PART 1  NETTY CONCEPTS AND ARCHITECTURE ...............1

1 Netty—asynchronous and event-driven  3
   1.1 Networking in Java  4
      Java NIO  6 • Selectors  6
   1.2 Introducing Netty  7
      Who uses Netty?  8 • Asynchronous and event-driven  9
   1.3 Netty’s core components  10
      Channels  10 • Callbacks  10 • Futures  11
      Events and handlers  12 • Putting it all together  13
   1.4 Summary  14
2 Your first Netty application 15

2.1 Setting up the development environment 15
   Obtaining and installing the Java Development Kit 16
   Downloading and installing an IDE 16
   Downloading and installing Apache Maven 16
   Configuring the toolset 17

2.2 Netty client/server overview 17

2.3 Writing the Echo server 18
   ChannelHandlers and business logic 18
   Bootstrapping the server 20

2.4 Writing an Echo client 22
   Implementing the client logic with ChannelHandlers 23
   Bootstrapping the client 24

2.5 Building and running the Echo server and client 26
   Running the build 26
   Running the Echo server and client 28

2.6 Summary 31

3 Netty components and design 32

3.1 Channel, EventLoop, and ChannelFuture 33
   Interface Channel 33
   Interface EventLoop 33
   Interface ChannelFuture 34

3.2 ChannelHandler and ChannelPipeline 34
   Interface ChannelHandler 34
   Interface ChannelPipeline 35
   A closer look at ChannelHandlers 37
   Encoders and decoders 37
   Abstract class SimpleChannelInboundHandler 38

3.3 Bootstrapping 38

3.4 Summary 40

4 Transports 41

4.1 Case study: transport migration 42
   Using OIO and NIO without Netty 42
   Using OIO and NIO with Netty 44
   Non-blocking Netty version 45

4.2 Transport API 46

4.3 Included transports 49
   NIO—non-blocking I/O 49
   Epoll—native non-blocking transport for Linux 51
   OIO—old blocking I/O 51
   Local transport for communication within a JVM 52
   Embedded transport 53
CONTENTS

5 ByteBuf 55
5.1 The ByteBuf API 56
5.2 Class ByteBuf—Netty’s data container 56
   How it works 56 • ByteBuf usage patterns 57
5.3 Byte-level operations 60
   Random access indexing 60 • Sequential access indexing 60
   Discardable bytes 61 • Readable bytes 61 • Writable bytes 62
   Index management 62 • Search operations 63
   Derived buffers 64 • Read/write operations 65
   More operations 69
5.4 Interface ByteBufHolder 69
5.5 ByteBuf allocation 70
   On-demand: interface ByteBufAllocator 70 • Unpooled buffers 71 • Class ByteBufUtil 72
5.6 Reference counting 72
5.7 Summary 73

6 ChannelHandler and ChannelPipeline 75
6.1 The ChannelHandler family 75
   The Channel lifecycle 76 • The ChannelHandler lifecycle 76
   Interface ChannelInboundHandler 77 • Interface ChannelOutboundHandler 78 • ChannelHandler adapters 79
   Resource management 80
6.2 Interface ChannelPipeline 82
   Modifying a ChannelPipeline 83 • Firing events 85
6.3 Interface ChannelHandlerContext 87
   Using ChannelHandlerContext 88 • Advanced uses of ChannelHandler and ChannelHandlerContext 91
6.4 Exception handling 92
   Handling inbound exceptions 92 • Handling outbound exceptions 93
6.5 Summary 95
CONTENTS

7 EventLoop and threading model 96
  7.1 Threading model overview 97
  7.2 Interface EventLoop 98
    I/O and event handling in Netty 4 99 • I/O operations
    in Netty 3 100
  7.3 Task scheduling 100
    JDK scheduling API 100 • Scheduling tasks
    using EventLoop 101
  7.4 Implementation details 103
    Thread management 103 • EventLoop/thread allocation 104
  7.5 Summary 105

