Serverless Frameworks for Kubernetes_

Gustavo Marin

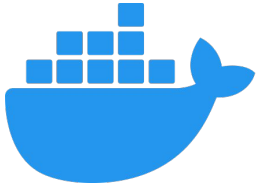
Software Developer
@Intelygenz

guumaster

SERVERLESS FOR K8S_

Introduction

What You Need To Know



docker

kubernetes

Amazon Lambda

(not) A Crash Course

Set of tools to pack and deploy
OS + Software + Your Code

A way to manage lots of docker containers with yaml files

A service where you can deploy functions into AWS infrastructure

SERVERLESS FOR K8S_

The Frameworks

Our Picks

Selection Criteria

- Available Docs
- Active Community
- Number of Solved Issues
- Github Stars

openfaas.com

kubeless.io

fission.io

nuclio.io

PYCON ES ALICANTE 2019

Features

Common

- ➡️ Support multiple languages (Python, NodeJS, Go, etc)
- ➡️ Abstracts away containers and infrastructure
- ➡️ Binds HTTP requests to function execution
- ➡️ Simplify deployment, routing, scalability, availability
- ➡️ A nice CLI

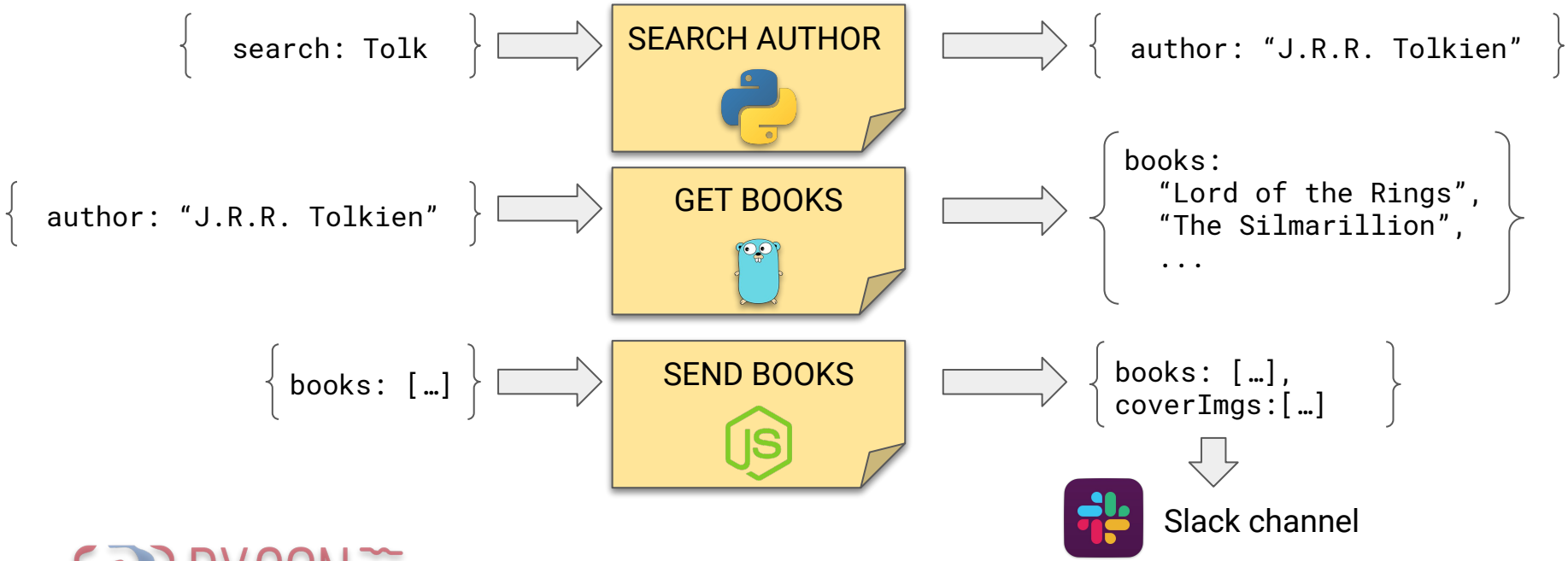
Unique

- ➡️ Types of events handlers (Kafka, Redis, Webhooks, etc)
- ➡️ Underlying architecture and monitoring
- ➡️ Function interface (context, logging, etc)
- ➡️ CLI features and usage

