

УДК 004.438.NET
ББК 32.973.26-018.2
Г79

Г79 Голдштейн С., Зурбалева Д., Флатов И. и др.
 Оптимизация приложений на платформе .NET. – Пер. с англ. Киселев
 А. Н. – М.: ДМК Пресс, 2017. – 524 с.: ил.

ISBN 978-5-97060-487-8

Увеличение производительности алгоритмов и приложений является чрезвычайно важным аспектом разработки и может дать вам преимущество перед конкурентами, а вашим пользователям обеспечить низкую стоимость владения и удовольствие от использования быстрых и отзывчивых приложений. Данная книга описывает внутренние особенности ОС Windows, среды выполнения CLR и аппаратного обеспечения, влияющие на производительность приложений, а также дает вам знания и инструменты для измерения производительности вашего кода в изоляции от внешних факторов.

Книга наполнена примерами кода на C# и рекомендациями, которые помогут вам выжать максимум возможного из вашего приложения – низкое потребление памяти, согласованную нагрузку на процессор и минимальное количество операций ввода/вывода с сетью и диском.

Издание предназначено для программистов, знакомых с языком C# и платформой .NET.

УДК 004.438.NET
ББК 32.973.26-018.2

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.

Материал, изложенный в данной книге, многократно проверен. Но, поскольку вероятность технических ошибок все равно существует, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. В связи с этим издательство не несет ответственности за возможные ошибки, связанные с использованием книги.

ISBN 978-143-024-458-5 (англ.)

© by Sasha Goldshtein, Dima Zurbalev, and
 Ido Flatow

ISBN 978-5-97060-487-8 (рус.)

© Оформление, перевод на русский язык, ДМК
 Пресс



ОГЛАВЛЕНИЕ

Предисловие	13
Об авторах	16
О научных редакторах	18
Благодарности	19
Введение	20
ГЛАВА 1.	
Характеристики производительности	23
Требования к производительности	24
Характеристики производительности	28
В заключение	31
ГЛАВА 2.	
Измерение производительности	32
Подходы к измерению производительности	32
Встроенные инструменты Windows	33
Счетчики производительности	34
Механизм трассировки событий для Windows	42
Профилировщики времени	58
Дискретный профилировщик Visual Studio	59
Инструментированный профилировщик Visual Studio	64
Дополнительные приемы использования профилировщиков времени	67
Профилировщики выделения памяти	71
Профилировщик выделения памяти Visual Studio	72
CLR Profiler	75
Профилировщики памяти	81
Другие профилировщики	86
Профилировщики доступа к данным и базам данных	87
Профилировщики конкуренции	88

Профилировщики ввода/вывода	91
Микрохронометраж	92
Пример неправильного микрохронометража	92
Рекомендации по проведению хронометража	96
В заключение	99

ГЛАВА 3.

Внутреннее устройство типов 102

Пример.....	102
Семантические отличия между ссылочными типами и типами значений.....	104
Хранение, размещение и удаление	105
Внутреннее устройство ссылочных типов	108
Таблица методов.....	109
Вызов методов экземпляров ссылочных типов.....	114
Блоки синхронизации и ключевое слово lock.....	122
Внутреннее устройство типов значений.....	128
Ограничения типов значений	130
Виртуальные методы типов значений.....	132
Упаковка	133
Предотвращение упаковки типов значений с помощью метода Equals	136
Метод GetHashCode	140
Эффективные приемы использования типов значений	144
В заключение	144

ГЛАВА 4.

Сборка мусора 145

Назначение сборщика мусора	146
Управление свободным списком	146
Сборка мусора на основе подсчета ссылок	148
Сборка мусора на основе трассировки	150
Фаза маркировки	151
Фазы чистки и сжатия	158
Закрепление	161
Разновидности сборщиков мусора	163
Приостановка потоков для сборки мусора.....	163
Сборщик мусора для сервера	170
Выбор разновидности сборщика мусора.....	172
Поколения	175
Предположения в основе модели поколений.....	176
Реализация поколений в .NET	177

Куча больших объектов	183
Ссылки между поколениями	185
Фоновый сборщик мусора	188
Сегменты сборщика мусора и виртуальная память	189
Финализация	194
Детерминированная финализация вручную	194
Автоматическая недетерминированная финализация	195
Ловушки недетерминированной финализации	198
Шаблон реализации метода Dispose	202
Слабые ссылки	205
Взаимодействие со сборщиком мусора	208
Класс System.GC	209
Взаимодействие с применением интерфейсов размещения CLR	213
Триггеры сборщика мусора	215
Эффективные приемы повышения производительности сборки мусора	216
Модель поколений	216
Закрепление	218
Финализация	219
Разные советы и рекомендации	220
В заключение	226

ГЛАВА 5.

Коллекции и обобщенные типы 230

Обобщенные типы	230
Обобщенные типы в .NET	234
Ограничения обобщенных типов	236
Реализация обобщенных типов в CLR	239
Коллекции	249
Параллельные коллекции	252
Проблемы, связанные с кешем	254
Собственные коллекции	261
Система непересекающихся множеств	261
Список с пропусками	263
Одноразовые коллекции	265
В заключение	269

ГЛАВА 6.