8 Bootstrapping 107
  8.1 Bootstrap classes 108
  8.2 Bootstrapping clients and connectionless protocols 109
    Bootstrapping a client 110 • Channel and EventLoopGroup
    compatibility 111
  8.3 Bootstrapping servers 113
    The ServerBootstrap class 113 • Bootstrapping a server 114
  8.4 Bootstrapping clients from a Channel 115
  8.5 Adding multiple ChannelHandlers during a
    bootstrap 117
  8.6 Using Netty ChannelOptions and attributes 118
  8.7 Bootstrapping DatagramChannels 119
  8.8 Shutdown 120
  8.9 Summary 120

9 Unit testing 121
  9.1 Overview of EmbeddedChannel 122
  9.2 Testing ChannelHandlers with EmbeddedChannel 123
    Testing inbound messages 124 • Testing outbound
    messages 126
  9.3 Testing exception handling 128
  9.4 Summary 130
PART 2 CODECS .....................................................................................131

10 The codec framework 133
10.1 What is a codec? 133
10.2 Decoders 134
   Abstract class ByteToMessageDecoder 134 • Abstract class ReplayingDecoder 136 • Abstract class MessageToMessageDecoder 137 • Class TooLongFrameException 138
10.3 Encoders 139
   Abstract class MessageToByteEncoder 139
   Abstract class MessageToMessageEncoder 141
10.4 Abstract codec classes 142
   Abstract class ByteToMessageCodec 142
   Abstract class MessageToMessageCodec 143
   Class CombinedChannelDuplexHandler 146
10.5 Summary 147

11 Provided ChannelHandlers and codecs 148
11.1 Securing Netty applications with SSL/TLS 148
11.2 Building Netty HTTP/HTTPS applications 151
   HTTP decoder, encoder, and codec 151 • HTTP message aggregation 153 • HTTP compression 154
   Using HTTPS 155 • WebSocket 155
11.3 Idle connections and timeouts 158
11.4 Decoding delimited and length-based protocols 160
   Delimited protocols 160 • Length-based protocols 163
11.5 Writing big data 165
11.6 Serializing data 167
   JDK serialization 167 • Serialization with JBoss Marshalling 167 • Serialization via Protocol Buffers 169
11.7 Summary 170

PART 3 NETWORK PROTOCOLS .........................................................171

12 WebSocket 173
12.1 Introducing WebSocket 174
12.2 Our example WebSocket application 174
CONTENTS

12.3 Adding WebSocket support 175
   Handling HTTP requests 175 • Handling WebSocket frames 178 • Initializing the ChannelPipeline 180
   Bootstrapping 182
12.4 Testing the application 183
   What about encryption? 184
12.5 Summary 186

13 Broadcasting events with UDP 187
13.1 UDP basics 188
13.2 UDP broadcast 188
13.3 The UDP sample application 188
13.4 The message POJO: LogEvent 189
13.5 Writing the broadcaster 190
13.6 Writing the monitor 195
13.7 Running the LogEventBroadcaster and LogEventMonitor 198
13.8 Summary 199

PART 4 CASE STUDIES ................................................................. 201

14 Case studies, part 1 203
14.1 Droplr—building mobile services 203
   How it all started 204 • How Droplr works 204
   Creating a faster upload experience 204 • The technology stack 206 • Performance 210 • Summary—standing on the shoulders of giants 211
14.2 Firebase—a real-time data synchronization service 211
   The Firebase architecture 212 • Long polling 213
   HTTP 1.1 keep-alive and pipelining 215 • Control of SslHandler 216 • Firebase summary 218
14.3 Urban Airship—building mobile services 218
   Basics of mobile messaging 219 • Third-party delivery 220
   Binary protocol example 220 • Direct to device delivery 223
   Netty excels at managing large numbers of concurrent connections 224 • Summary—Beyond the perimeter of the firewall 225
14.4 Summary 225
15 Case studies, part 2 226

15.1 Netty at Facebook: Nifty and Swift 226

What is Thrift? 227 • Improving the state of Java
Thrift using Netty 228 • Nifty server design 228
Nifty asynchronous client design 231 • Swift: a faster
way to build Java Thrift service 233 • Results 233
Facebook summary 235

15.2 Netty at Twitter: Finagle 236

Twitter’s growing pains 236 • The birth of Finagle 236
How Finagle works 237 • Finagle’s abstraction 242
Failure management 243 • Composing services 244
The future: Netty 245 • Twitter summary 245

15.3 Summary 245

appendix Introduction to Maven 246

index 265