Dockerize your Windows application

Ron Bruintjes

Agenda

- What is Docker?
- Docker for Windows
- Our simple Windows app
- Dockerize the Windows app
- What’s next?

What is Docker

“Docker is an open-source project that automates the deployment of applications inside software containers.”

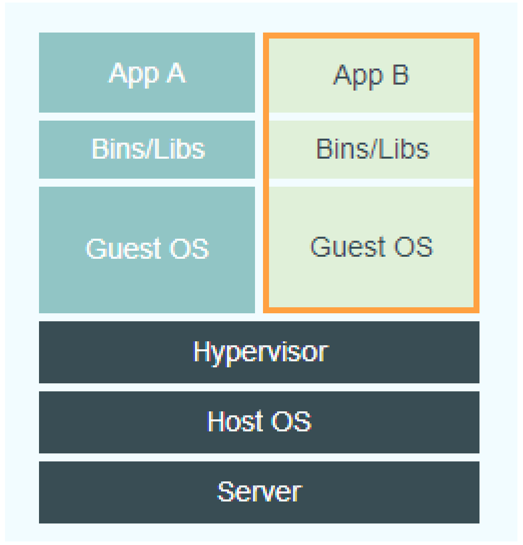
-Wikipedia

What is Docker

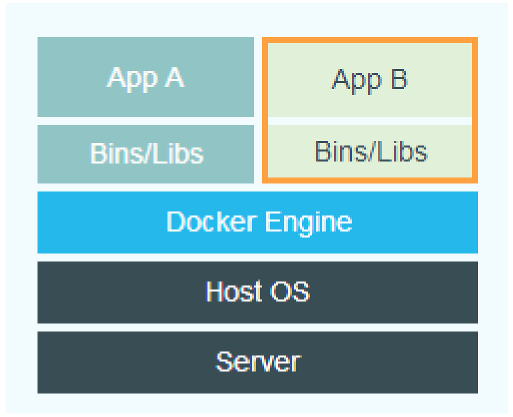
What is Docker

- Eliminates the “works on my machine” problem
- Run several apps side by side in isolated containers
- Support agile software delivery pipelines

