# C++ Web Servers and APIs

Silicon Valley Code Camp 13/14 October 2018

bit.ly/

rockncoder-2018-svcc-cpp

# Troy Miles

- Troy Miles
- Nearly 40 years of experience
- Coder, Author, Speaker
- LinkedIn Learning Author
- rockncoder@gmail.com
- @therockncoder

# Agenda

- Docker and C++
- Boost Beast & The Beast HTTP Server
- Creating a build container
- Running locally
- Deploy a container Heroku

# C++ Web Frameworks

| Framework | Notes |
|---|---|
| Civetweb | C/C++ embeddable web server |
| CppCMS | Free High Perf. web development |
| Crow | C++ micro web framework |
| Cutelyst | C++ web framework built on top of Qt |
| Kore | Ultra fast/flexible web server developed in C |
| libOnion | library to create web servers in C |
| Iwan | experimental scalable high perf. HTTP server |
| QDjango | web framework in C++ on top of Qt |
| TreeFrog | hi-speed full stack web app. in C++ on Qt |
| Wt | C++ library for developing web apps |

# C++ Web Frameworks

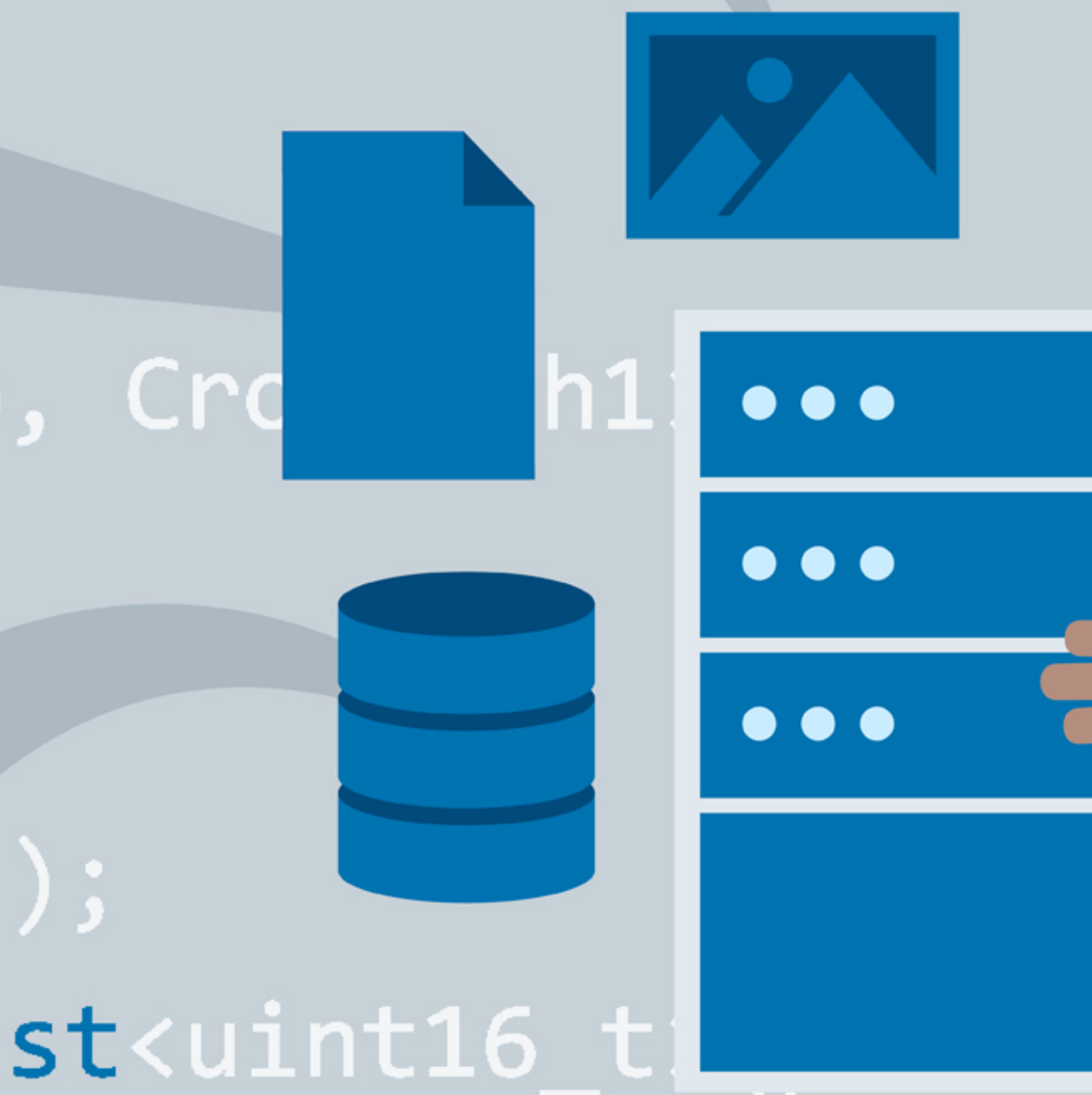
| Framework | Notes |
|---|---|
| Civetweb | C/C++ embeddable web server |
| CppCMS | Free High Perf. web development |
| Crow | C++ micro web framework |
| ~~Cutelyst~~ | ~~C++ web framework built on top of Qt~~ |
| ~~Kore~~ | ~~Ultra fast/flexible web server developed in C~~ |
| ~~libOnion~~ | ~~library to create web servers in C~~ |
| ~~lwan~~ | ~~experimental scalable high perf. HTTP server~~ |
| ~~QDjango~~ | ~~web framework in C++ on top of Qt~~ |
| ~~TreeFrog~~ | ~~hi-speed full stack web app. in C++ on Qt~~ |
| ~~Wt~~ | ~~C++ library for developing web apps~~ |


