Docker Beyond the Classroom
Meltwater

```
$ whoami
```

Meltwater

$ whoami

János Csorvási

Senior Engineer, Developer Productivity @ Meltwater

https://www.linkedin.com/in/jcsorvasi/

janos.csorvasi@meltwater.com

```
$ whois meltwater.com
```

Meltwater

$ whois meltwater.com

Media monitoring company

Get data, enrich, analyze, search

~350 engineers

Using docker in production since 2014

Agenda

Docker @ Class

Demos

Beyond the Classroom

How Can Docker Help

Kubernetes?

Discussion

Docker @ Class

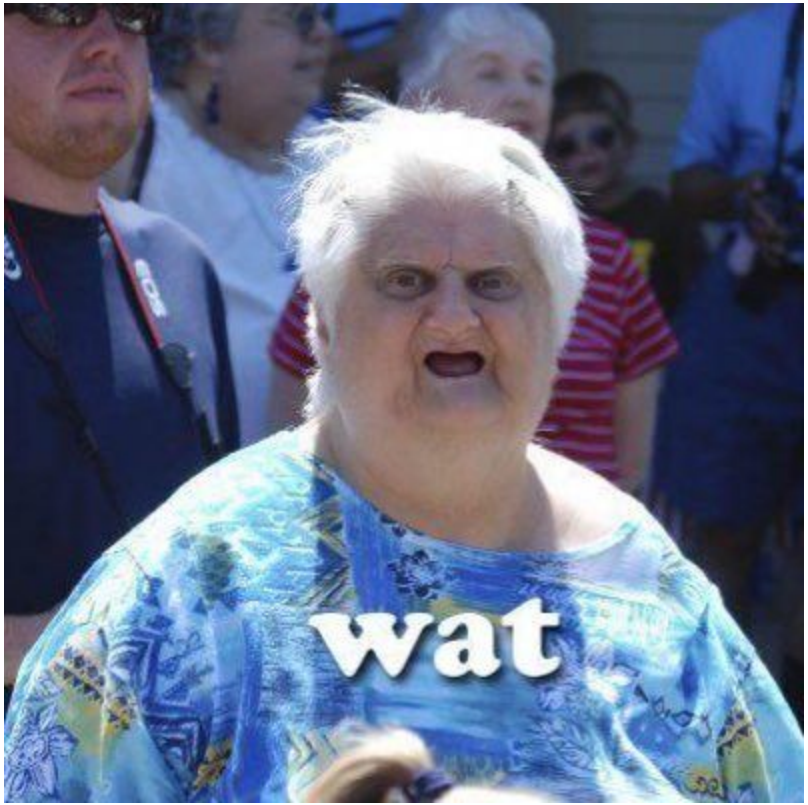
What is Docker?

Docker is a tool to create container images and manage container instances

What is a container?

A standard unit of software

Docker @ Class



Docker @ Class

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another.

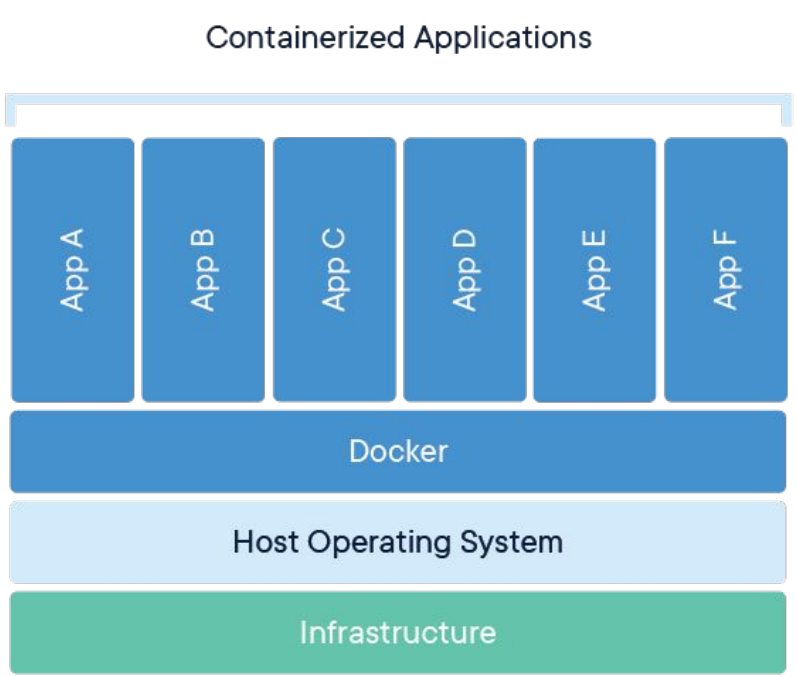
A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.

