УДК 004.43
Осійчук І. В. – ст. гр. СНнм-51
Тернопільський національний технічний університет імені Івана Пулюя

ОСОБЛИВОСТІ ФУНКЦІЙНОГО ПРОГРАМУВАННЯ НА SCALA

Науковий керівник: старший викладач Шимчук Г.

Osiichuk I. V.
Ternopil Ivan Puluj National Technical University

FEATURES OF FUNCTIONAL PROGRAMMING ON SCALA

Supervisor: Senior Lecturer Shymchuk G.

Ключові слова: функційне, програмування, scala, монади, функції, особливості.
Key words: functional, programming, scala, monads, functions, features.

Функційне програмування є одним з підходів до розробки програмного забезпечення, де основний акцент робиться на використанні функцій як основного засобу обчислення. Scala - це мова програмування, яка добре підтримує функційний підхід і надає багато функціональних можливостей для розробки складних програм.

Функції першого класу: Функції в Scala є об'єктами першого класу, що означає, що вони можуть бути передані як аргументи в інші функції, повернуті як значення з інших функцій, а також збережені в змінних.

Незмінність даних: В Scala, багато об'єктів є незмінними, що означає, що після створення їх значення не може бути змінено. Замість цього, нові об'єкти створюються на основі старих, що дозволяє уникнути побічних ефектів і забезпечує більш безпечну і передбачувану роботу з даними. Високорівневі функції: Scala підтримує високорівневі функції, такі як каррінг (currying) і часткове застосування (partial application), що дозволяє зручно створювати нові функції шляхом комбінування існуючих.

Рекурсія: Функціональне програмування сприяє використанню рекурсії як основного засобу ітерації, замість циклів. Scala надає підтримку для хвостової рекурсії, що дозволяє ефективніше вирішувати завдання, що потребують рекурсивного підходу.Функції вищих порядків: Функціональне програмування в Scala дозволяє використовувати функції вищих порядків - це такі функції, які приймають інші функції в якості аргументів або повертають їх як результати. Це дозволяє писати загальні функції, які можуть бути параметризовані різними функціями і використовуватись в різних контекстах.

Непрозорі типи даних: Scala підтримує непрозорі типи даних (opaque types), що дозволяють визначати власні типи з декларованим інтерфейсом, але приховують реалізацію. Це допомагає створювати безпечні абстракції, що підтримують принципи функціонального програмування, такі як незмінність даних і відсутність побічних ефектів.

Потоки (streams): Scala надає вбудовану підтримку для створення та операцій з потоками даних, які можуть бути безкінечними і лінивими, що дозволяє ефективно опрацьовувати великі об'єми даних. Безпечні функції: Функціональний підхід в Scala дозволяє писати функції, які не мають побічних ефектів та не змінюють стану програми. Це сприяє більшій безпеці та передбачуваності програмного коду, допомагає уникати багів, пов'язаних з змінним станом.

Імутабельні дані: Scala підтримує не змінювані дані за замовчуванням, що означає, що дані не можуть бути змінені після їх створення. Це дозволяє уникати

багатьох проблем, пов'язаних з конкурентністю та розділенням даних між різними функціями. Композиція функцій: Функціональний підхід дозволяє композицію функцій, що спрощує створення складних операцій з використанням простих функцій, які можуть бути повторно використані.

Обробка помилок: Scala надає механізми для обробки помилок, такі як Option, Either, та Try, які дозволяють більш безпечний та експресивний спосіб обробки помилок у функціональному стилі. Паралельне програмування: Scala надає вбудовану підтримку для паралельного програмування, таку як актори з бібліотекою Akka, що дозволяє створювати ефективні багатопотокові програми з використанням функціонального підходу.

Модульність: Функціональне програмування в Scala сприяє високій модульності коду, оскільки функції можуть бути розбиті на менші функції, що реалізують окремі функціональності. Це спрощує тестування, підтримку та розширення коду. Підтримка високого рівня абстракції: Scala надає можливості для використання високого рівня абстракції, таких як монади, з використанням функціонального підходу. Це дозволяє писати більш елегантний та зрозумілий код. Інтероперабельність: Scala є мовою, яка працює на віртуальній машині Java (JVM), що дозволяє використовувати багато бібліотек та інфраструктури, що вже існують у екосистемі Java. Це дозволяє Scala-розробникам використовувати багато різних інструментів, бібліотек та фреймворків, які розроблені для Java.

Розширюваність: Функціональний підхід в Scala дозволяє розширювати мову та створювати нові абстракції з використанням механізмів, таких як макроси. Це дозволяє розробникам розширювати мову та додавати свої власні конструкції мови, що допомагає вирішувати складні завдання.

Це лише загальний огляд функціонального програмування в Scala. Ця мова програмування надає багато функціональних концепцій та можливостей, що дозволяють писати експресивний, стійкий до помилок та ефективний код. Використання функціонального програмування в Scala може бути корисним для розробки розширених та складних додатків, а також для підвищення продуктивності розробника та покращення якості програмного коду. Для вивчення деталей функціонального програмування в Scala рекомендується ознайомитися зі специфічними функціональними конструкціями мови, такими як високорівневі функції, незмінність даних, зіставлення візерунків (pattern matching), монади, а також з іншими функціональними бібліотеками, такими як Cats, Scalaz, ZIO, тощо.

Функціональне програмування в Scala відкриває широкі можливості для розробки складних, масштабованих та високопродуктивних додатків, але потребу. Використання функціональних концепцій може сприяти покращенню структури коду, робити його більш читабельним, модульним та тестованим.

Література:

1. Paul Chiusano. Functional Programming in Scala / Paul Chiusano – Manning (1th edition), 2014. – 320 p.
2. Martin Odersky. Programming in Scala / Martin O. – Artima (4th edition), 2019. – 845 p.
3. Haoyi Li. Hands-on Scala Programming. / Haoyi Li – Haoyi Li (1th edition), 2020. – 414 p.