

31.8.–3.9.2015
in Nürnberg



Herbstcampus

Wissenstransfer
par excellence

REST-Services ruck-zuck fertig

Microservices mit Dropwizard entwickeln

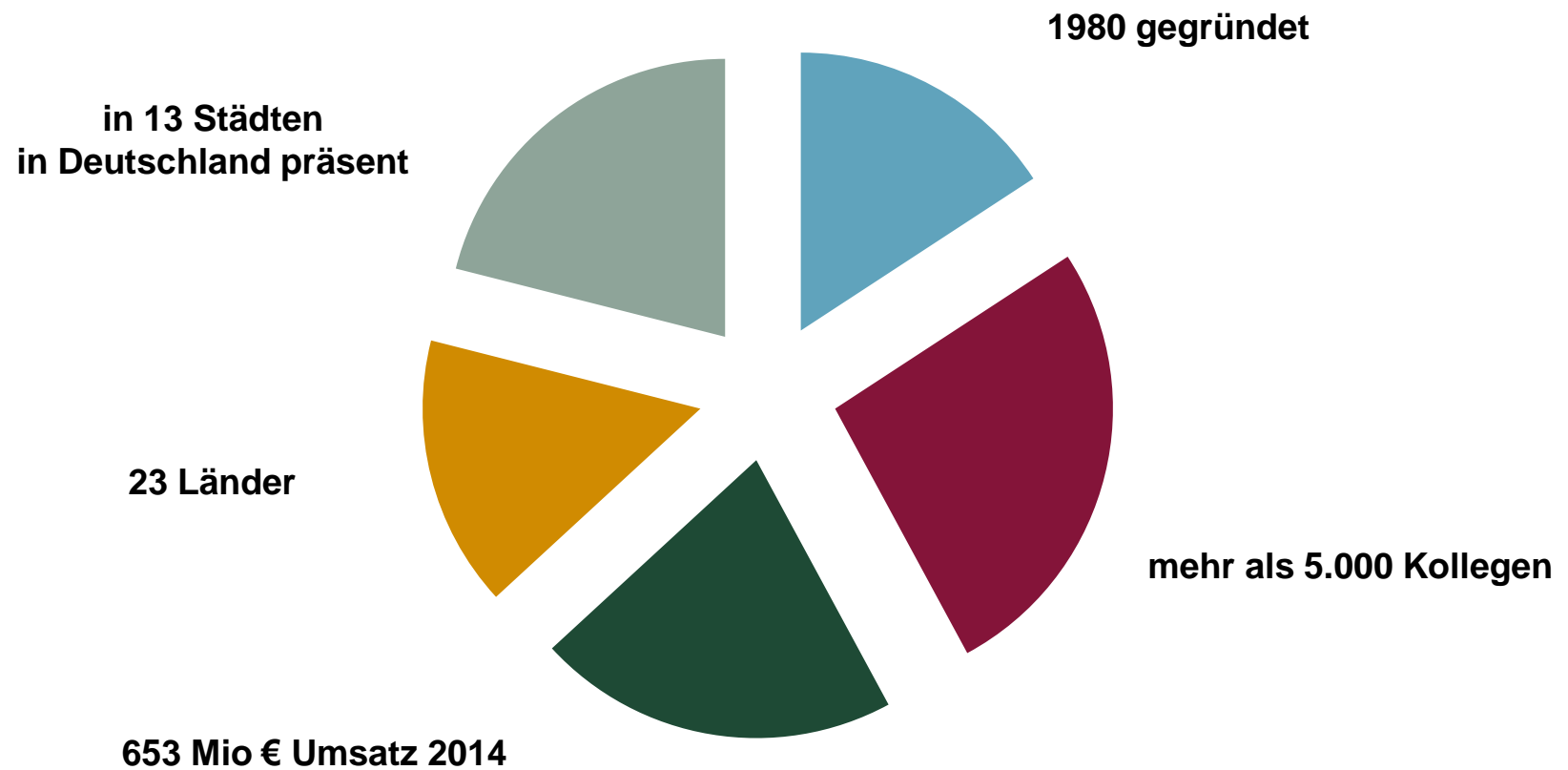
Alexander Schwartz

msg systems ag

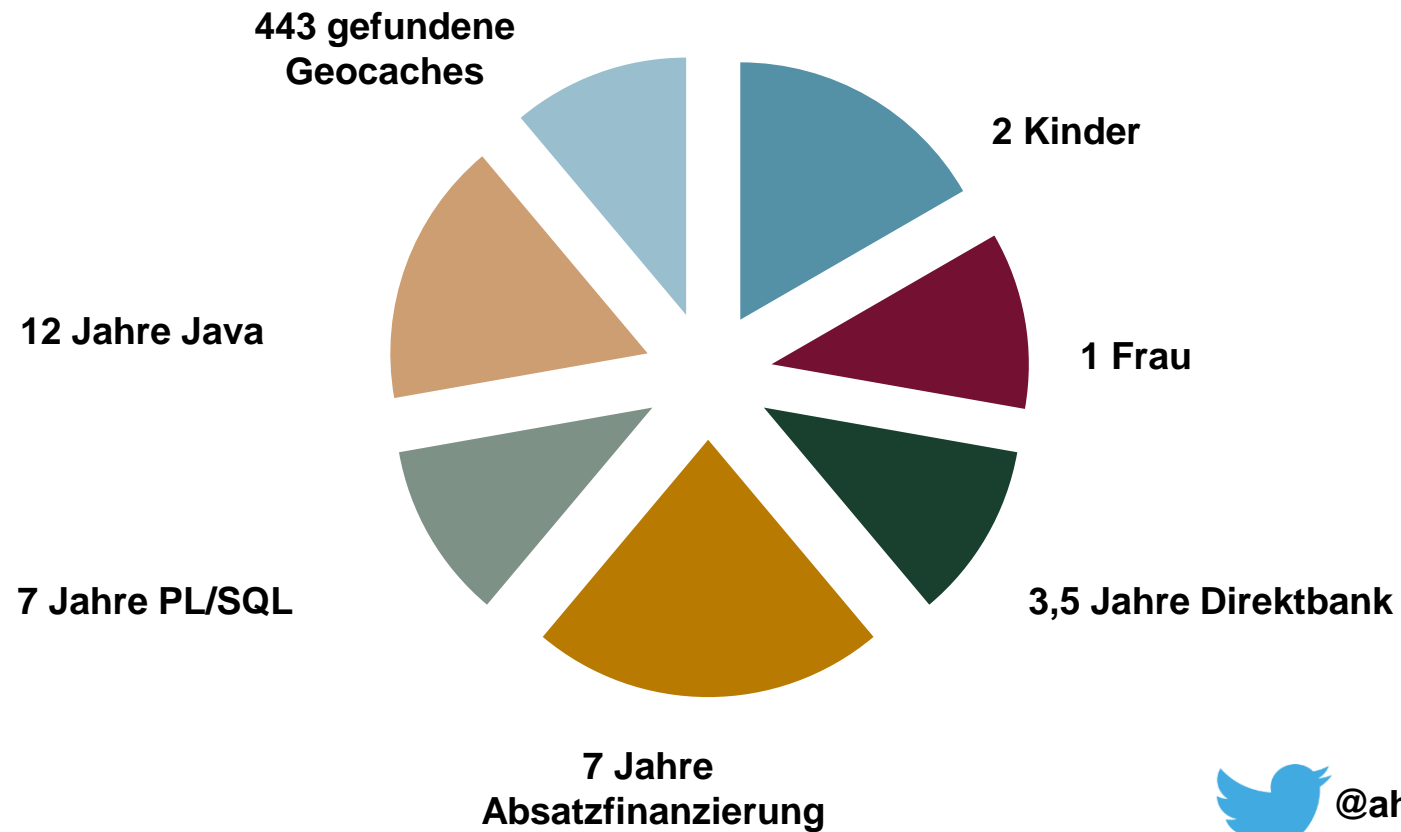
REST-Services ruck-zuck fertig mit Dropwizard

