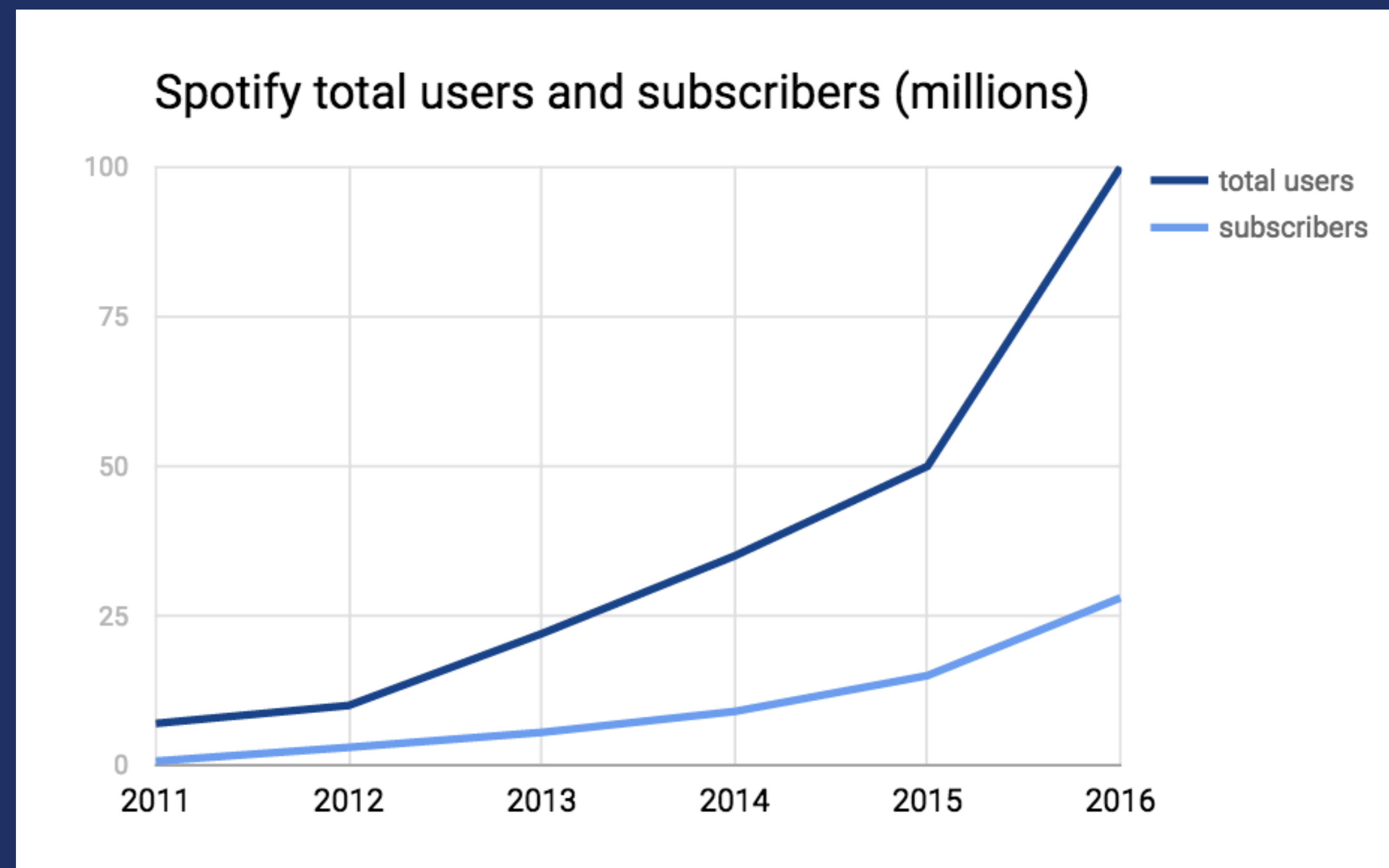


Reach Production Faster with Containers in Testing

David Xia



Spotify's Scale



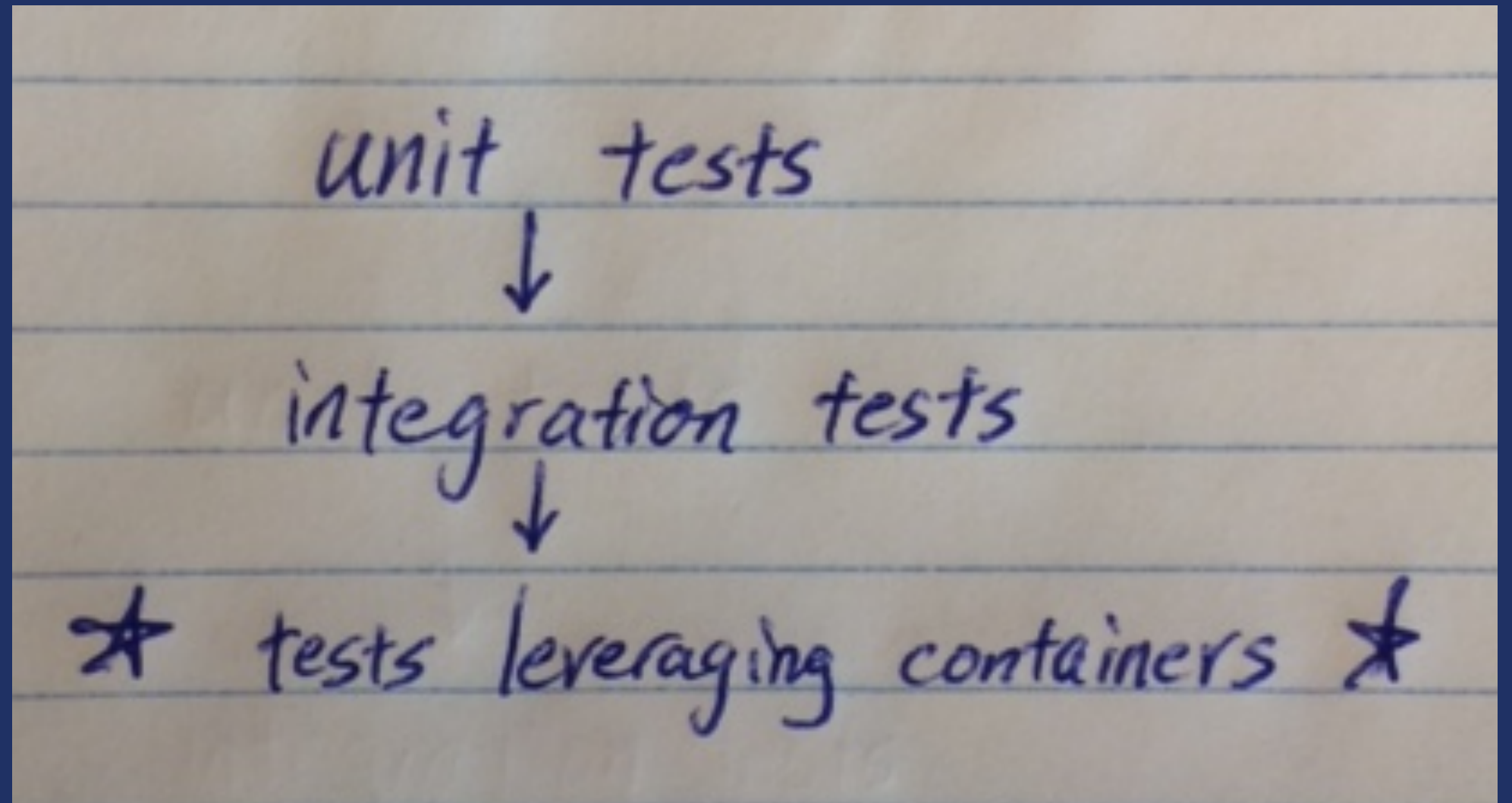
- **150+ people in infrastructure**
- **Thousands of hosts, 2000+ running containers**
- **1500+ deploys in past month, majority were containers**

About David Xia

- Work on deployment infrastructure
- Work on open-source Docker orchestration tool [Helios](#)

Prerequisites for This Talk

