

O'REILLY®

Programming Kubernetes

Developing Cloud Native Applications



Michael Hausenblas &
Stefan Schimanski

Table of Contents

Preface.....	vii
1. Introduction.....	1
What does Programming Kubernetes Mean?	1
A Motivational Example	3
Extension Patterns	3
2. Kubernetes API basics.....	5
The API Server	5
Terminology	8
The Kubernetes API	9
Using the API from the Command Line	10
3. Basics of client-go.....	13
The Repositories	13
The Client Library	13
Kubernetes API Types	15
API Machinery	15
Creating and Using a Client	16
Versioning and Compatibility	18
API Versions and Compatibility Guarantees	20
Kubernetes Objects in Go	21
TypeMeta	23
ObjectMeta	25
Spec and Status	26
Client sets	26
Client options	30
Informers and Caching	31

API Machinery in Depth	34
Kinds	35
Resources	35
REST Mapping	35
Scheme	36
Vendoring	37
Glide	37
Dep	38
4. Using Custom Resources.....	41
Discovery Information	42
Type definitions	44
Advanced Features of Custom Resources	46
Validating Custom Resources	46
Short Names and Categories	47
Printer Columns	48
Subresources	49
A Developers View on Custom Resources	53
Dynamic Client	54
Typed Clients	55
Controller-runtime Client of Operator SDK and Kubebuilder	59
5. Automating Code Generation.....	63
Why Code Generation	63
Calling the Generators	63
Controlling the generators with tags	65
Global Tags	66
Local Tags	67
deepcopy-gen tags	67
runtime.Object and DeepCopyObject	68
client-gen tags	69
informer-gen and lister-gen	70
Further Material	70
6. Controllers and Operators.....	71
The Controller Loop	71
Optimistic Concurrency	72
Edge Versus Level Driven Triggers	72
7. Solutions For Writing Operators.....	75
Kubebuilder	75
The Operator SDK	80

Metacontroller	81
Other Approaches	83
8. Custom API Servers.....	85
Use Cases For Custom API Servers	85
The Architecture: Aggregation	87
API Services	88
Inner Structure of a Custom API Server	91
Delegated Authentication and Trust	92
Delegated Authorization	93
Writing Custom API Servers	95
Options and Config Pattern and Startup Plumbing	95
The First Start	102
Internal Types and Conversion	103
Writing the API Types	107
Roundtrip Testing	108
Validation	110
Registry and Strategy	112
API Installation	115
Admission	120
Deploying Custom API Servers	126
Certificates and Trust	127
Sharing Etcd	127
9. Cloud Native Programming Languages.....	129
The Approach	129
Metaparticle	131
Ballerina	131
Pulumi	131
10. Packaging.....	133
Packaging: The Challenge	133
Helm	133
Ksonnet	134
Other Packaging Options	134
11. Advanced Topics.....	137
Custom Resources Versioning	137
Conversion	138
Admission Webhooks	138
Production-ready Deployments Of Custom Resources	138
Getting The Permissions Right For Custom Resources	139

Performance Considerations For Custom Resources	139
Observability And Instrumentation	139
A. Resources.....	141
Index.....	143

A

- admission configuration, 124
- admission plugin, 120
- admission plugin initializer, 124
- admission webhook, 86
- aggregation, 85
 - API aggregation, 87
- alpha version, 20
- API
 - invoke via command line, 10
 - proxy, 10
- API group, 18
- API Machinery, 15
- API server, 5
 - API group, 8
 - kind, 8
 - resource, 9
 - version, 8
- apixtensions-apiserver, 41, 91
- APIService, 126
- apps group, 18
- auditing, 92, 122

B

- Ballerina, 131
- bearer token, 93
- beta version, 20
- builder pattern, 17
- burst, 31

C

- CA bundle, 127
- category, 48
- client set, 26

- client-gen, 37, 53, 57
- client-set, 17
- clientset, 57
- CNPL (cloud native programming language), 129
- cobra command, 100
- code examples from this book, x
- cohabitation, 86
- control plane, 5
 - API server, 5
 - controller manager, 5
 - etcd, 5
 - scheduler, 5
- control vs. convenience, 129
- controller
 - loop, 71
 - optimistic concurrency, 72
 - triggers, 72
- controller-runtime, 53
- controllers
 - definition, 71
- conversion, 21
- core group
 - legacy group, 24
- CRDs, 107
- custom API server
 - aggregated API server, 85
- CustomResourceDefinition
 - CRD, 42
- CustomResources
 - CRDs, 17
 - CRs, 41

