

# Getting Started With Docker!



<http://bit.ly/2ErpzDa>

@BMcCoy04







# Who am I?

Name: Bryan

Occupation: Web Dev

Employer: Bcbsla

Twitter: @BMcCoy04

Email: [BryanMcCoy04@gmail.com](mailto:BryanMcCoy04@gmail.com)

Github: [github.com/bmccoy04](https://github.com/bmccoy04)

Slides: <http://bit.ly/2ErpzDa>



# Who likes?

- .Net Core
- Command Line commands
- Visual Studio Code
- Linux environments in general
- This demo is full of these things!



# What is Docker?

- It's a container platform to **build**, secure, and maintain applications on premise or in the cloud.
- It's made of up layered containers that run in the kernel of the host OS.
- Starts with a base OS system such as Windows Nano Server, Windows Server Core, or some flavor of Linux.

\*Everything in this demo will run a linux base image.



# Why should I use docker?

- Scalability, Scalability, Scalability!!!
- Easier/Faster deployments
- Cost Savings(?)
- Platform Agnostic
- Greater Separation of Concerns
- Containerized Applications
- Faster on-boarding
- And finally Scalability

**Who Doesn't love Microservices!?!?!?**

The background features a series of dark grey, 3D-style rectangular blocks arranged in a descending staircase pattern from the top right towards the bottom left. A single light green block is positioned above the text, and a single blue block is positioned below it, both appearing to be part of the staircase structure.





# Who should use docker?

- Software Developers/Engineers
- System Admins
- DevOps Engineers
- DBAs?



# Why I use Docker!

- Developing .Net Core apps across several platforms (MacOS, Windows 10, Ubuntu, Debian, and CentOS)
- Evaluating software
- Proof of concept apps
- Avoid installing applications



So what do I need to know?

# Docker lingo



# Docker Lingo

## Image:

What you build to run in a container (think VM). Can but pulled from a public repo or custom built.



# Docker Lingo

## Container:

The thing that runs your image. Kind of like a VM but runs in the host OS Kernel not in a Virtualizer.



# Docker Lingo

## Repository:

A collection of images. Can be public or private and hosted on premise or in a cloud solution.



# Docker Lingo

## Registry:

A service that contains repositories. Can be public or private.



# Docker Lingo

## Dockerfile:

File that contains information on how to build your image.





# Docker Lingo

## Tags:

A way to name and version your images.



# Docker Lingo

## Orchestrator:

VERY simply put, it's what controls and monitors your running containers.

(WTH is Kubernetes?)



So what do I need to know? Cont...

# Command line commands



So what do I need to know? Cont...

Build:

```
docker build -t webapp-dev:demo1 .
```



So what do I need to know? Cont...

Run:

```
docker run --rm -p 5001:80 --name web-dev-demo  
web-dev:demo1
```



So what do I need to know? Cont...

Container start/stop:

```
docker container stop web-dev-demo
```



## So what do I need to know? Cont...

### Tag:

```
docker tag webapp-dev:demo1 bmccoy04/coreDemo:part2
```

```
(docker tag "appname/image" "username"/"repo": "tag used for  
versioning")
```



So what do I need to know? Cont...

Push:

```
docker push bmmcoy04/coreDemo:part2
```





So what do I need to know? Cont...

Prune:

```
docker image prune -a
```



## So what do I need to know? Cont...

```
docker run -e 'ACCEPT_EULA=Y' -e \  
'MSSQL_SA_PASSWORD=YourStrong!Passw0rd' \  
-p 1433:1433 --name sql1 \  
-d microsoft/mssql-server-linux:2017-latest
```



# Example Dockerfile

```
FROM microsoft/aspnetcore-build:2.0 AS build-env  
WORKDIR /app
```

```
# Copy csproj and restore as distinct layers  
COPY *.csproj ./  
RUN dotnet restore
```

```
# Copy everything else and build  
COPY ./ ./  
RUN dotnet publish -c Release -o out
```