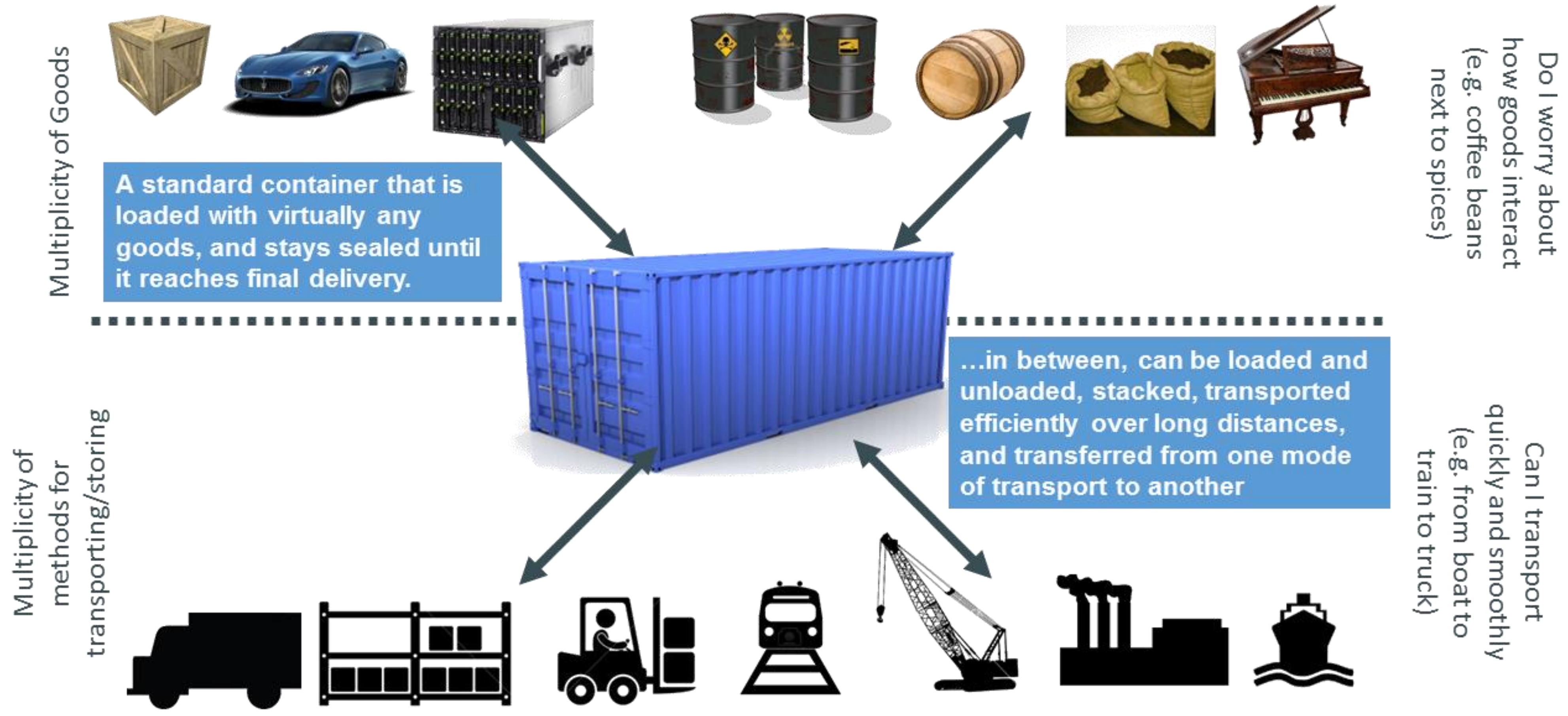
What is Docker



Docker



What is Docker



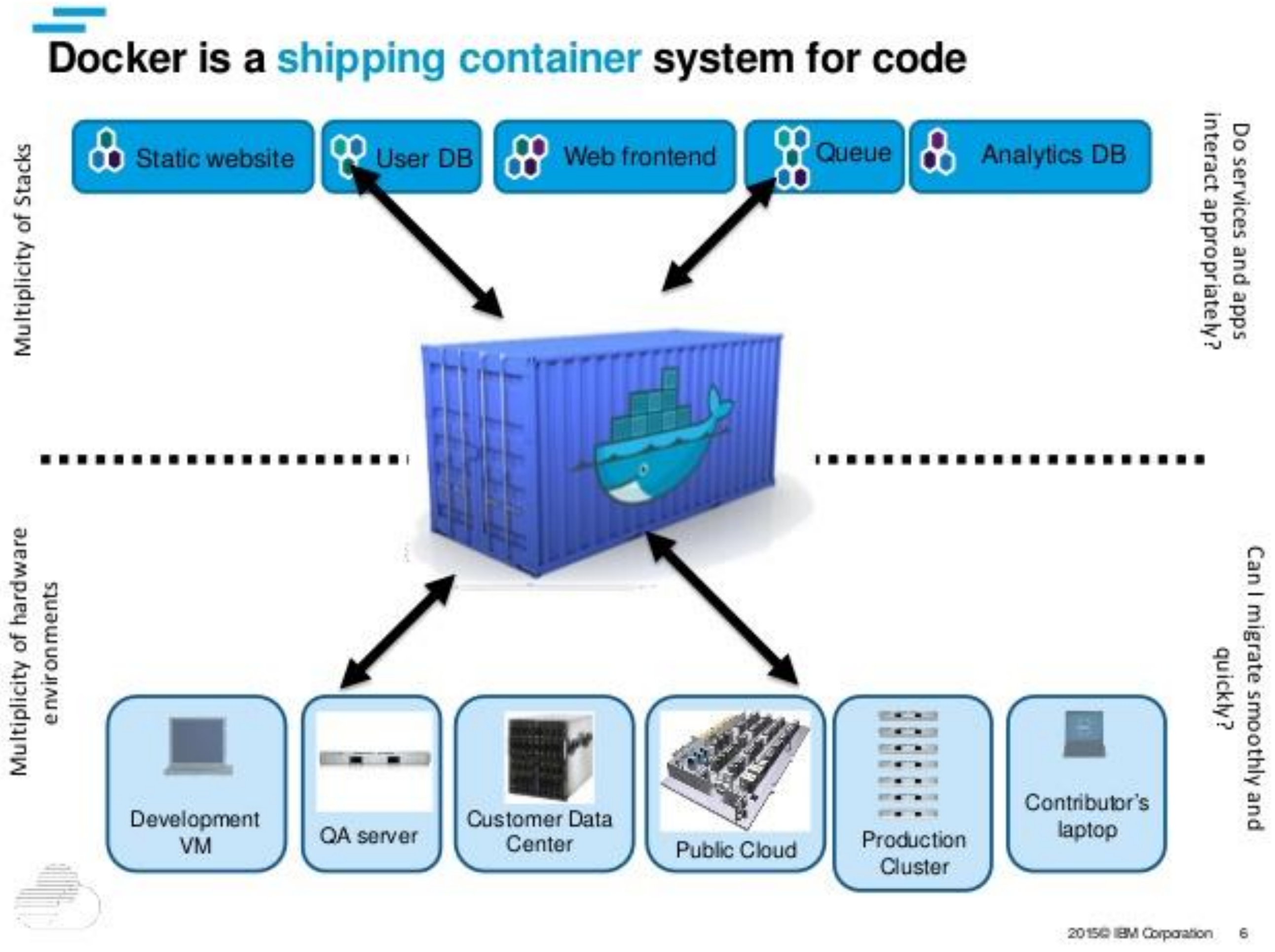
A standard container that is loaded with virtually any goods, and stays sealed until it reaches final delivery.