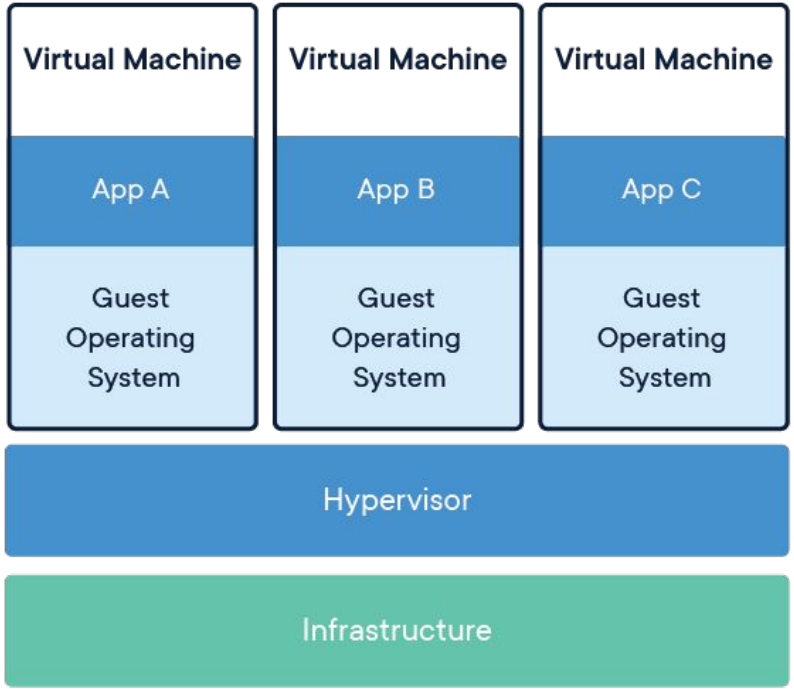
A bunch of code together = An application

An application = A container image

An application instance = A container instance

Docker @ Class





Containers are

- Standardized
- Isolated
- Lightweight
- Secure
- Reproducible
- Cheap

Run a container (spawn a container instance from an image)

```
$ docker run [<FLAGS>] <IMAGE> [<COMMAND>]
```

example:

```
$ docker run -it -p 8080:80 nginx
```

Manage a container

```
$ docker ps
$ docker [start|stop|pause|kill] <ID>
```

DEMO

Build a container

```
$ docker build [<FLAGS>] <PATH>
```

example

```
$ docker build -f path/to/Dockerfile .
```

Anatomy of a Dockerfile

A Dockerfile is a simple text file containing instructions on how to assemble a Docker image

```
# comment
INSTRUCTION arguments
```