- You're familiar with containers
- You like tests
- That's it!



Prerequisites for This Talk

You don't need to use containers in production for this talk to be useful!

Three Problems, Three Ways to Solve with Containers

How can I enable developers to:

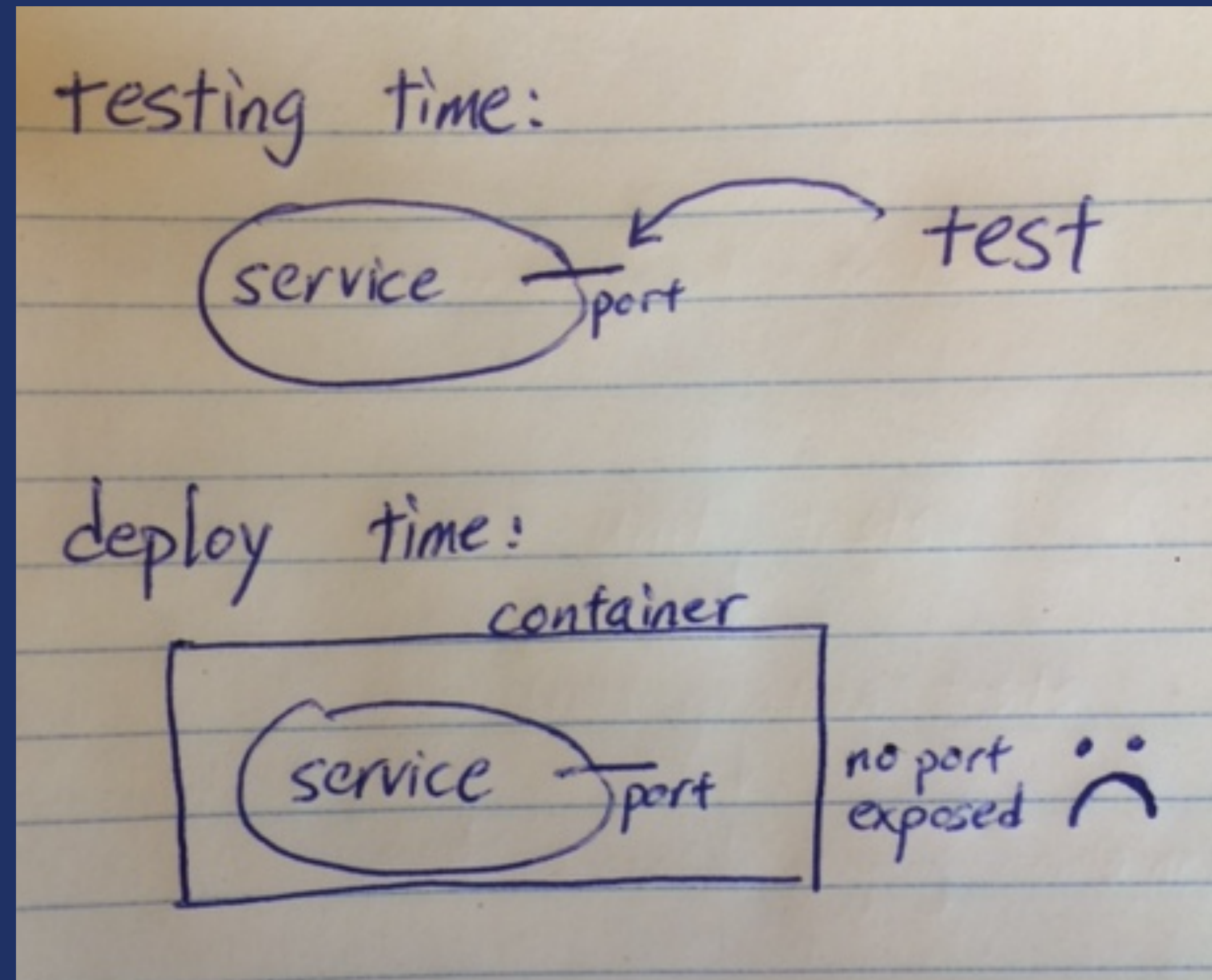
- catch container misconfiguration in tests?
- easily install and start non-trivial test dependencies?
- make their tests isolated and reproducible?