...in between, can be loaded and unloaded, stacked, transported efficiently over long distances, and transferred from one mode of transport to another

Do I worry about how goods interact (e.g. coffee beans next to spices)

Can I transport quickly and smoothly (e.g. from boat to train to truck)

What is Docker



Why Docker?

- About 5 times more efficient than a VM
- 1 Windows license (for the host) covers all containers
- Portability – consistent deployment
- Security
  - Scanning of images
  - Signing of images

Docker for Windows

- You will need Windows 10, and use Hyper-V
- www.docker.com, follow links to developer section
- After installation Docker commands available in shell
- `docker version`
- `docker run hello-world`

Docker for Windows

Choose your base image

- microsoft/nanoserver
  - Preferred, most minimal environment
- microsoft/windowsservercore
  - .NET Framework apps
  - MSI installers for apps and dependencies
  - 32-bit runtime support
- Or a derived image

Docker for Windows

Docker for Windows

Derived images

- microsoft/iis – basic Windows with IIS installed
- microsoft/apsnet – ASP.NET installed on top of IIS
- microsoft/aspnet:3.5 - .NET 3.5 installed and ASP.NET set up
- openjdk – OpenJDK Java runtime installed
- golang – Go runtime and SDK installed
- microsoft/dotnet - .NET runtime and SDK installed

