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 №6

з дисципліни
«Об’єктно орієнтоване програмування»

Виконав:

Студент групи ІП-04

Пащенко Дмитро Олексійович
номер у списку групи: 19

Перевірив:

Порєв Віктор Миколайович

Київ 2021

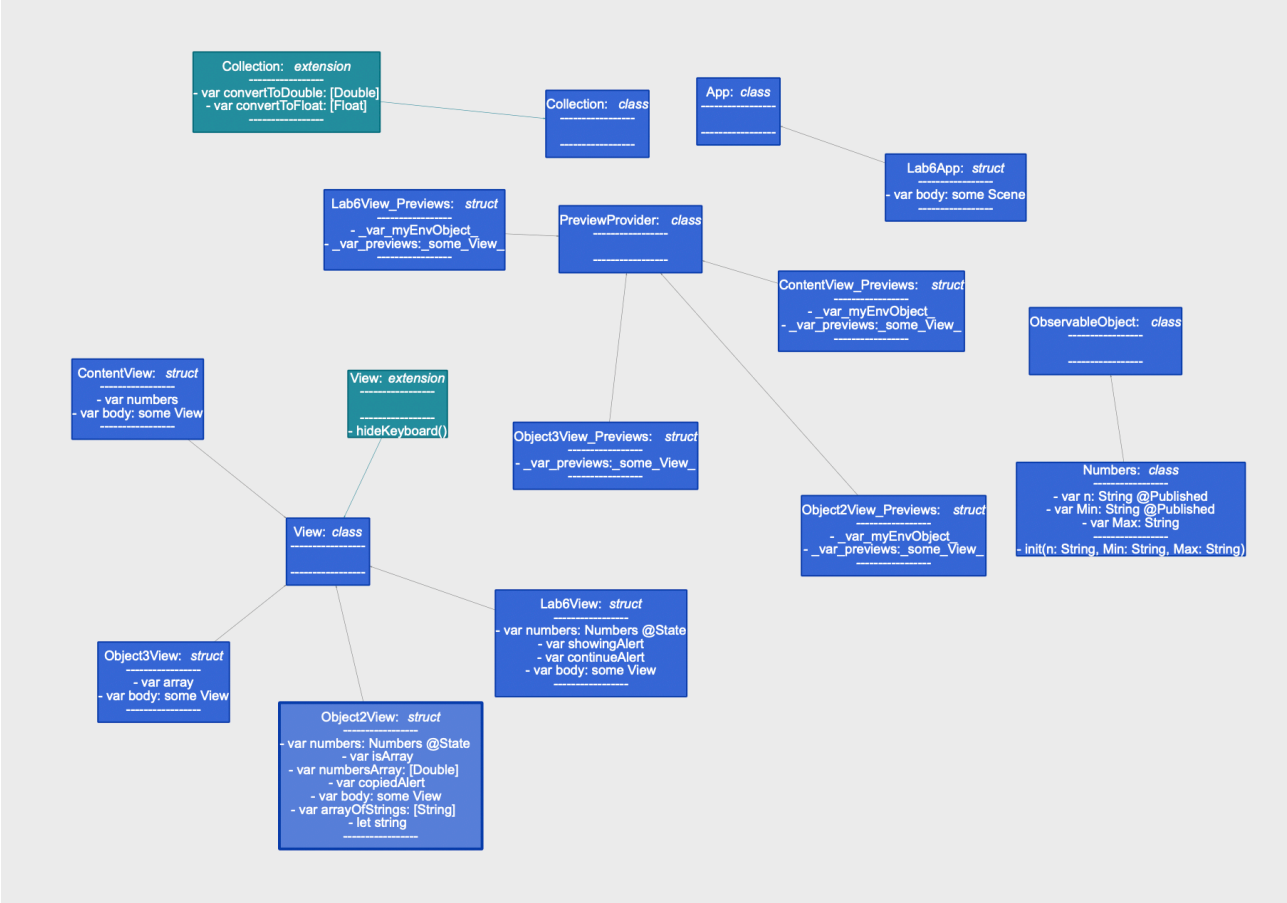
Мета:

Мета роботи – отримати вміння та навички використовувати засоби обміну інформацією та запрограмувати взаємодію незалежно працюючих програмних компонентів.

Завдання:

1. Створити у середовищі MS Visual Studio C++ проект Win32 з ім’ям Lab6.
2. Написати вихідні тексти усіх програм-компонентів згідно варіанту завдання.
3. Скомпілювати вихідні тексти і отримати виконувані файли програм.
4. Перевірити роботу програм. Налагодити взаємодію програм.
5. Проаналізувати та прокоментувати результати та вихідні тексти програм.
6. Оформити звіт.

Діаграма класів:



Висновок

Ми отримали вміння та навички використовувати засоби обміну інформацією та запрагромувати взаємодію незалежно працюючих програмних компонентів.
Навчились працювати із буфером обміну.