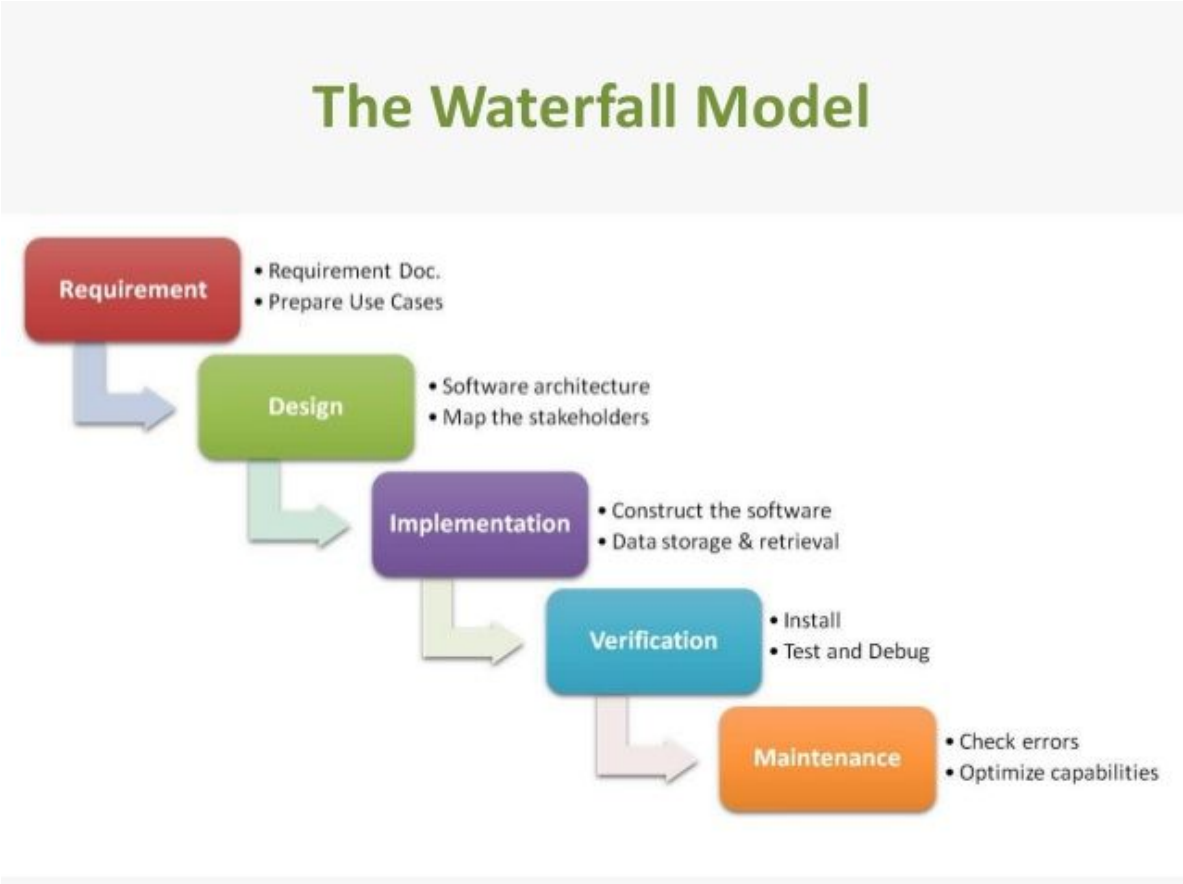
```
$ cat Dockerfile
```

```
FROM python:3.7-slim

LABEL maintainer="attila.nagy@meltwater.com"

COPY requirements.txt /tmp/requirements.txt

RUN pip install -r /tmp/requirements.txt

COPY my_source_code /my_app/my_source_code

ENTRYPOINT ["python3"]

CMD ["/my_app/my_source_code/main.py"]
```

