)**.

Протокол має містити: титульний аркуш, постановку задачі, UML діаграму класів, фрагменти коду, які демонструють виконання поставлених задач, висновки.

|  |
| --- |
|  |
| Рис. UML діаграма класів |

**Фрагменти коду**

|  |
| --- |
| 1. Множина інтерфейсів для класів першої лабораторної |
| public interface IFarmPet  {      void WaterThePet(FeedingOperator oper, FeedingEventArgs fArgs);  }  public interface IHomePet  {      void FeedThePet(FeedingOperator oper, FeedingEventArgs fArgs);  } |
| public interface IHorse  {      void StrokeTheHorse();  }  public interface IDog  {    void GiveTheCommandToGiveAPaw (int pawNumber);  } |
| class Dog : HomePet, IHomePet, IDog  {  …  } |
| void IHomePet.FeedThePet(FeedingOperator oper, FeedingEventArgs fArgs)  {  …  } |

|  |
| --- |
| 2. Абстрактний клас з абстрактними методами, його наслідування |
| abstract class Pet : IDisposable  {  public abstract void Jump(int times);  } |
| abstract class FarmPet : Pet, IFarmPet  {  …  } |

|  |
| --- |
| 3. Механізм “делегат - подія - обробник подій” |
| public delegate void Order(FeedingOperator oper, FeedingEventArgs args); |
| public class FeedingEventArgs : EventArgs  {  …  } |
| public class PetRestaurant  {      public PetRestaurant(FeedingOperator oper)      {          IFarmPet[] farmPets = new IFarmPet[2];          farmPets[0] = Horse.GetPet(120);          farmPets[1] = Horse.GetPet("Yaroslav Scherbatyuk", "OO0000OO", 10000, 30000, 200);          IHomePet[] iceCars = new IHomePet[2];          iceCars[0] = Dog.GetPet(120);          iceCars[1] = Dog.GetPet("Not Yaroslav Scherbatyuk", "AA1234AA", 300, 700, 120);          for (int i = 0; i < 2; i++)          {              oper.FeedingEvent += new Order(farmPets[i].WaterThePet);              oper.FeedingEvent += new Order(iceCars[i].FeedThePet);          }      }  } |
| public class FeedingOperator  {     public event Order FeedingEvent;     public FeedingEventArgs fArgs;     …     public void FeedUp()     {        double foodAmount, waterAmount;         FeedingEventArgs feedArgs = null;         …             if (FeedingEvent != null)             {                 FeedingEvent((FeedingOperator)this, feedArgs);             }         …     }  } |

|  |
| --- |
| 4. Анонімні методи, lambda - вирази, Action та Func |
| Order currentFeedInfo = delegate(FeedingOperator oper, FeedingEventArgs fArgs)  {      WriteLine($"Current feeding operator: {oper.Fullname}; Current water amount: {fArgs.WaterAmount} milliliters; Current food amount: {fArgs.FoodAmount} grams");  }; |
| Action<FeedingOperator> newClientReact = oper => WriteLine($"{oper.Fullname} see new client pet, it's Home Pet");  Func<double, double, double> calcPercent = (currValue, maxValue) => currValue / maxValue \* 100; |

|  |
| --- |
| 5. Новий клас виключної ситуації, новий клас аргументів для передачі їх до  обробника помилок, обробка помилки |
| public class FeedArgsException : Exception  {      private FeedingEventArgs args;      public FeedArgsException(FeedingEventArgs args)      {          this.args = args;      }      public override string Message => $"Incorrectly entered data! Only positive arguments allowed!";  } |
| public class FeedingEventArgs : EventArgs  {      private double foodAmount;      private double waterAmount;  } |
| try  {      Write("Enter amount of food: ");      foodAmount = Double.Parse(ReadLine());      Write("Enter amount of water: ");      waterAmount = Double.Parse(ReadLine());      feedArgs = new FeedingEventArgs(foodAmount, waterAmount);  }  catch (Exception ex)  {      ForegroundColor = ConsoleColor.Red;      WriteLine(ex.Message);      WriteLine("Set to default values:\n Amount of water: 1 milliliter\n Amount of food: 1 grams");      ResetColor();      feedArgs = new FeedingEventArgs();  }  finally  {      …  } |

|  |
| --- |
| 6. Метод - розширення будь-якого класу |
| static class DogExtension  {      public static void GiveCookie(this Dog dog)      {          WriteLine("Dog ate cookie you gave and now it is happy!");      }  } |

|  |
| --- |
| Результати |
| Enter amount of food: 140  Enter amount of water: 2000  Operator Elon Mask is responsible for feeding pet.  Elon Mask see new client pet, it's Farm Pet  Pet of noOwner serviced by Elon Mask. Current lvl of water of pet:  100 %  Elon Mask see new client pet, it's Home Pet  Pet of noOwner serviced by Elon Mask. Current lvl of satiety of pet: 28,00 %  Elon Mask see new client pet, it's Farm Pet  Pet of Yaroslav Scherbatyuk serviced by Elon Mask. Current lvl of water of pet: 20,00 %  Elon Mask see new client pet, it's Home Pet  Pet of Not Yaroslav Scherbatyuk serviced by Elon Mask. Current lvl  of satiety of pet: 46,67 %  Current feeding operator: Elon Mask; Current water amount: 2000 milliliters; Current food amount: 140 grams |

**Висновки**

В результаті виконання лабораторної роботи відбулось ознайомлення з такими можливостями мови С#: абстрактні класи, інтерфейси, делегати. Також зрозумів механізми оброблення подій у С# і можливості, що надають методи-розширення.