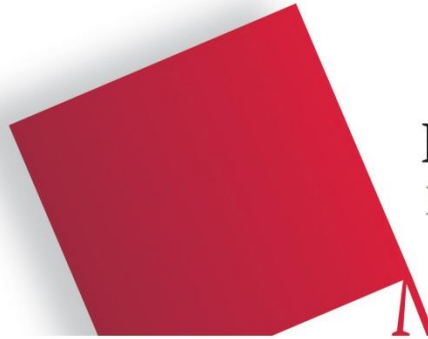


HOCHSCHULE
FÜR ANGEWANDTE
WISSENSCHAFTEN
MÜNCHEN

Ablauf, Inhalt und Themen 2019

FWP Aktuelle Technologien zur Entwicklung
verteilter Java-Anwendungen

Kontakt

Michael Theis
Lehrbeauftragter Hochschule München

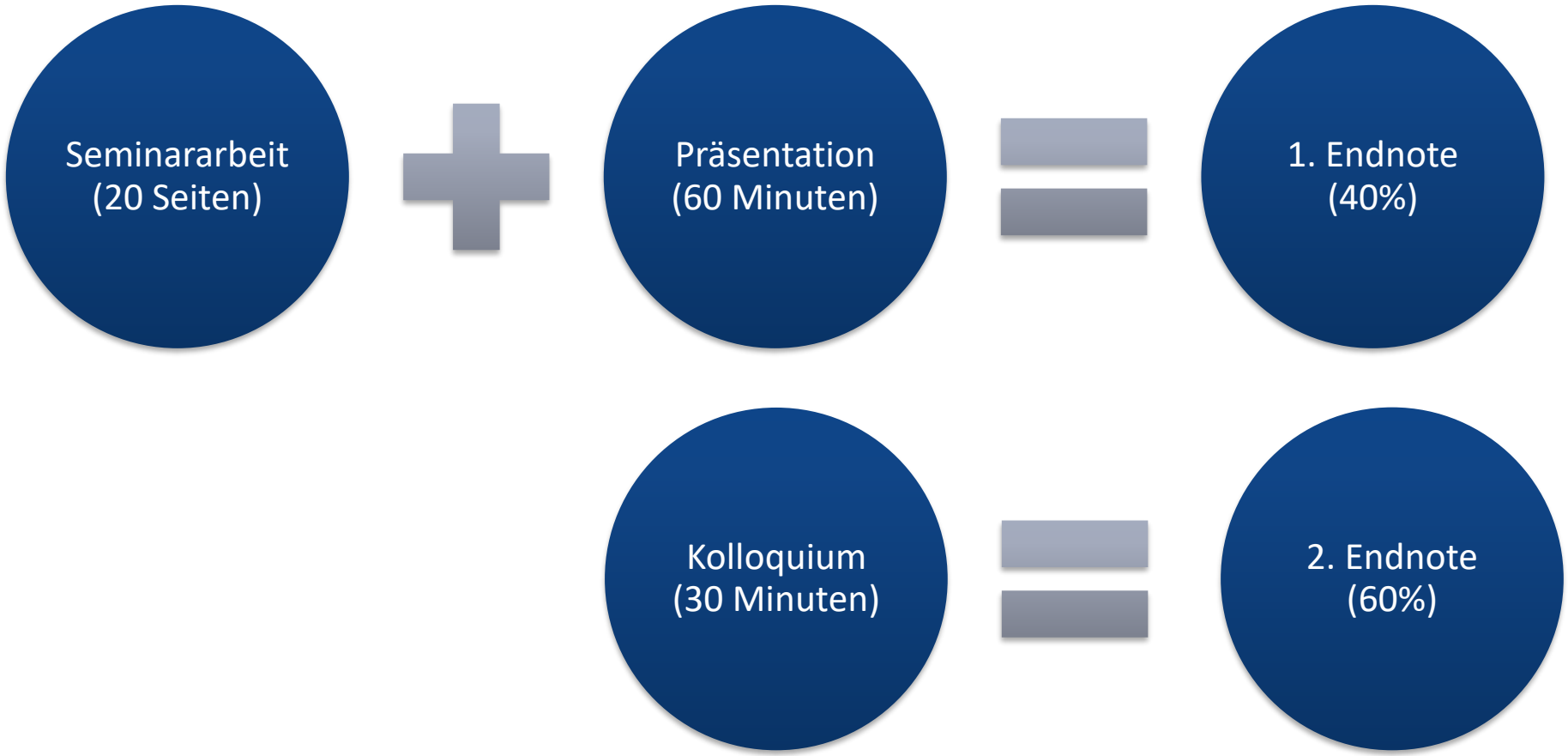
email michael.theis@hm.edu
mobile + 49 170 5403805
web https://tschutschu.de/Lehrauftrag.html

