tarent ACADEMY

tarent solutions GmbH

Testcontainers: einfach Docker Container in Tests einbinden

Kurz zu mir ...

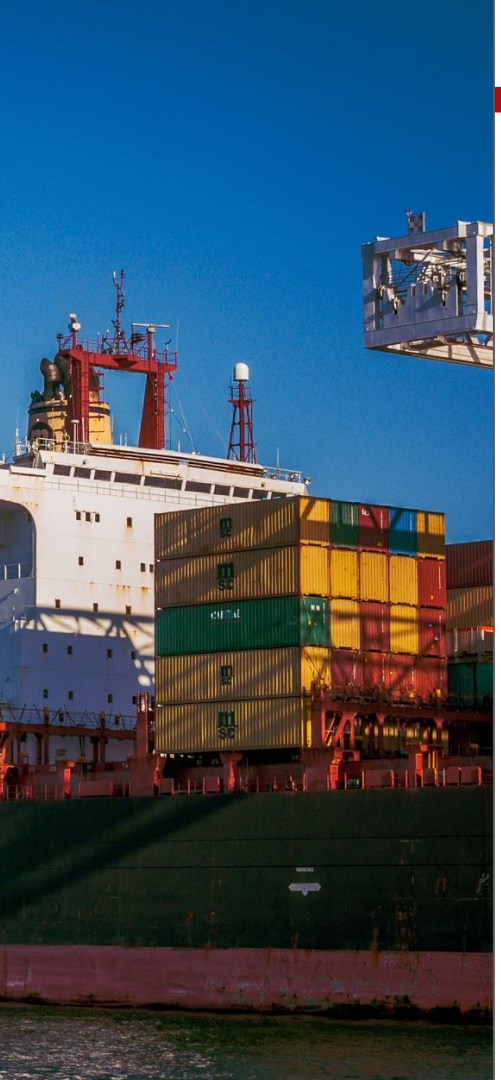
- Tim Steffens, **Softwareentwickler** und **Trainer** bei der tarent
- Interessen
  - **Funktionale Programmierung**
  - **Qualität** von Software
  - **Wandern** & Klettern
- die **tarent**
  - **hilft Kunden** gute Software zu bauen
    - in-house oder vor Ort
    - Schulungen
  - baut **eigene Software** (z.B. snabble.io)
  - hat Standorte in **Bonn, Köln, Berlin & Bukarest**


Wer seid Ihr?


Agenda
- Docker
- Testcontainers
- Live Coding Session #1
- Selenium
- Live Coding Session #2

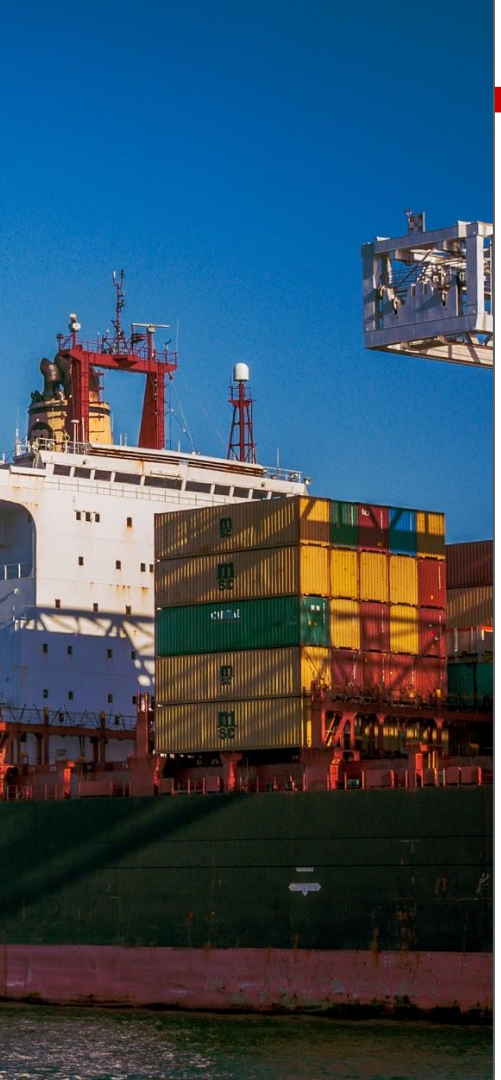

Unterbrecht mich!


BRITISH AIRWAYS
Docker



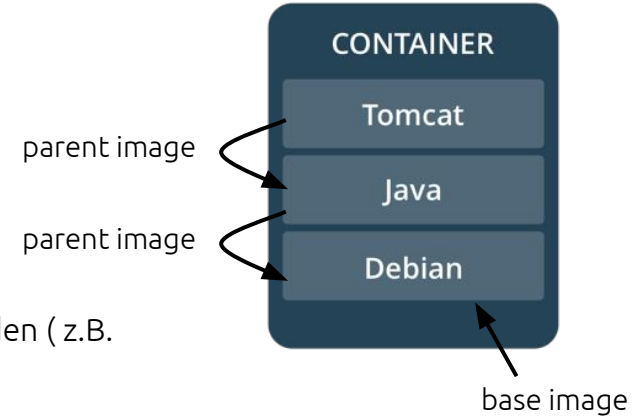
Was ist ein Docker Container?

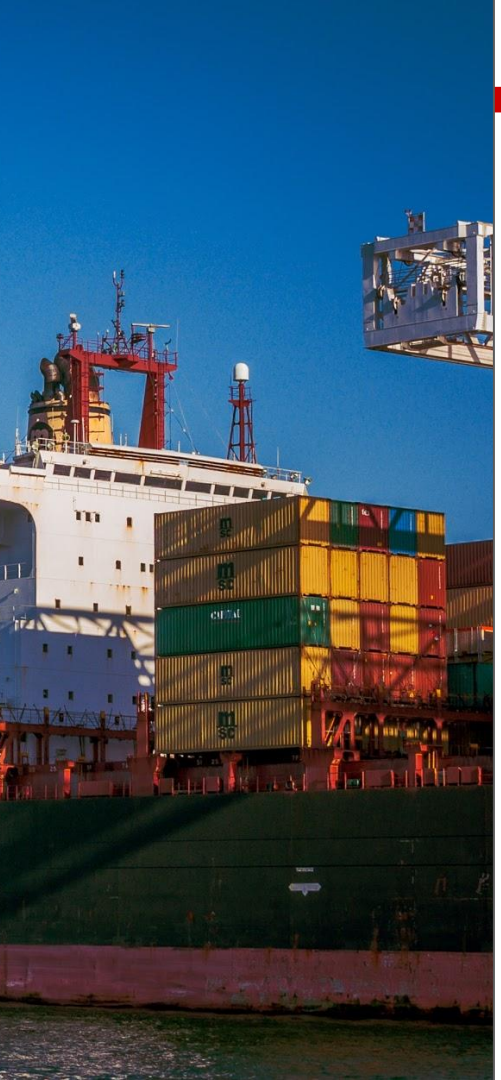
- ein **Linux Prozess** in einer einer **isolierten Umgebung** ausgeführt wird (kernel namespaces und cgroups)
- **Sandbox** mit definierten Ressourcen (**Dateisystem**, Netzwerk, ...)
    - "*Sandbox ist die englischsprachige Bezeichnung für Sandkiste oder Sandkasten und bezeichnet allgemein einen isolierten Bereich, innerhalb dessen jede Maßnahme keine Auswirkung auf die äußere Umgebung hat.*" - Wikipedia
- benutzt den **Kernel** vom zugrunde liegenden OS
    - ist also **keine virtuelle Maschine**
- basiert auf einem **Docker Image**



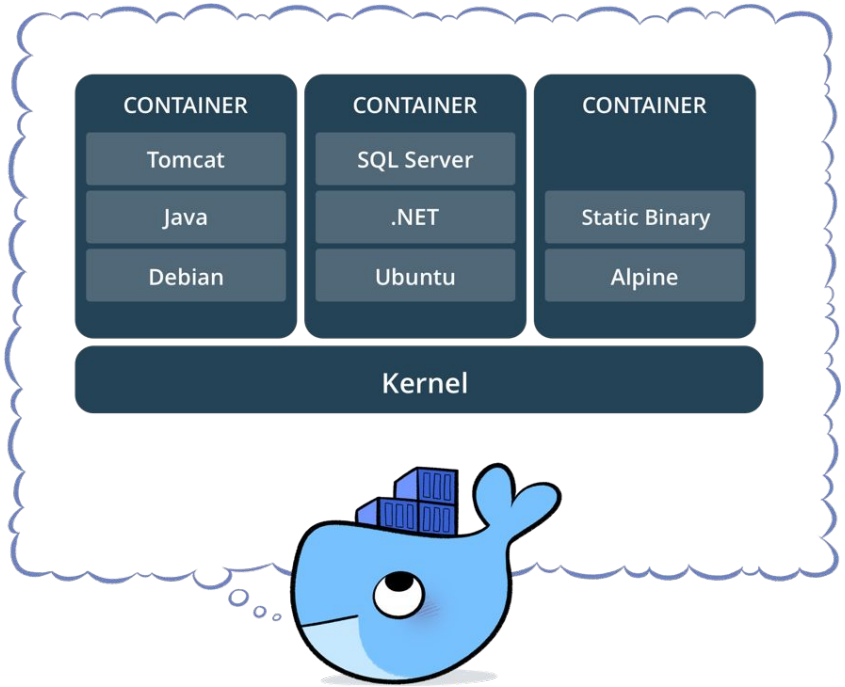
Was ist ein Docker Image?