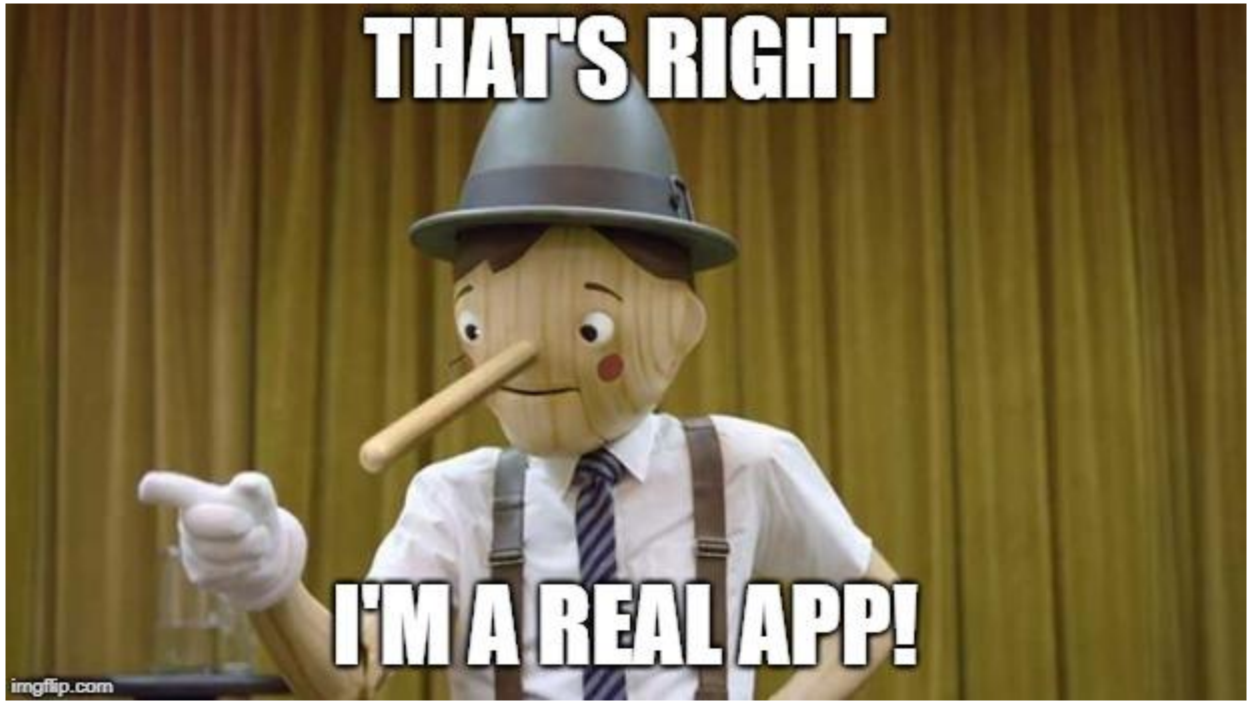
Docker for Windows

Our Simple Windows app

Visual C#, Web, ASP.NET Web Application, Single Page

OK, not that simple, more like a real app…(?)

Our Simple Windows app



Dockerize Windows app

- Publish your app locally
- Create a Dockerfile
- Describe the steps taken to install the app
- Build the image
- Run the container

Dockerize the Windows app – Publish locally

Dockerize the Windows app – Publish locally

- Create a publish profile to a local directory
  - Bin\Release\PublishOutput
- Change settings
  - Precompile during publishing: yes
- Publish app to PublishOutput

Dockerize Windows app - Dockerfile

- Based on microsoft/aspnet
  - Windows Server Core
  - IIS
  - ASP.NET
- Copy your published app into the /inetpub/wwwroot

Dockerize Windows app – Dockerfile

```
# The `FROM` instruction specifies the base image. You are
# extending the `microsoft/aspnet` image.

FROM microsoft/aspnet

# The final instruction copies the site you published earlier into the container.
COPY ./bin/Release/PublishOutput/ /inetpub/wwwroot
```

Dockerize Windows app – Build the image

```
docker build -t mvcrandomanswers .
```

```
REPOSITORY                    TAG                 IMAGE ID            CREATED             SIZE
mvcrandomanswers              latest              86838648aab6        2 minutes ago       10.1 GB
```

Dockerize Windows app – Run the container

```
docker container run --detach --name randomanswers mvcrandomanswers
```

```
docker container inspect randomanswers
```

Dockerize Windows app – See the app

Dockerize Windows app – See the app

What’s next?

How about not relying on Visual Studio to build your code?

Let Docker container build

```
FROM microsoft/dotnet-framework:4.7.2-sdk as build
WORKDIR /app

#copy csproj and restore in distinct layers
COPY *.sln .
COPY MVCRandomAnswerGenerator/*.csproj ./MVCRandomAnswerGenerator/
COPY MVCRandomAnswerGenerator/*.config ./MVCRandomAnswerGenerator/
RUN nuget restore

# copy everything else and build app
COPY MVCRandomAnswerGenerator/. ./MVCRandomAnswerGenerator/
WORKDIR /app/MVCRandomAnswerGenerator
RUN msbuild /p:Configuration=Release

FROM microsoft/aspnet:4.7.2 AS runtime
WORKDIR /inetpub/wwwroot
COPY --from=build /app/MVCRandomAnswerGenerator/. ./
```

Run the result

```
docker run –d –p 80:80 –name myapp randomanswerbuild:latest
```

then

```
docker inspect myapp
```

or

```
docker exec –ti myapp powershell
ipconfig
```

OK, what else is next?

- Turn this into a complete CI/CD chain
- Deploy some (all?) of it on Azure
- Learn more about containers
  - http://www.microsoft.com/containers
  - http://www.docker.com
  - http://www.katacoda.com/courses/docker
  - http://blog.sixeyed.com

Thank you! Questions?

http://docker.com

@docker

Contact info

- Twitter: @ronaldb
- http://automationadventures.com