ABLAUF UND TERMINE

Ablauf und Termine

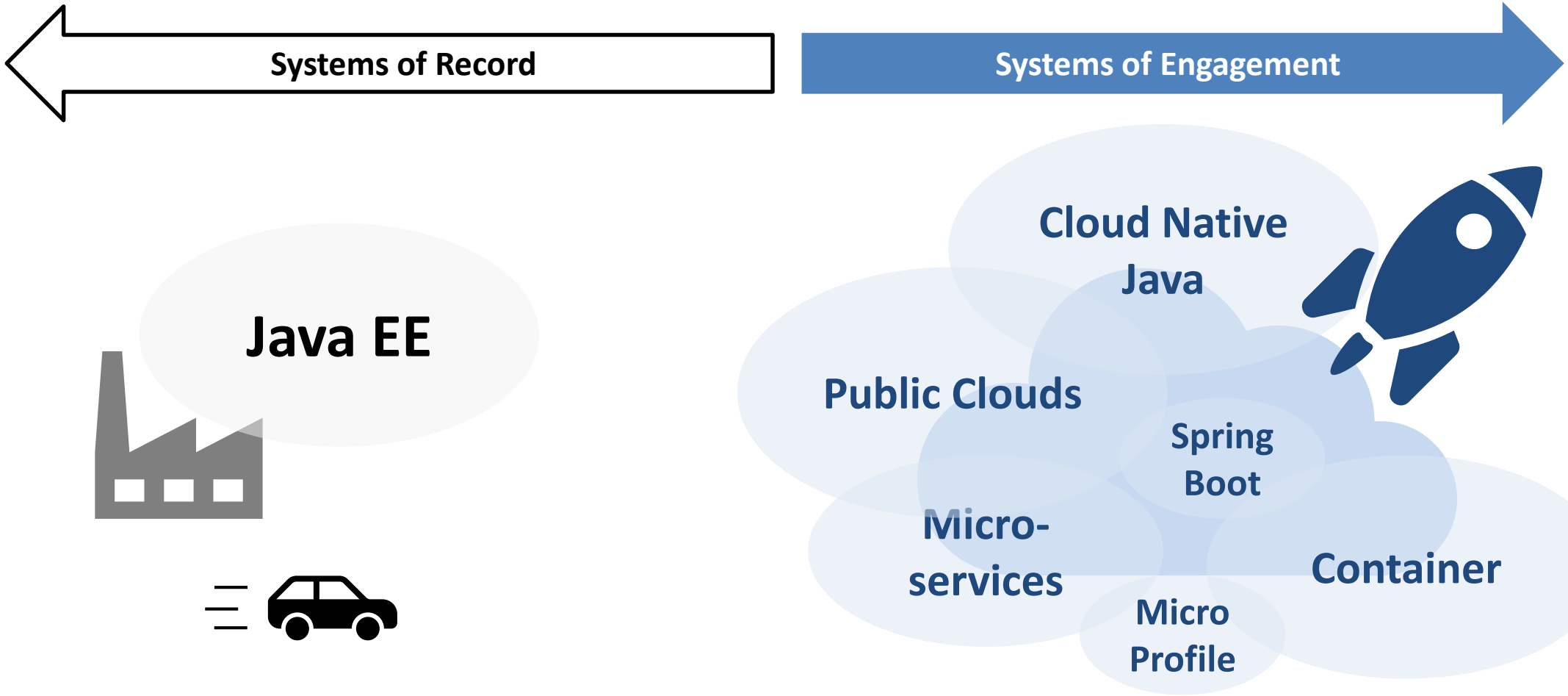
- Wöchentliche Vorlesung (4 SWS)
  - Start am 22.03.2019
  - Freitag von 15.15 – 18.30 Uhr Raum 1.008
- Themen werden zu Semesterbeginn vergeben und einzeln bearbeitet
- Pro Thema eine Seminararbeit (20 Seiten) und eine Präsentation (60 Minuten)
- Abschließend Kolloquium (30 Minuten)