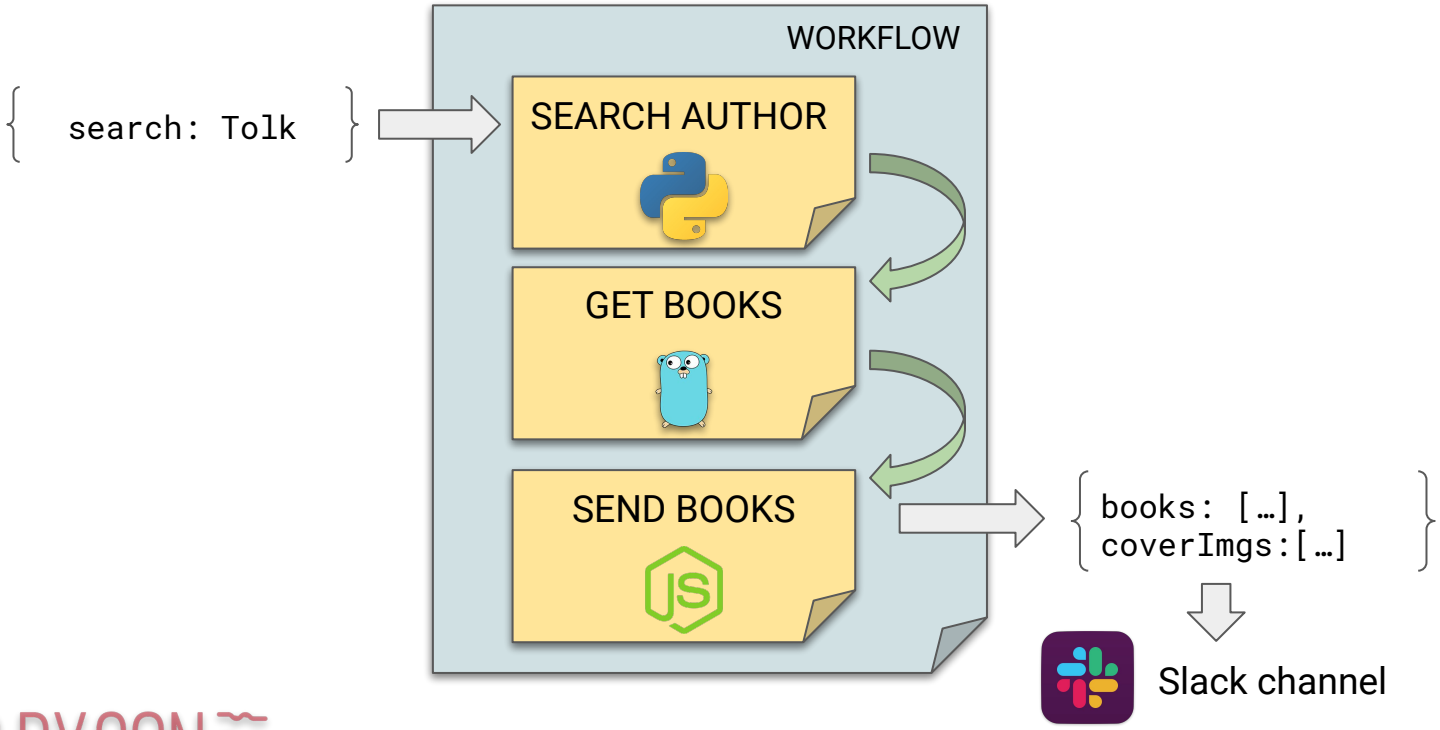
SERVERLESS FOR K8S_

The Proof of Concept

Proof Of Concept



Proof Of Concept



Source Code

```python
import os
import json

authors_file = os.path.dirname(os.path.realpath(__file__)) + '/authors.json'
authors_db = json.load(open(authors_file))

def get_author(context, event):
    input = event.body

    query = input.get('author')

    if not query:
        return ''

    res = [author for author in authors_db if query.lower() in author.lower()]

    response = {
       "author": res[0] if len(res) else ''
    }
    print('RES', response)

    return response
```

