

Spis treści

O autorze 7

O recenzencie 8

Wstęp 9

Rozdział 1. Co nowego w .NET Core 2 i C# 7? 13

Rozwój frameworka .NET 13

Nowości w .NET Core 2.0 15

Poprawki wydajności 15

Uproszczony system pakietów 17

Ścieżka aktualizacji z .NET Core 1.x do 2.0 17

1. Instalacja .NET Core 2.0 17

2. Zaktualizowanie TargetFramework 17

3. Aktualizacja wersji .NET Core SDK 18

4. Aktualizacja .NET Core CLI 18

Zmiany w ASP.NET Core Identity 18

Odkrywanie .NET Core CLI i szablonów nowych projektów 18

.NET Standard 22

Wersjonowanie .NET Standard 24

Nowości w .NET Standard 2.0 25

Tworzenie biblioteki .NET Standard 27

Co nowego w ASP.NET Core 2.0? 27

ASP.NET Core Razor Pages 27

Uproszczona konfiguracja Application Insights 28

Pule połączeń w Entity Framework Core 2.0 29

Nowe funkcje w C# 7.0 29

Krotki 30

Wzorce 31

Zwracanie referencji 32

Rozszerzone wyrażenia typu expression bodied member 32

Tworzenie lokalnych funkcji 33

Zmienne wyjściowe 33

Asynchroniczna metoda Main 34

Pisanie kodu wysokiej jakości 35

Podsumowanie 38

Rozdział 2. Mechanizmy wewnętrzne .NET Core i mierzenie wydajności 39

Mechanizmy wewnętrzne .NET Core 40

CoreFX 40

CoreCLR 40

Działanie MSIL, CLI, CTS i CLS 41

Jak działa CLR? 42

Od kompilacji do wykonania - pod maską 42

Mechanizm odzyskiwania pamięci (ang. garbage collection) 43

.NET Native i kompilacja JIT 46

Wykorzystywanie wielu rdzeni CPU dla większej wydajności 46

Jak kompilacje w trybie wydania zwiększają wydajność 48

Testy porównawcze aplikacji .NET Core 2.0 49

Poznawanie BenchmarkDotNet 49

Jak to działa 51

Ustawianie parametrów 51

Diagnostyka pamięci z użyciem BenchmarkDotNet 53

Dodawanie konfiguracji 53

Podsumowanie 55

Rozdział 3. Wielowątkowość i programowanie asynchroniczne w .NET Core 57

Wielowątkowość kontra programowanie asynchroniczne 58

Wielowątkowość w .NET Core 60

Zastrzeżenia w wielowątkowości 60

Wątki w .NET Core 61

Synchronizacja wątków 64

Task parallel library (TPL) 70

Wzorce projektowe programowania równoległego 77

Podsumowanie 83

Rozdział 4. Struktury danych i pisanie zoptymalizowanego kodu C# 85

Czym są struktury danych? 86

Notacja wielkiego O do mierzenia wydajności i złożoności algorytmu 88

Logarytmy 90

Wybieranie odpowiedniej struktury danych do optymalizacji wydajności 91

Tablice 91

Listy 92

Stosy 93

Kolejka 94

Listy łączone 95

Słowniki, tablice haszujące i zbiory haszujące 96

Listy generyczne 96

Najlepsze praktyki pisania zoptymalizowanego kodu C# 97

Narzędzia pakowania i rozpakowywania 98

Konkatenacja łańcuchów znaków 100

Obsługa wyjątków 101

For i foreach 102

Delegaty 103

Podsumowanie 104

Rozdział 5. Wytyczne projektowania wydajnych aplikacji .NET Core 105

Zasady kodowania 106

Konwencje nazewnicze 106

Komentarze 107

Jedna klasa na plik 107

Jedna logika na metodę 107

Zasady projektowania 108

KISS (Keep It Simple, Stupid) 108

YAGNI (You Aren't Gonna Need It) 109

DRY (Don't Repeat Yourself) 109

Podział odpowiedzialności 109

Zasady SOLID 110

Buforowanie 121

Struktury danych 122

Komunikacja 122

Zarządzanie zasobami 123

Współbieżność 124

Podsumowanie 125

Rozdział 6. Techniki zarządzania pamięcią w .NET Core 127

Przegląd zarządzania alokacją pamięci 128

Analizowanie mechanizmów wewnętrznych CLR przez debugger SOS w .NET Core 128

Fragmentacja pamięci 132

Unikanie destruktorów 133

Najlepsze praktyki zwalniania obiektów w .NET Core 135

Wstęp do interfejsu IDisposable 135

Czym są niezarządzane zasoby? 135

Wykorzystywanie IDisposable 136

Kiedy implementować interfejs IDisposable? 137

Destruktor i Dispose 138

Podsumowanie 140

Rozdział 7. Stosowanie zabezpieczeń i implementowanie odporności na błędy w aplikacjach .NET Core 141

Wprowadzenie do aplikacji odpornych na błędy 142

Polityki odporności 142

Przechowywanie danych wrażliwych z wykorzystaniem Application Secrets 158

Zabezpieczanie API w ASP.NET Core 161

SSL (ang. Secure Socket Layer) 161

Zapobieganie atakom CSRF (ang. Cross-Site Request Forgery) 163

Wzmacnianie nagłówków bezpieczeństwa 163

Uwierzytelnianie i autoryzacja 168

Uwierzytelnianie 169

Autoryzacja 169

Implementacja uwierzytelniania i autoryzacji z użyciem frameworka ASP.NET Core Identity 169

Podsumowanie 173

Rozdział 8. Architektura mikrousług 175

Architektura mikrousług 176

Zalety architektury mikrousług 177

Standardowe praktyki podczas tworzenia mikrousług 178

Typy mikrousług 179

DDD 179

Manipulowanie danymi w mikrousługach 179

Spójność w różnych scenariuszach biznesowych 180

Komunikacja z mikrousługami 181

Architektura baz danych w mikrousługach 182

Czym jest kompozycja API? 183

CQRS 184

Tworzenie architektury mikrousług w .NET Core 185

Tworzenie przykładowej aplikacji w .NET Core z wykorzystaniem architektury mikrousług 185

Wdrażanie mikrousług w kontenerach Docker 210

Czym jest Docker? 211

Korzystanie z Dockera w .NET Core 212

Uruchamianie obrazów Dockera 214

Podsumowanie 214

Rozdział 9. Monitorowanie wydajności aplikacji z wykorzystaniem narzędzi 215

Kluczowe wskaźniki wydajności aplikacji 216

Średni czas odpowiedzi 216

Apdex 216

Odsetek błędów 216

Liczba żądań 216

Przepustowość/punkty końcowe 217

Wykorzystanie procesora i pamięci 217

Narzędzia i techniki monitorowania wydajności 217

Wstęp do App Metrics 217

Konfigurowanie App Metrics w ASP.NET Core 217

Śledzące oprogramowanie pośredniczące 218

Dodawanie raportów graficznych 220

Podsumowanie 229

Skorowidz 231