

УДК 004.438.NET
ББК 32.973.26-018.2
Т18

Танвар Ш.

Т18 Параллельное программирование на C# и .NET Core / пер. с англ. А. Д. Ворониной; ред. В. Н. Черников. – М.: ДМК Пресс, 2022. – 272 с.: ил.

ISBN 978-5-97060-851-7

Книга представляет подход к параллельному программированию с учетом современных реалий. Информация структурирована таким образом, чтобы она легко усваивалась, даже если читатель не обладает специальными знаниями. Рассматриваются общие принципы написания параллельного и асинхронного кода; реализация параллелизма данных показана на коротких и простых примерах. В конце глав приводятся вопросы для повторения пройденного.

Издание предназначено для программистов C#, которые хотят изучить концепции параллельного программирования и многопоточности, а затем использовать полученные знания для приложений, построенных на базе .NET Core. Также оно пригодится специалистам, желающим ознакомиться с принципами работы параллельного программирования на современном оборудовании.

УДК 004.438.NET
ББК 32.973.26-018.2

First published in the English language under the title 'Hands-On Parallel Programming with C# 8 and .NET Core 3 – (9781789132410)'. Russian language edition copyright © 2021 by DМК Press. All rights reserved.

Все права защищены. Любая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

ISBN 978-1-78913-241-0 (англ.)
ISBN 978-5-97060-851-7 (рус.)

© Packt Publishing, 2019
© Перевод, оформление, издание,
ДМК Пресс, 2022

Содержание

От издательства	14
Об авторе	15
О переводе	16
О рецензентах	17
Предисловие	18

Часть I. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ РАБОТЫ С ПОТОКАМИ, МНОГОЗАДАЧНОСТИ И АСИНХРОННОСТИ	22
---	-----------

Глава 1. Введение в параллельное программирование	23
Технические требования.....	24
Подготовка к многоядерным вычислениям	24
Процессы.....	24
Дополнительно об ОС	24
Многозадачность	25
Hyper-threading	25
Классификация Флинна.....	26
Потоки	27
Типы потоков	27
Многопоточность.....	30
Класс Thread	31
Класс ThreadPool	35
BackgroundWorker	38
Многопоточность и многозадачность	41
Сценарии, при которых полезно параллельное программирование	42
Преимущества и недостатки параллельного программирования	42
Резюме	43
Вопросы	44

Глава 2. Параллелизм задач	45
Технические требования.....	45
Задачи	46
Создание и запуск задачи	46
Класс System.Threading.Tasks.Task	47
Синтаксис лямбда-выражений.....	47
Делегат Action	47
Делегат	47

Метод <code>System.Threading.Tasks.Task.Factory.StartNew</code>	48
Синтаксис лямбда-выражений	48
Делегат <code>Action</code>	48
Делегат	48
Метод <code>System.Threading.Tasks.Task.Run</code>	49
Синтаксис лямбда-выражений	49
Делегат <code>Action</code>	49
Делегат	49
Метод <code>System.Threading.Tasks.Task.Delay</code>	49
Метод <code>System.Threading.Tasks.Task.Yield</code>	50
Метод <code>System.Threading.Tasks.Task.FromResult<T></code>	52
Методы <code>System.Threading.Tasks.Task.FromException</code> и <code>System.Threading.Tasks.Task.FromException<T></code>	53
Методы <code>System.Threading.Tasks.Task.FromCanceled</code> и <code>System.Threading.Tasks.Task.FromCanceled<T></code>	53
Результаты выполнения задач	54
Отмена задач	55
Создание метки	55
Создание задач с использованием меток	56
Опрос состояния метки через свойство <code>IsCancellationRequested</code>	56
Регистрация отмены запроса с помощью делегата обратного вызова	57
Ожидание выполнения задач	58
<code>Task.Wait</code>	59
<code>Task.WaitAll</code>	59
<code>Task.WaitAny</code>	60
<code>Task.WhenAll</code>	60
<code>Task.WhenAny</code>	61
Обработка исключений в задачах	61
Обработка исключений из одиночных задач	62
Обработка исключений из нескольких задач	62
Обработка исключений задач с помощью обратного вызова	63
Преобразование шаблонов APM в задачи	64
Преобразование EAP в задачи	66
И еще о задачах	67
Цепочки задач	67
Продолжение выполнения задач с помощью метода <code>Task.ContinueWith</code>	68
Продолжение выполнения задач с помощью <code>Task.Factory.ContinueWhenAll</code> и <code>Task.Factory.ContinueWhenAll<T></code>	69
Продолжение выполнения задач с помощью <code>Task.Factory.ContinueWhenAny</code> и <code>Task.Factory.ContinueWhenAny<T></code>	69
Родительские и дочерние задачи	70
Создание отсоединенной задачи	70
Создание присоединенной задачи	71
Очереди с перехватом работы	72
Резюме	74

