Секция «Информационные технологии»

Асинхронное программирование: всестороннее изучение его преимуществ, недостатков и лучших практик

Islam Asabaev
Студент (бакалавр)
Чеченский государственный университет, Факультет информационных технологий, Грозный, Россия
E-mail: 7576457@mail.ru

УДК 004.42

Синхронное и Асинхронное программирование

Две различные парадигмы программирования с различными подходами к выполнению и использованию ресурсов. В синхронном программировании инструкции выполняются последовательно, при этом каждая инструкция ожидает завершения предыдущей, прежде чем перейти к следующей. Это может привести к блокировке, когда программа не может продолжить выполнение до завершения текущей операции, что приводит к медленному реагированию и снижению производительности, особенно при наличии медленных операций ввода-вывода.

С другой стороны, асинхронное программирование позволяет выполнять несколько задач одновременно, позволяя программе продолжать выполнение в ожидании завершения медленных операций ввода-вывода. Это приводит к повышению быстродействия, сокращению времени ожидания и повышению общей производительности.

С точки зрения использования ресурсов синхронное программирование может привести к неэффективному использованию ресурсов, поскольку программа не может продолжить выполнение до завершения текущей операции. Напротив, асинхронное программирование позволяет выполнять несколько задач одновременно, что позволяет более эффективно использовать ресурсы и повышать производительность.

Стоит отметить, что асинхронное программирование может быть достигнуто различными способами, такими как использование потоков, управляемыми событиями, или неблокирующим вводом-выводом. [1][5]

Управляемые события

Управление событиями в асинхронном программировании относится к процессу управления потоком выполнения в программе, основанному на событиях, инициируемых различными входными данными. Эти входные данные могут поступать из различных источников, включая взаимодействие с пользователем, системные уведомления или другие асинхронные процессы.

В асинхронном программировании управление событиями обычно осуществляется через цикл событий, который является центральным компонентом программы, ответственным за обработку входящих событий и запуск соответствующих ответов. Цикл событий отвечает за выполнение функций обратного вызова или других обработчиков, связанных с конкретными событиями, и за управление потоком выполнения в программе.

Цикл событий предназначен для обработки нескольких событий одновременно, не блокируя выполнение остальной части программы. Это позволяет программе быстро реагировать на события, не дожидаясь завершения других процессов.

Управление событиями является важнейшим аспектом асинхронного программирования, поскольку оно обеспечивает возможность обрабатывать входящие события своевременно и эффективно, избегая при этом узких мест в производительности.[2][4]

Потоки программирования

Потоки программирования в асинхронном программировании относятся к порядку, в котором различные задачи или операции выполняются в рамках асинхронной программы. В асинхронном программировании операции выполняются асинхронно, что означает, что они не блокируют выполнение других операций. Это требует иного подхода к проектированию и управлению потоком программы по сравнению с традиционным синхронным программированием.

В асинхронном программировании существует несколько распространенных программных потоков, в том числе:

· Программирование на основе обратного вызова - включает в себя выполнение функции обратного вызова при завершении задачи или операции, что позволяет программе продолжать выполнять другие задачи или операции параллельно.

· Программирование на основе обещаний - включает в себя использование обещаний для представления результата асинхронной операции, что позволяет программе управлять потоком выполнения и обрабатывать ошибки более организованным образом.

· Программирование Async / Await. включает в себя использование ключевых слов async и await для объявления асинхронных функций и ожидания их завершения более читаемым и интуитивно понятным способом.

Каждый из этих потоков программирования имеет свои собственные преимущества и недостатки, и выбор того, какой из них использовать, будет зависеть от конкретных требований программы и предпочтений программиста.