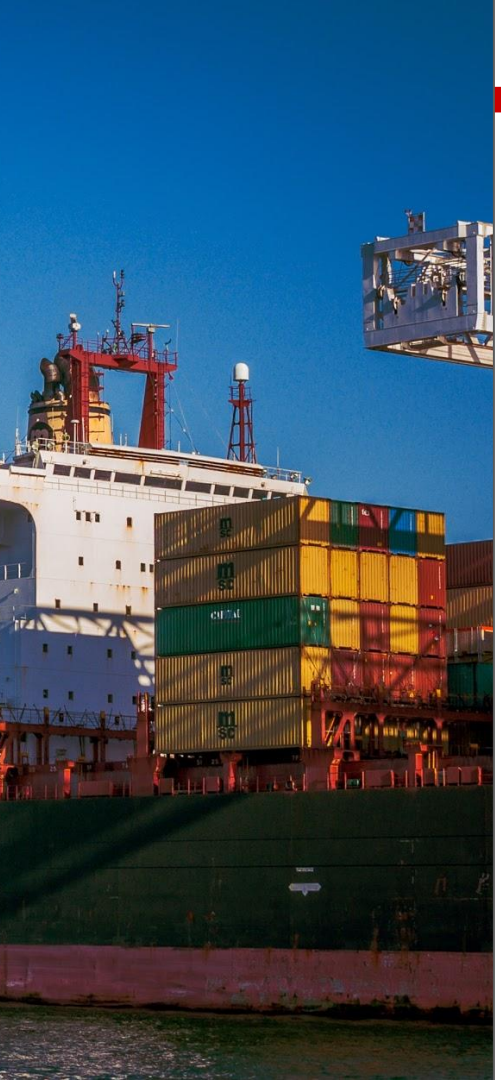
- **Template** für Container
- schreibgeschütztes **Speicherabbild**
- **FS-Diff** zu dem **Parent Image**
- kann aus einer **Registry** geladen werden (z.B. https://hub.docker.com/)
- **Identifikation** über **<repository>:<Tag>** , z.B.: postgres:11.5
  - Default für “Tag”: latest
  - bei abweichender Registry: my-registry.com:4711/my-image:0.1





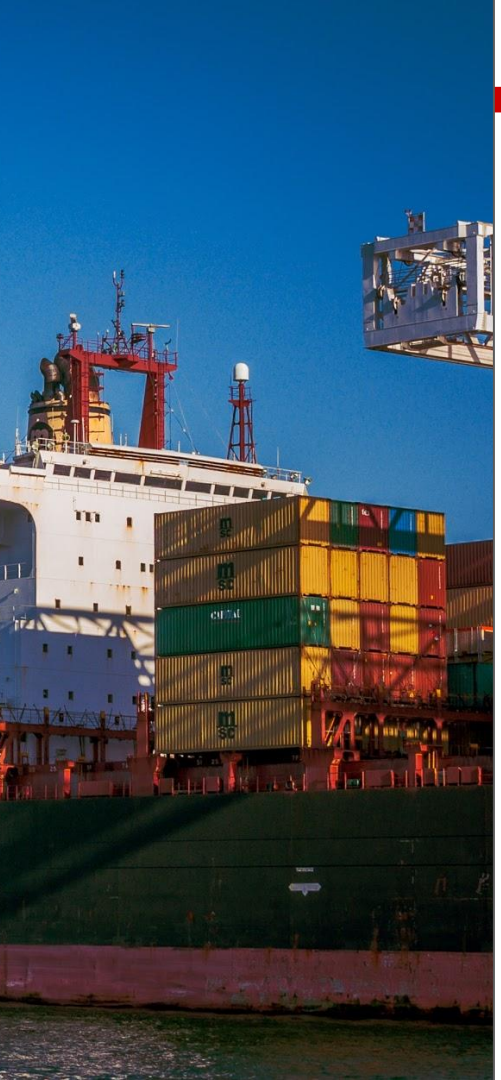
Container & Images





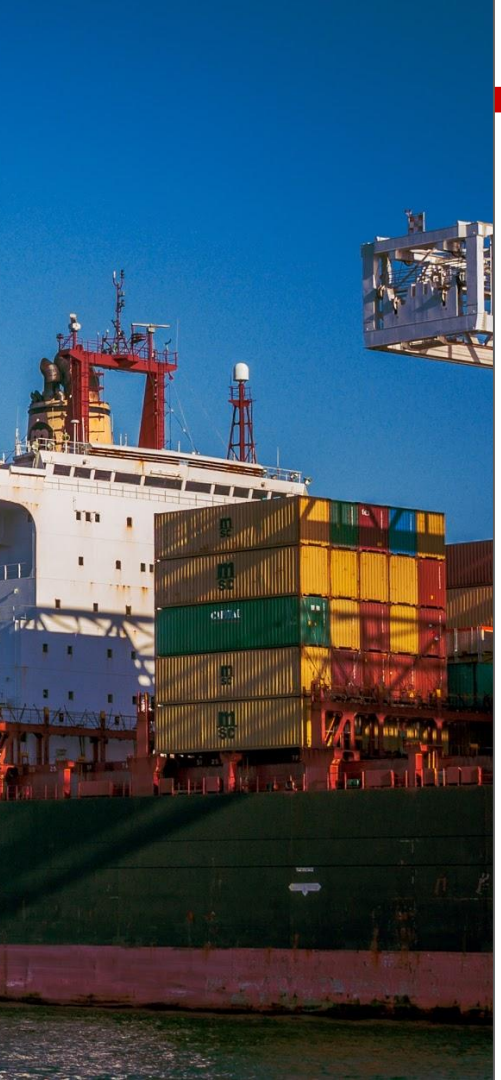
Docker Engine

- Sammlung von **Tools & Bibliotheken**
- u.a. **docker**-Kommando
- **Container verwalten**
  - starten / stoppen
  - auflisten
  - löschen
  - …
- **Images verwalten**
  - downloaden
  - erstellen, auflisten, löschen
  - …
- ...



Container Starten

```
docker run -p 8080:80 httpd:2.4
```

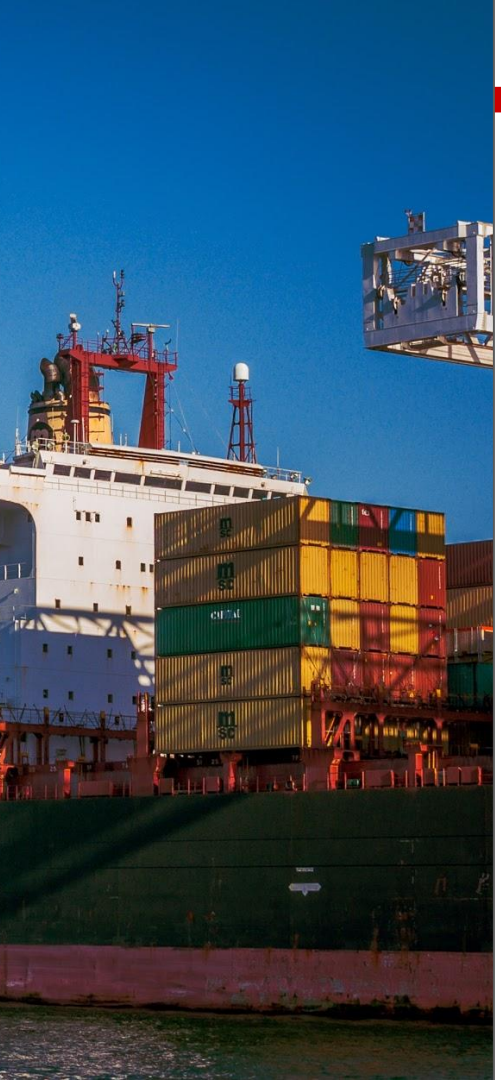


Container Starten

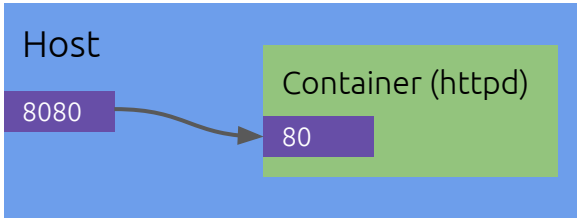
Port-Forwarding

```
docker run -p 8080:80 httpd:2.4
```

Image



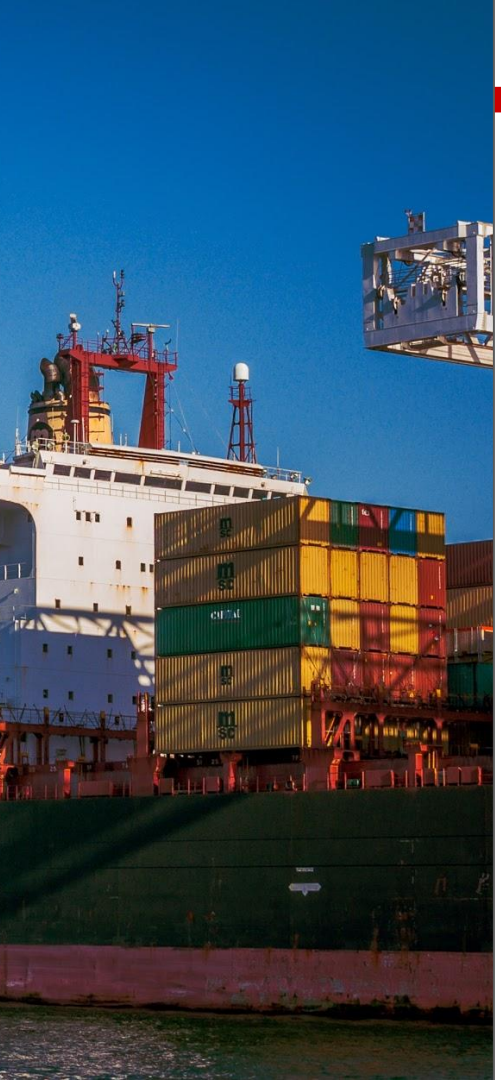
Container Starten



Port-Forwarding

```
docker run -p 8080:80 httpd:2.4
```

Image



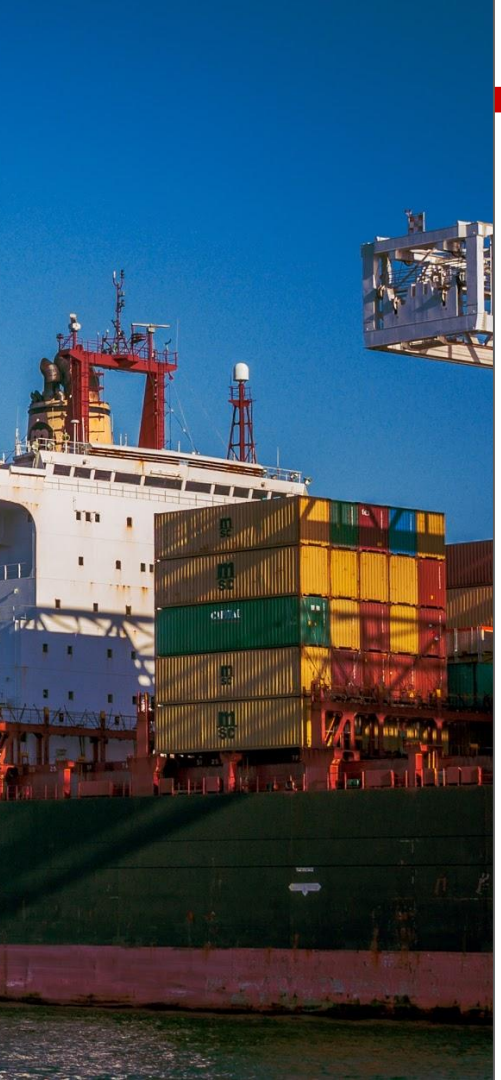
Images bauen

- **Dockerfile** anlegen

```
FROM nginx
COPY index.html /usr/share/nginx/html/
```