Глава 3. Реализация параллелизма данных	75
Технические требования.....	75
От последовательных циклов к параллельным.....	75
Метод <code>Parallel.Invoke</code>	76
Метод <code>Parallel.For</code>	78
Метод <code>Parallel.ForEach</code>	79
Степень параллелизма	80
Создание своей стратегии разделения данных.....	82
Разделение данных по диапазону.....	83
Разделение данных по блокам.....	83
Отмена циклов	84
Использование метода <code>Parallel.Break</code>	85
Использование <code>ParallelLoopState.Stop</code>	86
Использование <code>CancellationToken</code> для отмены циклов.....	87
Хранение данных в параллельных циклах	88
Локальная переменная потока	89
Локальная переменная блока данных.....	90
Резюме	91
Вопросы	91
 Глава 4. Использование PLINQ	93
Технические требования.....	93
LINQ-провайдеры в .NET	93
Создание PLINQ-запросов	94
Знакомство с классом <code>ParallelEnumerable</code>	94
Наш первый запрос PLINQ	95
Сохранение порядка в PLINQ при параллельном исполнении.....	96
Последовательное выполнение с использованием метода <code>AsUnOrdered()</code>	97
Параметры объединения данных в PLINQ.....	98
Параметр <code>NotBuffered</code>	98
Параметр <code>AutoBuffered</code>	99
Параметр <code>FullyBuffered</code>	100
Отправка и обработка исключений с помощью PLINQ	102
Объединение параллельных и последовательных запросов LINQ.....	104
Отмена запросов PLINQ.....	104
Недостатки параллельного программирования с помощью PLINQ	106
Факторы, влияющие на производительность PLINQ (ускорения).....	106
Степень параллелизма.....	107
Настройка объединения данных.....	107
Тип разделения данных.....	107
Когда нужно сохранять последовательное исполнение в PLINQ?.....	107
Порядок работы	108
<code>ForAll</code> против вызова <code>ToArray()</code> или <code>ToList()</code>	108
Принудительный параллелизм	108
Генерация последовательностей.....	108
Резюме	109
Вопросы	110

Часть II. СТРУКТУРЫ ДАННЫХ .NET CORE, КОТОРЫЕ ПОДДЕРЖИВАЮТ ПАРАЛЛЕЛИЗМ 111

Глава 5. Прimitives синхронизации 112

Технические требования.....	112
Что такое примитивы синхронизации?	113
Операции со взаимоблокировкой	113
Барьеры доступа к памяти в .NET	115
Что такое изменение порядка?.....	115
Типы барьеров памяти.....	116
Как избежать изменения порядка.....	117
Введение в примитивы блокировки	118
Как работает блокировка	118
Состояния потока	118
Блокировка или вращение?	119
Блокировка, мьютекс и семафор	120
Lock	120
Mutex	123
Semaphore	124
ReaderWriterLock	126
Введение в сигнальные примитивы	126
Thread.Join.....	126
EventWaitHandle	128
AutoResetEvent	128
ManualResetEvent.....	129
WaitHandles	131
Легковесные примитивы синхронизации	134
Slim locks	134
ReaderWriterLockSlim	135
SemaphoreSlim.....	136
ManualResetEventSlim	137
События Barrier и CountdownEvent	137
Примеры использования Barrier и CountdownEvent	138
SpinWait	140
SpinLock.....	141
Резюме	142
Вопросы	142

Глава 6. Использование параллельных коллекций 144

Технические требования.....	144
Введение в параллельные коллекции.....	144
Знакомство с IProducerConsumerCollection<T>.....	145
Использование ConcurrentQueue <T>.....	145
Производительность Queue<T> в сравнении с ConcurrentQueue<T>	148
Использование ConcurrentStack <T>.....	148
Создание параллельного стека.....	149

Использование ConcurrentBag<T>	150
Использование BlockingCollection<T>	151
Создание BlockingCollection<T>	151
Сценарий с несколькими производителями и потребителями.....	153
Использование ConcurrentDictionary<TKey,TValue>.....	154
Резюме	155
Вопросы	156

Глава 7. Повышение производительности с помощью

отложенной инициализации	157
Технические требования.....	157
Что такое отложенная инициализация?.....	157
Введение в System.Lazy<T>	160
Логика создания объекта реализуется в конструкторе.....	161
Логика создания объекта передается в качестве делегата в Lazy<T>	162
Обработка исключений с помощью шаблона отложенной инициализации	163
Отсутствие исключений в ходе инициализации	163
Случайное исключение при инициализации с кэшированием исключений	163
Некэшируемые исключения	165
Отложенная инициализация с локальным хранилищем потоков	166
Сокращение издержек при помощи отложенной инициализации.....	168
Резюме	170
Вопросы	170

Часть III. АСИНХРОННОЕ ПРОГРАММИРОВАНИЕ

С ИСПОЛЬЗОВАНИЕМ C#	172
----------------------------------	-----

Глава 8. Введение в асинхронное программирование

Технические требования.....	174
Типы выполнения программ	174
Синхронное выполнение программ	174
Асинхронное выполнение программ	176
Случаи использования асинхронного программирования.....	177
Написание асинхронного кода	177
Использование метода BeginInvoke класса Delegate.....	178
Использование класса Task.....	179
Использование интерфейса IAsyncResult	179
Когда не следует использовать асинхронное программирование	181
В базе данных без пула обработки подключений.....	181
Когда важно, чтобы код легко читался и поддерживался.....	181
Для простых и быстрых операций	181
Для приложений с большим количеством разделяемых данных	182
Проблемы, решаемые асинхронным кодом	182
Резюме	183
Вопросы	183

