Розроблено програмний продукт, що відповідає всім стандартним вимогам тривимірної комп'ютерної гри. Додаток має зрозумілий інтерфейс, звукове супроводження ігрових подій, тривимірну графічну складову з ефектами та повноцінний ігровий цикл.

МІРОШНИЧЕНКО Д.В., ДЕМКІВСЬКА Т.І.

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ СТВОРЕННЯ СИСТЕМИ КОНВЕРТАЦІЇ ФОРМАТІВ .XLS, XLS X, HTML, PDF НА БАЗІ TELEGRAM BOT API

MIROSHNYCHENKO D.V., DEMKIVSKA T.I.

DEVELOPENT OF SOFTWARE FOR CREATING THE FORMAT XLS, XLS X, HTML, PDF CONVERT SYSTEM BASED ON TELEGRAM BOT API

The article touches the subject of to use the telegram platform and its telegram api bot to implement an automated bot that will convert files of different formats.

It also explains the principles of developing an automated bot that are embedded in the design structure, which allows you to build the work process as quickly and efficiently as possible. Which in turn allows you to reduce the load on the server and minimize its work.

Вступ

У сучасному світі ключовим фактором розробки програмного забезпечення є якість і швидкість розробки. Великі компанії розуміючи це, створюють великі платформи, які дозволяють самостійно реалізовувати будь-які програмні рішення.

Платформа мессенджера ―Telegram" дозволяє на своїй базі програмно створити бота, який може реалізовувати різні програмні рішення. Це дає змогу заощадити час та кошти.

Постановка завдання

Головною метою даного дослідження є використання платформи мессенджера ―Telegram" в створенні бота, який буде конвертувати файли з одного формату в інший швидко, просто і максимально ефективно для користувача, а також розробка архітектури проекту та створення дружнього інтерфейсу взаємодії з користувачем. Бот має бути виконаний за правилами патернів Strategy та State. В залежності від вибору користувача формату для конвертації, реалізація конвертування має бути представлена по правилам паттерну Strategy, де в залежності від контексту ми можемо змінювати реалізацію виконання. Завданням роботи є демонстрація того, як телеграмм-бот конвертує файли з одного формату в інший де результатом виконання конвертації будет файл с потрібним для вас форматом сформований ботом в процесі конвертації.

Основна частина

Розроблено бот, який наглядно демонструє використання патерну Strategy для більш ефективної реалізації алгоритму швидкого конвертування з одного формату в інший.

При виконанні даної роботи було досліджено засоби та правила патерну Strategy та API Telegram-Bot. Предметом вивчення являються патерни, автоматизовані боти та засоби які потрібні для їх реалізації. Методами дослідження виступають аналіз та експеримент роботи фреймворку, створеного по правилам патерну. Засобами дослідження є інструменти створення автоматичних ботів, такі як API Telegram Bot, .Net Core, а також мова програмування C#. Новизна цієї роботи полягає у слідуванні класичних правил патерну і реалізація його на сучасній платформі для створення ботів. Практичне значення дослідження полягає в тому, що класичні правила є актуальні у розробці навіть через 20 років і дозволяють реалізовувати швидко та зручно на різних платформах, що запобігає дублюванню кода та дозволяє швидко вносити зміні до коду.

Strategy – це патерн поведінки, являє шаблон проектування, що визначає набір алгоритмів, інкапсулює кожен з них і забезпечує їх взаємозамінність.

Головним правилом цього патерну є те що залежно від ситуації ми можемо легко замінити один використовуваний алгоритм іншим. При цьому заміна алгоритму відбувається незалежно від об'єкта, який використовує даний алгоритм.

Зазвичай, Strategy виступає у якості об‘єктно-орієнтованого класу, що допомагає взаємодіяти інтерфейсом від якого успадковуються всі конкретні реалізації для кожного процесу конвертації.

Коли використовувати стратегію:

− коли є кілька споріднених класів, які відрізняються поведінкою. Можна задати один основний клас, а різні варіанти поведінки винести в окремі класи і при необхідності їх застосовувати;

− коли необхідно забезпечити вибір з кількох варіантів алгоритмів, які можна легко міняти в залежності від умов;

− коли необхідно змінювати поведінку об'єктів на стадії виконання програми;

− коли клас, який застосовує певну функціональність, нічого не повинен знати про її реалізації.

Учасники паттерну:

− інтерфейс IStrategy, який визначає метод Algorithm (). Це загальний інтерфейс для всіх реалізуючих його алгоритмів. Замість інтерфейсу тут також можна було б використовувати абстрактний клас;

− класи ConcreteStrategy1 і ConcreteStrategy, які реалізують інтерфейс IStrategy, надаючи свою версію методу Algorithm (). Подібних класів-реалізацій може бути безліч;

− клас Context зберігає посилання на об'єкт IStrategy і пов'язаний з інтерфейсом IStrategy зв'язком агрегації.

На Рис.1 представлено об'єкт IStrategy який є властивістю ContextStrategy, для якого можна виділити приватну змінну, щоб потім, за допомогою динамічної установки, використовувати спеціальний метод.