D

- deep-copy, 57
- defaulting, 106
- DeferredDiscoveryRESTMapper
 - RESTMapper, 36
- delegated authentication, 93
- delegated authorization, 93
- discovery endpoint, 102
- dynamic client, 53

E

- edge-driven trigger, 72
- etcd, 95
- etcd operator, 127
- etcdproxy-controller, 128
- event, 29
- external version, 105

F

- field selector, 28
- fuzzer, 109

G

- GA, 20
- generic registry, 112
- Git, viii
- Go (build system), viii
- graceful termination, 101
- GroupVersion, 27
- GroupVersionKind
 - GVK, 22, 35
- GroupVersionResource
 - GVR, 35
- GVR
 - GroupVersionResource, 54

H

- Helm, 133
 - chart, 133
- HTTP/2, 86

I

- informers, 31
- internal clients, 27
- internal version
 - hub version, 103

J

- JSON, 9

K

- kind, 35
- ksonnet, 134
- kube-aggregator, 87
- kube-apiserver, 46, 87
- kubebuilder, 53, 59
- Kubebuilder, 75
- kubeconfig, 17
- kubectrl, 17, 83
- Kubernetes
 - releases, viii
- Kubernetes API, 9

L

- label selector, 28
- level-driven trigger, 72

M

- man-in-the-middle, 89
- manifests
 - GitHub repository, x
- master node, 5
- meta/v1, 16
- Metacontroller, 81
- Metaparticle, 131
- mutating admission webhooks, 120

N

- NameGenerator, 114

O

- ObjectMeta, 111
- ObjectTyper, 114
- OpenAPI, 100
- OpenAPI Schema, 46
- OpenAPI), 48
- OpenShift, 86, 86
- operator
 - alternatives, 83
- Operator SDK, 59, 80
- operator-sdk
 - operator-framework, 53
- operators
 - definition, 71

optimistic concurrency, 51
options and config pattern, 95

P

package management, viii
packaging, 133

- Ansible, 134
- Chef, 134
- custom, 134
- Puppet, 134
- sed, 134
- YAML, 133

post start hook, 100
Protobuf

- protocol buffers, 17

protobuf

- protocol buffers, 24

Protocol buffer

- protobuf, 30

Protocol Buffer

- protobuf, 85

Protocol Buffers, 9
Pulumi, 131

Q

QPS, 31

R

RBAC

- role based access control, 50, 93

recommended options, 96
relist period, 32
request header, 92
request header client CA, 92
resource, 35
REST, 112
rest config, 17
REST mapping, 35
RESTMMapper, 27, 35
roundtrippable, 107
runtime.Object, 36

S

scale subresource, 51, 70
scheme, 25, 36
semantic versioning

- semver, 14, 19

server-side printing, 48
shared informer factory, 32
short name, 47
spec-status split, 50
status subresource, 28, 50

- subresource, 69

storage version, 21, 104
strategy, 113
subject access review

- SubjectAccessReview
 - SAR, 93

SubjectAccessReview, 92, 112
subresource, 49
subresources, 31

T

throttling

- rate limiting, 30

timeout, 30
TokenAccessReview, 93
TokenReview, 92
typed client, 53
TypeMeta, 23, 55, 55

U

unstructured.Unstructured, 54
UserAgent, 30

V

validating admission webhooks, 120

W

watch, 29
webhook admission plugins

- admission webhook, 120

websocket, 86

About the Authors

Michael Hausenblas is a developer advocate for containers at AWS. His background is in large-scale data processing and container orchestration and he is experienced in advocacy and standardization at W3C and IETF. Before Amazon, Michael worked at Red Hat, Mesosphere, MapR, and two research institutions in Ireland and Austria. He contributes to open source software mainly using Go, blogs, writes books, and hangs out on Twitter too much.

Stefan Schimanski is a principal software engineer for Go, Kubernetes, and OpenShift at Red Hat. His focus is the Kubernetes API server, especially the implementation of CustomResourceDefinitions, API Machinery in general and the publishing of the Kubernetes staging repositories client-go, apimachinery, api, etc. Before Red Hat, Stefan worked at Mesosphere on Marathon, Spark and their Kubernetes offering and before as freelancer and consultant in high availability and distributed systems. In a former life Stefan did research in Mathematical Logic about constructive mathematics, type systems and lambda calculus.