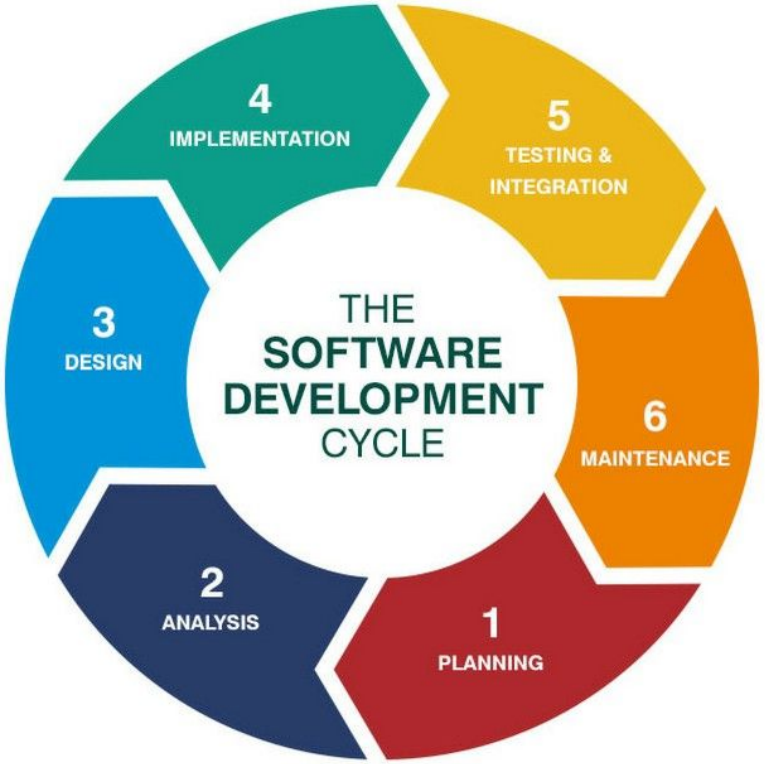
DEMO

Beyond the Classroom

Docker Beyond the Classroom - Story

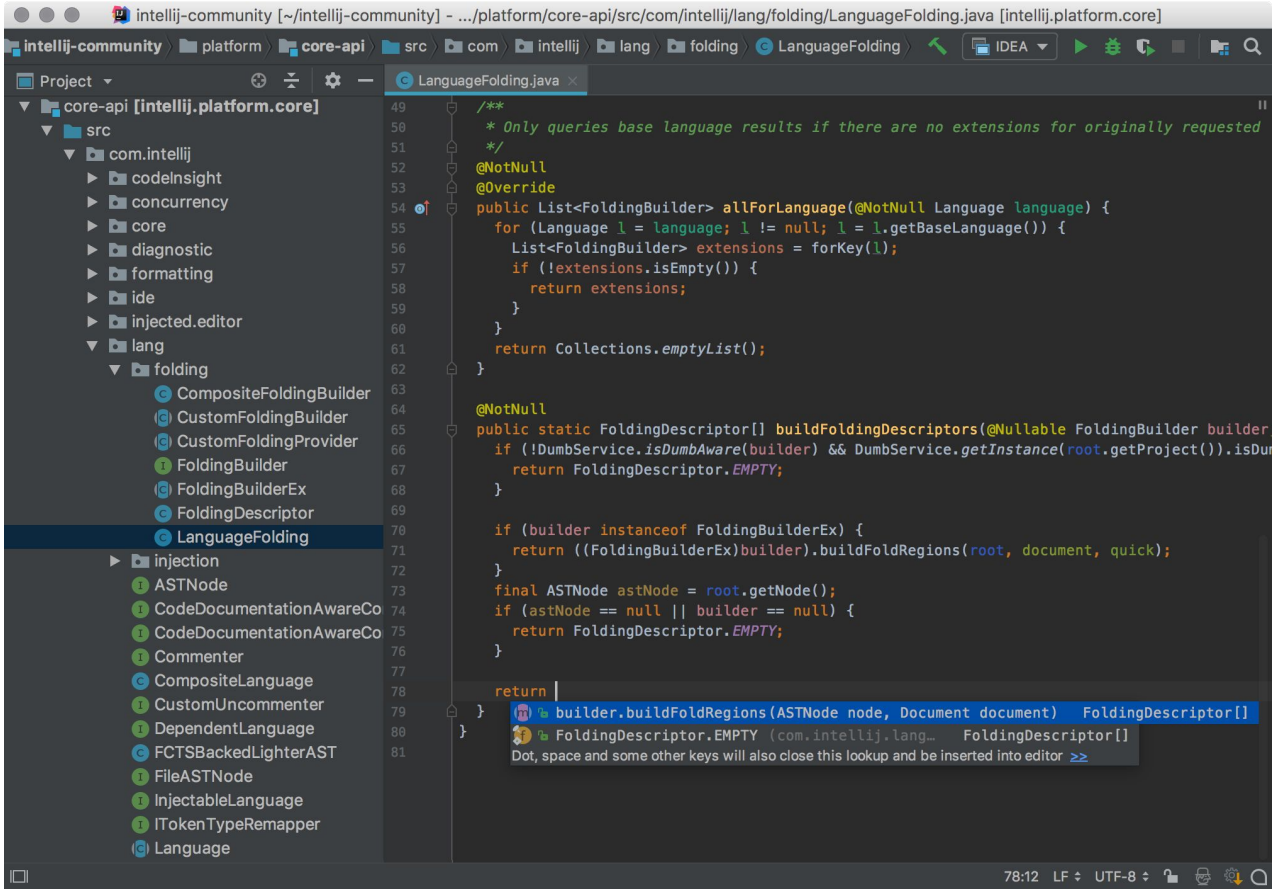


Docker Beyond the Classroom - Story

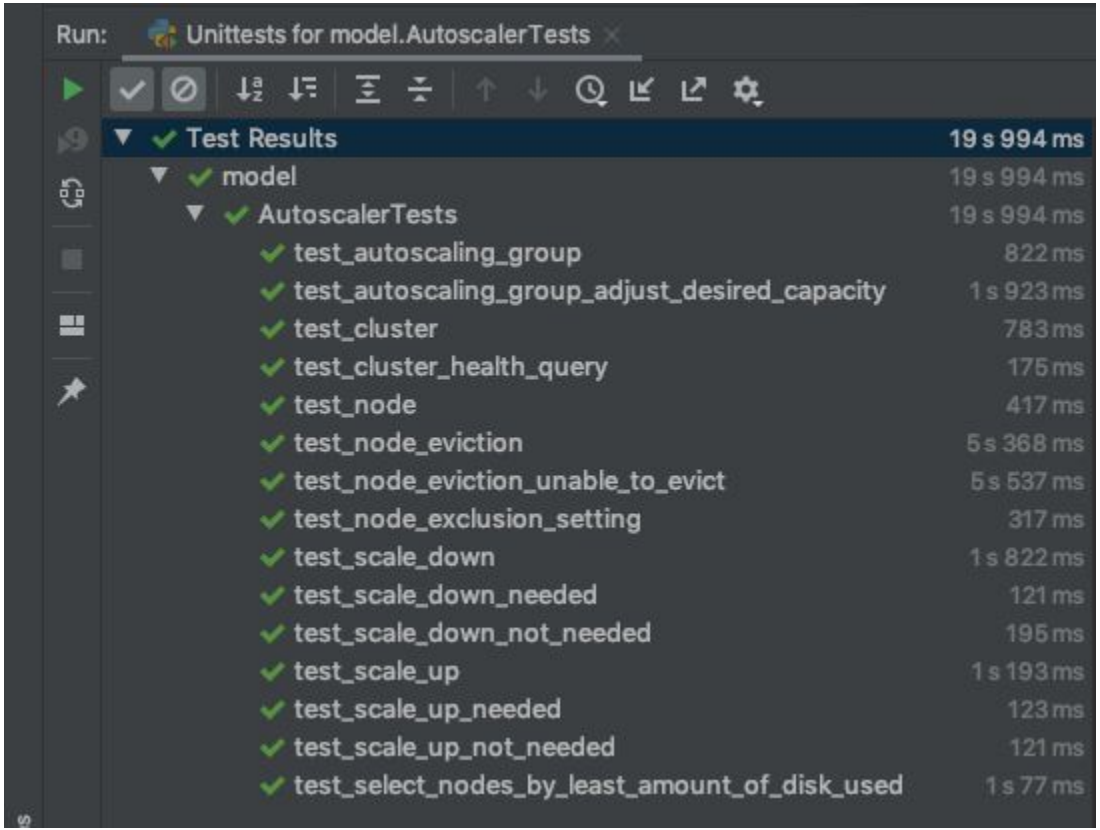


Docker Beyond the Classroom - Story

Docker Beyond the Classroom - Story



Docker Beyond the Classroom - Story



Docker Beyond the Classroom - Story

Docker Beyond the Classroom - Story

Docker Beyond the Classroom - Story

```
java.lang.RuntimeException
        at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
        at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:39)
        at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:27)
        at java.lang.reflect.Constructor.newInstance(Constructor.java:513)
        at org.codehaus.groovy.reflection.CachedConstructor.invoke(CachedConstructor.java:77)
        at org.codehaus.groovy.runtime.callsite.ConstructorSite$ConstructorSiteNoUnwrapNoCoerce.callConstructor(ConstructorSite
        at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallConstructor(CallSiteArray.java:52)
        at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callConstructor(AbstractCallSite.java:192)
        at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callConstructor(AbstractCallSite.java:196)
        at newifyTransform$_run_closure1.doCall(newifyTransform.gdsl:21)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
        at java.lang.reflect.Method.invoke(Method.java:597)
        at org.codehaus.groovy.reflection.CachedMethod.invoke(CachedMethod.java:86)
        at groovy.lang.MetaMethod.doMethodInvoke(MetaMethod.java:234)