Meilensteine und ihre Bewertungen



THEMEN 2019

Themenschwerpunkte



Jakarta EE löst Java EE ab: Was können wir erwarten?

Die Java-Plattform Enterprise Edition (Java EE) ist im September 2017 von Oracle an die Eclipse Foundation übergeben worden. Eine der ersten Maßnahmen der Eclipse Foundation war die Umbennenung von Java EE zu Jakarta EE. Zeigen Sie mit dieser Studienarbeit, dass hinter Jakarta EE mehr als nur ein neuen Name für Java EE steckt:

- Erläutern Sie die wesentlichen Ziele, die die Eclipse Foundation mit zukünftigen Versionen des Standards für Enterprise Java verfolgt.
- Schildern Sie, welche Features die erste Version von Jakarta EE enthalten soll.
- Zeigen Sie auf, wann neue Versionen von Jakarta EE zu erwarten sind.
- Bewerten Sie Jakarta EE kurz auf Tauglichkeit hinsichtlich der Entwicklung und dem Betrieb von cloud-fähigen Java-Applikationen

Einstiegsliteratur und Internetquellen:

Jakarta EE Homepage der Eclipse Foundation
https://jakarta.ee/

SpringBoot vs Java EE Application Server

SpringBoot hat sich in den letzten Jahren einen Namen als leichtgewichtige Alternative zu Java EE Application Servern gemacht. Die Java EE-Community hat versucht, mit dem Java EE Web Profile die Java EE Plattform auf eine vergleichbare schlanke Laufzeitumgebung für server-seitige Java-Applikationen zu reduzieren.

- Vergleichen Sie die Konzepte von SpringBoot mit denen eines Java EE Web Profile Application Servers und stellen Sie die Stärken und Schwächen der beiden Plattformen gegenüber.
- Gehen Sie dabei insbesondere auf die unterschiedlichen Deployment-Konzepte ein (WAR + Application Server vs. Über-JAR mit eingebautem Servet Container)

Einstiegsliteratur und Internetquellen:

SpringBoot Homepage
https://projects.spring.io/spring-boot/

JSR 342: Java™ Platform, Enterprise Edition 7 (Java EE 7) Web Profile Specification
https://www.jcp.org/en/jsr/detail?id=342
(Runterladen über Final Release > Download page > unterer Button Download > WebProfile.pdf

Charakteristiken einer Microservice-Architektur

Kaum ein anderes Buzzword wird in der IT-Branche derzeit so inflationär gebraucht wie das Wort Microservice: Selbst bisher eher konservativ agierende Unternehmen wollen ihre konventionellen Applikations-Monolithen in schlanke voneinander unabhängige Microservices zerlegen. Was dabei gerne aus den Augen verloren wird, ist die Tatsache, dass Microservices mit einem nicht immer offensichtlichen Preis bezahlt werden müssen.

- Versuchen Sie eine vernünftige Definition des Begriffes Microservice zu finden.
- Zeigen Sie die wesentlichen Charakteristiken einer Microservice-Architektur auf.
- Arbeiten Sie die wesentlichen Herausforderungen beim Übergang zu einer Microservice-Welt heraus.