Problem 1: Container Misconfiguration

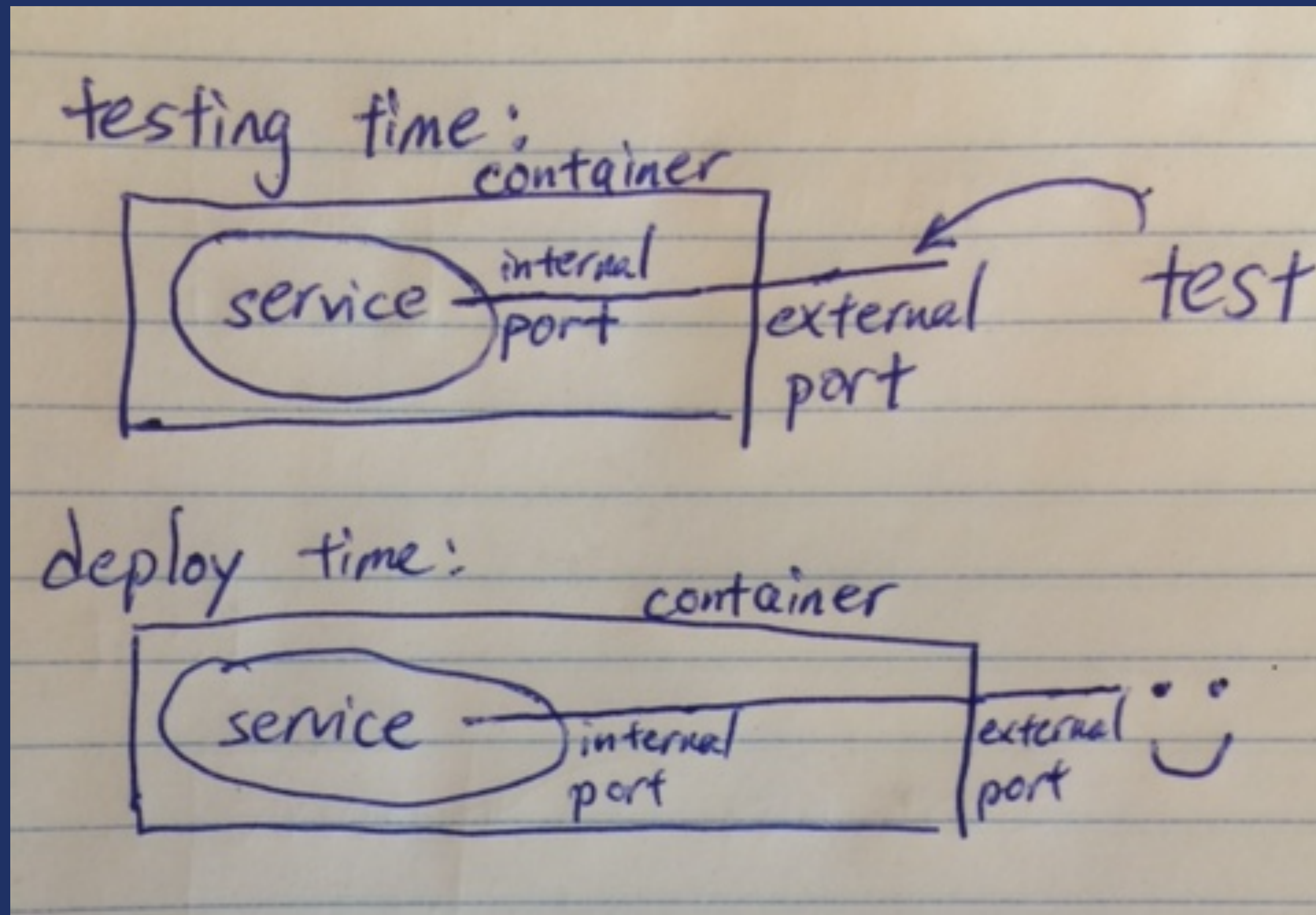
“Why did my service pass integration tests but fail when I deployed it as a container?”