1	Application Server Evolution	5
2	Dropwizard	10
3	Beispielanwendung	14
4	Vorteile Drowizard	19

Mein Sponsor und Arbeitgeber – msg systems ag



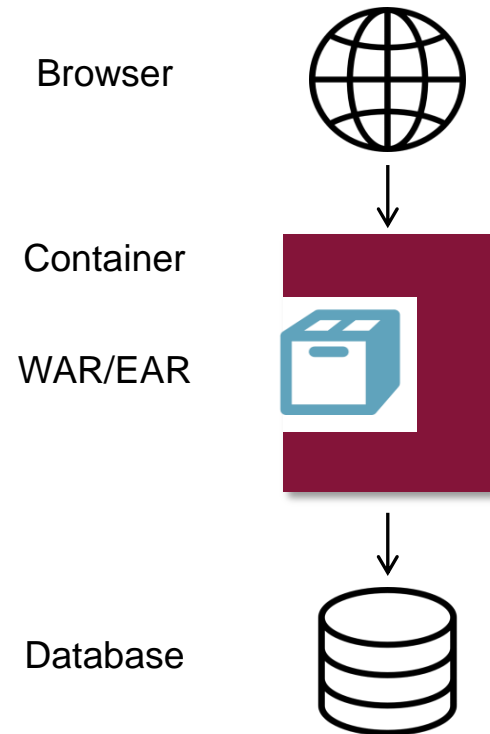
Wer ich bin – Alexander Schwartz



REST-Services ruck-zuck fertig mit Dropwizard

1	Application Server Evolution	5
2	Dropwizard	10
3	Beispielanwendung	14
4	Vorteile Drowizard	19

Traditionelle JEE Anwendungen brauchen einen Container



Best Practices für Application Server

#1: Eine Anwendung pro Container

Vorteile:

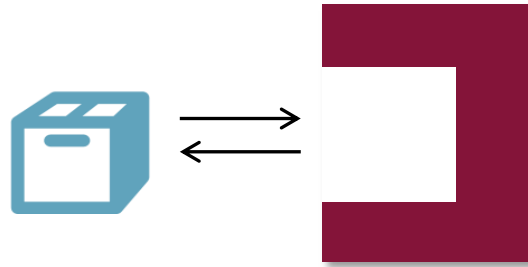
- Anwendungsspezifische Konfiguration für Datenbank, Treiber, SSL, Thread Pools, ...
- Neustart des Containers betrifft nur eine Anwendung
- Redeployment der Anwendung mit Neustart möglich

#2: Ein Container pro Anwendung

Vorteile:

- Konfiguration kann beim Deployment der Anwendung angepasst werden
- Anwendung läuft genau auf der getesteten Container-Version
- Anwendung bekommt genau die gewünschte Container-Version und JEE-Version

Wechselseitige Abhängigkeit zwischen Anwendung und Container



Folgen:

- Nicht genutzt Features in traditionellen Containern
- Unnötig komplizierte Prozesse für Continuous Integration / Continuous Delivery

„Is JEE Dead?“ (Brian O’Neill 03/2008)

„JEE is dead“ (Brian O’Neill 10/2012)

„Applikationsserver sind tod“ (Eberhardt Wolff 05/2014)

Application Server

Lösung: Kombiniere Anwendung mit Container



Dropwizard



Spring Boot



Vert.x

Effekt:

- Alles in einem JAR: einfache Installation und Staging
- Schneller Start: angenehme Entwicklung

```
# migrate database
java -jar application.jar db migrate config.yml
# start application
java -jar application.jar server config.yml
```

REST-Services ruck-zuck fertig mit Dropwizard

1	Application Server Evolution	5
2	Dropwizard	10
3	Beispielanwendung	14
4	Vorteile Drowizard	19

Entstehung von Dropwizard

Yammer migrierte von einer monolithischen Rails-Anwendung zu modularen Dropwizard-Microservices.