Код програми

Lab6App.swift

```
import SwiftUI

@main
struct Lab6App: App {
    var body: some Scene {
        WindowGroup {
            ContentView()
        }
    }
}
```

Model.swift

```
import Foundation

class Numbers: ObservableObject {
    @Published var n: String
    @Published var Min: String
    @Published var Max: String

    init(n: String, Min: String, Max: String) {
        self.n = n
        self.Min = Min
        self.Max = Max
    }
}
```

ContentView.swift

```
import SwiftUI

struct ContentView: View {
    var numbers = Numbers(n: "", Min: "", Max: "")

    var body: some View {
        TabView {
            Lab6View()
                .tabItem {
                    Text("Lab6")
                    Image(systemName: "1.square")
                }

            Object2View()
                .tabItem {
                    Text("Object2")
                    Image(systemName: "2.square")
                }

            Object3View()
                .tabItem {
                    Text("Object3")
                    Image(systemName: "3.square")
                }
        }
        .environmentObject(numbers)
    }
}

struct ContentView_Previews: PreviewProvider {
    static var myEnvObject = Numbers(n: "", Min: "", Max: "")
```

```
    static var previews: some View {
        ContentView()
            .environmentObject(myEnvObject)
    }
}
```

Lab6View.swift

```
import SwiftUI
extension View {
    func hideKeyboard() {

UIApplication.shared.sendAction(#selector(UIResponder.resignFirstResponder), to:
nil, from: nil, for: nil)
    }
}

struct Lab6View: View {
    @EnvironmentObject var numbers: Numbers
    @State private var showingAlert = false
    @State private var continueAlert = false

    var body: some View {
        VStack {
            TextField("n", text: $numbers.n)
                .padding()
                .frame(width: 200)
                .background(.bar)
                .cornerRadius(15)
                .keyboardType(.decimalPad)

            TextField("Min", text: $numbers.Min)
                .padding()
                .frame(width: 200)
                .background(.bar)
                .cornerRadius(15)
                .keyboardType(.decimalPad)

            TextField("Max", text: $numbers.Max)
                .padding()
                .frame(width: 200)
                .background(.bar)
                .cornerRadius(15)
                .keyboardType(.decimalPad)

            Button("Start") {
                hideKeyboard()

                if numbers.n.isEmpty || numbers.Min.isEmpty ||
numbers.Max.isEmpty {
                    showingAlert = true
                } else {
                    continueAlert = true
                }
            }
            .alert("Something is not filled", isPresented: $showingAlert) {
                Button("OK", role: .cancel) {  }
            }
            .alert("Success! Continue to Object2", isPresented: $continueAlert)
{
                Button("OK", role: .cancel) {  }
            }
            .padding()
            .background(.blue)
            .foregroundColor(.white)
            .cornerRadius(15)
```

```
        }
    }
}

struct Lab6View_Previews: PreviewProvider {
    static var myEnvObject = Numbers(n: "", Min: "", Max: "")

    static var previews: some View {
        Lab6View()
            .environmentObject(myEnvObject)
    }
}
```

Object2View.swift

```
import SwiftUI

struct Object2View: View {
    @EnvironmentObject var numbers: Numbers
    @State var isArray = false
    @State var numbersArray: [Double] = []
    @State private var copiedAlert = false

    var body: some View {
        List {
            ForEach(numbersArray, id: \.self) { item in
                HStack {
                    Text("\(numbersArray.firstIndex(of: item)!)")
                    Spacer()
                    Text("\(item)")
                }
            }
        }
        .onAppear {
            if !isArray {
                for _ in 0 ... (Int(numbers.n)! - 1) {
                    numbersArray.append(Double.random(in:
Double(numbers.Min)! ... Double(numbers.Max)!))
                }
                isArray = true
            } else {
                numbersArray = []
                for _ in 0 ... (Int(numbers.n)! - 1) {
                    numbersArray.append(Double.random(in:
Double(numbers.Min)! ... Double(numbers.Max)!))
                }
            }
        }
        .onTapGesture {
            var arrayOfStrings: [String] = []
            for item in numbersArray {
                arrayOfStrings.append(String(item))
            }

            let string = arrayOfStrings.joined(separator: ", ")

            UIPasteboard.general.string = string

            copiedAlert = true
        }
        .alert("Array is copied to clipboard", isPresented: $copiedAlert) {
            Button("OK", role: .cancel) {  }
        }
    }
}
```

```
struct Object2View_Previews: PreviewProvider {
    static var myEnvObject = Numbers(n: "", Min: "", Max: "")

    static var previews: some View {
        Object2View()
            .environmentObject(myEnvObject)
    }
}
```

Object3View.swift

```
import SwiftUI
import SwiftUICharts

extension Collection where Iterator.Element == String {
    var convertToDouble: [Double] {
        return compactMap{ Double($0) }
    }
    var convertToFloat: [Float] {
        return compactMap{ Float($0) }
    }
}

struct Object3View: View {
    @State var array = UIPasteboard.general.string!.components(separatedBy: ",
").convertToDouble

    var body: some View {
        LineView(data: array, title: "Line chart", legend: "Full screen")
    }
}

struct Object3View_Previews: PreviewProvider {
    static var previews: some View {
        Object3View()
    }
}
```