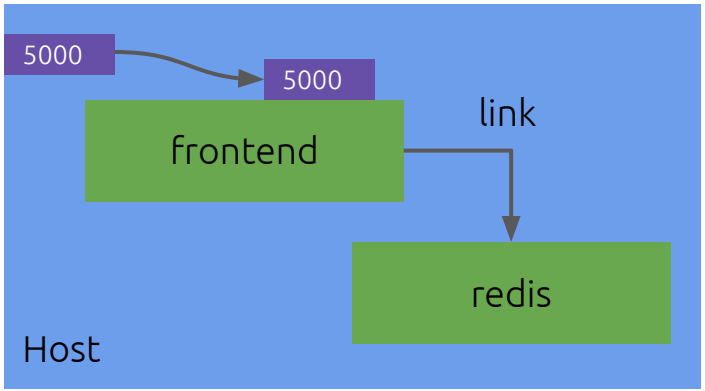
- **Image bauen** und Namen geben

```
docker build --pull -t my-nginx .
```

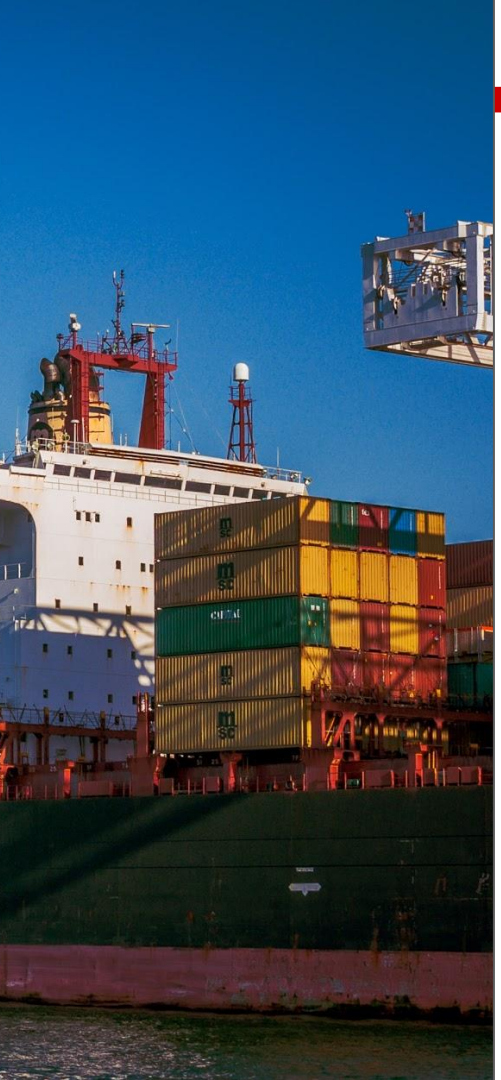


Docker Compose

- Tool um eine **Multi-Container-Applikationen** zu **definieren** und **auszuführen**
- **YAML Datei** um Container zu **konfigurieren**
- **einzelnes Kommando** um **alle Container** der Konfiguration zu erstellen und zu **starten**

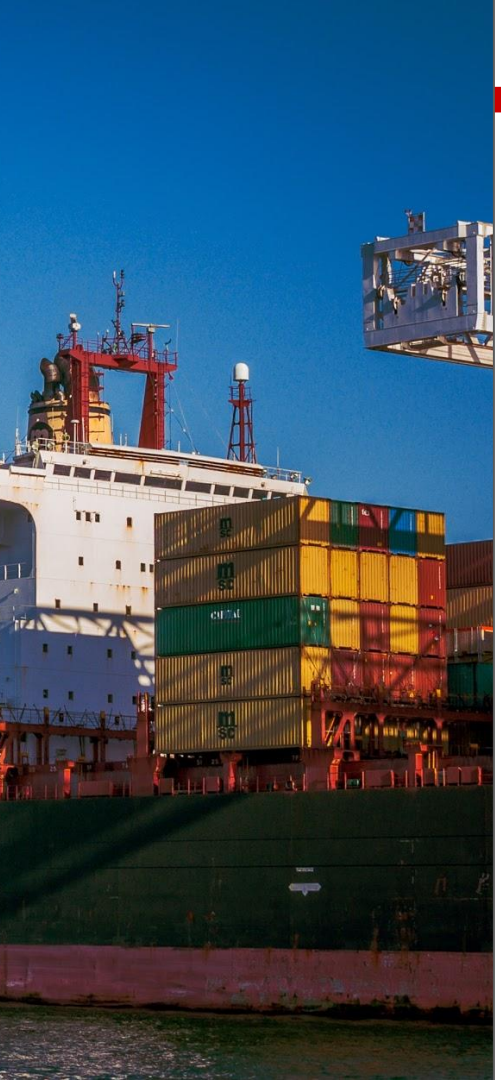



```
version: '3'
services:
  frontend:
    build: .
    ports:
    - "5000:5000"
    links:
    - redis
  redis:
    image: redis
```



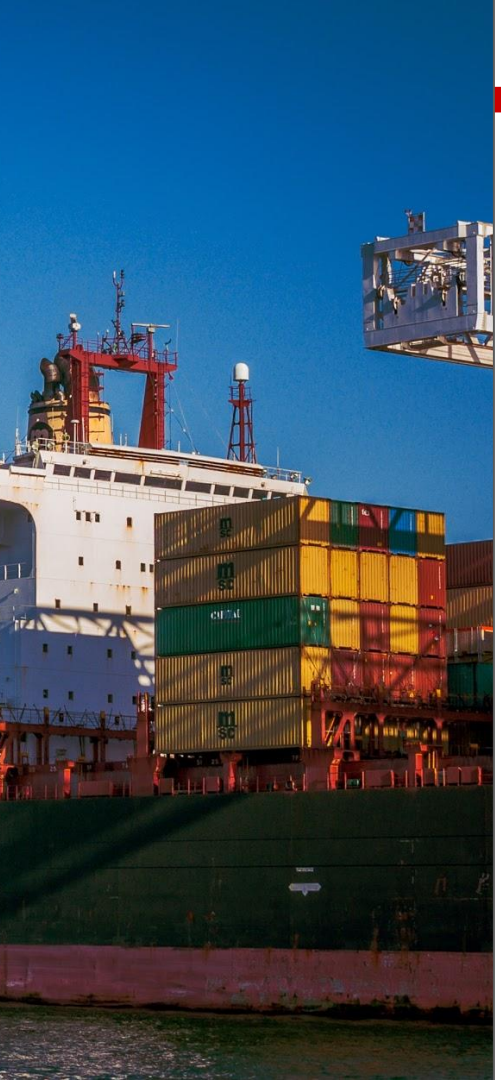
Installation

- docker engine: https://docs.docker.com/install/
- docker compose: https://docs.docker.com/compose/install/



Warum Container?

- **einfaches Handling**
  - einzige Installations-Abhängigkeit zum Start eines Containers: **docker** - das System bleibt ansonsten unverändert!
  - für Entwicklung: leichtgewichtig z.B. unterschiedliche Datenbanken ausprobieren
- **portabel**
  - Java App: Java Version + ggf. JCE Policies
  - C++ App: libs & header files in richtiger Version
- Es ist einfach die **exakte Version** aus **PROD** zu bekommen
- **einfaches Deployment** & **einfache Skalierung** in der **Cloud**



Warum Container für Tests?

- Direkt mit der **richtigen Version** testen
  - z.B. **DB**-Version aus **Prod** anstelle einer **In-Memory-Datenbank** testen (z.B. H2)
  - Verhalten kann sich stark unterscheiden!
- **Kompatibilität** z.B. mit **verschiedenen Datenbanken** lässt sich einfach prüfen
- **Selenium Grid** funktioniert **out of the box** - auf jedem Rechner!
- mit **docker-compose** lässt sich **lokal** einfach eine **komplett integrierte Umgebung** simulieren


Testcontainers

Testcontainers

- Sammlung von Bibliotheken um **Docker Container einfach** aus einem Test **starten** zu können
- Verfügbar für verschiedene Sprachen / Plattformen:
  - **java**, go, js, python, scala etc.
  - https://github.com/testcontainers/
- **Open Source** (MIT License)
- am meisten verwendet: **testcontainers-java**
  - https://github.com/testcontainers/testcontainers-java
  - Dokumentation: https://www.testcontainers.org/
  - Einfache Einbindung in **JUnit** und **Spock**


Spock
weder noch!
JUnit

Testcontainers

- Unterstützte Container-Typen / Module
  - Docker Compose Module
  - Elasticsearch container
  - Kafka Containers
  - Mockserver Module
  - Nginx Module
  - RabbitMQ Module
  - **Webdriver Containers**
  - …
  - **Generic Container**

Testcontainers Einbinden

- Gradle:

```
testCompile "org.testcontainers:testcontainers:1.12.1"
```

- Maven

```
<dependency>
   <groupId>org.testcontainers</groupId>
   <artifactId>testcontainers</artifactId>
   <version>1.12.1</version>
   <scope>test</scope>
</dependency>
```

- neuste Version: https://search.maven.org/search?q=a:testcontainers


Maven
weder noch!
Gradle

Mit Containern arbeiten

```
import org.testcontainers.containers.GenericContainer

GenericContainer apache =
       new GenericContainer("httpd:2.4")
               .withExposedPorts(80)

apache.start()

println("Accessible via
     http://localhost:${apache.getMappedPort(80)} ")

System.in.newReader().readLine()

apache.stop()
```

Mit Containern arbeiten

```
import org.testcontainers.containers.GenericContainer

GenericContainer apache =
       new GenericContainer("httpd:2.4")
               .withExposedPorts(80)

apache.start()

println("Accessible via
     http://localhost:${apache.getMappedPort(80)} ")

System.in.newReader().readLine()

apache.stop()
```

Ports werden automatisch auf freien Random-Port gemapped

Postgres Container

```
import org.testcontainers.containers.PostgreSQLContainer


PostgreSQLContainer postgres =
        new PostgreSQLContainer()

postgres.start()

println("Accessible via ${postgres.getJdbcUrl()} ")

System.in.newReader().readLine()

postgres.stop()
```

Integration in Junit 4

```
@ClassRule
public static GenericContainer apache =
    new GenericContainer("httpd:2.4")
            .withExposedPorts(80);
```

- Start und Stop wird von Testcontainers mittels @ClassRule vorgenommen
- Der Ryuk Sidecar-Container stellt sicher, dass alle networks, volumes und evtl. nicht gestoppte Container entfernt werden: https://github.com/testcontainers/moby-ryuk

Integration in Spock

```
testCompile "org.testcontainers:spock:1.12.1"
```

```
--
```