- Sad Developer 1

Problem 1: Container Misconfiguration



Solution 1: Container Misconfiguration

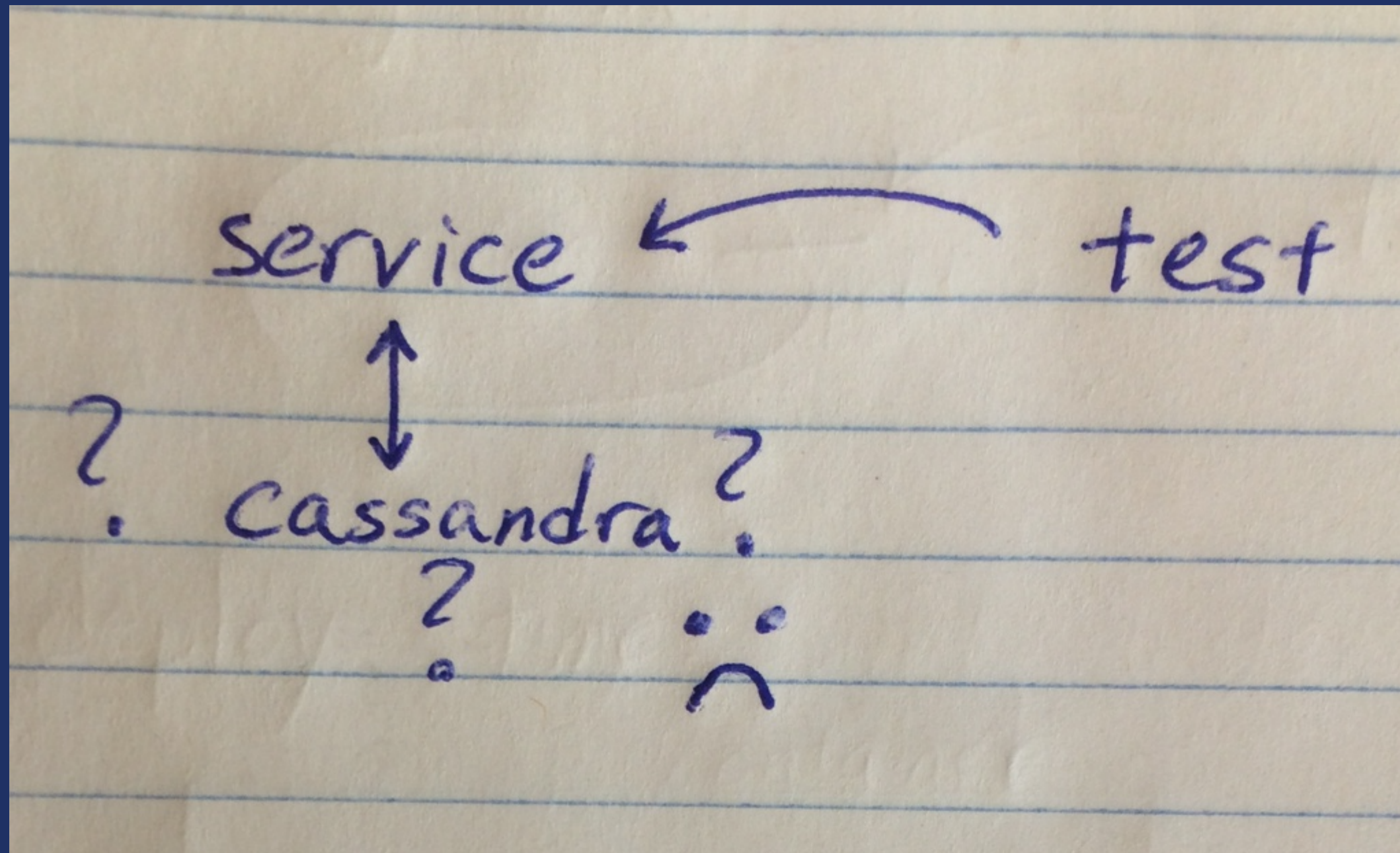


Problem 2: Non-trivial dependencies

“I want to run my project's integration tests locally. The tests need a local Cassandra/other DB. How do I set everything up?”