        at org.codehaus.groovy.runtime.metaclass.ClosureMetaClass.invokeMethod(ClosureMetaClass.java:272)
        at groovy.lang.MetaClassImpl.invokeMethod(MetaClassImpl.java:893)
        at org.codehaus.groovy.runtime.callsite.PogoMetaClassSite.callCurrent(PogoMetaClassSite.java:66)
        at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:151)
        at newifyTransform$_run_closure1.doCall(newifyTransform.gdsl)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
        at java.lang.reflect.Method.invoke(Method.java:597)
        at org.codehaus.groovy.reflection.CachedMethod.invoke(CachedMethod.java:86)
        at groovy.lang.MetaMethod.doMethodInvoke(MetaMethod.java:234)
        at org.codehaus.groovy.runtime.metaclass.ClosureMetaClass.invokeMethod(ClosureMetaClass.java:272)
        at groovy.lang.MetaClassImpl.invokeMethod(MetaClassImpl.java:893)
        at org.codehaus.groovy.runtime.callsite.PogoMetaClassSite.call(PogoMetaClassSite.java:39)
        at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:121)
        at org.jetbrains.plugins.groovy.dsl.GroovyDslExecutor$_processVariants_closure1.doCall(GroovyDslExecutor.groovy:54)
        at sun.reflect.GeneratedMethodAccessor61.invoke(Unknown Source)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
        at java.lang.reflect.Method.invoke(Method.java:597)
        at org.codehaus.groovy.reflection.CachedMethod.invoke(CachedMethod.java:86)
        at groovy.lang.MetaMethod.doMethodInvoke(MetaMethod.java:234)
        at org.codehaus.groovy.runtime.metaclass.ClosureMetaClass.invokeMethod(ClosureMetaClass.java:272)
```

