Министерство образования

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Отчет по лабораторной работе №4

«Использование языка программирования Swift: наследование, протоколы»

Выполнил:Крукович А.В.

Студент группы 310902

Проверил: Усенко Ф.В.

Минск 2024

Цель работы – Выполнить разработку приложения с использованием языка программирования Swift: наследование, протоколы.

Задание. Протокол Отправки уведомлений. Реализовать метод, который оправляет уведомление. Создать класс, который будет иметь в себе список различных способов доставки уведомлений. Добавить несколько различных способов доставки уведомлений.

**Код на языке Swift:**

**Package.swift**

**// swift-tools-version:5.7**

**import PackageDescription**

**let package = Package(**

**name: "NotificationSenderProject",**

**targets: [**

**.executableTarget(**

**name: "MainModule",**

**dependencies: ["NotificationModule", "DeliveryMethodsModule", "NotificationProtocolsModule"]**

**),**

**.target(**

**name: "NotificationModule",**

**dependencies: ["DeliveryMethodsModule", "NotificationProtocolsModule"]**

**),**

**.target(**

**name: "DeliveryMethodsModule",**

**dependencies: ["NotificationProtocolsModule"]**

**),**

**.target(**

**name: "NotificationProtocolsModule",**

**dependencies: []**

**),**

**]**

**)**

**DeliveryMethodsModule\EmailDelivery.swift**

**import Foundation**

**import NotificationProtocolsModule**

**public class EmailDelivery: NotificationSender {**

**public init() {}**

**public func sendNotification(message: String, to recipient: String) {**

**print("Отправка уведомления по email: \(message) на адрес \(recipient)")**

**}**

**}**

**DeliveryMethodsModule\PushNotificationsDelivery.swift**

**import Foundation**

**import NotificationProtocolsModule**

**public class PushNotificationDelivery: NotificationSender {**

**public init() {}**

**public func sendNotification(message: String, to recipient: String) {**

**print("Отправка push-уведомления: \(message) на устройство \(recipient)")**

**}**

**}**

**DeliveryMethodsModule\SmsDelivery.swift**

**import Foundation**

**import NotificationProtocolsModule**

**public class SmsDelivery: NotificationSender {**

**public init() {}**

**public func sendNotification(message: String, to recipient: String) {**

**print("Отправка уведомления по SMS: \(message) на номер \(recipient)")**

**}**

**}**

**NotificationProtocolsModule\NotificationSender.swift**

**import Foundation**

**public protocol NotificationSender {**

**func sendNotification(message: String, to recipient: String)**

**}**

**NotificationsModule\NotificationSenderMessage.swift**

**import Foundation**

**import DeliveryMethodsModule**

**import NotificationProtocolsModule**

**public class NotificationSenderManager {**

**private var deliveryMethods: [NotificationSender]**

**public init() {**

**self.deliveryMethods = []**

**}**

**public func addDeliveryMethod(\_ method: NotificationSender) {**

**deliveryMethods.append(method)**

**}**

**public func sendAllNotifications(message: String, to recipient: String) {**

**for method in deliveryMethods {**

**method.sendNotification(message: message, to: recipient)**

**}**

**}**

**}**

**MainModule\main.swift**

**import Foundation**

**import NotificationModule**

**import NotificationProtocolsModule**

**import DeliveryMethodsModule**

**let notificationSenders: [NotificationSender] = [**

**EmailDelivery(),**

**SmsDelivery(),**

**PushNotificationDelivery()**

**]**

**let notificationManager = NotificationSenderManager()**

**for sender in notificationSenders {**

**notificationManager.addDeliveryMethod(sender)**

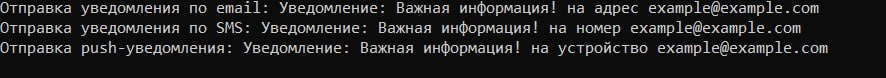
**}**

**let message = "Уведомление: Важная информация!"**

**let recipient = "example@example.com"**

**notificationManager.sendAllNotifications(message: message, to: recipient)**

**Скриншоты работающей программы:**

****

**Рисунок 1 - Результат выполнения программы**

**Вывод: В этой лабораторной работе мы разработали приложение на языке программирования Swift, используя наследование и протоколы. Функционал приложения был разбит на несколько пакетов, учитывая логику работы, и была сделана валидация вводимых значений, работа программы происходила с экземплярами протоколов, а не экземплярами классов.**