УДК 004.42:519.681

ТЕХНОЛОГІЇ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ У АКТУАЛЬНИХ МОВАХ ПРОГРАМУВАННЯ

Савенко О.О., Полякова Т.В.

Харківський національний автомобільно-дорожній університет, Харків

Об'єктно-орієнтоване програмування є поширеним підходом розробки різноманітних додатків. Об'єктом дослідження є реалізація технології об'єктно-орієнтованого програмування у актуальних мовах програмування на прикладі розробки мобільних додатків на платформі Android.Програмування Android-пристроїв ділиться на дві частини – програмування додатків користувача та системне програмування (драйвери та модулі системи).

Програмування додатків користувача здійснюють на декількох мовах програмування, насамперед Java та останнім часом Kotlin, що є розробкою компанії JetBrains як заміна Java із спрощеним синтаксисом. Також можлива розробка на мовах C# (Xamarin) та C++ (Qt QML, native).

Особливості реалізації ООП:

– Java – об'єктно-орієнтована мова програмування, в Java все є об'єктом (однокореневі ієрархія, оскільки все виходить від java.lang.Object). Java підтримує абстрактні класи та інтерфейси. Один клас може успадковувати декілька інтерфейсів, що є деяким аналогом множинного успадковування. Елементи функціонального програмування на основі використання лямбда функцій також присутні в останніх версіях мови. [1]

– Kotlin є модифікованим нащадком Java, та має такі корисні речі, як lambda вирази та функції розширення. Kotlin не підтримує статичні методи класів, Null-references контролюються системою типів мови. В Kotlin реализовано концепцию data class для класів, призначених для зберігання даних, при цьому в них автоматично додається деякий корисний функціонал, наприклад методи для порівняння або копіювання. [2]

– C# - це мова програмування загального призначення, яка

підтримує різні технології програмування, елементи функціонального програмування та є основною мовою програмування для Windows, також підтримується кросплатформна розробка. У C# немає механізму множинного успадковування класів, але є множинне успадковування інтерфейсів. [3]

– C++ це універсальна мова програмування. В C ++ не існує кореневої ієрархії об’єктів. C ++ підтримує як процедурне, так і об'єктно-орієнтоване програмування. C++11 та новіші версії мови мають розширені можливості метапрограмування на основі використання шаблонів, елементи функціонального програмування на основі використання лямбда-функцій. C++ підтримує множинне успадковування та механізми для вирішення проблем ромбовидних ієрархій. [4]

Таким чином, сучасні об’єктно-орієнтовані програмування підтримують реалізацію основних концепцій ООП - інкапсуляцію, успадковування, поліморфізм. Множинне успадковування, інтерфейси, абстрактні класи, метапрограмування та ін. можуть бути відсутніми у конкретній мові. [5]

Інтроспекція - це здатність програми досліджувати тип або властивості об'єкта під час роботи програми, іноді визначити всю ієрархію класів..

Рефлексія - це здатність комп'ютерної програми вивчати і модифікувати свою структуру і поведінку (значення, мета-дані, властивості і функції) під час виконання, тобто викликати методи об'єктів, створювати нові об'єкти, модифікувати, не знаючи імен інтерфейсів, полів, методів під час компіляції.

Ці можливості мови хоча і порушують стандартні підходи до ООП, але в деяких випадках можуть бути дуже корисними для того, аби програма могла змінюватися під час виконання та реалізувати алгоритми більш складного рівня.

Проведений аналіз показав, що об’єктно-орієнтоване програмування все ще залишається актуальною технологією програмування, що має особливості реалізації у різних мовах програмування. Сучасними тенденціями є спрощення мовних конструкцій (Kotlin vs Java), відмова або заміна від суперечливих механізмів на кшталт множинного успадковування, спонукання до

використання змішаних технологій – елементів функційного програмування в ООП-мовах.

Таблиця 1 – Особливості реалізації мов програмування

	Java	Kotlin	C#	C++
Інкапсуляція	+	+	+	+
Успадковування	+	+	+	+
Поліморфізм	+	+	+	+
Множинне успадковування	-	-	-	+
Абстрактні класи	+	+	+	+*
Інтерфейси	+	+	+	-
Множинне успадковування інтерфейсів	+	+	+	-
Метапрограмування	+	+	+	+
Рефлексія	+	+	+	-
Інтроспекція	+	+	+	+**

Примітка: ** - чисті віртуальні класи; * RTTI.

Список використаних джерел

[1] Java Language and Virtual Machine Specifications, URL: https://docs.oracle.com/javase/specs/

[2] M. Akhin, V/ Belyaev, Kotlin language specification, URL: https://kotlinlang.org/spec/introduction.html

[3] C# language specification | Microsoft Docs, URL: https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/language-specification/introduction

[4] Standard C++, URL: https://isocpp.org/std/the-standard

[5] О.В. Мнушка, В.М. Савченко, О.Б. Маций, *Об'єктно-орієнтоване програмування мовою Python*, Харків, ХНАДУ, 2021. – 228 с. – ISBN: 978-617-7912-88-9.