CROW

```
in(int argc, char* argv[]) {
::SimpleApp app;


_ROUTE(app, "/
](){
return "<div><h1>Hello, Cro      h1
;


* port = getenv("      ");
16_t iPort = static_cast<uint16_t
 << "PORT = " << iPort << "\n";
```



# Is Crow Dead?

- Last commit Dec. 27, 2017
- List of issues growing
- List of pull request growing too

# Is this project still maintained? #308

Open **jimlloyd** opened this issue on Jul 15 · 2 comments

**jimlloyd** commented on Jul 15

This project looks like a very good fit for what I want, but it shows signs of being abandoned.

If it is not abandoned, I wonder if the author and/or active contributors have given any thought to adapting Crow to use Boost::beast rather than using Boost::asio directly.

Also, if the project is maintained, it seems like there should be another release. The one release tagged `v0.1` is several months older than the most recent commit.

👍 2

**slyshykO** commented on Jul 17

Same as #227

**moneroexamples** commented on Aug 7

My guess, its not. Thus for my next project I will be looking for alternatives to crow and probably when find time, to replace crow in my current ones.

Recently found this one ( https://github.com/0xdead4ead/beast_http_server ). Maybe worth checking it out?

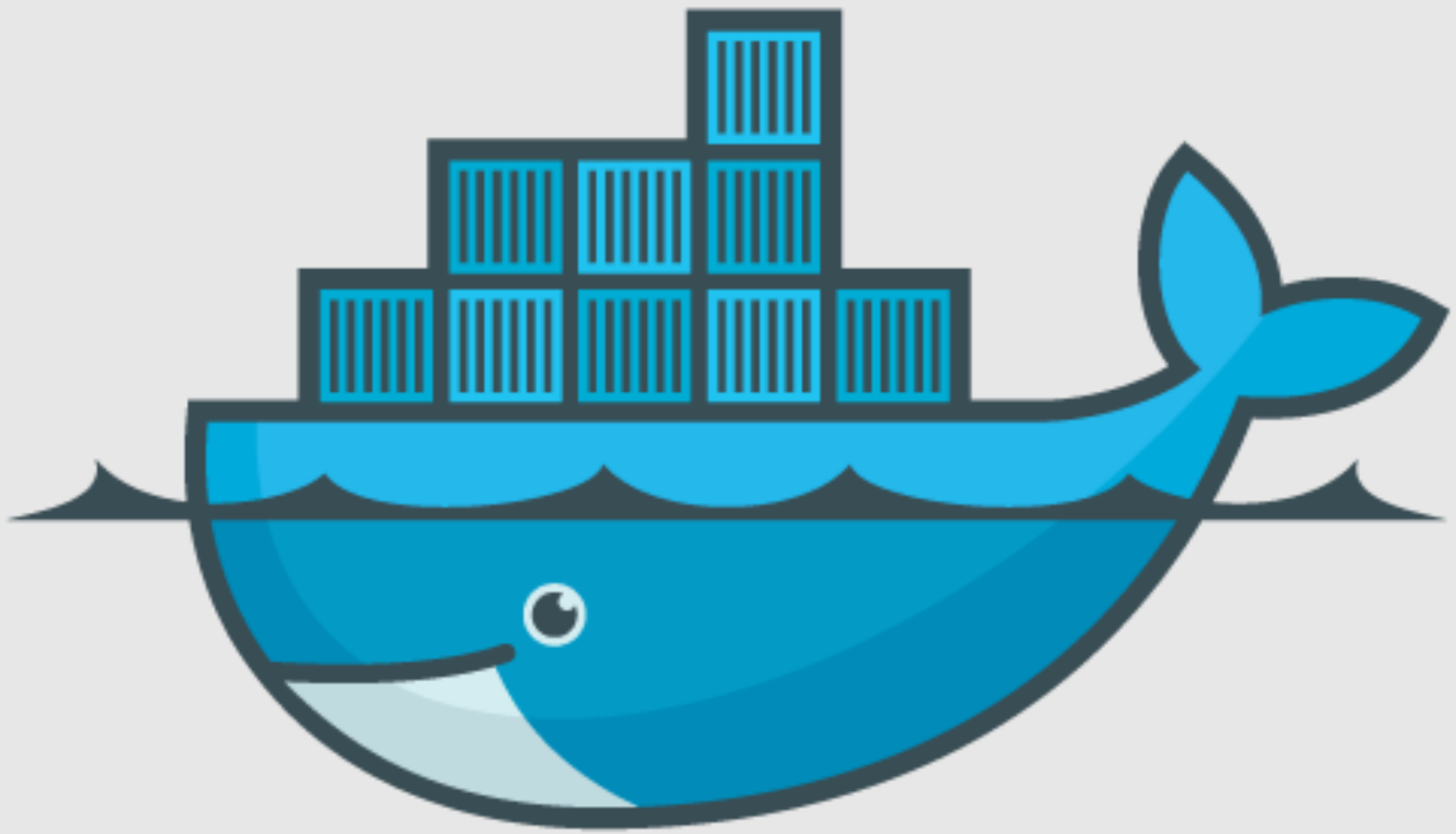
As
Nc
La
Nc
Pr
Nc
Mi
Nc
Nc
Yo
frc
3 p

# Boost Beast

- Added to Boost in 2016
- Created by Vinnie Falco
- Header-only
- But low-level, not a server or a client
- https://www.youtube.com/watch?v=uJZgRcvPFwI

# Beast HTTP Server

- Uses C++14 and Boost Beast
- Header-only
- Begun Jul 24, 2018
- Last commit Oct 8, 2018
- https://github.com/0xdead4ead/beast_http_server



docker

# Docker

- Operating system level virtualization
- Also know as containerization
- Initial release Mar 13, 2013
- Apache License 2.0

# What Docker Brings?

- Solves development problems
- Solves deployment problems

# Development Problems

- Polluting you dev box
- Installing the right libraries

# Deployment Problems

- It doesn't work in production
- Deploying too much stuff
- Knowing if it is safe to upgrade

# Beast & Docker

Dockerfile

```
FROM ubuntu:18.10

RUN apt-get -qq update && apt-get -qq upgrade
WORKDIR /usr/src
RUN apt-get -qq install libboost-all-dev cmake build-essential \
  libtcmalloc-minimal4 libssl-dev openssl git && \
  ln -s /usr/lib/libtcmalloc_minimal.so.4 /usr/lib/libtcmalloc_minimal.so && \