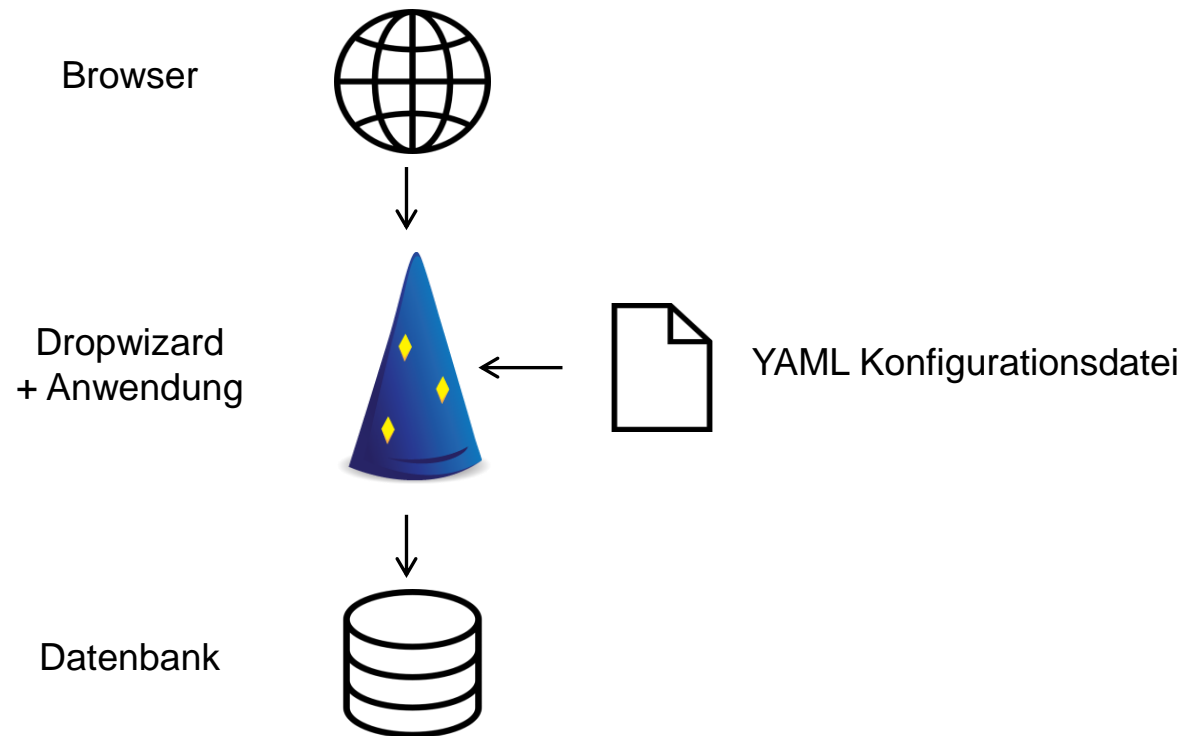
Version	Datum	Ereignis
0.1	12/2011	Initiales Release
	06/2012	Microsoft kauft Yammer
0.7	04/2014	Forms, Java 7, Jetty 9
0.8	03/2015	JAX RS 2.0, Jersey 2.x

Dropwizard Komponenten

- Jetty für HTTP
- Jersey für REST
- Jackson für JSON
- Metrics für Monitoring

„Freunde“: Hibernate Validator, JDBI, Hibernate, Liquibase, Freemarker, ...