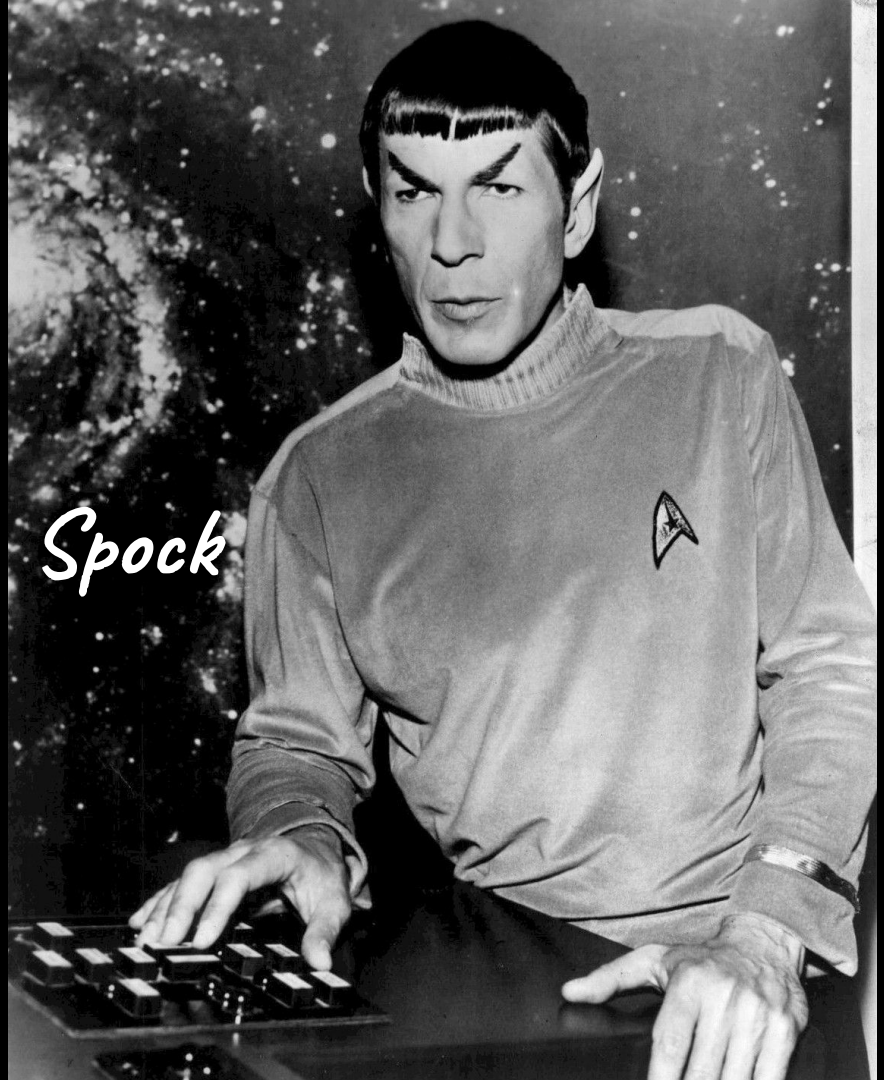
```
@Testcontainers
class DatabaseTest extends Specification {

    @Shared
    GenericContainer apache =
         new GenericContainer("httpd:2.4")
                 .withExposedPorts(80)
```

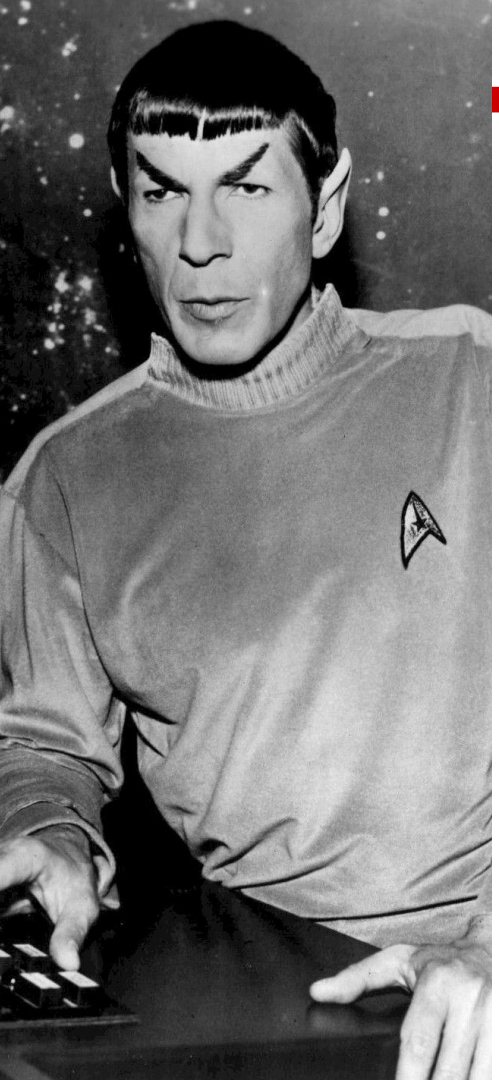
- @Shared gibt an, dass Container zwischen den einzelnen Tests nicht neu gestartet werden sollen

Integration in Spring Boot

in application.yml:

```
spring:
 jpa.hibernate.ddl-auto : none
 datasource:
   driver-class-name :
org.testcontainers.jdbc.ContainerDatabaseDriver
   url: jdbc:tc:mysql:5.7.22://localhost/ta_youtrain
   username: tarent
   password: tarent
```


Spock

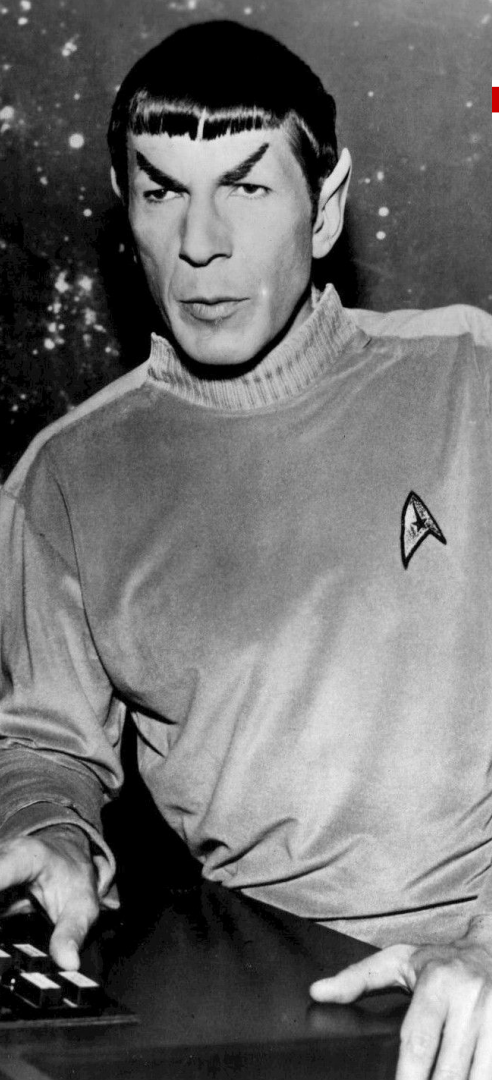


Spock

*"Spock is a **testing** and specification **framework** for **Java** and **Groovy** applications. What makes it stand out from the crowd is its **beautiful** and **highly expressive** specification **language**. Thanks to its **JUnit runner**, Spock is compatible with most IDEs, build tools, and continuous integration servers."*

- http://spockframework.org/spock/docs/1.3/introduction.html

- basiert auf der JVM Sprache Groovy: http://www.groovy-lang.org/

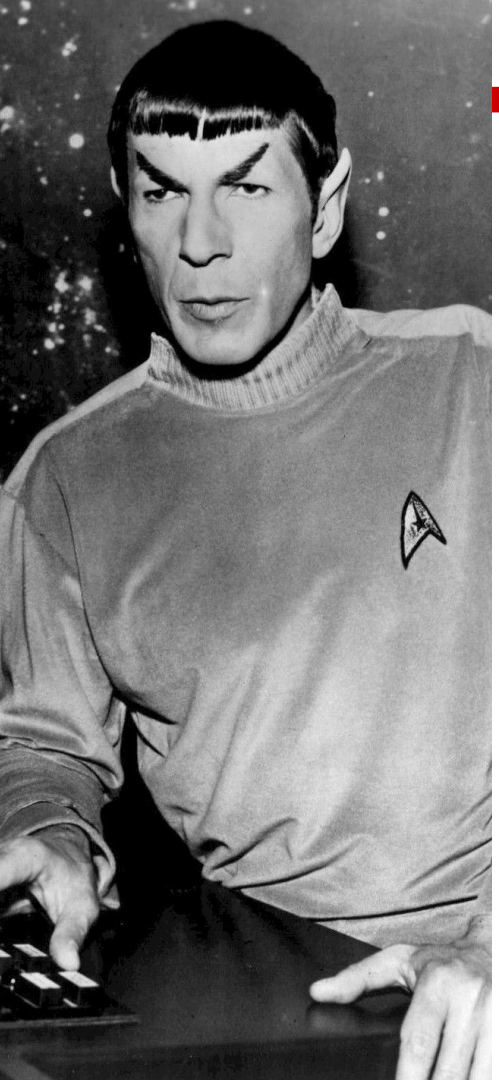


Groovy

- **Script-Sprache**, die dynamische und statische typisierung zulässt
- läuft Java Virutal Machine (**JVM**)

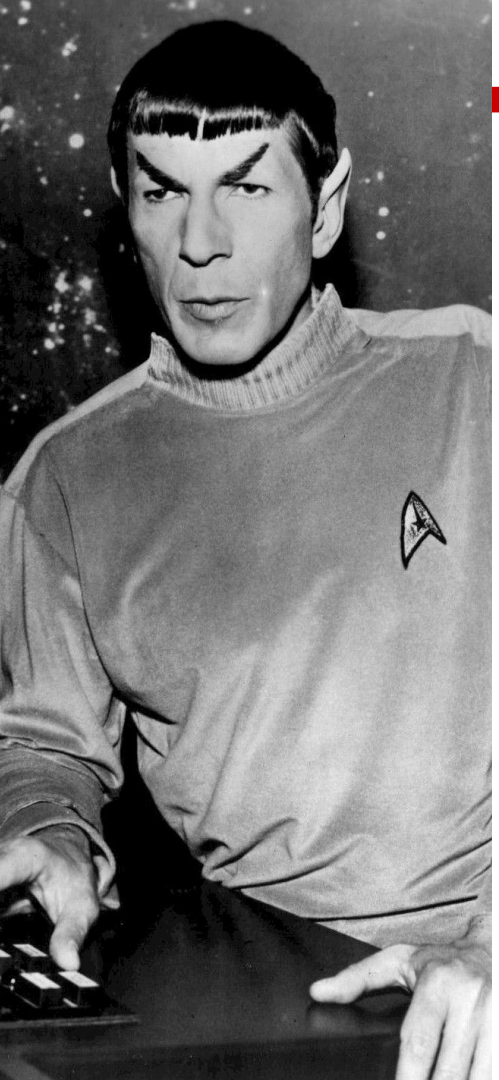
```
def list = [1,2,3,4]
[1,2,3,4] == [1,2,3,4] // = true!!

def map = [name: "Arnold", role: "Agile Tester"]
map.name // = "Arnold"

class Employee { String name; String role }
def employee = new Employee(name: "Arnold", role: "Agile
Tester")
employee.name // = "Arnold"
```


