1 Определение многопоточного программирования Java

В отличие от многих других языков программирования, Java предлагает встроенную поддержку многопоточного программирования. Многопоточная программа содержит две или более частей, которые могут выполняться одновременно. Каждая часть такой программы называется потоком (thread), и каждый поток задает отдельный путь выполнения.

Практически все современные операционные системы поддерживают многозадачность. Существуют две формы многозадачности. Одна основана на процессах, а другая – на потоках.

Процесс (process) – это объект, который создается операционной системой, когда пользователь запускает приложение. Процессу выделяется отдельное адресное пространство, причем это пространство физически недоступно для других процессов.

Процесс может работать с файлами или с каналами связи локальной или глобальной сети. Чаще всего одна программа состоит из одного процесса, но бывают и исключения. Процессы изолированы друг от друга, поэтому прямой доступ к памяти чужого процесса невозможен (взаимодействие между процессами осуществляется с помощью специальных средств).

Для каждого процесса ОС создает «виртуальное адресное пространство», к которому процесс имеет прямой доступ. Это пространство принадлежит процессу, содержит только его данные и находится в полном его распоряжении. Операционная система же отвечает за то, как виртуальное пространство процесса проецируется на физическую память. Схема этого взаимодействия представлена на рисунке 1.

Операционная система оперирует так называемыми страницами памяти, которые представляют собой просто область определенного фиксированного размера. Если процессу становится недостаточно памяти, система выделяет ему дополнительные страницы из физической памяти.

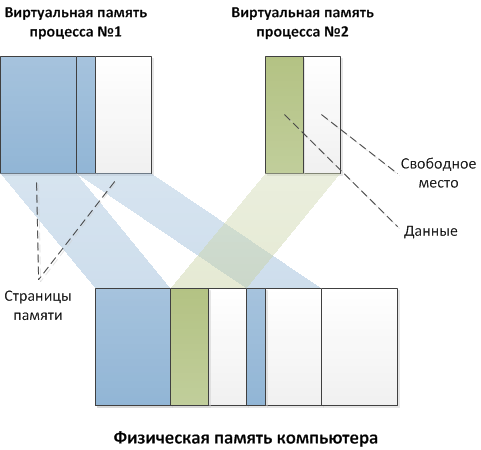


Рисунок 1 – Схема распределения виртуального адресного пространства в физической памяти компьютера

Страницы виртуальной памяти могут проецироваться на физическую память в произвольном порядке. При запуске программы операционная система создает процесс, загружая в его адресное пространство код и данные программы, а затем запускает главный поток созданного процесса.

Исполнительная система Java во многом зависит от потоков, и все библиотеки классов разработаны с учетом многопоточности. Java использует потоки для обеспечения асинхронности во всей среде. Это помогает уменьшить неэффективность, предотвращая затраты циклов CPU.

Для каждого процесса операционная система создает один главный поток (thread), который является потоком выполняющихся по очереди команд центрального процессора. При необходимости главный поток может создавать другие потоки, пользуясь для этого программным интерфейсом операционной системы.

Все потоки, созданные процессом, выполняются в адресном пространстве этого процесса и имеют доступ к ресурсам процесса. Однако поток одного процесса не имеет никакого доступа к ресурсам потока другого процесса, так как они работают в разных адресных пространствах.

Один поток – это одна единица исполнения кода. Каждый поток последовательно выполняет инструкции процесса, которому он принадлежит, параллельно с другими потоками этого процесса.

Известно, что на одно ядро процессора, в каждый момент времени, приходится одна единица исполнения. Однако запуск нескольких параллельных потоков возможен и в системах с одноядерными процессорами. В этом случае система будет периодически переключаться между потоками, поочередно давая выполняться то одному, то другому потоку. Такая схема называется псевдо-параллелизмом, представлена на рисунке 2.

Цветные квадраты на рисунке – это инструкции процессора (зеленые – инструкции главного потока, синие – побочного).

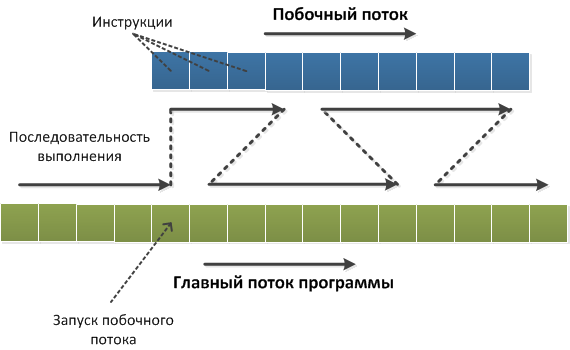


Рисунок 2 – Схема переключения между потоками

Выполнение идет слева направо. После запуска побочного потока его инструкции начинают выполняться вперемешку с инструкциями главного потока. Кол-во выполняемых инструкций за каждый подход не определено.

Система запоминает состояние (контекст) каждого потока, перед тем как переключиться на другой поток, и восстанавливает его по возвращению к выполнению потока. В контекст потока входят такие параметры, как стек, набор значений регистров процессора, адрес исполняемой команды и другие.

Потоки существуют в нескольких состояниях. Поток может быть в состоянии выполнения. Может находиться в состоянии готовности к выполнению, как только он получит время CPU. Выполняющийся поток может быть приостановлен, что временно притормаживает его действие.

Затем приостановленный поток может быть продолжен с того места, где он был остановлен. Поток может быть блокирован в ожидании ресурса. В любой момент выполнение потока может быть завершено, что немедленно останавливает его выполнение. После завершения поток не может быть продолжен.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. <https://habr.com/ru/articles/164487/>
2. <http://www.uic.unn.ru/~tarasov/JavaLectures/TarasovVLJavaAndEclipse_10_MultiThreads.pdf>
3. <https://www.helloworld.ru/texts/comp/lang/java/java5/vol10/ch2.html>
4. <https://metanit.com/java/tutorial/8.1.php>
5. <https://habr.com/ru/sandbox/167189/>
6. <http://www.linkex.ru/java/multithreaded-programming.php>