Einstiegsliteratur und Internetquellen:

Martin Fowler, James Lewis: *Microservices*
https://martinfowler.com/articles/microservices.html

Sam Newman: *Building Microservices*
O´Reilly, 1. Auflage, 17. Februar 2015
ISBN: 978-1491950357

Richard Rodger: *The Tao of Microservices*
Manning, 28. September 2016
ISBN: 978-1617293146

Eberhard Wolff: Das Microservice-Praxisbuch: Grundlagen, Konzepte und Rezepte
dpunkt.verlag GmbH, 1. Auflage, 29. Januar 2018
ISBN: 978-3864905261

Sam Newman: *DevoXX 2015 • Principles of Microservices*
https://youtu.be/PFQnNFe27kU

R. Meshenberg: *GOTO 2016 • Microservices at Netflix Scale: Principles, Tradeoffs & Lessons Learned*
https://youtu.be/57UK46qfBLY

Cloud Native Java Apps mit Spring Boot/Spring Cloud

Bei der Umsetzung von cloud-nativen Java-Applikationen hat zur Zeit Spring Boot im Vergleich zu Java EE deutlich die Nase vorn. Für die Integration der Spring Boot in Cloud-Umgebungen sowie die Umsetzung wichtiger Features wie Configuration Management, Service Discovery und Resilienz sorgt dabei Spring Cloud.

- Geben Sie zunächst einen Überblick über die verschiedenen Projekte unter dem Schirm von Spring Cloud.
- Wählen Sie ein bestimmtes Spring Cloud Projekt aus und zeigen Sie, welches Feature sich damit wie in einer Spring Boot-Applikation umsetzen lässt.
- Vergessen Sie dabei nicht, die wesentlichen Stärken und Schwächen des von Ihnen ausgewählten Spring Cloud Projektes aufzuzeigen

Einstiegsliteratur und Internetquellen:

Spring Cloud Project Homepage
http://projects.spring.io/spring-cloud/

Josh Long, Kenny Bastani
Cloud Native Java: Designing Resilient Systems with Spring Boot, Spring Cloud, and Cloud Foundry
O´Reilly, 1. Auflage, 22. August 2017
ISBN: 978-1449374648

Progressive Web-Frontends für Cloud Applikationen

Web-Frontends auf Basis von JavaScript oder TypeScript sind inzwischen für der Standard für Benutzer-schnittstellen von Cloud Applikationen oder Microservices. Die Ansätze reichen hierbei von einer Web-App für mehrere Microservices bis zu einer Kombination von mehreren Web-Apps zu einer einheitlichen Benutzerschnittstelle (wie es zum Beispiel Amazon macht).

- Beschreiben Sie zunächst, welche verschiedenen Ansätze bei der Umsetzung von Benutzerschnittstellen für Cloud Native Applikationen existieren.
- Schildern Sie anschließend, welchen Herausforderungen sich diese Web-Frontends in einem Cloud-Umfeld mit beliebigen mobilen Endgeräten stellen müssen.
- Zeigen Sie dann, wie diese Web-Frontends in einem Cloud-Umfeld normalerweise bereitgestellt und betrieben werden.
- Belegen Sie Ihre Ausführungen mit einem kleinen Frontend basierend auf der Technologie Ihrer Wahl (Angular, React, Vue).

Einstiegsliteratur und Internetquellen:

Google: Progressive Web Apps Homepage
https://developers.google.com/web/progressive-web-apps/

Sam Newman
Pattern: Backends for Frontends
https://samnewman.io/patterns/architectural/bff/

Bereitstellen und Konsumieren von REST APIs mit Java EE

Leichtgewichtige Webservices auf Basis von REST haben inzwischen herkömmliche SOAP-Webservices fast völlig verdrängt. Im Rahmen dieser Studienarbeit soll nun nicht auf die Einzelheiten von REST eingegangen, sondern konkret beschrieben werden, wie sich REST-Endpunkte in einer modernen Java EE Applikation mit JAX-RS und JSON-B umsetzen lassen.

- Zeigen Sie, wie sich mit dem Standard JAX-RS über Java-Klassen mit ein paar Annotationen REST-Endpunkte bereitstellen lassen.
- Erläutern Sie, wie sich dabei mit dem neuen Standard JSON-B Java-Objekte an JSON-Nachrichten binden lassen. Dabei soll explizit auf den Einsatz der proprietären Lösung Jackson verzichtet werden.
- Demonstrieren Sie dann, wie sich REST-Endpunkte mit REST-Clients aufrufen lassen.