Java
weder noch!
Groovy



Beispiel

```
class ListSpec extends Specification {

    def "Should be able to remove from list"() {
        given:
        def list = [1, 2, 3, 4]

        when:
        list.remove(0)

        then:
        list == [2, 3, 4]
    }

}
```



Einbinden

- gradle

```
apply plugin: "groovy"

implementation "org.codehaus.groovy:groovy-all:2.5.7"

testCompile
"org.spockframework:spock-core:1.3-groovy-2.5"
```

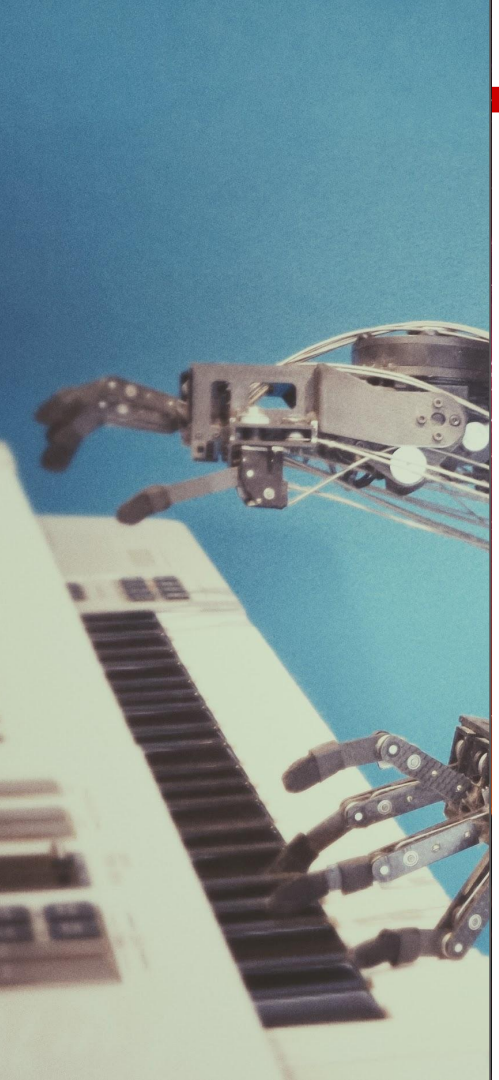
- maven

  see https://github.com/spockframework/spock-example/blob/master/pom.xml


Live Coding Session #1

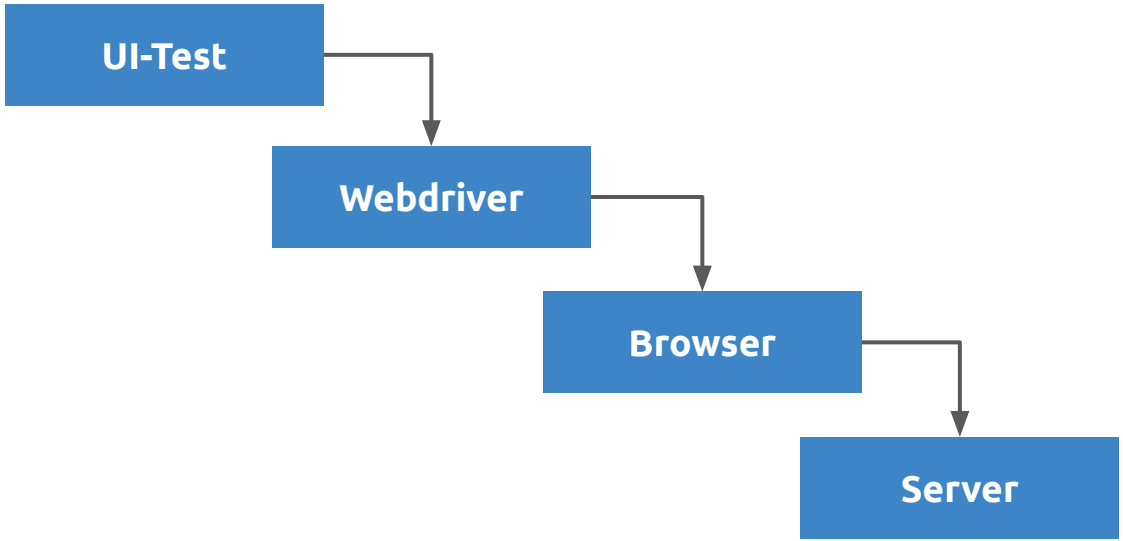
Selenium

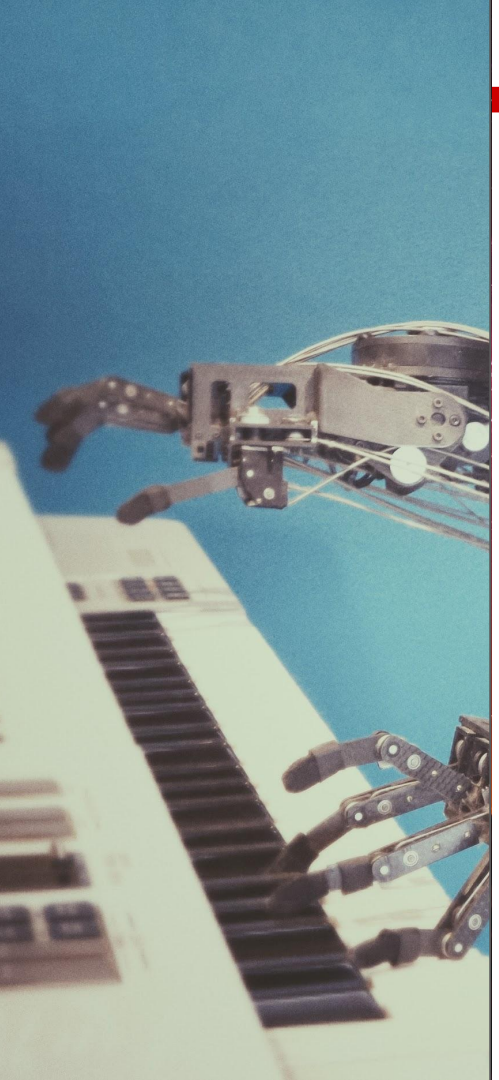

Selenium
weder noch!
Geb



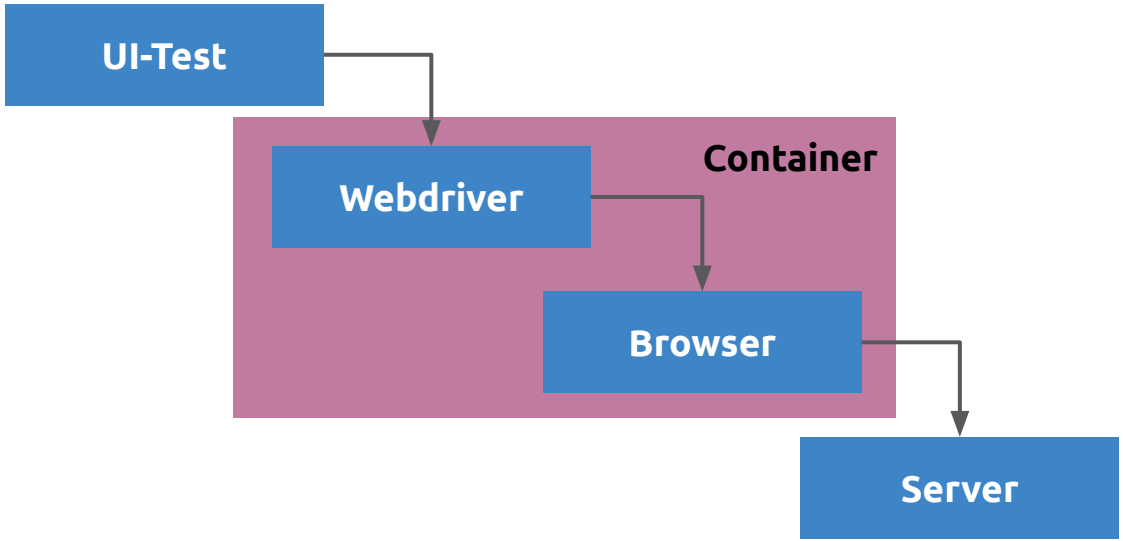
Was ist Selenium?

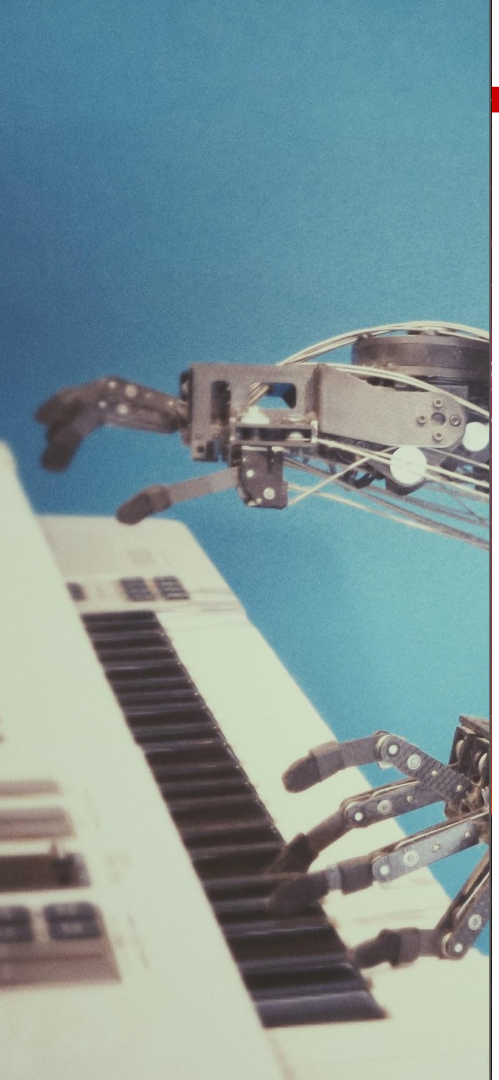
- *Selenium automates browsers*. That's it! - https://www.seleniumhq.org/