THE POC_

Source Code

```go
func Handler(context *nuclio.Context, event nuclio.Event) (interface{}, error) {
    var input map[string]string
    _ = json.Unmarshal(event.GetBody(), &input)

    author := input["author"]
    // author check omitted

    jsonFile, _ := os.Open("books.json")
    defer jsonFile.Close()

    byteValue, _ := ioutil.ReadAll(jsonFile)

    // Book struct omitted for brevity
    var response []Book
    var filtered []Book

    _ = json.Unmarshal(byteValue, &response)

    for _, book := range response {
        if isValueInList(author, book.Authors) {
            filtered = append(filtered, book)
        }
    }

    res, _ := json.Marshal(map[string]interface{}{
        "author": author,
        "books":  filtered,
    })

    return nuclio.Response{
        StatusCode:  200,
        ContentType: "application/json",
        Body:        res,
    }, nil
}
```

PYCON ES ALICANTE 2019

THE POC_

Source Code

```
const format = require('date-format')
const slack = require('slack-notify')(MY_SLACK_WEBHOOK_URL)
const toDate = str => format('yyyy-MM-dd', new Date(str))

exports.handler = function(context, event) {
    var req = JSON.parse(event.body);
    context.logger.infoWith('Body', { req });

    const attachments = req.books.map(book => {
        const published = toDate(book.publishedDate.$date)
        return {
            fallback: '',
            fields: [
                { title: 'Title', value: book.title, short: true },
                { title: 'Authors', value: book.authors.join(", "), short: true },
                { title: 'Published', value: published, short: true },
            ],
            "image_url": book.thumbnailUrl,
            "thumb_url": book.thumbnailUrl,
            "footer": "Book Cover",
        }
    })

    slack.send({
        channel: "#kst-test",
        text: `[Nuclio] Books by *${req.author}*`,
        username: 'SearchBot',
        attachments
    })

    context.callback({ status: "done" });
}
```

THE POC_

Source Code

```
const getAuthor = query => fetch( 'http://search-author:8080', opts)
const getBooks = author => fetch( 'http://search-books:8080', opts)
const sendResultNotification = ({ author, books }) => fetch( 'http://send-books:8080', opts)

exports.handler = async function(context, event) {
  try {
    const input = JSON.parse(event.body)

    const authorResponse = await getAuthor(input.author)
    if (!authorResponse || !authorResponse.author) {
      context.logger.infoWith('NO AUTHOR FOUND ', authorResponse)
      return context.callback({ status: 'error' })
    }
    context.logger.infoWith('AUTHOR', authorResponse)

    const authorBooks = await getBooks(authorResponse.author)
    context.logger.infoWith('BOOKS', typeof authorBooks, authorBooks)

    if (!authorBooks || !authorBooks.books.length) {
      context.logger.infoWith('NO BOOKS FOUND ', authorResponse)
      return context.callback({ status: 'error' })
    }

    await sendResultNotification(authorBooks)
    context.callback({ status: 'done' })
  } catch (err) {
    context.logger.infoWith('Exception', err)
    context.callback({ status: 'error', err })
  }
}
```

Our Experience

First Impressions

				
Open/Closed Issues	50/550	200/400	150/400	50/250
Github Stars	★ 15.6k	★ 4.6k	★ 5.0k	★ 2.9k
Installation				
CLI Usage				
Available Docs				
Debugging				
Python (with deps)				

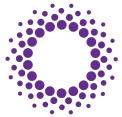
Some Remarks

- ✓ Useful CLI
- ✓ Single yaml multiple functions

- ✗ Bad I/O handling
- ✗ Broke with heavy load

- ✓ Based on CRD
- ✓ Simple Docker image

- ✗ CLI is too verbose
- ✗ Hard to find internal URL
- ✗ Customization is hard



- ✓ Environments
- ✓ Have a Workflow concept

- ✗ Workflow doesn't work
- ✗ In the middle of a rewrite
- ✗ Outdated docs

- ✓ Rich context
- ✓ Nice logs
- ✓ Useful CLI

- ✗ Maybe too much focus on IDE

PYCON ES ALICANTE 2019

SERVERLESS FOR K8S_

Conclusions

¿Can I Use them Today?

Things that needs improvement

- Low maturity level, not production ready
- Hard to debug a function
- Long deploy lifecycle
- Not easy to see logs
- Not a good Developer Experience

¿Why Serverless On K8s?

Interesting Benefits

- ✓ Focus on Code and not Infra
- ✓ Choose the right language for each task
- ✓ All CLI and IDE are very useful
- ✓ No vendors lock-in
- ✓ Fully portable to different environments

PYCON ES ALICANTE 2019

Try It Yourself

openfaas.com

nuclio.io

```
print("done")_
```

guumaster

INTELYGENZ