Конкуренция и параллелизм 270

Перспективы и преимущества	270
----------------------------------	-----

Зачем использовать приемы параллельного программирования?	272
От потоков к пулам потоков и задачам	273
Параллелизм задач	281
Параллелизм данных	290
Асинхронные методы в C# 5	295
Дополнительные шаблоны в TPL	300
Синхронизация	302
Код без блокировок	304
Механизмы синхронизации Windows	311
Вопросы оптимального использования кеша	314
Использование GPU для вычислений	318
Введение в C++ AMP	318
Умножение матриц	322
Моделирование движения частиц	323
Мозаики и разделяемая память	325
В заключение	331

ГЛАВА 7.

Сети, ввод/вывод и сериализация 332

Общие понятия	333
Синхронный и асинхронный ввод/вывод	333
Порты завершения ввода/вывода	335
Пул потоков в .NET	340
Копирование памяти	341
Чтение вразброс и запись со слиянием	342
Файловый ввод/вывод	343
Управление кешированием	343
Небуферизованный ввод/вывод	344
Сети	345
Сетевые протоколы	346
Сетевые сокеты	348
Сериализация и десериализация данных	351
Тестирование производительности средств сериализации	352
Сериализация объектов DataSet	354
Windows Communication Foundation	356
Пороговые значения	356
Модель обработки	357
Кеширование	359
Асинхронные клиенты и серверы WCF	359
Привязки	361
В заключение	362

ГЛАВА 8.

Небезопасный код и взаимодействие с ним ... 364

Небезопасный код	365
Закрепление объектов в памяти и дескрипторы сборщика мусора	366
Управление жизненным циклом	368
Выделение неуправляемой памяти	368
Использование пулов памяти	368
P/Invoke	370
PInvoke.net и P/Invoke Interop Assistant	372
Привязка	374
Заглушки маршалера	375
Двоично совместимые типы	380
Направление маршалинга, ссылочные типы и типы значений	382
Code Access Security	383
Взаимодействие с COM-объектами	384
Управление жизненным циклом	386
Маршалинг через границы подразделений	386
Импортирование библиотек типов и Code Access Security	389
NoPIA	390
Исключения	391
Расширения языка C++/CLI	392
Вспомогательная библиотека marshal_as	395
Код на языке IL и неуправляемый код	397
Взаимодействие со средой выполнения WinRT в Windows 8 ...	397
Эффективные приемы взаимодействий	398
В заключение	399

ГЛАВА 9.

Оптимизация алгоритмов 400

Систематизация сложности	401
Большое O	401
Машины Тьюринга и классы сложности	403
Мемоизация и динамическое программирование	409
Расстояние Левенштейна	411
Кратчайший путь между всеми парами вершин	413
Аппроксимация	416
Задача коммивояжера	417
Задача о максимальном разрезе	418
Вероятностные алгоритмы	419
Вероятностное решение задачи о максимальном разрезе	419

Тест простоты Ферма	420
Индексирование и сжатие	421
Кодировка переменной длины	421
Сжатие индексов	423
В заключение	425

ГЛАВА 10.

Шаблоны оптимизации производительности ... 426

Оптимизации JIT-компилятора	426
Стандартные оптимизации	427
Встраивание методов	428
Отключение проверки границ	430
Хвостовые вызовы	432
Производительность на этапе запуска	436
Предварительная JIT-компиляция с помощью NGen (Native Image Generator)	438
Фоновая JIT-компиляция в многопроцессорных системах	441
Упаковщики образов	442
Управляемая оптимизация на основе профилирования	443
Различные советы по оптимизации времени запуска	445
Аппаратно-зависимые оптимизации	447
Единственный поток команд и множество потоков данных	448
Распараллеливание инструкций	452
Исключения	457
Механизм рефлексии	458
Генерация кода	459
Генерация из исходного кода	460
Генерация кода с использованием легковесного генератора кода	462
В заключение	467

ГЛАВА 11.

Производительность веб-приложений 468

Измерение производительности веб-приложений	469
Тестирование производительности и нагрузочное тестирование веб-приложений в среде Visual Studio	469
Инструменты мониторинга HTTP	471
Инструменты анализа веб-взаимодействий	473
Увеличение производительности веб-сервера	473
Кеширование часто используемых объектов	474
Использование асинхронных страниц, модулей и контроллеров	476

Настройка окружения ASP.NET	481
Отключение механизмов трассировки и отладки в ASP.NET	481
Отключение механизма ViewState	483
Кеш вывода на стороне сервера	485
Предварительная компиляция приложений ASP.NET	488
Тонкая настройка модели процесса в ASP.NET	488
Настройка IIS	491
Кеширование вывода	491
Настройка пула приложения	493
Оптимизация сети	496
Включение HTTP-заголовков кеширования	496
Включение сжатия в IIS	501
Минификация и объединение	504
Использование сетей доставки содержимого (CDN)	507
Масштабирование приложений ASP.NET	509
Горизонтальное масштабирование	510
Механизмы масштабирования в ASP.NET	511
Ловушки горизонтального масштабирования	512
В заключение	513
Предметный указатель	514