

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 №** 3

з дисципліни “Об'єктно-орієнтоване програмування та шаблони проєктування”

тема “Структурні шаблони проектування”

|  |  |  |
| --- | --- | --- |
| Виконав  студент II курсу  групи КП-02  Щербатюк Ярослав Сергійович  (*прізвище, ім’я, по батькові*)  варіант №21 |  | Перевірив  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладач  Заболотня Тетяна Миколаївна  (*прізвище, ім’я, по батькові*) |

Київ 2020

**Мета роботи**

Навчитися на практиці визначати, який структурний шаблон проектування слід використовувати в залежності від умов задачі.

**Постановка завдання**

1. Реалізувати виведення системних повідомлень програми (попередження про помилки, інформаційні вікна, діалогові вікна тощо) мовою, зрозумілою користувачу, який працює з даною системою. Мова є параметром профілю користувача та встановлюється для кожного користувача окремо. Нехай українська мова буде мовою, що встановлюється за замовчанням.
2. За допомогою шаблону проєктування забезпечити виведення (і тільки виведення) на шахову дошку фігур в їх початковому розташуванні.

**UML-діаграми до задач**

|  |
| --- |
| **Задача 1** |
| **C:\Users\Yarik Scherbatyuk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\UMLDiagram1.png** |

|  |
| --- |
| **Задача 2** |
|  |

**Тексти коду програми**

|  |
| --- |
| **ProgramUI.cs (1 задача)** |
| using System;  using System.Collections.Generic;  public abstract class ProgramUI  {      protected UILanguageInterface languageInterface;      public UILanguageInterface LanguageInterface      {          set { this.languageInterface = value; }      }      public abstract void Notificate();      public abstract void ShowInfo();      public abstract void ShowDialog();  }  public class MyProgramUI : ProgramUI  {      public override void Notificate()      {          this.languageInterface.Notificate();      }      public override void ShowDialog()      {          this.languageInterface.ShowDialog();      }      public override void ShowInfo()      {          this.languageInterface.ShowInfo();      }      public MyProgramUI()      {          this.languageInterface = new UkrainianLanguageInterface();      }      public MyProgramUI(UILanguageInterface languageInterface)      {          this.languageInterface = languageInterface;      }  }  public abstract class UILanguageInterface  {      public abstract void Notificate();      public abstract void ShowInfo();      public abstract void ShowDialog();  }  public class EnglishLanguageInterface : UILanguageInterface  {      public override void Notificate()      {          Console.WriteLine("Notification");      }      public override void ShowDialog()      {          Console.WriteLine("Dialog window");      }      public override void ShowInfo()      {          Console.WriteLine("Window with information");      }  }  public class RussianLanguageInterface : UILanguageInterface  {      public override void Notificate()      {          Console.WriteLine("Уведомление");      }      public override void ShowDialog()      {          Console.WriteLine("Диалоговое окно");      }      public override void ShowInfo()      {          Console.WriteLine("Окно с информацией");      }  }  public class UkrainianLanguageInterface : UILanguageInterface  {      public override void Notificate()      {          Console.WriteLine("Сповіщення");      }      public override void ShowDialog()      {          Console.WriteLine("Діалогове вікно");      }      public override void ShowInfo()      {          Console.WriteLine("Вікно з інформацією");      }  }  public class LanguageGenerator  {      private Dictionary<string, UILanguageInterface> langDict;      public LanguageGenerator()      {          this.langDict = new Dictionary<string, UILanguageInterface>();          this.langDict.Add("eng", new EnglishLanguageInterface());          this.langDict.Add("rus", new RussianLanguageInterface());          this.langDict.Add("ukr", new UkrainianLanguageInterface());      }      public UILanguageInterface GetLanguageInterface(string langName)      {          UILanguageInterface lang ;          try          {              lang = this.langDict[langName];          }          catch(Exception)          {              throw new Exception($"Language '{langName}' do not supports");          }          return lang;      }  } |