Dropwizard Komponenten



REST-Services ruck-zuck fertig mit Dropwizard

1	Application Server Evolution	5
2	Dropwizard	10
3	Beispielanwendung	14
4	Vorteile Drowizard	19

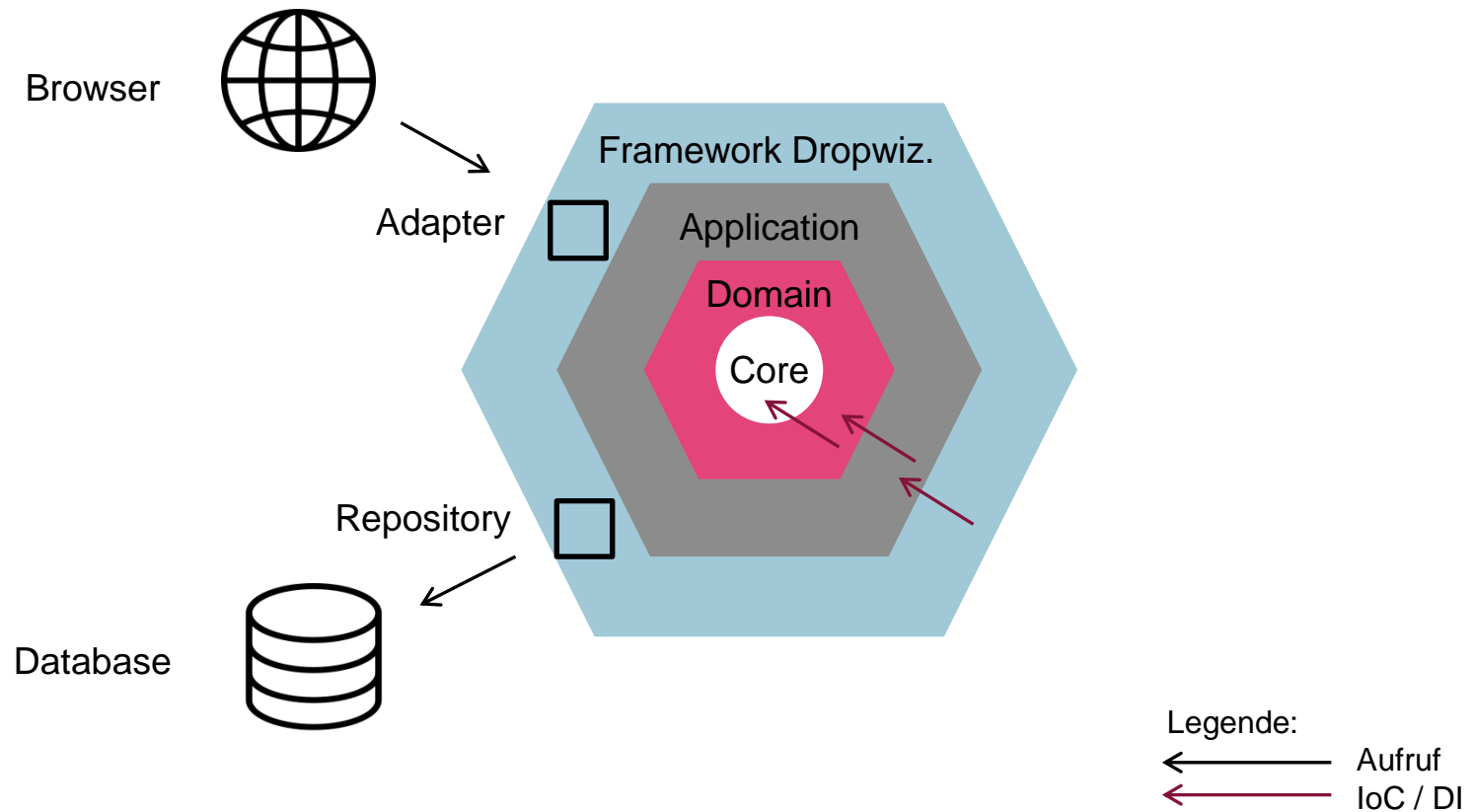
(Ausgewählte) Ideen des Domain Driven Designs

