

Оглавление

Предисловие	15
Об авторе.....	19
Контактная информация	19
Благодарности.....	20
От издательства	21
Введение во второе издание	22
Введение.....	24
Цель этой книги.....	24
В чем смысл выбора управляемого кода.....	27
Работает ли управляемый код медленнее нативного?.....	29
Стоит ли овчинка выделки?	31
Я что, теряю контроль?	31
Работа с CLR, а не против нее	32
Уровни оптимизации.....	32
Коварная соблазнительность простоты	34
Хронология совершенствования производительности среды .NET.....	36
.NET Core	38
Учебный исходный код.....	39
Глава 1. Измерение производительности и инструменты	41
Выбор предмета измерения.....	41
Преждевременная оптимизация	43
Сравнение усредненных и процентных показателей	44
Эталонное тестирование.....	46

Полезные инструменты	47
Visual Studio.....	49
Профилирование центрального процессора	51
Профилирование с помощью командной строки	54
Счетчики производительности	56
ETW-события	64
PerfView.....	67
Интерфейс и представления данных в PerfView	68
Профилер CLR Profiler	73
Анализатор производительности Windows Performance Analyzer	76
WinDbg.....	78
CLR MD	83
Анализаторы IL	87
MeasureIt.....	88
BenchmarkDotNet	89
Оснащение кода инструментами.....	91
Утилиты SysInternals.....	92
База данных.....	94
Другие инструменты.....	94
Издержки измерений.....	94
Резюме.....	95
Глава 2. Управление памятью.....	96
Выделение памяти	96
Операция сборки мусора.....	99
Параметры конфигурации	105
Сравнение сборки мусора в режиме рабочей станции и в режиме сервера.....	105
Сборка мусора в фоновом режиме.....	107
Режимы задержки.....	108
Большие объекты.....	110
Дополнительные параметры	111
Советы по повышению производительности.....	113
Сокращайте размеры выделяемой памяти.....	113
Самое важное правило	114

Сокращайте время существования объекта	115
Сбалансируйте выделение.....	116
Сократите количество ссылок между объектами.....	116
Избегайте закреплений.....	117
Избегайте финализаторов.....	118
Избегайте выделения больших объектов	120
Избегайте копирования буферов	121
Объединяйте долгоживущие и большие объекты в пулы	124
Сокращайте степень фрагментации кучи больших объектов	131
При определенных обстоятельствах выполняйте принудительную полную сборку мусора.....	131
Уплотняйте кучу больших объектов по требованию	133
Получайте уведомление о намечающейся сборке мусора	133
Применяйте для кэширования слабые ссылки	137
Динамически выделяйте память в стеке.....	144
Исследование памяти и сборки мусора.....	145
Счетчики производительности	145
События ETW	147
Как выглядит куча памяти моего приложения	148
Сколько времени занимает сборка мусора	151
Где именно происходит выделение памяти	155
Что за объекты находятся в куче.....	158
Где именно допущена утечка памяти.....	165
Каков размер моих объектов.....	170
Каким объектам выделена память в ЛОН.....	173
Какие объекты были закреплены	175
Где происходит фрагментация	177
Фрагментация виртуальной памяти	180
В каком поколении находится объект	182
Какие объекты выжили в поколении gen 0.....	182
Откуда был сделан явный вызов метода GC.Collect.....	185
Какие слабые ссылки имеются в моем процессе	186
Какие финализируемые объекты имеются в куче.....	186
Резюме.....	187

Глава 3. JIT-компиляция	189
Преимущества JIT-компиляции.....	190
JIT в действии	191
JIT-оптимизации	193
Сокращение времени JIT-компиляции и запуска	194
Оптимизация JIT-компиляции с помощью профилирования (Multicore JIT)	197
Когда следует применять NGEN	197
.NET Native	200
Настраиваемая предварительная подготовка	201
Когда JIT-компиляция не может составить конкуренцию	202
Исследование поведения JIT-компилятора	203
Счетчики производительности	203
ETW-события	204
Какой код подвергся JIT-компиляции	204
На какие методы и модули затрачивается больше всего времени при JIT-компиляции	207
Исследование кода, полученного после JIT-компиляции	208
Резюме.....	209
Глава 4. Асинхронное программирование	210
Пул потоков	212
Библиотека распараллеливания задач	213
Отмена задачи	217
Обработка исключений.....	218
Дочерние задачи	222
Среда TPL Dataflow	223
Параллельно выполняемые циклы	229
Советы по повышению производительности.....	232
Избегайте использования блокировок.....	232
Избегайте конвоев при блокировке и диспетчеризации.....	233
Использование объектов Tasks для неблокирующего ввода-вывода.....	233
async и await	238
О структуре программы.....	240
Правильно используйте таймеры.....	242