- Sad Developer 2

Problem 2: Non-trivial dependencies

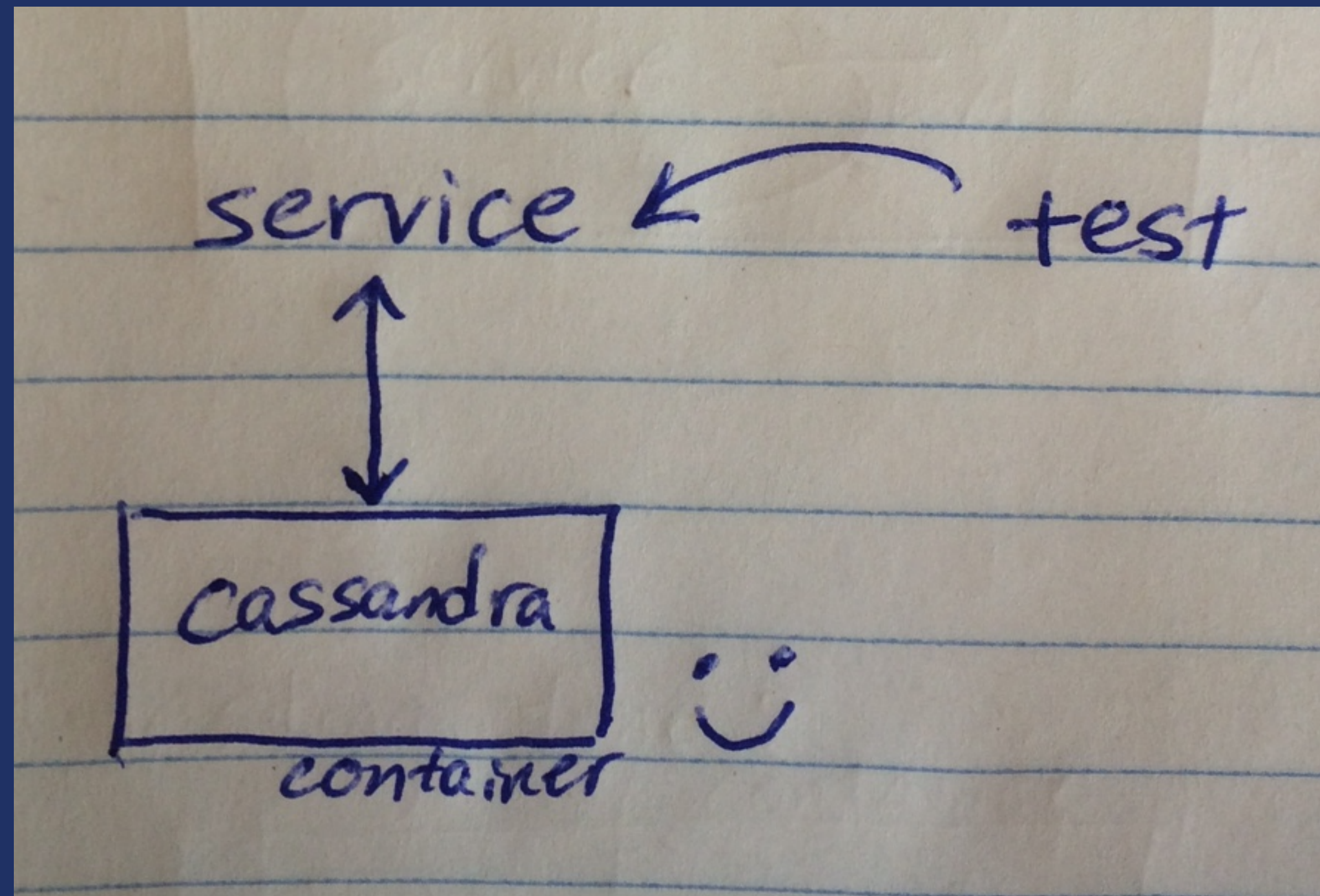


Solution 2: Non-trivial dependencies

```
docker run --name foo -d cassandra
```

- Happy Developer 2

Solution 2: Non-trivial dependencies



Problem 3: Reproducible Tests

**“How can I easily restore
my test dependencies to
a clean state?”**

- Sad Developer 3

Solution 3: Reproducible Tests

```
docker stop <container ID>
```

```
docker run --name foo -d cassandra
```

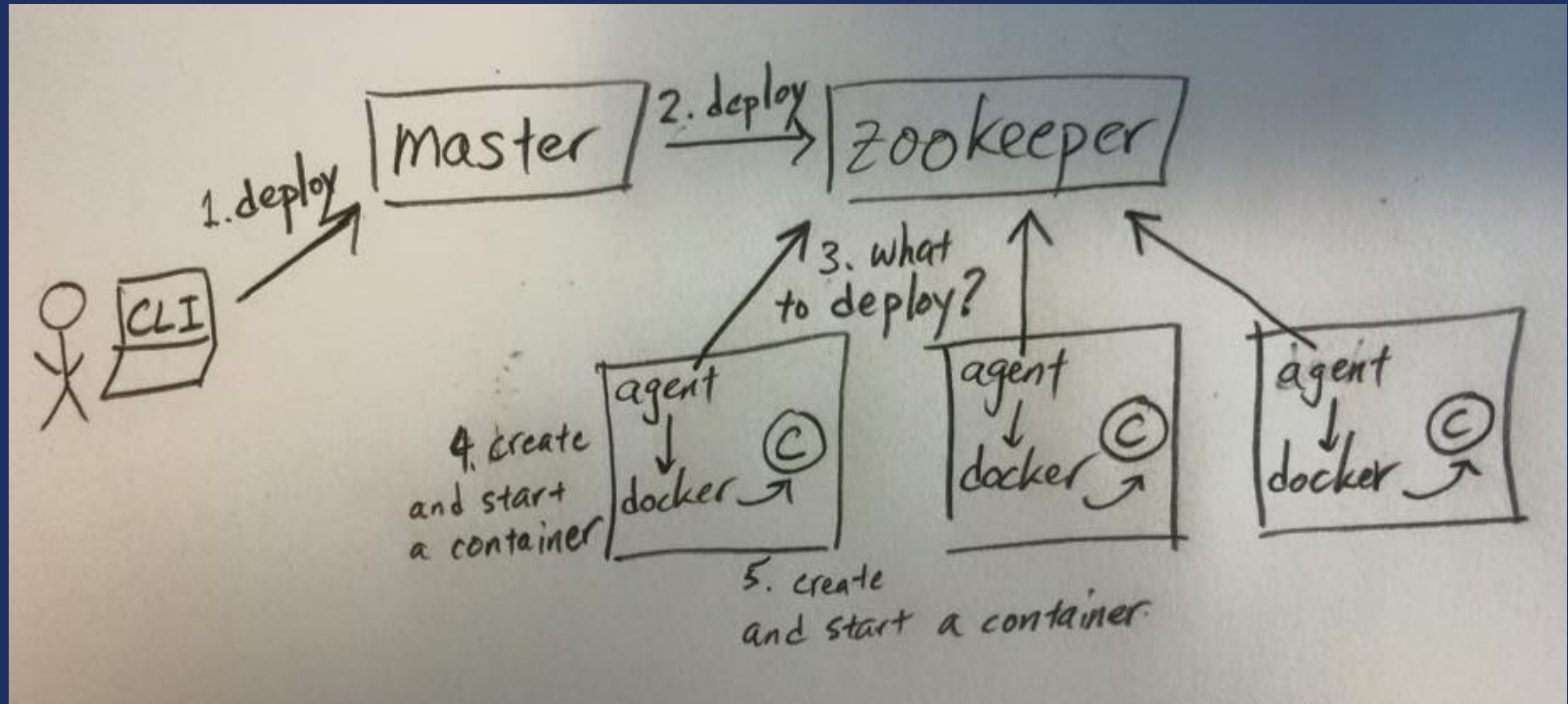
- Happy Developer 3

Key Takeaways - Using Containers in Tests Can Help You:

- Test more of the stack in an env resembling production
- Easily start real dependencies
- Ensure tests are reproducible and isolated