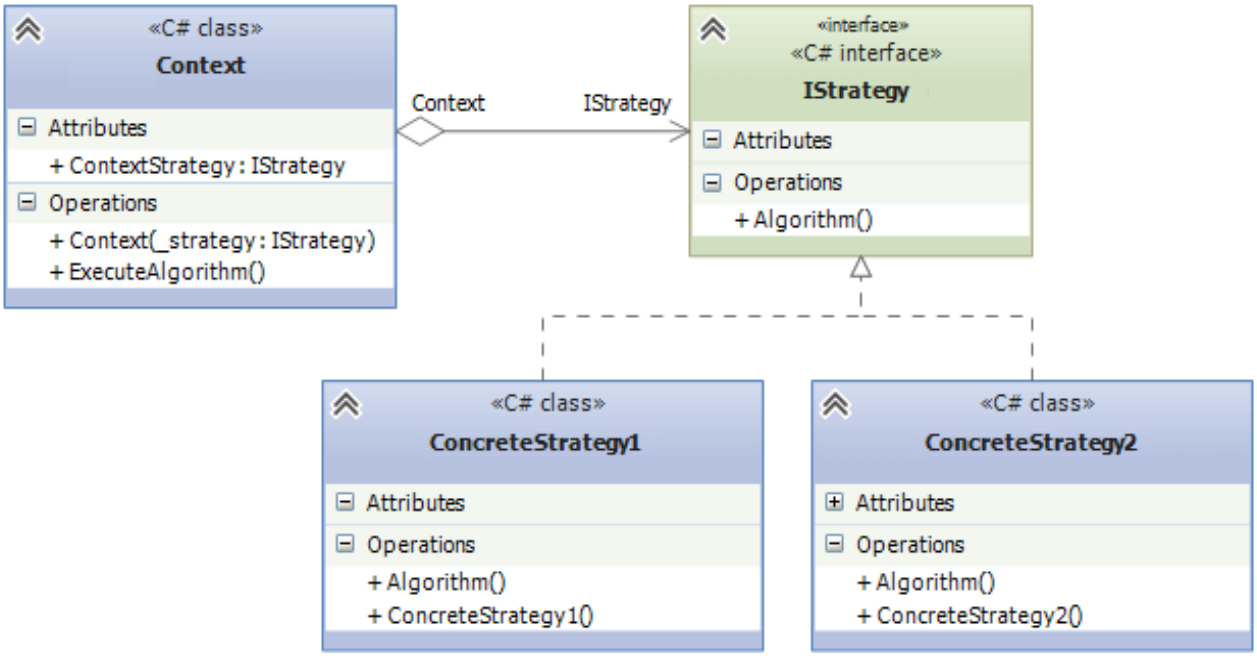

Рис. 1

Автоматизований бот - спеціальні акаунти в Telegram, створені для того, щоб автоматично обробляти і відправляти повідомлення. Користувачі можуть взаємодіяти з ботами за допомогою повідомлень, що відправляються через звичайні або групові чати. Логіка бота контролюється за допомогою HTTPS запитів до нашого API для пошукових роботів.

Висновки

В результаті даного дослідження було проведено вивчення та аналіз патерну Strategy та документації Telegram Bot Api. Завдяки відокремленню реалізації конвертування різних форматів одне від одного, за допомогою автоматизованого бота, ми отримуємо можливість легкої, швидкої та ефективної конвертації файлів різних форматів.

Розроблено автоматизований чат-бот, реалізований за правилами паттерна Strategy.

Література

1. Герберт Шилдт C# 4.0: повне керівництво. – М.: Вільямс, 2019. – 1056

2. Джефрі Ріхтер CLR via C #. Програмування на платформі Microsoft .NET Framework 4.5 мовою C#. – М.: Питер, 2016. – 896

ПОСВІСТАК В.С., ДЕМКІВСЬКА Т.І.

КЛІЄНТ-СЕРВЕРНА АРХІТЕКТУРА ТА ЇЇ ВИКОРИСТАННЯ ПРИ РОЗРОБЦІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

POSVISTAK V.S., DEMKIVSKA T.I.

CLIENT-SERVER ARCHITECTURE AND ITS USAGE IN SOFTWARE DEVELOPMENT

Since the invention of programming architecture per se, it never was in stagnation. It has always been evolving to fix old inconveniences and make software more secure, expandable and productive. This article touches the subject of client-server architecture as a solution which brought a numerous amount of possibilities. Client-server architecture is used in the vast majority of modern applications and it is a great example of simple and laconic design pattern providing the ability to create powerful and complex software products.

Developed an Android referential application using the client-server architecture

Вступ

Винайдення програмування створило багато труднощів. Питання комунікації програмних модулів, розширюваності системи, а також продуктивності мали бути вирішені за допомогою архітектури або дизайну програмного коду певним чином.

Клієнт-серверна архітектура відіграє важливу роль у розробці ПЗ. На сьогодні вона використовується у більшості програмних продуктів через те, що вирішує проблему побудови комунікації різних модулів кінцевого продукту.

Постановка завдання

Метою даного дослідження є розробка Android-застосунку та ознайомлення з клієнт-серверною архітектурою, що включає в себе аналіз проблем, які вона вирішує, в першу чергу питання комунікації програмних модулів. Розглянуті можливі альтернативи та приклади використання архітектури в типовому застосунку, що потребує централізоване джерело даних. Такі застосунки широко представлені на сучасному ринку програмного забезпечення. Існує тенденція до зберігання даних в ~~“~~хмарі” (cloud), тобто не на локальному накопичувачі даних, а на віддаленому. В такому випадку комунікація відбувається через мережу Інтернет. Окрім зберігання даних стали можливі хмарні обчислення, де роль клієнта полягає лише у делегуванні певних операцій на сервер і отримання