Einstiegsliteratur und Internetquellen:

Java EE 8 Tutorial Kapitel 32:
Building RESTful Webservices with JAX-RS
https://javaee.github.io/tutorial/jaxrs.html#GIEPU

Java EE 8 Tutorial Kapitel 33:
Accessing REST Resources with the JAX-RS Client API
https://javaee.github.io/tutorial/jaxrs-client.html#BABEIGIH

Java EE 8 Tutorial Kapitel 21: JSON Binding
https://javaee.github.io/tutorial/jsonb.html

Bereitstellen und Konsumieren von REST APIs mit Spring

Leichtgewichtige Webservices auf Basis von REST haben inzwischen herkömmliche SOAP-Webservices fast völlig verdrängt. Im Rahmen dieser Studienarbeit soll nun nicht auf die Einzelheiten von REST eingegangen, sondern konkret beschrieben werden, wie sich REST-Endpunkte in einer modernen Spring Boot Applikation umsetzen lassen.

- Zeigen Sie, wie sich mit ein paar Annotationen Java-Klassen als REST-Endpunkte bereitstellen lassen.
- Erläutern Sie, wie dabei im Spring Boot das Binding von Java-Objekte an JSON-Dokumente realisieren lässt.
- Demonstrieren Sie dann, wie sich REST-Endpunkte mit RestTemplates aufrufen lassen.

Einstiegsliteratur und Internetquellen:

SpringBoot Homepage
https://projects.spring.io/spring-boot/

Selbstdokumentierende REST-APIs mit Open API

In einer service-orientierten Welt mit einer Vielzahl von REST APIs ist eine konsistente Spezifikation und Dokumentation einer REST API ein wichtiger Erfolgsfaktor für die Nutzung der Schnittstelle. Dafür hat sich inzwischen die Open API Spezifikation von Swagger als Standard etabliert.

- Zeigen Sie, wie sich mit OpenAPI REST APIs in einem einheitlichen Format spezifizieren und anschließend sich aus der Spezifikation Java-Code generieren lässt (Contract First).
- Demonstrieren Sie, wie sich bereits in Java implementierte REST API durch den Einsatz von Annotationen in selbstdokumentierende REST APIs verwandeln lassen (Contract Last).

Einstiegsliteratur und Internetquellen:

Open API Homepage auf Swagger.io
https://swagger.io/docs/specification/about/

Eclipse MicroProfile Open API Homepage
https://github.com/eclipse/microprofile-open-api

Reactive Web Backends mit Spring WebFlux

Moderne Web-Backends müssen eine große Flut von Requests von unter Umständen Millionen von Benutzern bewältigen können. Will man diese Last mit einem vertretbaren Aufwand an Ressourcen stemmen, so ist ein Wechsel von den herkömmlichen blockierenden Web-Stacks (wie die Java EE Servlet API) zu reaktiven nicht-blockierenden Web-Stacks empfehlenswert. Mit Spring Version 5 ist nun ein neuer Vertreter dieser reaktiven Web-Stacks auf der Bildfläche erschienen: **Spring WebFlux**.

- Erklären Sie zunächst die Konzepte des reaktiven Programmiermodells.
- Erläutern Sie dann, wie Spring WebFlux diese Konzepte für die Verarbeitung von HTTP-Requests umsetzt.
- Vergleichen Sie Spring WebFlux mit der herkömmlichen Servlet API aus dem Java EE-Standard.
- Bewerten abschließend Spring WebFlux nach seinen Stärken und Schwächen.

Einstiegsliteratur und Internetquellen:

Spring WebFlux Referenz
https://docs.spring.io/spring/docs/current/spring-framework-reference/web-reactive.html#webflux

Integration von relationalen Datenbanken mit Spring Data

Bei Spring Applikationen hat sich inzwischen Spring Data als Standard für Persistenztechnologie durchgesetzt. Spring Data ist dabei im Vergleich zur Java Persistence API (JPA) deutlich leichter einzusetzen und unterstützt neben relationalen Datenbanken auch NoSQL Datenspeicher.

- Beschreiben Sie zunächst die Konzepte von Spring Data.
- Erläutern Sie, wie sich mit Spring Data relationale Daten in eine Spring Boot-Applikation integrieren lassen.