Независимо от используемого потока программирования, асинхронное программирование требует иного мышления и подхода к разработке и выполнению кода по сравнению с традиционным синхронным программированием. Тщательное рассмотрение потока и порядка операций имеет решающее значение для обеспечения правильного и эффективного выполнения программы.[3]

Неблокирующий (асинхронный) ввод-вывод

Неблокирующий ввод-вывод — это фундаментальная концепция в асинхронном программировании, которая позволяет программе выполнять несколько операций ввода-вывода одновременно, не блокируясь медленными операциями ввода-вывода. В традиционном синхронном программировании, когда программа выполняет операцию ввода-вывода, такую как чтение из файла или ожидание ответа сети, она блокируется до завершения операции. Это может привести к снижению производительности и длительному времени ожидания, особенно когда несколько операций ввода-вывода выполняются одна за другой. В отличие от этого, неблокирующий ввод-вывод позволяет программе инициировать операцию ввода-вывода и продолжать выполнять другие задачи, ожидая завершения операции. Когда операция ввода-вывода завершается, программа получает уведомление, позволяющее ей получить результаты и продолжить обработку. Неблокирующий ввод-вывод реализуется с помощью обратных вызовов или событий, которые представляют собой функции, запускаемые при возникновении определенного события. Например, когда чтение файла завершено, запускается событие и выполняется функция обратного вызова, позволяющая программе извлечь результаты и продолжить обработку.

Такой подход позволяет асинхронному программированию обрабатывать несколько операций ввода-вывода одновременно, повышая общую производительность системы за счет сокращения времени ожидания и избегая узких мест. Кроме того, это позволяет создавать высокочувствительные системы, способные эффективно обрабатывать обмен данными в режиме реального времени и большие объемы данных.[6]

Плюсы асинхронного программирования

• Масштабируемость. Выполнение нескольких задач одновременно, что позволяет

им лучше масштабироваться и обрабатывать большие объемы данных и пользователей.

- Производительность. Вытекающий из первого, так как выполняя несколько задач одновременно, сокращается время ожидания и повышается общая производительность.
- Оперативность. Позволяет программам продолжать выполнение в ожидании медленных операций ввода-вывода, таких как доступ к диску или сети, повышая оперативность системы.
- Использование ресурсов. При асинхронном программировании, программы лучше используют системные ресурсы, поскольку они могут продолжать выполняться в ожидании завершения операций ввода-вывода, а не блокироваться.
- Параллелизм. Асинхронное программирование позволяет создавать сложные параллельные системы, которые могут обрабатывать несколько задач и событий одновременно, повышая общую производительность и быстроту реагирования системы.
- Уменьшения задержки в системе, поскольку оно позволяет программам продолжать выполняться в ожидании завершения операций ввода-вывода, а не блокироваться.

Минусы асинхронного программирования

- Сложность. Асинхронное программирование может быть более сложным в написании и обслуживании, чем синхронное программирование, особенно при работе с условиями гонки и взаимоблокировками.
- Отладка. Отладка кода может быть сложной задачей из-за сложных взаимосвязей между задачами, что затрудняет выявление и исправление ошибок.
- Взаимоблокировки. В асинхронном программировании взаимоблокировки могут возникать, когда две задачи ожидают завершения друг друга, что приводит к взаимоблокировке, препятствующей дальнейшему прогрессу.
- Тестирование асинхронного кода может быть более сложным, чем тестирование синхронного кода, поскольку оно требует моделирования нескольких параллельных задач и обработки асинхронных событий.
- Обработка ошибок в асинхронном программировании может быть сложной задачей, поскольку она требует обработки ошибок в нескольких местах и обеспечения правильного распространения ошибок по всей программе.
- Управление памятью. В асинхронном программировании важно обеспечить надлежащее высвобождение ресурсов, когда они больше не нужны, чтобы избежать утечек памяти и других проблем с производительностью.

Несмотря на перечисленные минусы и плюсы, многие разработчики программного обеспечения считают, что преимущества асинхронного программирования намного перевешивают его недостатки и что этот подход хорошо подходит для широкого спектра приложений и вариантов использования. При создании веб-приложений, игр или систем реального времени асинхронное программирование может помочь разработчикам вывести свое программное обеспечение на новый уровень производительности.