Helios in a Nutshell



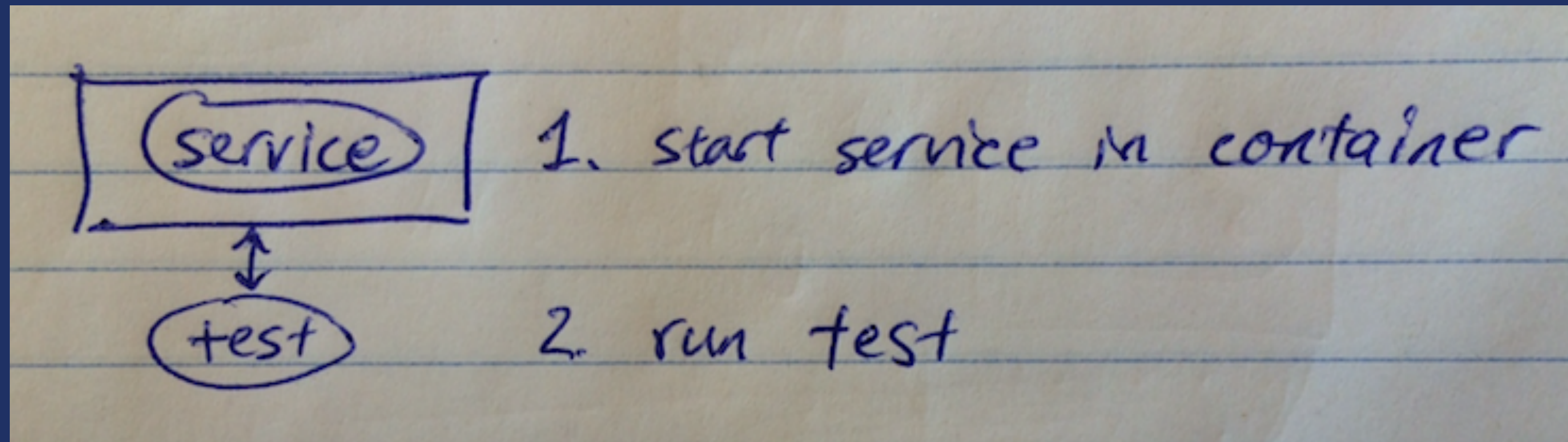
How helios-testing was born

- multitenancy
- docker
- container orchestration (helios)
- container testing framework (helios-testing)



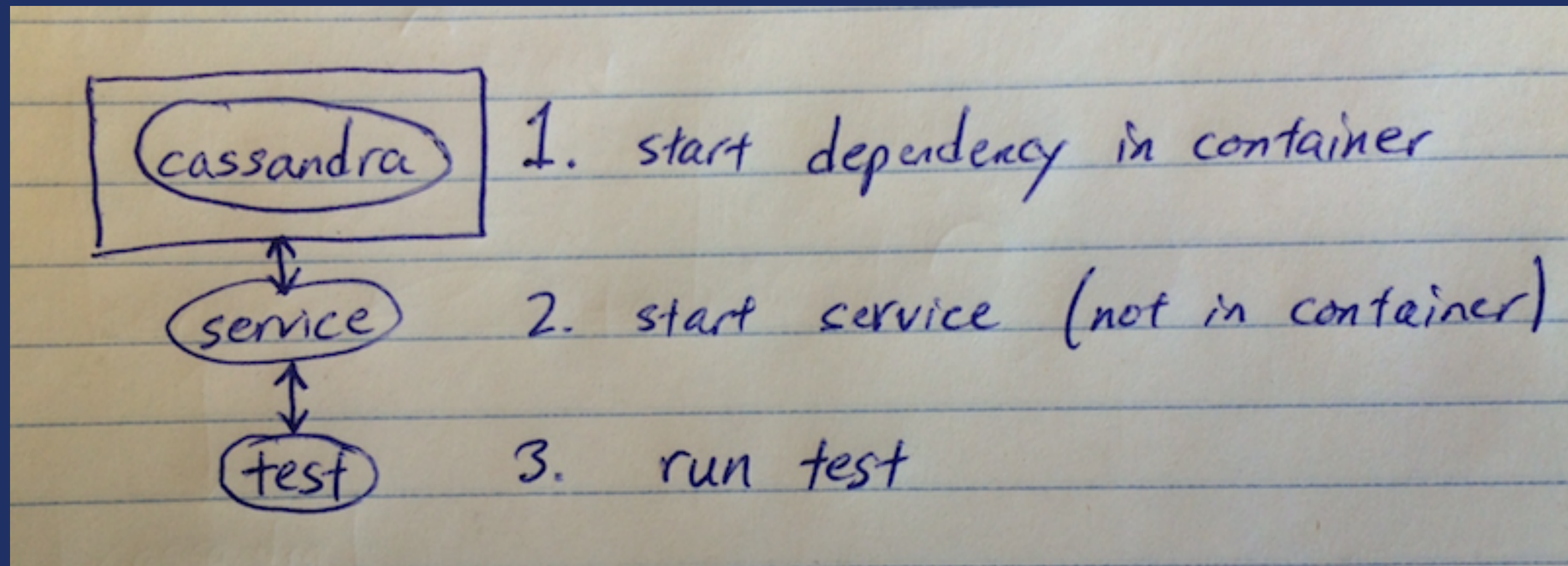
What does helios-testing do?

Let's you write code to start and stop containers.



