

# Table of Contents

Foreword.....	ix
Preface.....	xi
<b>1. Quarkus Overview.....</b>	<b>1</b>
Developer-Friendly	1
Integration with Kubernetes	2
Memory and First Response Time	3
A Basic Quarkus Workflow	3
<b>2. Scaffolding.....</b>	<b>5</b>
2.1 Scaffolding a Quarkus Project with Maven	5
2.2 Scaffolding a Quarkus Project with Gradle	7
2.3 Scaffolding a Quarkus Project with the Quarkus Start Coding Website	8
2.4 Scaffolding a Quarkus Project with Visual Studio Code	10
2.5 Live Reloading with Dev Mode	15
2.6 Serving Static Resources	16
<b>3. Developing RESTful Services.....</b>	<b>19</b>
3.1 Creating a Simple REST API Endpoint	19
3.2 Extracting Request Parameters	21
3.3 Using Semantic HTTP Response Status Codes	23
3.4 Binding HTTP Methods	25
3.5 Enabling Cross-Origin Resource Sharing (CORS)	27
3.6 Using Reactive Routes	28
3.7 Intercepting HTTP Requests	30
3.8 Secure Connections with SSL	33

<b>4. Configuration.....</b>	<b>35</b>
4.1 Configuring the Application with Custom Properties	35
4.2 Accessing Configuration Properties Programmatically	39
4.3 Overwriting Configuration Values Externally	40
4.4 Configuring with Profiles	41
4.5 Changing Logger Configuration	42
4.6 Adding Application Logs	43
4.7 Advanced Logging	45
4.8 Configuring with Custom Profiles	48
4.9 Creating Custom Sources	49
4.10 Creating Custom Converters	52
4.11 Grouping Configuration Values	54
4.12 Validating Configuration Values	56
<b>5. Programming Model.....</b>	<b>59</b>
5.1 Marshalling/Unmarshalling JSON	59
5.2 Marshalling/Unmarshalling XML	62
5.3 Validating Input and Output Values	65
5.4 Creating Custom Validations	69
5.5 Validating Objects Programmatically	72
5.6 Injecting Dependencies	74
5.7 Creating Factories	76
5.8 Executing Object Life Cycle Events	78
5.9 Executing Application Life Cycle Events	79
5.10 Using a Named Qualifier	80
5.11 Using Custom Qualifiers	82
5.12 Qualifying and Configuring Annotations	83
5.13 Creating Interceptors	84
5.14 Writing Behavioral Tests	86
5.15 Writing Unit Tests	90
5.16 Creating Mock Objects	93
5.17 Creating Mock Objects with Mockito	94
5.18 Grouping Several Annotations into One with a Meta-Annotation	95
5.19 Executing Code Before or After a Test	97
5.20 Testing the Native Executable	103
<b>6. Packaging Quarkus Applications.....</b>	<b>107</b>
6.1 Running in Command Mode	107
6.2 Creating a Runnable JAR File	109
6.3 Über-JAR Packaging	111
6.4 Building a Native Executable	111

6.5 Building a Docker Container for JAR File	113
6.6 Building a Docker Container for Native File	114
6.7 Build and Dockerize a Native SSL Application	115
<b>7. Persistence</b>	<b>119</b>
7.1 Defining a Datasource	119
7.2 Using Multiple Datasources	120
7.3 Adding Datasource Health Check	121
7.4 Defining Transaction Boundaries Declaratively	122
7.5 Setting a Transaction Context	123
7.6 Programmatic Transaction Control	124
7.7 Setting and Modifying a Transaction Timeout	125
7.8 Setup with Persistence.xml	126
7.9 Setup Without persistence.xml	127
7.10 Using Entities from a Different JAR	127
7.11 Persisting Data with Panache	128
7.12 Finding All Entity Instances with Panache listAll Method	129
7.13 Finding Individual Entities with Panache findById Method	130
7.14 Finding Entities Using Panache Find and List Methods	130
7.15 Obtaining a Count of Entities Using the Panache count Method	131
7.16 Paginating Through Entity Lists Using the Panache page Method	132
7.17 Streaming Results via the Panache Stream Method	132
7.18 Testing Panache Entities	133
7.19 Using a Data Access Object (DAO) or Repository Pattern	134
7.20 Using Amazon DynamoDB	135
7.21 Working with MongoDB	140
7.22 Using Panache with MongoDB	144
7.23 Using Neo4j with Quarkus	146
7.24 Flyway at Startup	149
7.25 Using Flyway Programmatically	150
<b>8. Fault Tolerance</b>	<b>153</b>
8.1 Implementing Automatic Retries	153
8.2 Implementing Timeouts	155
8.3 Avoiding Overloads with the Bulkhead Pattern	156
8.4 Avoiding Unnecessary Calls with the Circuit Breaker Pattern	158
8.5 Disabling Fault Tolerance	161
<b>9. Observability</b>	<b>163</b>
9.1 Using Automatic Health Checks	163
9.2 Creating Custom Health Checks	166
9.3 Exposing Metrics	168

