

**Міністерство освіти і науки України  
Національний технічний університет України  
«Київський політехнічний інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки  
Кафедра обчислювальної техніки**

**Лабораторна робота №1**  
з дисципліни  
«Об'єктно орієнтоване програмування»

Виконав:  
Студент групи ІМ-21  
Сірик Максим Олександрович  
номер у списку групи: 22

Перевірив:  
Порєв Віктор Миколайович

Київ 2023

# Зміст

<b>1</b>	<b>Мета:</b>	<b>2</b>
<b>2</b>	<b>Завдання:</b>	<b>2</b>
2.1	Варіант 2: . . . . .	2
2.2	Варіант 3: . . . . .	2
<b>3</b>	<b>Текст програми:</b>	<b>2</b>
3.1	Package: com.github.erotourtes.app . . . . .	2
3.2	Package: com.github.erotourtes.view . . . . .	2
3.3	Package: com.github.erotourtes.task1 . . . . .	3
3.4	Package: com.github.erotourtes.task2 . . . . .	4
<b>4</b>	<b>Ілюстрації:</b>	<b>6</b>
4.1	Task 1 . . . . .	6
4.2	Task 2 . . . . .	6
4.3	Діаграма класів . . . . .	7
<b>5</b>	<b>Висновки:</b>	<b>7</b>

## 1 Мета:

Отримати перші навички створення програм для Windows на основі проєктів для Visual C++ з використанням Windows API і навчитися модульному програмуванню на C++

## 2 Завдання:

1. Створити у середовищі MS Visual Studio C++ проєкт Win32 з ім'ям Lab1.
2. Написати вихідний текст програми згідно варіантів завдання.
3. Скомпілювати вихідний текст і отримати виконуваний файл програми.
4. Перевірити роботу програми

$$B_1 = 22 \bmod 4 = 2 \quad (1)$$

$$B_2 = (22 + 1) \bmod 4 = 3 \quad (2)$$

### 2.1 Варіант 2:

Два вікна діалогу. Спочатку з'являється перше, яке має дві кнопки: [Далі >] і [Відміна]. Якщо натиснути кнопку [Далі >], то воно закриється і з'явиться друге дліг вікно, яке має кнопки: [< Назад], [Так] і [Відміна]. Якщо натиснути кнопку [<Назад], вікно закриється і перехід до першого вікна.

### 2.2 Варіант 3:

Вікно діалогу з елементом списку (List Box) та двома кнопками: [Так] і [Відміна]. У список автоматично записуються назви груп нашого факультету. Якщо вибрати потрібний рядок списку і натиснути [Так], то у головному вікні повинен відображатися текст вибраного рядка списку.

## 3 Текст програми:

### 3.1 Package: com.github.erotourtes.app

Лістинг 1: MyApp.kt

```
package com.github.erotourtes.app

import com.github.erotourtes.view.MainView
import javafx.stage.Stage
import tornadofx.App

class MyApp: App(MainView::class) {
    override fun start(stage: Stage) {
        with(stage) {
            width = 600.0
            height = 400.0
        }
        super.start(stage)
    }
}
```

### 3.2 Package: com.github.erotourtes.view

Лістинг 2: MainView.kt

```

package com.github.erotourtes.view

import com.github.erotourtes.task1.TaskView
import javafx.scene.layout.HBox
import tornadofx.*

class MenuBar : View("Menu") {
    val changeView: (Int) -> Unit

    init {
        val changeView = params["changeView"] as? (Int) -> Unit
            ?: throw IllegalArgumentException("changeView_is_null")
        this.changeView = changeView
    }

    override val root = menubar {
        menu("Tasks") {
            item("Do_task_1", "Shortcut+1").action {
                changeView(1)
            }
            item("Do_task_2", "Shortcut+2").action {
                changeView(2)
            }
        }
    }
}

class MainView : View("Lab_1") {
    private val firstView = TaskView()
    private val centerView = HBox()

    private val changeView = { i: Int ->
        when (i) {
            1 -> centerView.replaceChildren(TaskView().root)
            2 -> centerView.replaceChildren(com.github.erotourtes.task2.TaskView().root)
            else -> throw IllegalArgumentException("Unknown_task_number")
        }
    }

    private val menuBar: MenuBar by inject(params = mapOf("changeView" to changeView))

    override val root = borderpane {
        top = menuBar.root
        center = centerView.apply { add(firstView.root) }
    }
}