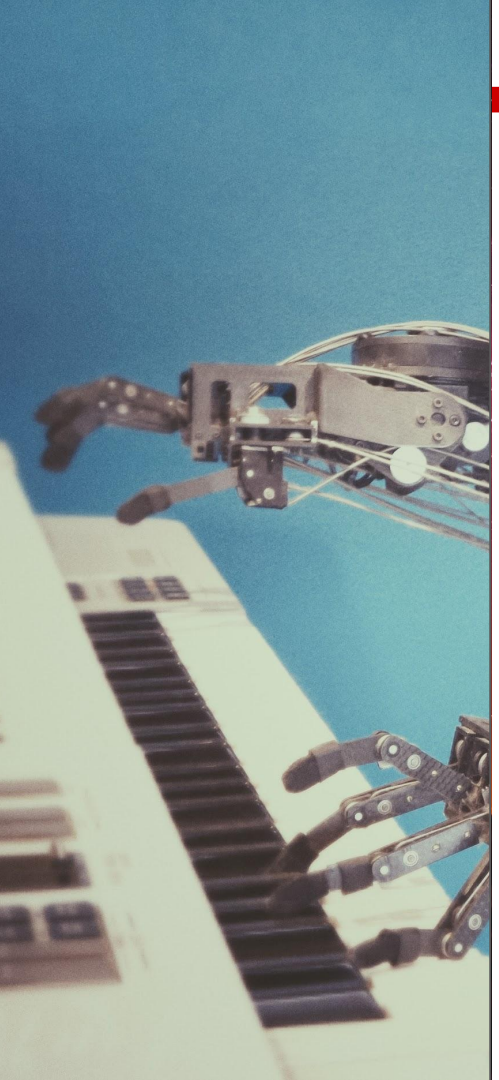
Selenium im Container





Selenium Webdriver Container

```
import org.openqa.selenium.remote.RemoteWebDriver
import org.testcontainers.containers.BrowserWebDriverContainer
import static org.testcontainers.containers.
      BrowserWebDriverContainer.VncRecordingMode. RECORD_ALL


BrowserWebDriverContainer chrome = new
      BrowserWebDriverContainer()
         .withCapabilities( new ChromeOptions())
         .withRecordingMode( RECORD_ALL, new File("./"))

chrome.start()

RemoteWebDriver driver = chrome.getWebDriver()
```



Geb

- https://gebish.org
- **DSL** für **UI-Tests**
- Setzt auf **Selenium** auf
- einfache und **schöne Syntax**

```
Browser.drive {
    go "http://myapp.com/login"
    assert $("h1").text() == "Please Login"
    $("form.login").with {
        username = "admin"
        password = "password"
        login().click()
    }
    assert $("h1").text() == "Admin Section"
}
```


Live Coding Session #2

tarent solutions GmbH

Wenn Ihr Hilfe braucht

- beim **automatisierten Testen**
  (Docker, Junit, Groovy/Spock, UI-Tests mit Selenium/Geb, Gatling, ...),
- einem **Software Projekt** im allgemeinen
  (Webdevelopment, Microservices, AI, Machine Learning, AR / VR)
- oder beim Aufbauen von **Wissen und Skills** in diesen Gebieten
  (**Training**)

Sprecht mich gerne an!

tarent solutions GmbH

Kotlin Schulung von Christian Schmidt @ JavaLand 2020

- JavaLand 2020
  - Konferenz rund um Java
  - 17. bis 19. März 2020
  - Phantasialand Brühl (bei Bonn)
  - https://www.javaland.eu/

- Kotlin – Eine praktische Einführung
  - Syntax & Konzepte
  - Übung: Webservice mit REST-Api

tarent solutions GmbH

Vielen Dank!

Tim Steffens
Softwareentwickler & Trainer
t.steffens@tarent.de
https://github.com/tmstff
@tmstff

Rochusstraße 2-4
53123 Bonn

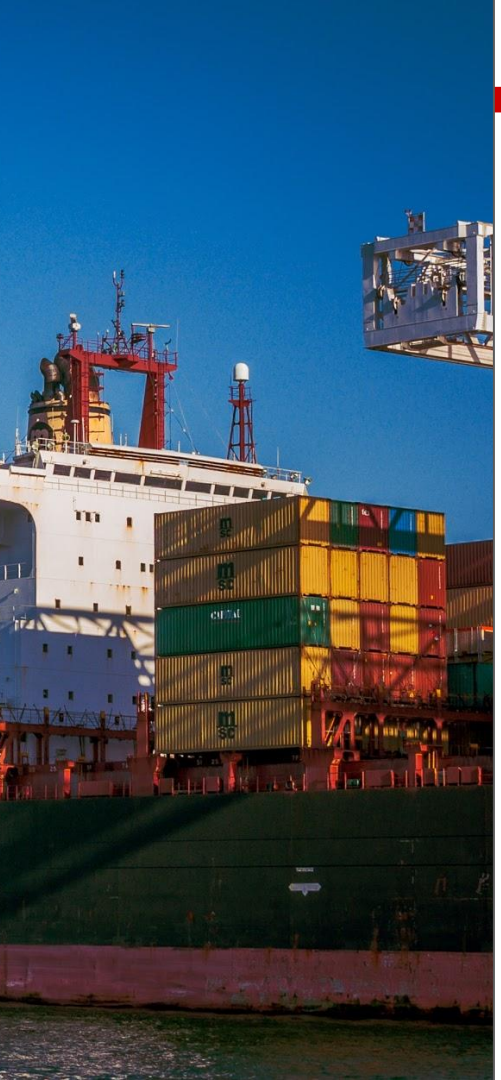
Telefon: 0228 54881-0
Telefax: 0228 54881-235

academy@tarent.de
www.tarent.de/academy

Feedback-Formular, Slides & Code:

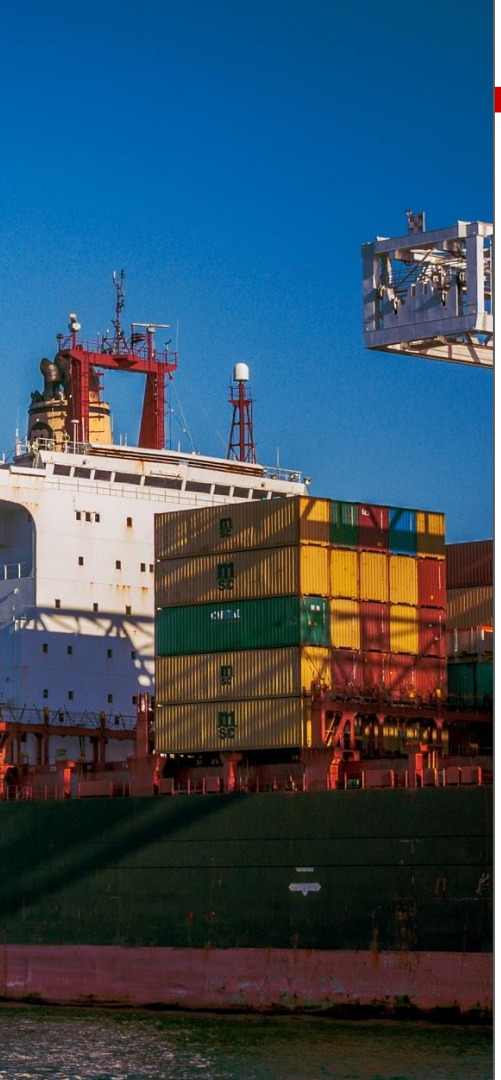
https://bit.ly/2sNeBXK


Backup



Tip: regelmäßig aufräumen

```
$ docker system prune

WARNING! This will remove:
        - all stopped containers
        - all networks not used by at least one container
        - all dangling images
        - all build cache
Are you sure you want to continue? [y/N] y
```



Tip: regelmäßig aufräumen

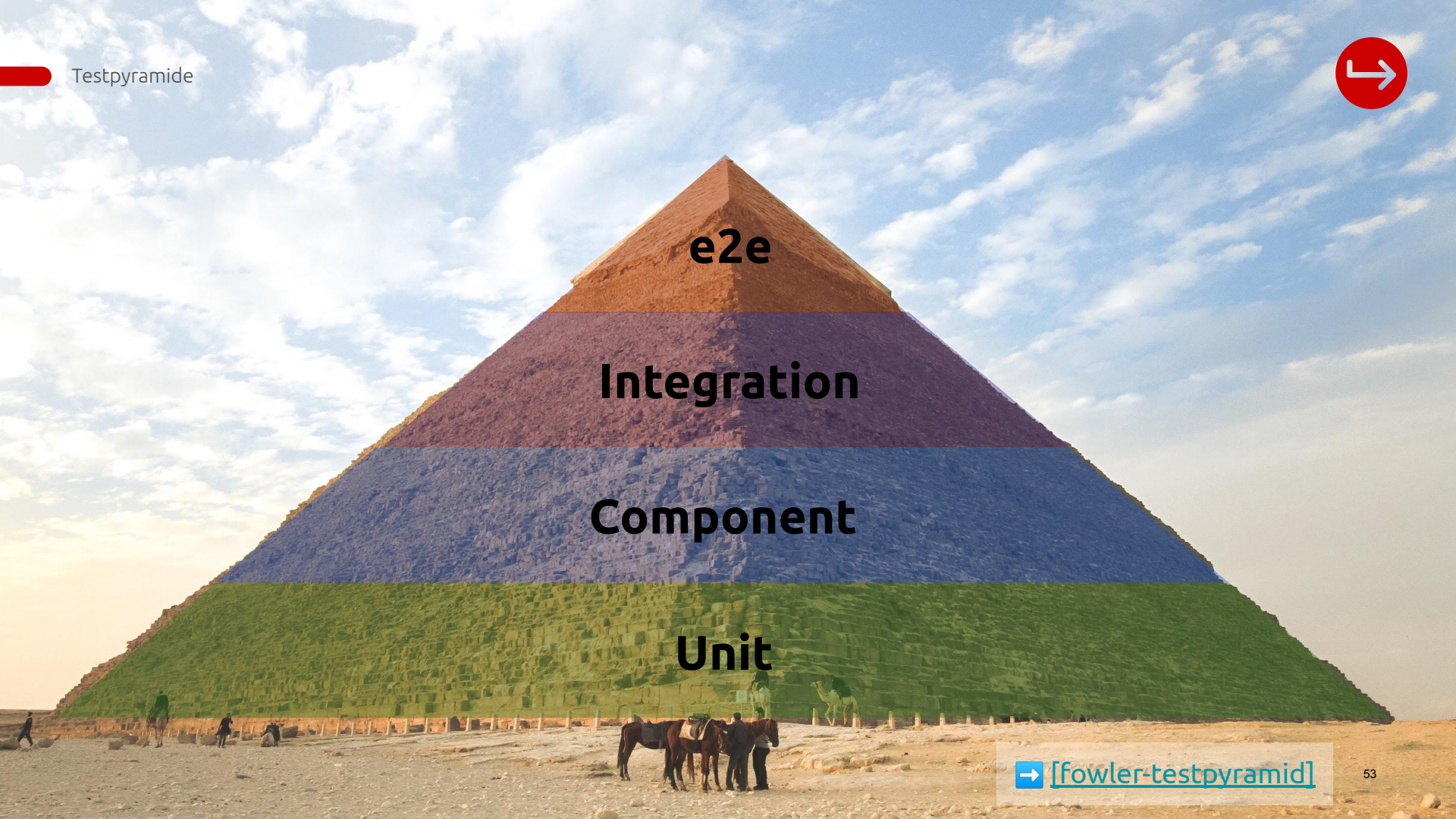
```
$ docker system prune -a --volumes

WARNING! This will remove:
        - all stopped containers
        - all networks not used by at least one container
        - all volumes not used by at least one container
        - all images without at least one container associated to them
        - all build cache
Are you sure you want to continue? [y/N] y
```