Подберите подходящий размер пула потоков	244
Не прерывайте потоки	245
Не меняйте приоритет потоков.....	245
Синхронизация потоков и блокировки.....	246
Нужно ли вообще заботиться о производительности?	246
А нужна ли вообще блокировка?.....	247
Порядок предпочтения синхронизации.....	249
Модели памяти	249
Использование volatile при необходимости.....	251
Использование Monitor (lock)	252
Использование методов Interlocked.....	255
Асинхронные блокировки	258
Другие механизмы блокировки	260
Конкурентность и коллекции.....	261
Копирование ресурса для каждого потока.....	264
Исследование потоков и конфликтов.....	265
Счетчики производительности	265
ETW-события	266
Получение информации о потоках.....	267
Визуализация задач и потоков с помощью Visual Studio	268
Использование PerfView для обнаружения конфликта блокировок	269
Где потоки блокируются на вводе-выводе	270
Резюме.....	270
Глава 5. Общие подходы к написанию кода и классов.....	272
Классы и структуры	272
Исключение из правил: изменяемая структура для хранения иерархии полей.....	274
Виртуальные методы и запечатанные классы	276
Свойства.....	276
Переопределение Equals и GetHashCode для структур	277
Потоковая безопасность	279
Кортежи.....	279
Диспетчеризация интерфейсов.....	280

Избегайте упаковки.....	281
Возвращения по ссылке (ref) и локальные значения.....	282
for или foreach.....	287
Приведение типов.....	289
P/Invoke.....	291
Делегаты.....	293
Исключения.....	295
dynamic.....	297
Отражение.....	299
Генерация кода.....	301
Создание шаблонов.....	301
Создание делегата.....	302
Аргументы метода.....	303
Оптимизация.....	305
Подведение итогов.....	305
Предварительная обработка.....	306
Исследование проблем производительности.....	307
Счетчики производительности.....	307
ETW-события.....	307
Поиск инструкций упаковки.....	308
Обнаружение исключений первого шанса.....	310
Резюме.....	311
Глава 6. Использование среды .NET Framework.....	313
Разберитесь с каждым вызываемым API.....	314
Множество API для решения одних и тех же задач.....	314
Коллекции.....	315
Какие коллекции лучше не использовать.....	315
Массивы.....	316
Сравнение ступенчатых и многомерных массивов.....	317
Обобщенные коллекции.....	319
Коллекции для многопоточной среды.....	321
Коллекции для работы с битами.....	323
Исходный объем.....	324

Сравнение ключей.....	325
Сортировка.....	326
Создание собственных типов коллекций	326
Строки.....	327
Сравнение строк.....	327
ToUpper и ToLower	328
Объединение.....	328
Форматирование	329
ToString.....	330
Избегайте разбора строк.....	330
Подстроки.....	331
Избегайте использования API, выдающих исключения при обычных обстоятельствах	331
Избегайте использования API, выделяющих память из кучи больших объектов	332
Применение ленивой инициализации.....	332
Удивительно высокие издержки от использования перечислений	334
Учет времени	335
Регулярные выражения	337
LINQ.....	339
Чтение и запись файлов.....	343
Оптимизация настроек HTTP и сетевых соединений	344
SIMD	347
Исследование причин возникновения проблем с производительностью.....	348
Резюме.....	349
Глава 7. Счетчики производительности.....	351
Использование существующих счетчиков.....	352
Создание пользовательского счетчика.....	352
Счетчики усредненных показателей.....	353
Счетчики мгновенных показателей.....	354
Дельта-счетчики	354
Процентные счетчики	354
Резюме.....	355

Глава 8. ETW-события	356
Определение событий	357
Потребление пользовательских событий в PerfView	360
Создание собственного слушателя ETW-событий	362
Получение подробных данных об EventSource	367
Потребление событий CLR и системы	368
Пользовательские аналитические расширения PerfView	370
Резюме	373
Глава 9. Безопасность и анализ кода	374
Представление об операционной системе, API и оборудовании	374
Ограничение использования API в определенных областях кода	375
Пользовательские правила FxCop	375
.NET Compiler Code Analyzers	382
Выполняйте централизацию и абстрагирование сложного и важного для повышения производительности кода	391
Изолируйте неуправляемый и небезопасный код	391
Отдавайте приоритет ясности кода, а не получению высокой производительности, пока нет веских причин для обратного	392
Резюме	393
Глава 10. Формирование команды, нацеленной на достижение высокой производительности	394
Выявление областей, требующих особо высокой производительности	394
Эффективное тестирование	395
Инфраструктура и автоматизация для оценки производительности	396
Доверяйте только конкретным числовым показателям	398
Эффективная система просмотра кода	399
Обучение	400
Резюме	401
Приложение А. Начало работы над повышением производительности приложения	402
Определение метрик	402
Анализ использования центрального процессора	402

Анализ использования памяти.....	403
Анализ JIT-компиляции.....	404
Анализ производительности в асинхронном режиме	404
Приложение Б. Увеличение производительности на более высоком уровне	406
ASP.NET.....	406
ADO.NET	407
WPF	408
Приложение В. Нотация «О» большое».....	409
«О» большое	409
Самые распространенные алгоритмы и их сложность	412
Сортировка.....	412
Графы	412
Поиск.....	413
Особый случай.....	413
Приложение Г. Библиография	414
Ценные источники информации	414
Люди и блоги	414