9.4	Creating Metrics	170
9.5	Using Distributed Tracing	176
9.6	Custom Distributed Tracing	182
<b>10.</b>	<b>Integrating with Kubernetes</b>	<b>187</b>
10.1	Building and Pushing Container Images	187
10.2	Generating Kubernetes Resources	191
10.3	Generating Kubernetes Resources with Health Checks	194
10.4	Deploying Services on Kubernetes	195
10.5	Deploying Services on OpenShift	197
10.6	Building and Deploying a Container Image Automatically	200
10.7	Configuring an Application from Kubernetes	202
10.8	Configuring an Application from Kubernetes with Config Extension	204
10.9	Interacting with a Kubernetes Cluster Programmatically	206
10.10	Testing Kubernetes Client Interactions	209
10.11	Implementing a Kubernetes Operator	211
10.12	Deploying and Managing Serverless Workloads with Knative	224
<b>11.</b>	<b>Authentication and Authorization</b>	<b>229</b>
	Quarkus Security Basics	229
11.1	Authentication and Authorization with Elytron Properties File Config	234
11.2	Authentication and Authorization with Elytron Security JDBC Config	237
11.3	Authorization with MicroProfile JWT	241
11.4	Authorization and Authentication with OpenId Connect	248
11.5	Protecting Web Resources with OpenId Connect	251
<b>12.</b>	<b>Application Secrets Management</b>	<b>253</b>
12.1	Storing Data Using Kubernetes Secrets	253
12.2	Store Configuration Secrets Securely with Vault	257
12.3	Cryptography as a Service	260
12.4	Generate Database Password as Secret	263
12.5	Authenticating Services Using Vault Kubernetes Auth	267
<b>13.</b>	<b>Quarkus REST Clients</b>	<b>273</b>
13.1	Using the JAX-RS Web Client	273
13.2	Using the MicroProfile REST Client	276
13.3	Implementing a CRUD Client	280
13.4	Manipulating Headers	282
13.5	Using REST Client for Multipart Messages	285
13.6	Using REST Client to Configure SSL	287

<b>14. Developing Quarkus Applications Using Spring APIs.....</b>	<b>291</b>
14.1 Using Spring Dependency Injection	291
14.2 Using Spring Web	294
14.3 Using Spring Data JPA	297
14.4 Using Spring Security	299
14.5 Using Spring Boot Properties	301
<b>15. Working with a Reactive Programming Model.....</b>	<b>305</b>
15.1 Creating Async HTTP Endpoints	305
15.2 Streaming Data Asynchronously	307
15.3 Using Messaging to Decouple Components	307
15.4 Reacting to Apache Kafka Messages	310
15.5 Sending Messages to Apache Kafka	313
15.6 Marshalling POJOs into/out of Kafka	314
15.7 Using Kafka Streams API	316
15.8 Using AMQP with Quarkus	322
15.9 Using MQTT	323
15.10 Query Using Reactive SQL	324
15.11 Insert Using Reactive SQL Client	327
15.12 Using the Reactive MongoDB Client	328
15.13 Using the Reactive Neo4j Client	330
<b>16. Additional Quarkus Features.....</b>	<b>333</b>
16.1 Creating Templates with the Qute Template Engine	333
16.2 Rendering HTML Using Qute	335
16.3 Changing the Location of Qute Templates	337
16.4 Extending Qute Data Classes	338
16.5 Describing Endpoints with OpenAPI	339
16.6 Customizing OpenAPI Spec	341
16.7 Sending Email Synchronously	346
16.8 Sending Email Reactively	349
16.9 Creating Scheduled Jobs	351
16.10 Using Application Data Caching	353
<b>A. Minikube.....</b>	<b>357</b>
<b>B. Keycloak.....</b>	<b>359</b>
<b>C. Knative.....</b>	<b>363</b>
<b>Index.....</b>	<b>365</b>