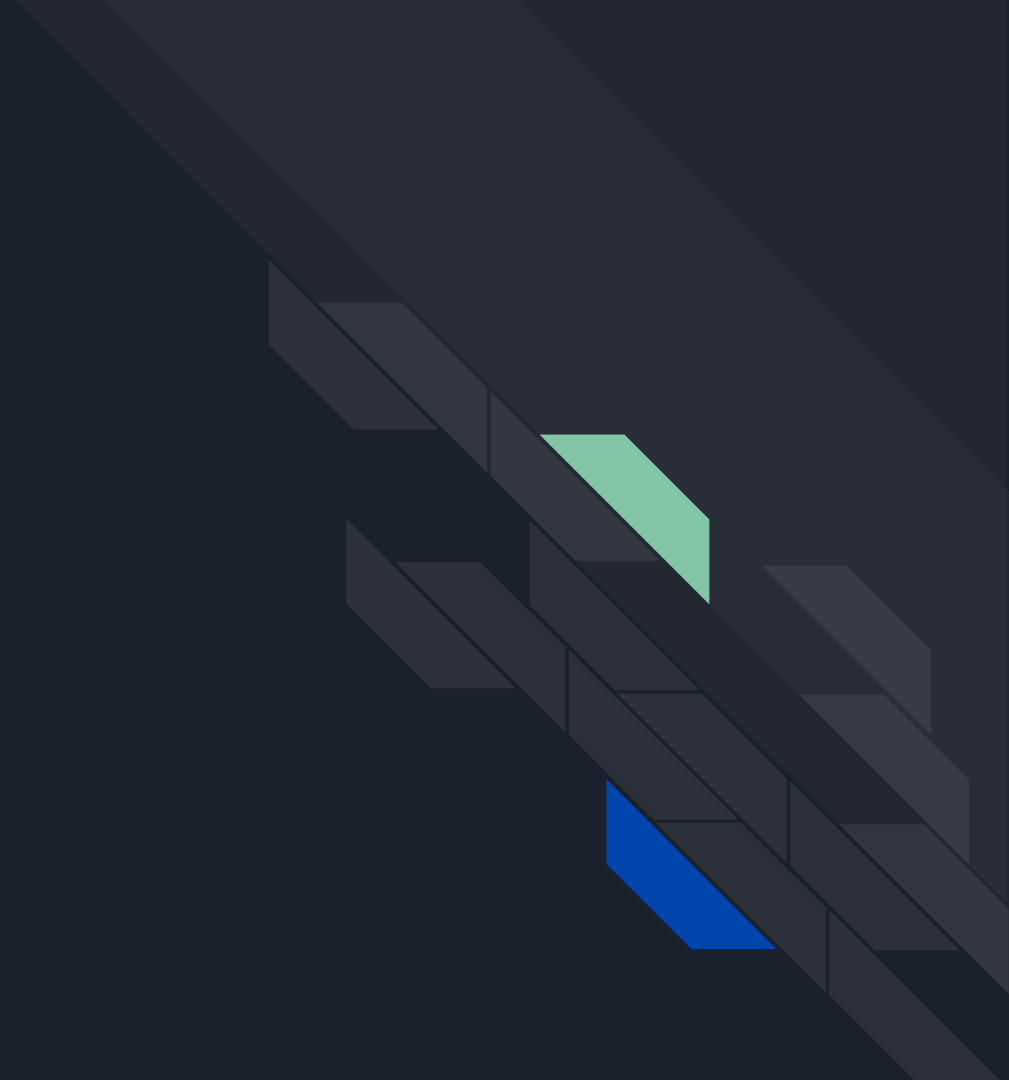
```
# Build runtime image  
FROM microsoft/aspnetcore:2.0  
WORKDIR /app  
COPY --from=build-env /app/out .  
ENTRYPOINT ["dotnet", "WebApp.dll"]
```



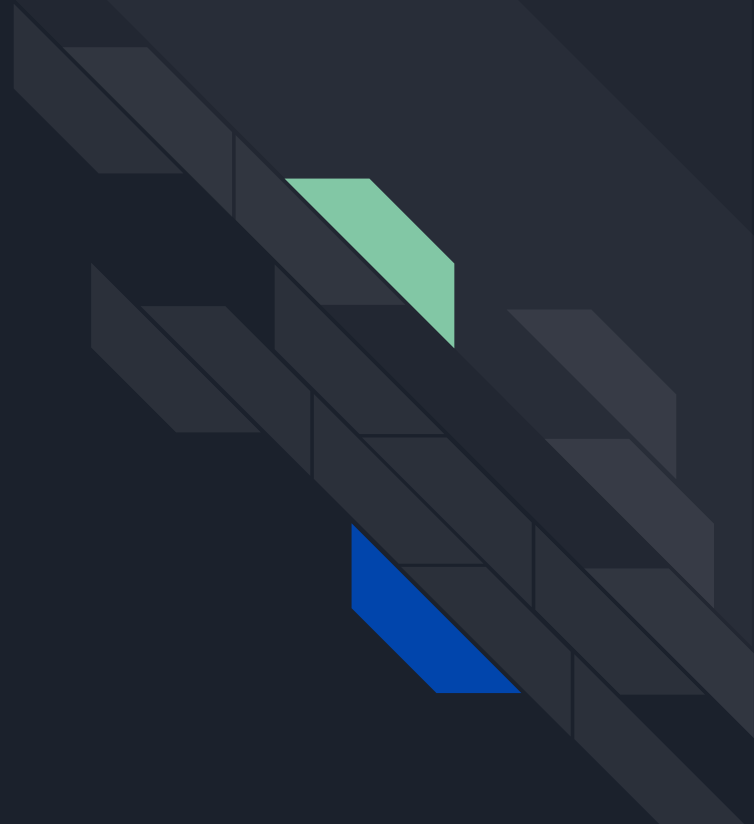
# Docker-Compose

- Run multi-tenant apps. (include webapps, webapis, caching servers, dbs?)
- Can actually start scaling stuff.
- Can be used for debugging in Visual Studio

DEMO



What Next?





# Volumes / Mounts

- Used for persistent storage.
- Can be created separately from containers.
- Can mount a volume to a container when a container is started.



# Orchestrators

- Docker Swarm.
- Kubernetes.
- Azure Service Fabric. (Not App Fabric)





# Helpful links

- <https://hub.docker.com/explore/>
- <https://www.docker.com/get-docker>
- <https://docs.docker.com/>
- <https://stackoverflow.com/questions/tagged/docker>
- <https://twitter.com/BMcCoy04>
- <http://bit.ly/2ErpzDa> (These slides)



# BR .Net, Sql Server, and Bi User groups!

- Meet 2nd wednesday of every month.
- Free Tech Talks.
- Free Food and prizes!
- Get to meet/talk to fellow developers in the area.

Meet up link: <https://bit.ly/brusergroups>

Questions?





# Who am I?

Name: Bryan

Occupation: Web Dev

Employer: Bcbsla

Twitter: @BMcCoy04

Email: [BryanMcCoy04@gmail.com](mailto:BryanMcCoy04@gmail.com)

Github: [github.com/bmccoy04](https://github.com/bmccoy04)

Slides: <http://bit.ly/2ErpzDa>