Zugriff Container Logs

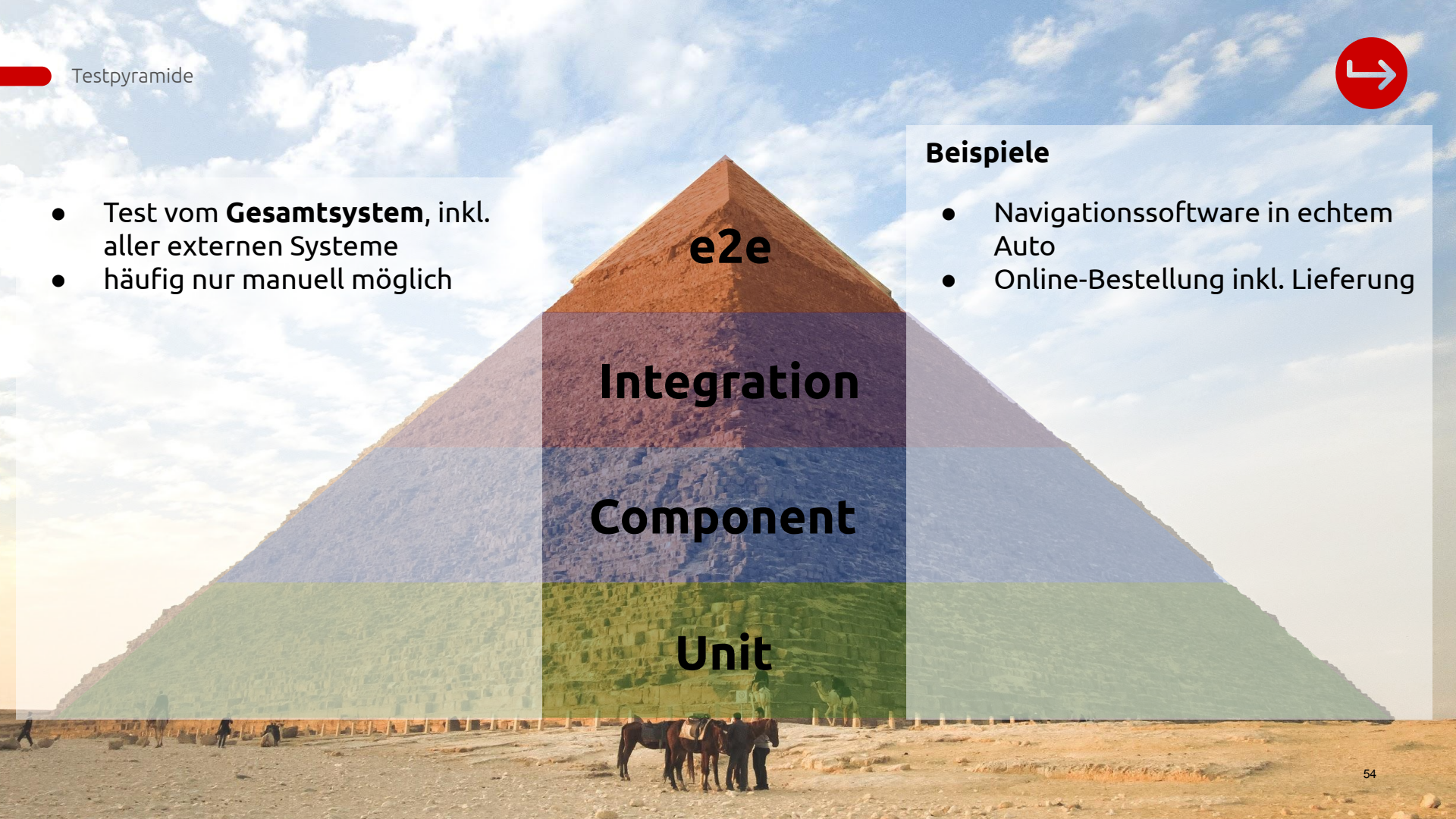
- `final String logs = container.getLogs();`
- Oder zu einem SLF4J logger schicken:

```
container.followOutput(new
    Slf4jLogConsumer(LOGGER));
```


Testpyramide



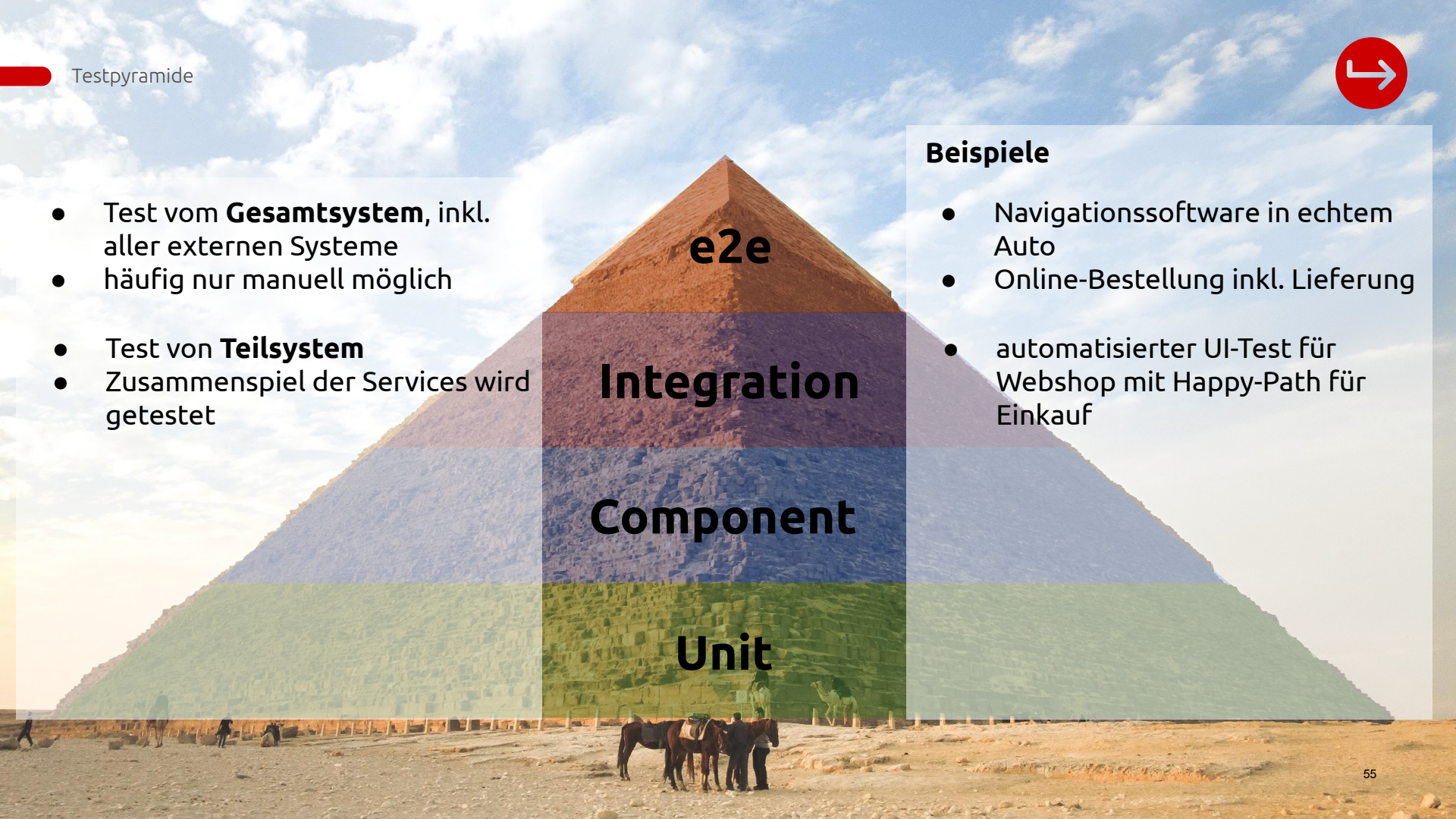
➡️ [fowler-testpyramid]



- Test vom **Gesamtsystem**, inkl. aller externen Systeme
- häufig nur manuell möglich

Beispiele

- Navigationssoftware in echtem Auto
- Online-Bestellung inkl. Lieferung



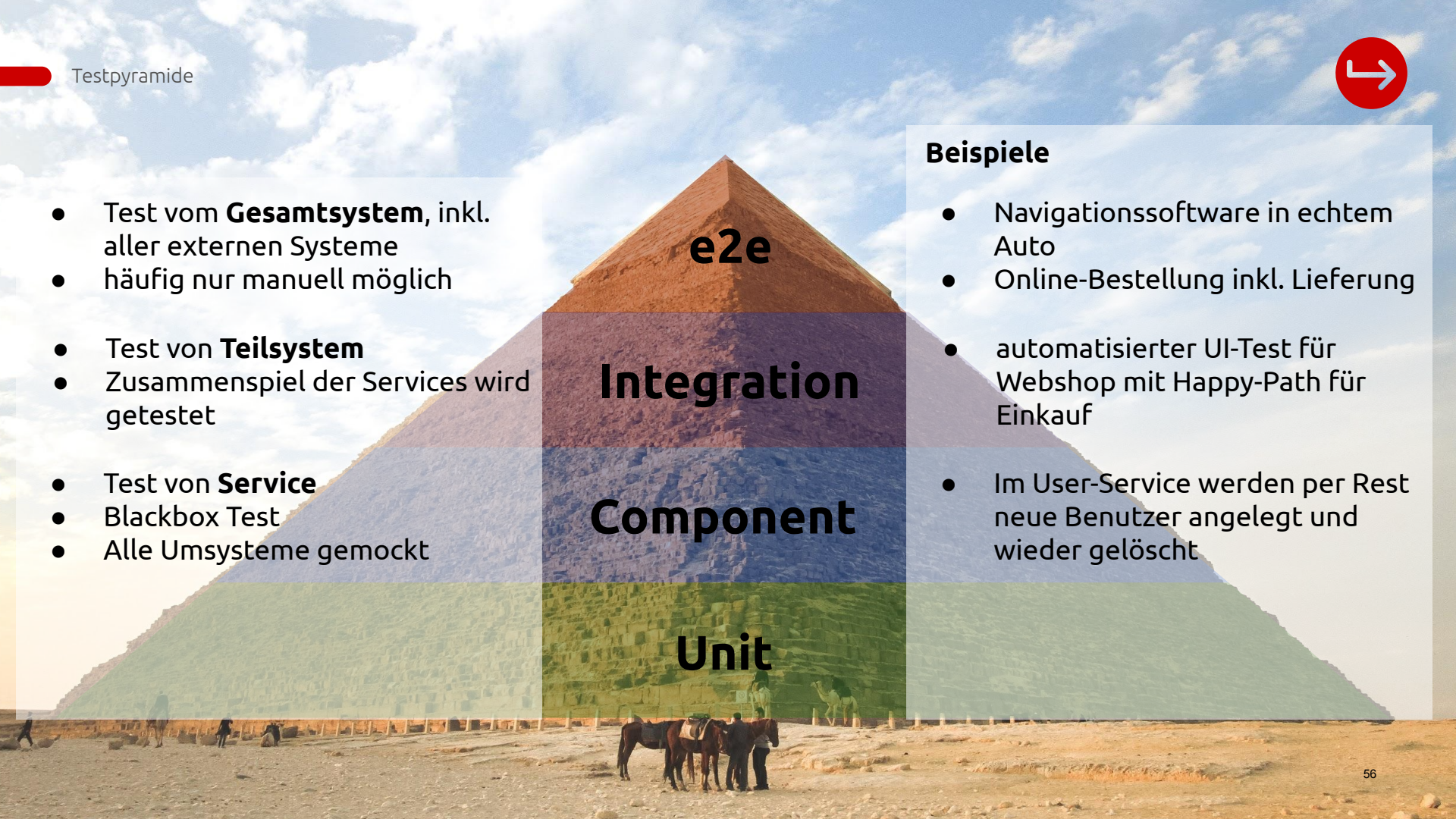
- Test vom **Gesamtsystem**, inkl. aller externen Systeme
- häufig nur manuell möglich