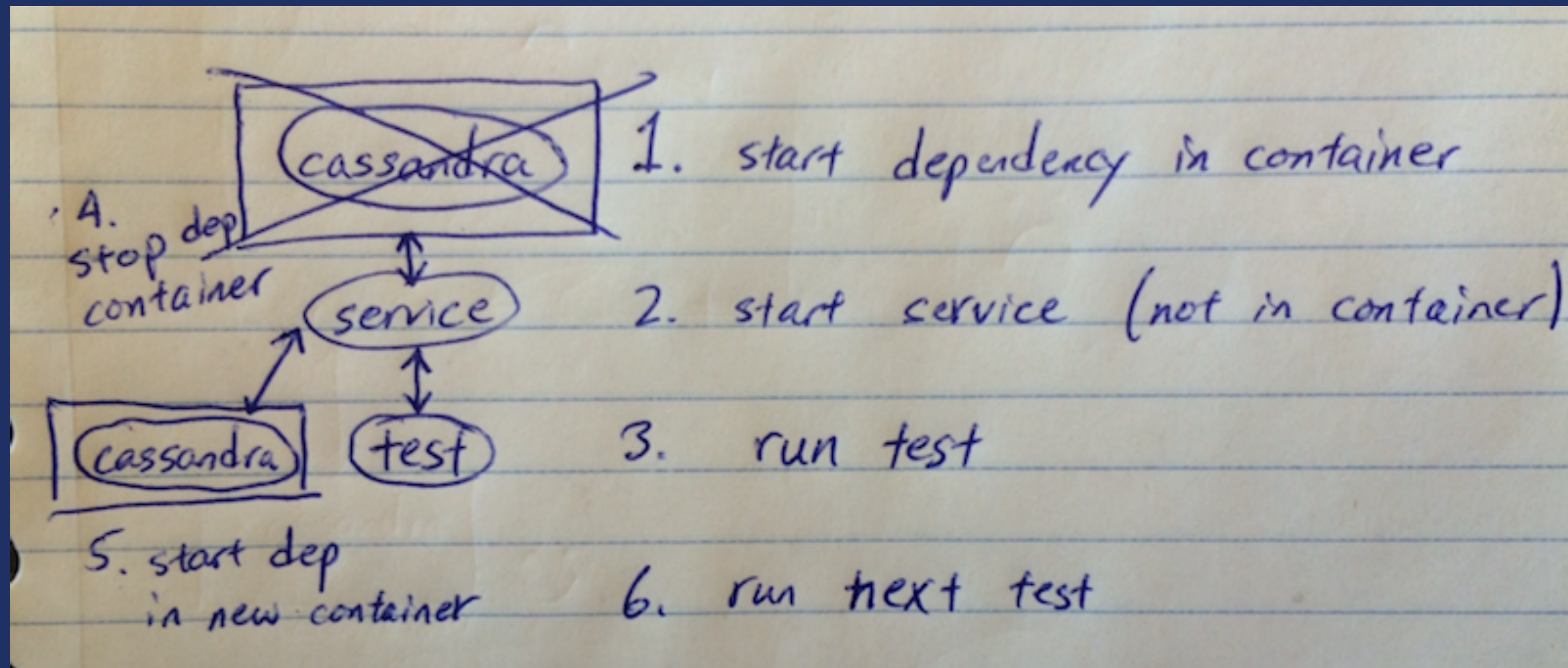
What does helios-testing do?

Let's you write code to start and stop containers.



What does helios-testing do?

Let's you write code to start and stop containers.



Demo! Solution 1: Container Configuration

The screenshot displays an IDE with two main panels. The left panel shows the source code for a Java web service, and the right panel shows the output of the containerization process.

Left Panel: Service.java

```
1 package com.davidxia.qcondemo;
2
3 import static spark.Spark.get;
4
5 public class Service {
6     public static void main(String[] args) {
7         get("/hello", (req, res) -> "Hello, World");
8         get("/:name", (req, res) -> {
9             final String name = req.params(":name");
10            return "Hello, " + name;
11        });
12     }
13 }
14
```

Right Panel: Build and Run Output

Build Output:

```
[INFO] Building image davidxia/qcon-demo:latest
Step 0 : FROM davidxia/qcon-demo:base
----> 06a61177fbf6
Step 1 : MAINTAINER David Xia <dxia@spotify.com>
----> Using cache
----> bc948291ecc6
Step 2 : ENTRYPOINT /bin/bash -c exec /usr/bin/java $JVM_DEFAULT_ARGS $JVM_ARGS -jar /usr/share/qcon-demo/qcon-demo.jar "$@" bash
----> Using cache
----> 7483fa250945
Step 3 : COPY target/lib /usr/share/qcon-demo/lib
----> 7b5fc2c33f6f
Removing intermediate container 992fed636e96
Step 4 : COPY target/qcon-demo-0.0.1-SNAPSHOT.jar /usr/share/qcon-demo/qcon-demo.jar
----> bc2657b193a5
Removing intermediate container cd2cfaad0357
Successfully built bc2657b193a5
[INFO] Built davidxia/qcon-demo:latest
[INFO] BUILD SUCCESS
[INFO] Total time: 8.706 s
[INFO] Finished at: 2016-06-11T08:49:18-04:00
[INFO] Final Memory: 45M/417M
```

Run Output:

```
dxia@Davids-MacBook-Air ~/s/qcon-demo master
mvn clean package

Run ContainerIT
1 test passed - 401ms
ContainerIT (com.davidxia.qcondemo) 401ms
testContainerHello 401ms
"name": "dxia", "timezone": "America/New_York"
09:01:08.687 [main] INFO com.spotify.helios.testing.HeliosConfig - No profile found at helios.solo.profile. Using an empty config.
09:01:13.342 [main] INFO com.spotify.helios.testing.HeliosSoloDeployment - image onescience/alpine:latest is present. Not pulling it.
09:01:13.345 [main] INFO com.spotify.docker.client.DefaultDockerClient - Creating container with ContainerConfig: ContainerConfig{hostname=null, domainname=null, username=null, attachStdin=null, attachStdout=null, attachStderr=null, portSpecs=null, exposedPorts=null, tty=null, openStdin=null, stdinOnce=null, env=[DOCKER_HOST=unix:///var/run/docker.sock], cmd=[curl, -f, --unix-socket, ///var/run/docker.sock, http://containers/bc18a131/json], image=onescience/alpine:latest, volumes=null, workingDir=null, entrypoint=null, networkDisabled=null, onBuild=null, labels=null, macAddress=null, hostConfig=HostConfig{binds=[/var/run/docker.sock:/var/run/docker.sock], containerIDFile=null, lxcConf=null, privileged=null, portBindings=null, links=null, publishAllPorts=null, dns=null, dnsSearch=null, extraHosts=null, volumesFrom=null, capAdd=null, capDrop=null, networkMode=null, securityOpt=null, devices=null, memory=null, memorySwap=null, cpuShares=null, cpusetCpus=null, cpuQuota=null, cgroupParent=null, restartPolicy=null,
```