|  |
| --- |
| **Flyweight.cs (2 задача)** |
| using System.Collections.Generic;  public abstract class Figure  {      protected char figureSymbol;      protected string role;      public string Role      {          get { return this.role; }      }      public char Symbol      {          get { return this.figureSymbol; }      }  }  public class Pawn : Figure  {      public Pawn()      {          this.figureSymbol = 'P';          this.role = "Pawn";      }  }  public class Rook : Figure  {      public Rook()      {          this.figureSymbol = 'R';          this.role = "Rook";      }  }  public class Knight : Figure  {      public Knight()      {          this.figureSymbol = 'N';          this.role = "Knight";      }  }  public class Bishop : Figure  {      public Bishop()      {          this.figureSymbol = 'B';          this.role = "Bishop";      }  }  public class Queen : Figure  {      public Queen()      {          this.figureSymbol = 'Q';          this.role = "Queen";      }  }  public class King : Figure  {      public King()      {          this.figureSymbol = 'K';          this.role = "King";      }  }  public class Empty : Figure  {      public Empty()      {          this.figureSymbol = ' ';          this.role = "Empty";      }  }  public class FigureFactory  {      private Dictionary<char, Figure> figures;      public FigureFactory()      {          this.figures = new Dictionary<char, Figure>();          this.figures.Add('P', new Pawn());          this.figures.Add('K', new King());          this.figures.Add('Q', new Queen());          this.figures.Add('B', new Bishop());          this.figures.Add('N', new Knight());          this.figures.Add('R', new Rook());          this.figures.Add('E', new Empty());      }      public Figure GetFigure(char c)      {          Figure f = this.figures[c];          return f;      }  } |

|  |
| --- |
| **ChessDesk.cs (2 задача)** |
| using System;  public class ChessDesk  {      private Figure[,] desk = new Figure[8, 8];      public ChessDesk(FigureFactory factory)      {          for(int i = 0; i < this.desk.GetLength(0); i++)          {              for(int j = 0; j < this.desk.GetLength(1); j++)              {                  if(i == 1 || i == 6)                  {                      this.desk[i, j] = factory.GetFigure('P');                      continue;                  }                  else if(i == 0 || i == 7)                  {                      if(j == 0 || j == 7)                      {                          this.desk[i, j] = factory.GetFigure('R');                          continue;                      }                      if(j == 1 || j == 6)                      {                          this.desk[i, j] = factory.GetFigure('N');                          continue;                      }                      if(j == 2 || j == 5)                      {                          this.desk[i, j] = factory.GetFigure('B');                          continue;                      }                      if(j == 3)                      {                          this.desk[i, j] = factory.GetFigure('Q');                          continue;                      }                      if(j == 4)                      {                          this.desk[i, j] = factory.GetFigure('K');                          continue;                      }                  }                  this.desk[i, j] = factory.GetFigure('E');              }          }      }      public void PrintDesk()      {          for(int i = 0; i < this.desk.GetLength(0); i++)          {              for(int j = 0; j < this.desk.GetLength(1); j++)              {                  Console.Write($"{this.desk[i, j].Symbol} ");              }              Console.WriteLine();          }      }  } |

**Приклади роботи програм**

|  |
| --- |
| **Приклади роботи першої програми** |
| PS C:\Users\Yarik Scherbatyuk\Documents\OOP\_2course\_scherbatyukk\lab3\src\1> dotnet run  Enter command: sfsfsdfsfs  The given key 'sfsfsdfsfs' was not present in the dictionary.  Enter command: notif  Сповіщення  Enter command: changeLang eng  Enter command: notify  The given key 'notify' was not present in the dictionary.  Enter command: notif  Notification  Enter command: dialog  Dialog window  Enter command: changeLang rus  Enter command: info  Окно с информацией  Enter command: dialog  Диалоговое окно  Enter command: exit |

|  |
| --- |
| **Виведена в консоль шахова дошка з другої задачі** |
| PS C:\Users\Yarik Scherbatyuk\Documents\OOP\_2course\_scherbatyukk\lab3\src\2> dotnet run  R N B Q K B N R  P P P P P P P P  P P P P P P P P  R N B Q K B N R |

**Обґрунтування використання шаблонів**

1. Вибір шаблону Міст обумовлюється тим, що потрібно розділити виклик вікон та спосіб отримання повідомлень певною мовою. Вибір мови, якою можуть виводитись текст на вікні, може змінюватись під час виконання програми. При зміні мови виведення тексту на вікнах програми код не повинен перекомпільовуватись.

2. Вибір шаблону Пристосуванець обумовлюється тим, що замість створення великої кількості об’єктів фігур, які будуть повторюватись, на шаховій дошці, ми тільки створюємо групу унікальних фігур. Оскільки за умовою задачі потрібно реалізувати тільки виведення фігур на шаховій на початку гри, тоді ідентичність кожного об’єкту не має значення.

**Висновки**

В ході лабораторної роботи я покращив знання про структурні шаблони проектування. Навчився використовувати шаблони проектування в залежності від умов задачі.