- Fachliche Domäne (Daten und Funktionalität) wird als Aggregates, Entities und Value Objects implementiert
- Funktionalität wird ggf. zusätzlich Services implementiert
- Fachlicher Code ist nicht von technischem Framework-Code abhängig
- Daten werden in Repositories abgelegt
- Interfaces des Repositories ist Teil der fachlichen Domäne, Implementierung des Repositories ist Teil der außenliegenden Schichten
- Umliegende Systeme werden über Adapter angebunden und können einen Anticorruption Layer beinhalten

Literatur:

Eric Evans: "Domain-Driven Design: Tackling Complexity in Software," 2004
Vaughn Vernon: "Implementing Domain-Driven Design", 2013

Dropwizard Komponenten



<http://fideloper.com/hexagonal-architecture>

Zeit für Programmcode!

Dropwizard Komponenten

	Bereich	Core	Add-On	Extra
Jersey / JAX 2.0	REST	X		
JDBI, Hibernate, Liquibase	Datenbank	X		
Freemarker, Mustache	View	X		
Metrics	Monitoring	X		
AssertJ, Basisklassen Test	Test	X		
Google Guice	Dependency Injection		X	
Swagger	Dokumentation		X	
Lombok	Entwicklung			X
JBoss Keycloak	Sicherheit			X
Arquillian Graphene	GUI Test			X
Shazam Gwen	BDD Test			X

REST-Services ruck-zuck fertig mit Dropwizard

1	Application Server Evolution	5
2	Dropwizard	10
3	Beispielanwendung	14
4	Vorteile Drowizard	19

Vorteile Dropwizard

- Dropwizard bringt viele Standardkomponenten mit
- Chance auf gleichartige, kleine Anwendungen
- Viele Wahlmöglichkeiten – junges, aktives Ökosystem
- Gute Dokumentation für die ersten Schritte
- Erprobte Komponenten
- Metriken integriert
- Wiederverwendung von JEE Knowhow
- Einfaches Deployment als JAR Datei
- Gute Vorbereitung für Tests



@ahus1de

Links

Dropwizard Homepage

<http://dropwizard.io/>

Eric Evans: Domain Driven Design Quickly

<http://www.infoq.com/minibooks/domain-driven-design-quickly>

Vaughn Vernon: Implementing Domain-Driven Design

https://vaughnvernon.co/?page_id=168

Guice für Dropwizard

<https://github.com/HubSpot/dropwizard-guice>

Swagger für Dropwizard

<https://github.com/federecio/dropwizard-swagger>

Lombok

<http://projectlombok.org/>

JBoss Keycloak

<http://keycloak.jboss.org/>

Keycloak Dropwizard Integration

<https://github.com/ahus1/keycloak-dropwizard-integration>

Arquillian Graphene

<http://arquillian.org/modules/graphene-extension/>

Shazam Gwen

<https://github.com/shazam/gwen>

Gwen: BDD-Framework für lesbare und refaktorisierbare Tests in Java

<http://heise.de/-2520872>

<https://github.com/ahus1/bdd-examples/>

JGiven: Behavior-Driven Development in Plain Java

<http://jgiven.org/>



@ahus1de



Alexander Schwartz
Principal IT Consultant

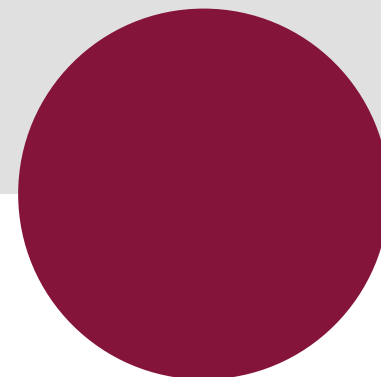
+49 171 5625767
alexander.schwartz@msg-systems.com



@ahus1de

msg systems ag (Headquarters)
Robert-Buerkle-Str. 1, 85737 Ismaning
Germany

www.msg-systems.com



.consulting .solutions .partnership