git clone https://github.com/nlohmann/json.git && \
git clone https://github.com/nlohmann/fifo_map.git && \
git clone https://github.com/0xdead4ead/beast_http_server.git && \
mkdir -p /usr/include/nlohmann/json/single_include && \
mkdir -p /usr/include/nlohmann/fifo_map && \
cp -r json/single_include /usr/include/nlohmann/json && \
cp -r fifo_map/src /usr/include/nlohmann/fifo_map
```

# Creating a volume

- `docker run -v <host>:<container> -ti <image> bash`
- `-v: the volume option`
- `host:container`
- `-ti: terminal interactive mode`
- `image: the name of the docker image to launch`
- `bash: the name of the app to run`

# Viewing the examples

- `docker build -t beastbox .`
- `docker run -v /beast:/usr/beast beatbox:latest bash`
- `docker ps`
- `docker cp <id>:/usr/src/beast_http_server ~/Desktop/beast`

# Building the examples

- cd /usr/beast/beast_http_server
- mkdir build && cd build
- cmake ..
- make

# Running an example

- examples/ex1_server/ex1_server

- ctrl-C to exit out of the server

# Fixing the example

- Address must be 0.0.0.0
- Port should be read from an environment variable
- To rebuild, run make again

# Exposing a Port

- -p 8080:80
- Translates the host port 8080 to the container port 80
- -e PORT:8080
- Creates an environment variable named PORT with the value “8080” (a string)


HEROKU

# Heroku

- Cloud Platform as a Service
- One of the first cloud platforms
- Available since June 2007
- Supports Ruby, Java, Node, Scala, Clojure, Python, PHP, and Go
- Applications run in Dynos

# Containerize It

- docker ps

- docker cp . <id>:/usr/beastweb

# Deploy It

- heroku login
- heroku container:login
- heroku create
- docker build -t hello_beast .
- heroku container:push web -a floating-headland-31085
- heroku container:release web -a floating-headland-31085
- heroku open -a floating-headland-31085

# Seeing the Logs

- heroku logs -a floating-headland-31085

# Useful Links

- https://www.docker.com/
- https://www.heroku.com/home
- https://www.linkedin.com/learning
- https://github.com/0xdead4ead/beast_http_server
- https://github.com/ipkn/crow
- https://github.com/boostorg/beast

bit.ly/

rockncoder-2018-svcc-cpp

# Summary

- There are C++ web servers
- The payoff is fast initial response and more bang for the buck
- Docker makes things easier, but debugging is challenging
- You can deploy your containers to Heroku for free