

# An intro to ko, developing the knative way

---

Andrea Frittoli  
Developer Advocate  
andrea.frittoli@uk.ibm.com  
@blackchip76

# Microservice(s) in Go

# From local...

- A few lines of code
- Build and run locally
- [github.com/afrittoli/examples/ms/go/helloworld](https://github.com/afrittoli/examples/ms/go/helloworld)

```
package main

import (
    "flag"
    "fmt"
    "net/http"
)

func main() {
    hwPort := flag.Int("port", 8080, "Listening port numbers")
    flag.Parse()

    http.HandleFunc("/", func(w http.ResponseWriter,
        r *http.Request) {
        fmt.Fprintf(w, "{ \"hello\": \"%s\" }", r.URL.
            Path)
    })

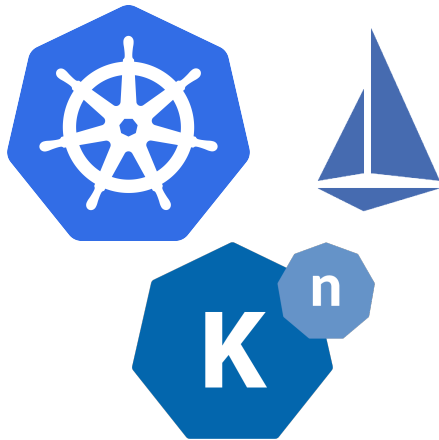
    http.ListenAndServe(fmt.Sprintf(":%d", *hwPort),
        nil)
}
```

```
#!/bin/bash
```

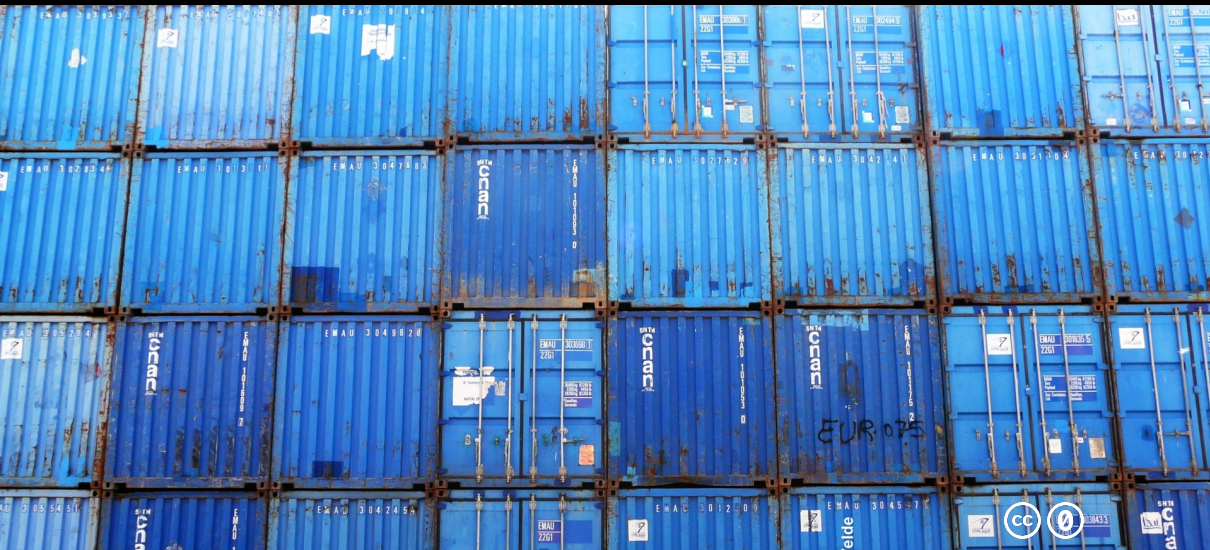
```
go build
./helloworld --port 8080 &
```

## ...to the Cloud

- Scaling
- Resiliency
- Security



Something is missing



## Containers!

- One to compile
- One to run
- A single Dockerfile for both

```
# Start by building the application.
FROM golang:1.8 as build

WORKDIR /go/src/github.com/afrittoli/go_helloworld
ADD . /go/src/github.com/afrittoli/go_helloworld

RUN go-wrapper download
    # "go get -d -v ./..."
RUN go-wrapper install

# Now copy it into our base image.
FROM gcr.io/distroless/base
COPY --from=build /go/bin/go_helloworld /helloworld
CMD ["/helloworld"]
```

Build the image  
Tag the image  
Push the image to the registry  
Update the image version

- Kubernetes manifests
- Helm chart values

```
#!/bin/bash

# Define variables
TAG=$(git log -1 --pretty=%H)
REGISTRY=registry.ng.bluemix.net/knative

# Build and push the image
docker build .
docker tag ${REGISTRY}/go_helloworld:${TAG}
docker push ${REGISTRY}/go_helloworld:${TAG}
```

# Getting Started with Ko

# Invisible containers

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: helloworld_ms
spec:
  replicas: 1
  selector:
    matchLabels:
      app: helloworld_ms
  template:
    metadata:
      labels:
        app: helloworld_ms
    spec:
      containers:
        - name: helloworld_ms
          # This is the import path for the Go binary to build and run.
          image: github.com/afrittoli/examples/ms/go/helloworld
          ports:
            - containerPort: 8080
```



# Installing

Install `go`  
and setup your environment:

```
# Sources go in ${GOPATH}/src
# Binaries go in ${GOPATH}/bin

export GOPATH=<root of GO projects>
export GOBIN=${GOPATH}/bin # This is the default

export PATH=${GOBIN}:${PATH}
```

Install `ko`:

```
go get -u github.com/google/go-containerregistry/cmd/ko
```

Congratulations!  
Ko is now available:

```
$ ko
Rapidly iterate with Go, Containers, and Kubernetes.
```

```
Usage:
  ko [flags]
  ko [command]
```

Available Commands:

```
  apply      Apply the input files with image references
              resolved to built/pushed image digests.
  delete     See "kubectl help delete" for detailed usage.
  help       Help about any command
  publish     Build and publish container images from the
              given importpaths.
  resolve     Print the input files with image references
              resolved to built/pushed image digests.
```

Flags:

```
-h, --help  help for ko
```

Use "`ko [command] --help`" for more information about a command.

# Container Registry

Import paths are hashed by default:

```
github.com/afrittoli/examples/ms/go/helloworld  
⇒ $KO_DOCKER_REPO/helloworld-<path-hash>
```

Import paths can be preserved  
as long as the registry supports it:

```
github.com/afrittoli/examples/ms/go/helloworld  
⇒ $KO_DOCKER_REPO/github.com/afrittoli/examples/ms/  
go/helloworld
```

Ko can use different container registries.

## Using IBM Cloud Container Registry:

```
# Login to IBM Cloud  
ibmcloud login  
ibmcloud cr login  
  
# Obtain the CR endpoint  
ibmcloud cr info  
  
# Define a namespace  
ibmcloud cr namespace-add knative  
  
# Create a write token  
ibmcloud cr token-add --readwrite --non-expiring --  
description "ko token"  
  
# Store credentials in ~/.docker/config.json  
docker login -u token -p <token> <endpoint>
```

Publish, Resolve, Apply,  
Delete

# Publish

# Resolve

# Apply & Delete

# Ko and Knative

# Developing Knative



# Knative & Ko

# Q&A

# Thank You! Questions?

Andrea Frittoli  
Developer Advocate  
andrea.frittoli@uk.ibm.com  
@blackchip76

—