Demo! Problem 2

The screenshot displays a development environment with two main windows. The left window is an IDE showing a Java file named `Service.java` in the package `com.davidxia.qcondemo`. The code defines a `Service` class with a `main` method that interacts with a Cassandra database. The right window shows the Docker Desktop interface with a table of running containers and a log viewer for the `CassandraContainerIT` container.

```
package com.davidxia.qcondemo;

import ...

public class Service {

    static final String KEYSpace = "qcondemo";

    public static void main(String[] args) {
        // Connect to the cluster
        final String contactPoint = firstNonNull(System.getenv("CASSANDRA_HOST"), "127.0.0.1");
        final Cluster cluster = Cluster.builder().addContactPoint(contactPoint).build();

        get("/hello", (req, res) -> "Hello, World");

        get("/:name", (req, res) -> {
            final String name = req.params(":name");
            return "Hello, " + name;
        });

        post("/users", (req, res) -> {
            final String username = req.body();

            // Insert one record into the users table
            try (final Session session = cluster.connect(KEYSPACE)) {
                session.execute(format(
                    "INSERT INTO %s.users (username) VALUES ('%s')", KEYSpace, username));
                return "";
            }
        });

        get("/users/:username", (req, res) -> {
            // Use select to get the user we just entered
            final String username = req.params(":username");

            try (final Session session = cluster.connect(KEYSPACE)) {
                final ResultSet resultSet = session.execute(format(
                    "SELECT * FROM %s.users WHERE username='%s'", KEYSpace, username));
                final Iterator<Row> iterator = resultSet.iterator();

                if (!iterator.hasNext()) {
                    res.status(404);
                    return "";
                }
            }
        });
    }
}
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
Mail -		Mail -			1-7 of 7
compose	dfpdxia/qc...	com.davidxia.qcondemo.CassandraContainerIT	10 minutes ago	Up 10s	
Inbox	dfpdxia/qc...	com.davidxia.qcondemo.CassandraContainerIT	10 minutes ago	Up 10s	
Drafts (1)	dfpdxia/qc...	com.davidxia.qcondemo.CassandraContainerIT	10 minutes ago	Up 10s	
Notes	dfpdxia/qc...	com.davidxia.qcondemo.CassandraContainerIT	10 minutes ago	Up 10s	
More -	dfpdxia/qc...	com.davidxia.qcondemo.CassandraContainerIT	10 minutes ago	Up 10s	

Run CassandraContainerIT

0 of 1 test

```
"boot":{"class":{"path":"/Library/Java/JavaVirtualMachines/jdk1.8.0_71
.jdk/Contents/Home/jre/lib/resources.jar:/Library/Java/JavaVirtualMachines/jdk1.8
.0_71.jdk/Contents/Home/jre/lib/rt.jar:/Library/Java/JavaVirtualMachines/jdk1.8
.0_71.jdk/Contents/Home/jre/lib/sunrsasign
.jar:/Library/Java/JavaVirtualMachines/jdk1.8.0_71.jdk/Contents/Home/jre/lib/jsse
.jar:/Library/Java/JavaVirtualMachines/jdk1.8.0_71.jdk/Contents/Home/jre/lib/jce
.jar:/Library/Java/JavaVirtualMachines/jdk1.8.0_71
.jdk/Contents/Home/jre/lib/charsets.jar:/Library/Java/JavaVirtualMachines/jdk1.8
.0_71.jdk/Contents/Home/jre/lib/jfr.jar:/Library/Java/JavaVirtualMachines/jdk1.8
.0_71.jdk/Contents/Home/jre/classes"},
"library":{"path":"/Library/Java/JavaVirtualMachines/jdk1.8.0_71
.jdk/Contents/Home/jre/lib}},"cpu":{"endian":"little","isalist":""},
"io":{"unicode":{"encoding":"UnicodeBig"},"java":{"command":"com.intelliJ.rt
.execution.application.AppMain com.intelliJ.rt.execution.junit.JUnitStarter
-ideVersion5 com.davidxia.qcondemo.CassandraContainerIT",
"launcher":"SUN_STANDARD"},"jnu":{"encoding":"UTF-8"},
"management":{"compiler":"HotSpot 64-Bit Tiered Compilers"},
"os":{"patch":{"level":"unknown"}},"user":{"country":"US",
"dir":"/Users/dxia/src/qcon-demo","home":"/Users/dxia","language":"en",
"name":"dxia","timezone":"America/New_York"}})
10:20:24.036 [main] INFO com.spotify.helios.testing.HeliosConfig - No profile found
at helios.solo.profile. Using an empty config.

Process finished with exit code 130
```

Successes

“Testing with a Cassandra container is the closest I can get to testing against Cassandra in reality.”

Successes

“I especially like the fact that I can test my image in its final state and be confident that it will work in production.”

Successes

“Using helios-testing to run datastores in containers has made the tests portable and setup free (by setup I mean no manual installation of the datastore on the test machine or locally).”

Lessons Learned

Make sure your testing framework and infrastructure are fast and reliable.

Lessons Learned

Make framework's interface and implementation as simple as possible.

Lessons Learned

Provide great test examples.

Key Takeaways - Using Containers in Tests Can Help You:

- Test more of the stack in an env resembling production
- Easily start real dependencies
- Ensure tests are reproducible and isolated



When Not to Use Containers in Tests

- Don't test functionality unrelated to containers that you can easily test separately
- When your container-based tests overlaps a lot with regular integration tests



Acknowledgements

Rohan Singh

Matt Brown

Staffan Gimåker

Mats Linander

Nic Cope



Q&A

@davidxia_

github.com/davidxia

github.com/spotify/helios

github.com/davidxia/qcon-demo

Demo videos: example 1 and 2

helios-testing