```

### 3.3 Package: com.github.erotourtes.task1

Лістинг 3: TaskView.kt

```

package com.github.erotourtes.task1

import javafx.stage.StageStyle

```

```

import tornadofx.*

class TaskView : View("Task_1") {
    override val root = vbox {
        label { text = "Doing_task_1,_@2" }

        button("Start") {
            action {
                find<Window1>().openModal(stageStyle = StageStyle.UTILITY)
            }
        }
    }
}

```

ЛІСТИНГ 4: Windows.kt

```

package com.github.erotourtes.task1

import javafx.stage.StageStyle
import tornadofx.Fragment
import tornadofx.action
import tornadofx.button
import tornadofx.hbox

class Window1 : Fragment("Window_1") {
    override val root = hbox {
        button("Next") {
            action {
                find<Window2>().openModal(stageStyle = StageStyle.UTILITY)
                close()
            }
        }
        button("Cancel") { action (:: close) }
    }
}

class Window2 : Fragment("Window_2") {
    override val root = hbox {
        button("Previous") {
            action {
                find<Window1>().openModal(stageStyle = StageStyle.UTILITY)
                close()
            }
        }
        button("Yes") { action (:: close) }
        button("Cancel") { action (:: close) }
    }
}

```

### 3.4 Package: com.github.erotourtes.task2

ЛІСТИНГ 5: TaskView.kt

```

package com.github.erotourtes.task2

import javafx.beans.property.SimpleStringProperty

```

```

import javafx.stage.StageStyle
import tornadofx.*

class TaskView : View("Task_2") {
    private val message = SimpleStringProperty("")

    override val root = vbox {
        label { text = "Doing_task_2,_@3" }

        button("Start") {
            action {
                find<Window>(mapOf(Window::message to message)).openModal(stageStyle = StageStyle.DECORATION_NONE)
            }
        }

        label(message)
    }
}

```

ЛІСТИНГ 6: Window.kt

```

package com.github.erotourtes.task2

```

```

import javafx.beans.property.SimpleStringProperty
import javafx.collections.FXCollections
import javafx.collections.ObservableList
import javafx.scene.control.ListView
import tornadofx.*

```

```

class ListController(
    private val fragment: Fragment,
    private val observable: SimpleStringProperty
) : Controller() {
    val items: ObservableList<String> = FXCollections.observableArrayList("IM-21", "IM-22")
    private var selectedMsg = ""

    fun handleChange(listview: ListView<String>) {
        listview.selectionModel.selectedItemProperty().addListener { _, _, newValue ->
            selectedMsg = newValue
        }
    }

    fun handleBtnYesChange() {
        observable.value = this.selectedMsg
    }

    fun handleBtnCancelChange() {
        observable.value = ""
        fragment.close()
    }
}

class Window : Fragment("Window") {
    val message: SimpleStringProperty by param()
    private val controller = ListController(this, message)
}