IT WORKS

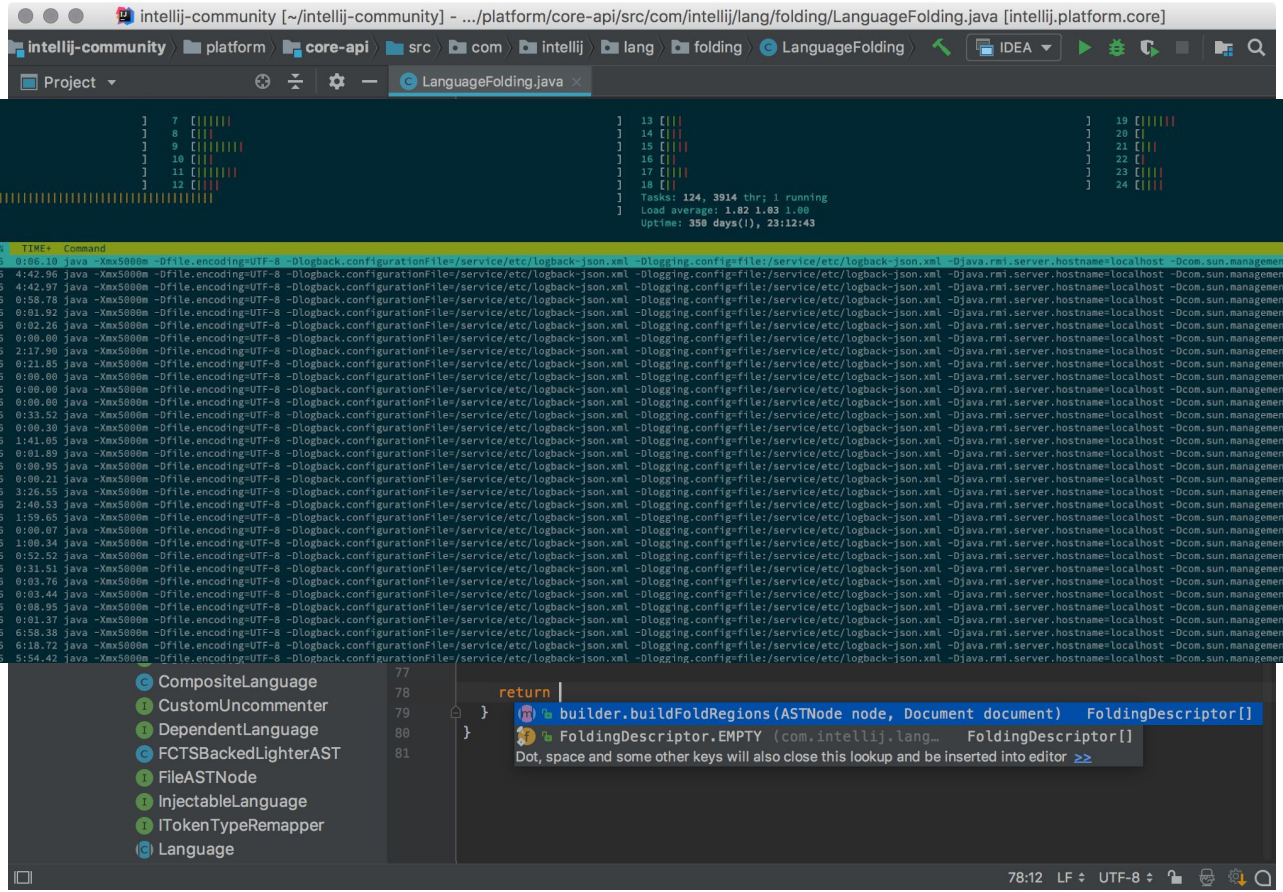
on my machine

Docker Beyond the Classroom - Story

Why?