- Test von **Teilsystem**
- Zusammenspiel der Services wird getestet

Beispiele

- Navigationssoftware in echtem Auto
- Online-Bestellung inkl. Lieferung

- automatisierter UI-Test für Webshop mit Happy-Path für Einkauf



- Test vom **Gesamtsystem**, inkl. aller externen Systeme
- häufig nur manuell möglich

- Test von **Teilsystem**
- Zusammenspiel der Services wird getestet

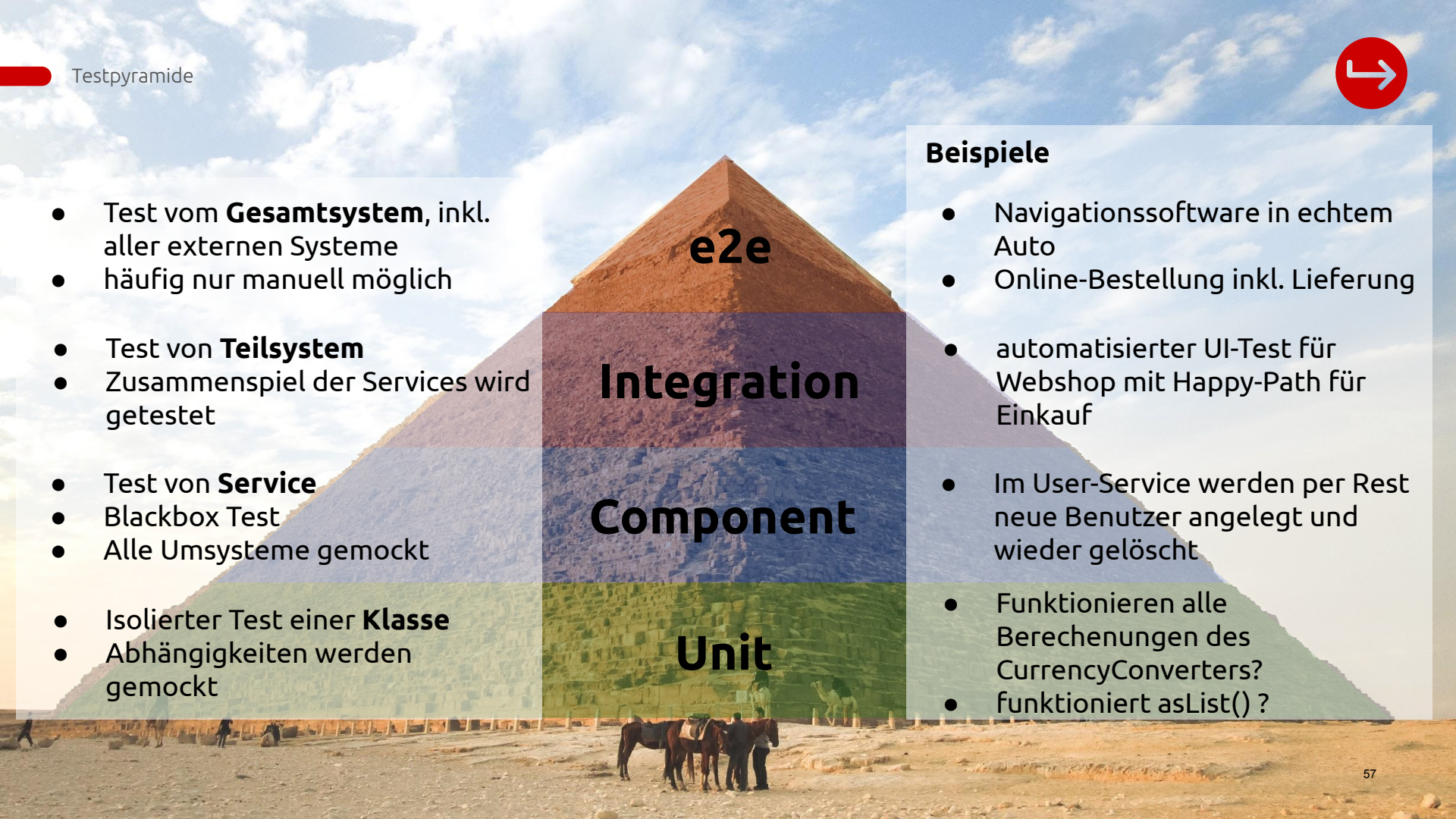
- Test von **Service**
- Blackbox Test
- Alle Umsysteme gemockt

Beispiele

- Navigationssoftware in echtem Auto
- Online-Bestellung inkl. Lieferung

- automatisierter UI-Test für Webshop mit Happy-Path für Einkauf

- Im User-Service werden per Rest neue Benutzer angelegt und wieder gelöscht



- Test vom **Gesamtsystem**, inkl. aller externen Systeme
- häufig nur manuell möglich

- Test von **Teilsystem**
- Zusammenspiel der Services wird getestet

- Test von **Service**
- Blackbox Test
- Alle Umsysteme gemockt

- Isolierter Test einer **Klasse**
- Abhängigkeiten werden gemockt

Beispiele

- Navigationssoftware in echtem Auto
- Online-Bestellung inkl. Lieferung

- automatisierter UI-Test für Webshop mit Happy-Path für Einkauf

- Im User-Service werden per Rest neue Benutzer angelegt und wieder gelöscht

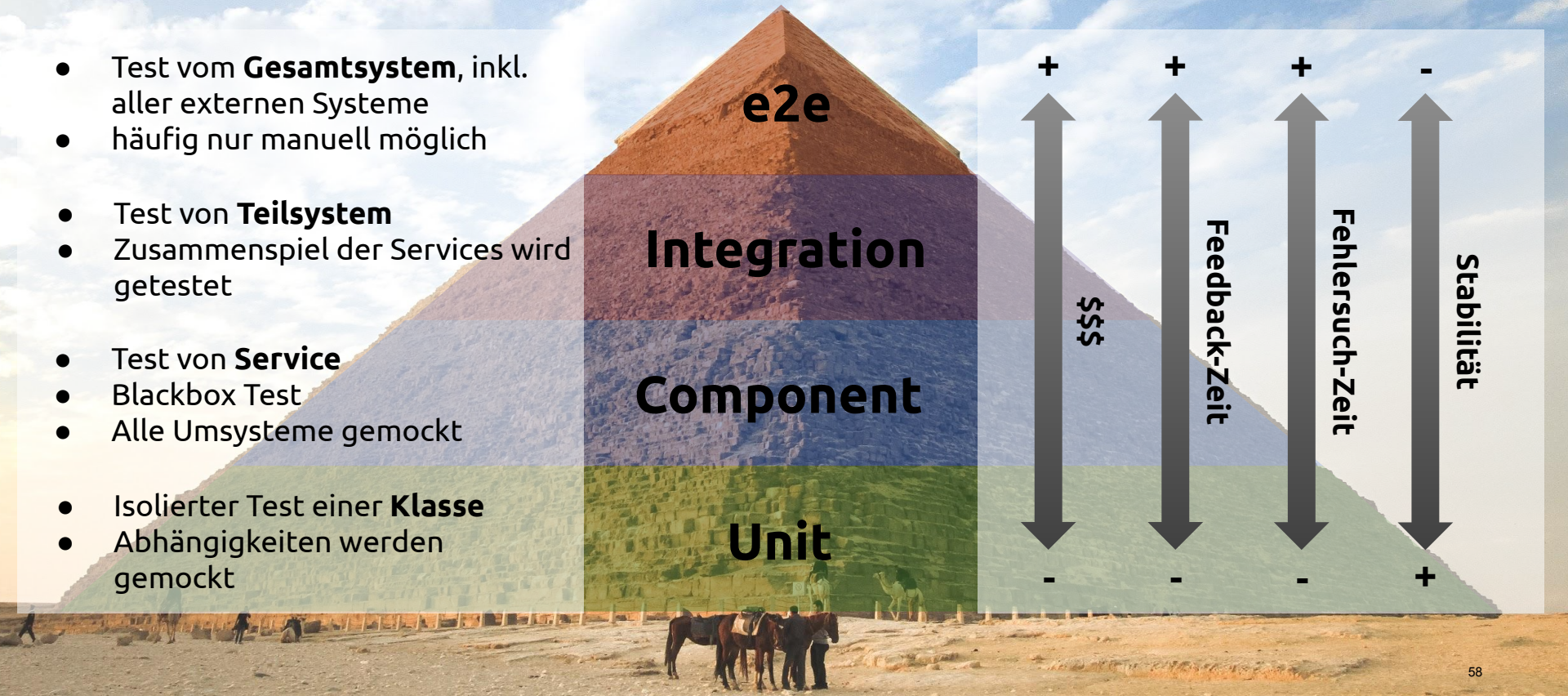
- Funktionieren alle Berechenungen des CurrencyConverters?
- funktioniert asList() ?

- Test vom **Gesamtsystem**, inkl. aller externen Systeme
- häufig nur manuell möglich

- Test von **Teilsystem**
- Zusammenspiel der Services wird getestet

- Test von **Service**
- Blackbox Test
- Alle Umsysteme gemockt

- Isolierter Test einer **Klasse**
- Abhängigkeiten werden gemockt



Für diesen Workshop besonders relevant