```

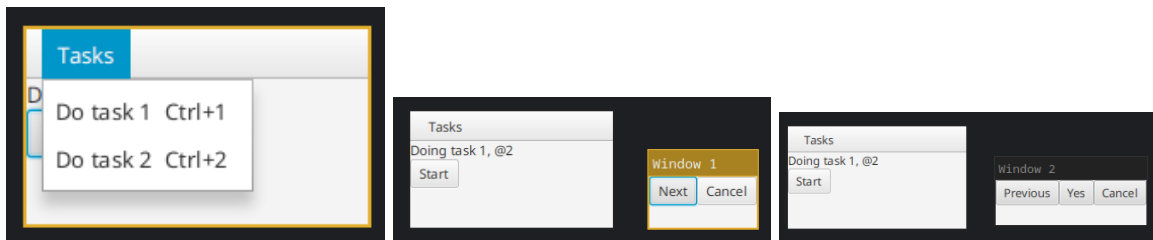
```

override val root = vbox {
    listview(controller.items) { controller.handleChange(this) }
    hbox {
        button("Yes") { action(controller::handleBtnYesChange) }
        button("Cancel") { action(controller::handleBtnCancelChange) }
    }
}
}

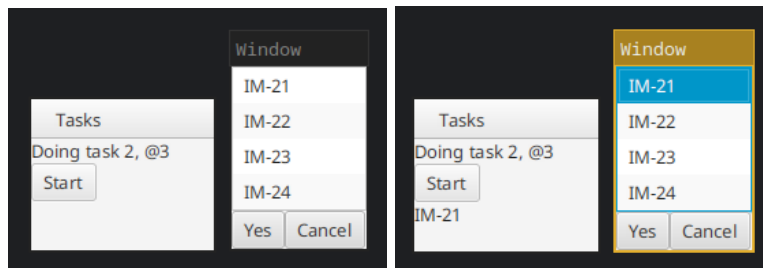
```

## 4 Ілюстрації:

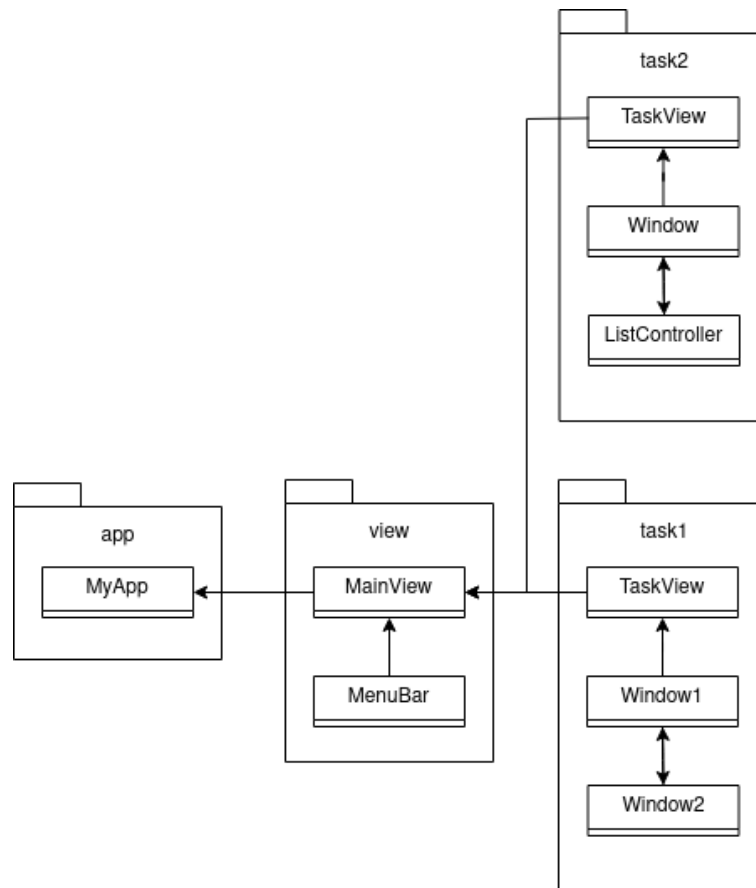
### 4.1 Task 1



### 4.2 Task 2



### 4.3 Діаграма класів



## 5 Висновки:

Отже, я отримав навички створення програм для Linux і навчився модульному програмуванню на Kotlin.