- The environment is too different
  - Hardware
  - OS
  - Network
  - Libs
  - Runtime (JVM)
  - Filesystem
  - Configuration
  - Service dependencies

Docker Beyond the Classroom - Story



How Can Docker Help

How could've Docker helped in the story?

- A container runs the same on the Dev's laptop and a server in prod
- Everything encapsulated in a package
- Standard way to run
- Standard way to configure
- Always start from scratch

Packaging

- A container encapsulates everything the app needs
  - With a few exceptions...

Isolation

- Libraries
  - No conflicts with other versions of the same lib needed by another app
- Filesystem
  - Files are not shared (by default) between containers
- From other processes

Reproducible environments

- Easy to do full system integration tests locally
  - Dependencies
  - Dependent systems
  - Databases

Docker Beyond the Classroom - How Can Docker Help

```
$ cat docker-compose.yaml

services:
  elasticsearch:
    image: elasticsearch:5.2.1
    ports:
      - "9200:9200"
    environment:
      - cluster.name=dummy-cluster
      - "ES_JAVA_OPTS=-Xms512m -Xmx512m"
    mem_limit: 1g
  redis:
    image: redis:3.2.4
    ports:
      - "6379:6379"
```

Docker Beyond the Classroom - How Can Docker Help

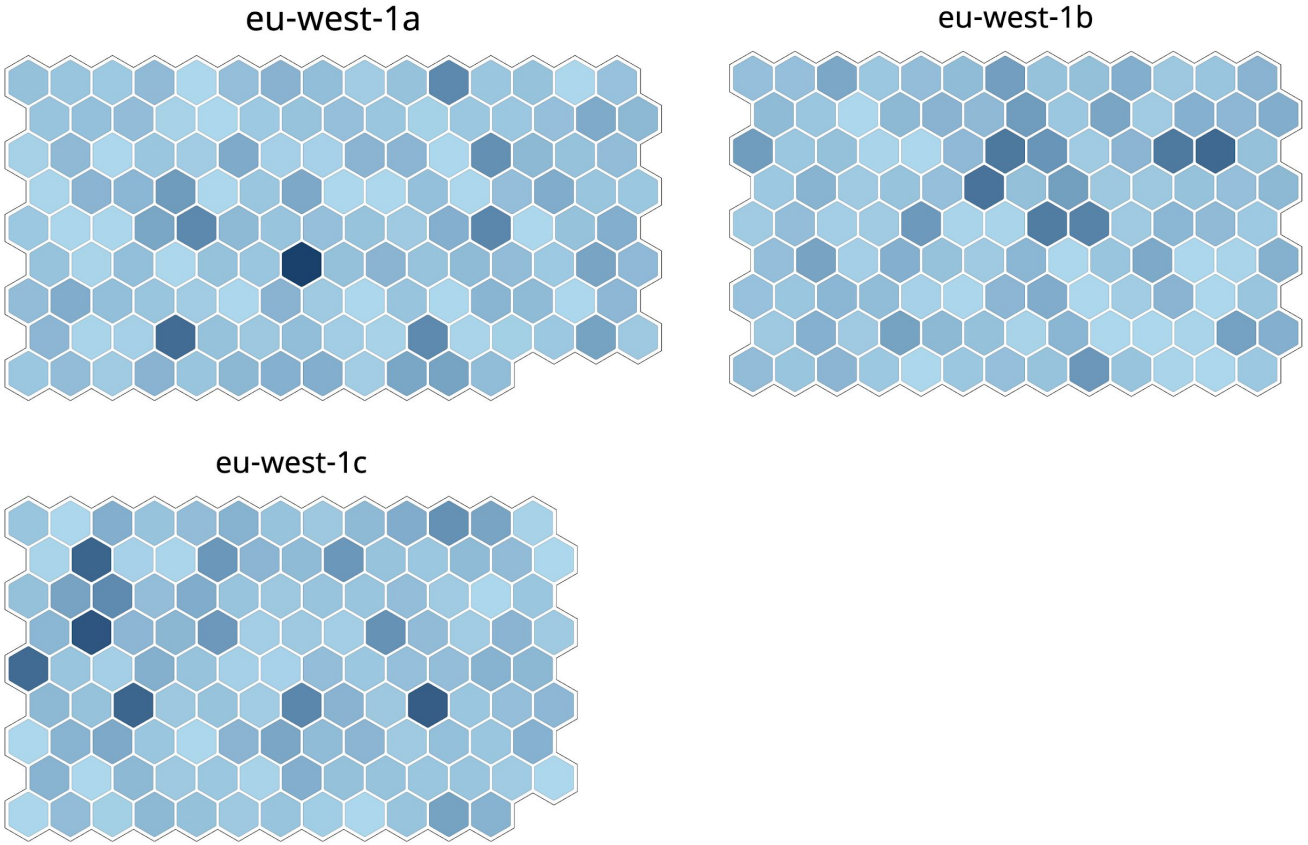
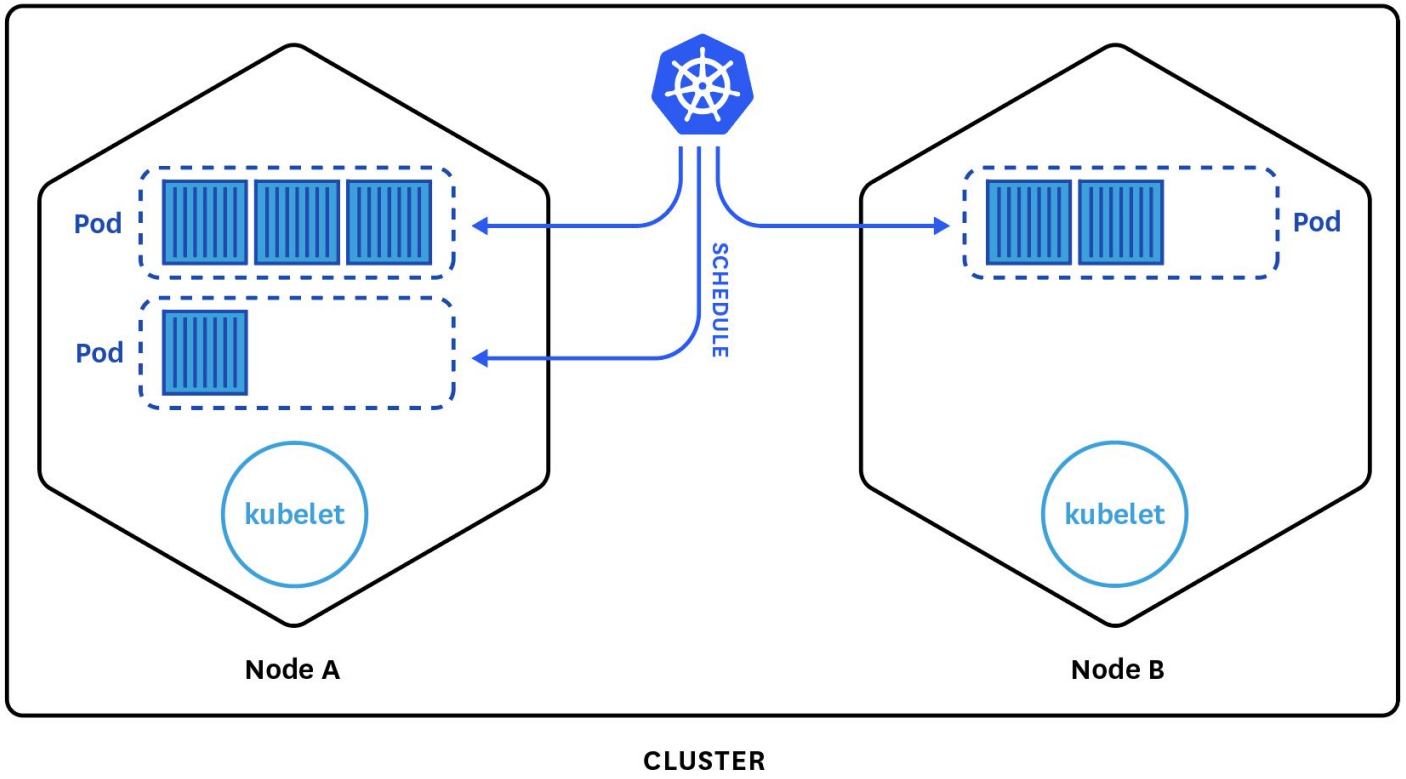
Multi-tenancy

- Machine dedicated to an app...
- Multiple instances of the same app on the same machine
- Different apps on the same machine
- Just a bunch of machines
    - Pets vs. Cattle

Container Orchestration

- “Make this happen, I don’t care how and where”

Docker Beyond the Classroom - How Can Docker Help



Docker Beyond the Classroom - How Can Docker Help