Глава 9. Основы асинхронного программирования с помощью async, await и задач	184
Технические требования.....	184
Введение в async и await	185
Возвращаемый тип асинхронных методов	188
Асинхронные делегаты и лямбда-выражения.....	189
Асинхронные шаблоны на основе задач	189
Метод компилятора с ключевым словом async.....	189
Ручная реализация TAP	189
Обработка исключений с помощью асинхронного кода	190
Метод, возвращающий Task и создающий исключение.....	190
Асинхронный метод вне блока try-catch без await.....	191
Вызов асинхронного метода из блока try-catch без await.....	192
Вызов асинхронного метода с await за пределами блока try-catch.....	194
Метод, возвращающий значение void	194
Асинхронность с PLINQ.....	195
Оценка производительности асинхронного кода.....	196
Рекомендации по написанию асинхронного кода	198
Не используйте async void	199
Все методы в цепочке вызовов должны быть асинхронными.....	199
По возможности используйте ConfigureAwait	200
Выводы	200
Вопросы	200

Часть IV. ОТЛАДКА, ДИАГНОСТИКА И МОДУЛЬНОЕ ТЕСТИРОВАНИЕ АСИНХРОННОГО КОДА..... 202

Глава 10. Отладка задач с Visual Studio	203
Технические требования.....	204
Отладка с VS 2019.....	204
Отладка потоков.....	204
Использование окон параллельных стеков	206
Отладка при помощи окон параллельных стеков	207
Представление потоков	207
Представление задач	209
Отладка с использованием окна контроля параллельных данных.....	209
Использование визуализатора параллелизма.....	211
Представление использования.....	212
Представление потоков	212
Представление ядер	213
Выводы	214
Вопросы	214
Дополнительные материалы для чтения	215

Глава 11. Создание модульных тестов для параллельного и асинхронного кодов	216
Технические требования.....	216

Модульное тестирование с .NET Core	217
Проблемы при написании модульных тестов для асинхронного кода	219
Создание модульных тестов для параллельного и асинхронного кодов	221
Проверка на успешный результат	221
Проверка результата исключения при нулевом делителе	222
Имитация обращений к реальным методам и данным с помощью Moq	222
Инструменты тестирования	224
Выводы	225
Вопросы	226
Дополнительные материалы для чтения	226

Часть V. ДОПОЛНИТЕЛЬНЫЕ СРЕДСТВА ПОДДЕРЖКИ ПАРАЛЛЕЛЬНОГО ПРОГРАММИРОВАНИЯ В .NET CORE

Глава 12. IIS и Kestrel в ASP.NET Core

Технические требования	228
Многопоточность в IIS и внутренние компоненты	229
Предотвращение нехватки ресурсов	229
Поиск восхождения к вершине	229
Многопоточность в Kestrel и внутренние компоненты	231
ASP.NET Core 1.x	232
ASP.NET Core 2.x	232
Лучшие практики использования многопоточности в микросервисах	233
Микросервисы с одним потоком и одним процессором	233
Микросервисы с одним потоком и несколькими процессорами	234
Микросервисы с несколькими потоками и одним процессором	234
Асинхронные сервисы	234
Выделенные пулы потоков	234
Введение асинхронности в ASP.NET MVC Core	235
Асинхронные потоки	238
Выводы	241
Вопросы	241

Глава 13. Шаблоны параллельного программирования

Технические требования	243
Шаблон MapReduce	243
Реализация MapReduce с помощью LINQ	244
Агрегация	246
Шаблон разделения/объединения	248
Шаблон спекулятивной обработки	248
Шаблон отложенной инициализации	249
Шаблон разделяемого состояния	252
Выводы	252
Вопросы	253

Глава 14. Управление распределенной памятью

Технические требования	255
------------------------------	-----

Введение в распределенные системы.....	255
Модель общей и распределенной памяти.....	256
Модель общей памяти.....	256
Модель распределенной памяти	257
Типы коммуникационных сетей	258
Статические коммуникационные сети	258
Динамические коммуникационные сети	259
Свойства коммуникационных сетей.....	259
Топология.....	260
Алгоритмы маршрутизации	261
Стратегия коммутации	261
Управление потоком	261
Исследование топологий	262
Линейная и кольцевая топологии	262
Линейные массивы.....	262
Кольцо или тор.....	263
Решетки и торы	263
Двумерные решетки	263
2D-тор.....	264
Программирование устройств с распределенной памятью с использованием передачи сообщений	264
Почему MPI?	265
Установка MPI на Windows	265
Пример программы с использованием MPI	265
Базовое использование отправки/приема сообщений	266
Коллективы	267
Выводы	267
Вопросы	268
Ответы на вопросы.....	269
Предметный указатель.....	270