Использование и примеры

Асинхронное программирование широко используется в различных областях, включая веб-разработку, разработку игр и системы реального времени. В веб-разработке он обычно используется для повышения отзывчивости и производительности веб-приложений, позволяя им обрабатывать несколько запросов одновременно. Например, Node.js, популярная платформа для создания веб-приложений, использует асинхронное программирование для обработки входящих запросов и выполнения операций с базой данных. Ruby on Rails, еще один популярный веб-фреймворк, также поддерживает асинхронное программирование, позволяя разработчикам писать эффективные и масштабируемые веб-приложения.

При разработке игр асинхронное программирование используется для повы-

шения производительности и отзывчивости игр, позволяя нескольким задачам, таким как рендеринг графики и обработка входных данных [7], выполняться одновременно. Это позволяет играм работать плавно и быстро реагировать на действия игрока, улучшая общий игровой опыт.

Асинхронное программирование в веб-разработке. Например, веб-приложению, которое позволяет пользователям загружать файлы, может потребоваться выполнить несколько задач одновременно, таких как обработка загруженного файла, отправка уведомлений и обновление учетной записи пользователя. Используя асинхронное программирование, приложение может выполнять эти задачи в фоновом режиме, повышая свою общую производительность и позволяя пользователям продолжать использовать приложение, не испытывая длительного ожидания.[8]

Системы реального времени, такие как торговые системы, финансовые системы и системы фондового рынка, также выигрывают от асинхронного программирования. Эти системы требуют обмена данными в режиме реального времени и обработки больших объемов данных, а асинхронное программирование позволяет им эффективно решать эти задачи.

Заключение.

В заключение следует отметить, что синхронное программирование лучше подходит для небольших и менее сложных проектов, где легкость понимания и простота имеют приоритет над производительностью и оперативностью реагирования. С другой стороны, асинхронное программирование лучше подходит для более крупных и сложных проектов, требующих высокой производительности и быстроты реагирования, где ключевым моментом являются методы повышения производительности, масштабируемости и отзывчивости программных приложений. Это позволяет нескольким задачам выполняться одновременно и независимо, что позволяет более эффективно использовать вычислительные ресурсы и улучшает пользовательский опыт. Однако это также сопряжено с дополнительными сложностями и может затруднить отладку и обработку ошибок. Понимание различных подходов к асинхронному программированию, таких как программирование, управляемое событиями, неблокирующий ввод-вывод и потоки, а также использование обратных вызовов и обещаний, имеет важное значение для эффективного использования этого метода.

Источники и литература

1) Мельников В.А. Понятие асинхронного программирования // история, современное состояние и перспективы инновационного развития науки // Сборник статей по итогам Международной научно-практической конференции. Стерлитамак, 2021 // Издательство: Общество с ограниченной ответственностью "Агентство международных исследований" (Уфа) // 2021. С.51-52.
2) Любимова Т.В. Асинхронность в программировании // Университетская наука. 2019. № 2 (8). С. 135-138.
3) Фешина Е.В., Омельченко Д.А., Гонатаев Р.Г. Многопоточность и асинхронность в языке программирования python // ИННОВАЦИИ. НАУКА. ОБРАЗОВАНИЕ. 2021 № 28. С.988-992.
4) Асинхронное программирование в .NET от Microsoft, [URL]: https://docs.microsoft.com/en-us/dotnet/standard/asynchronous-programming-patterns/
5) Синхронное и асинхронное Асинхронное программирование: чем они отличаются? [URL]: https://www.makeuseof.com/synchronous-asynchronous-programming-different/

6) «Асинхронное программирование в Node.js» по Node.js Фундамент [URL]: https://nodejs.org/en/docs/guides/blocking-vs-non-blocking/
7) Асинхронное программирование в веб-проектах – Григорий Добряков [URL]: https://www.dobryakov.com/blog/822/