Einstiegsliteratur und Internetquellen:

Spring Data Project Homepage
http://projects.spring.io/spring-data/

Mark Pollack, Oliver Gierke, u.a.
Spring Data
O´Reilly and Associates, 1. Auflage, 30. Nov. 2012
ISBN: 978-1449323950

NoSQL und Java EE – Ein perfektes Paar?

Seit Jahrzenten sind relationale Daten-banken die erste Wahl, wenn es um ernsthafte Datenspeicherung geht. Unser Informationszeitalter mit immer rasanter wachsenden Datenmengen bringt die relationalen Datenbanken jedoch an ihre Grenzen. Abhilfe schaffen sollen nicht-relationale „NoSQL“ Datenbanken, die der Datenflut mit neuen Konzepten entgegentreten.

- Erläutern Sie zunächst die grundliegenden Konzepte von NoSQL-Datenbanken und vergleichen Sie diese mit der relationalen Technik.
- Wählen Sie aus dem Angebot eine bekannte NoSQL Datenbank aus.
- Beschreiben Sie deren besonderen Eigenschaften.
- Zeigen Sie, wie sich diese NoSQL-Datenbank in eine Java EE Anwendung integrieren lässt.

Einstiegsliteratur und Internetquellen:

Prasmod J. Sadalage, Martin Fowler
NoSQL Distilled
Addison Wesley Pearson Education 2013
ISBN 978-0-321-82662-6

Java Applikationen mit token-basierter Sicherheit schützen

Token-basierte Sicherheitsmechanismen wie OAuth2 und OpenID Connect verdrängen zunehmend herkömmliche Sicherheitsmechanismen wie Basic Authentication, Form Based Authentication. Ziel dieser Studienarbeit ist es, konkret die Integration von token-basierter Sicherheit entweder auf der Basis von Java EE, Eclipse MicroProfile JWT Auth oder Spring Boot/Spring Security zu demonstrieren.

- Vergleichen Sie einleitend die herkömmlichen Sicherheitsmechanismen mit den modernen token-basierten Alternativen.
- Zeigen Sie anschließend, wie sich tokenbasierte Sicherheit auf der von Ihnen gewählten technologischen Plattform in eine Applikation einbauen lassen.

Einstiegsliteratur und Internetquellen:

OpenID Connect – Core 1.0 Spezifikation
http://openid.net/specs/openid-connect-core-1_0.html

RFC 7519 JSON Web Token (JWT)
https://tools.ietf.org/html/rfc7519

Java EE Security API Specification JSR 375
https://javaee.github.io/security-spec/

MicroProfile JWT Auth API
https://github.com/eclipse/microprofile-jwt-auth

Spring Security Homepage
https://spring.io/projects/spring-security

KeyCloak Homepage
https://www.keycloak.org/

Lang laufende Prozesse mit Java EE

In fast jeder nicht-trivialen Applikation muss man früher oder später Aktivitäten ausführen, die bezüglich ihrer Laufzeit den gewöhnlichen Rahmen sprengen (> 120 s).

Hier können die mit Java EE 7 neu hinzu gekommenen Concurrency Utilities (JSR-236) Abhilfe schaffen: ManagedExecutor-Services und ManagedScheduled-ExecutorServices lassen sich nun als Ressourcen im Application Server konfigurieren und aus Applikationen heraus nutzen.

- Beschreiben Sie die wesentlichen Features der Concurrency Utilities API.
- Erläutern Sie, wie sich damit lang-laufende Prozesse standard-konform umsetzen lassen.
- Vergleichen Sie Lösungen auf Basis der Concurrency Utilities mit @Schedule EJBs und @Asynchronous EJB-Aufrufen.

Einstiegsliteratur und Internetquellen:

The Java EE Tutorial Kapitel 60
https://javaee.github.io/tutorial/concurrency-utilities.html#GKJIQ8
Oracle September 2017

JSR 236: Concurrency Utilities for JavaTM EE
Homepage zu JSR 236 auf JCP.org
https://jcp.org/en/jsr/detail?id=236

Batchverarbeitung mit Java

Die Verarbeitung von großen Datenmengen ist nicht mehr ausschließlich mainframe-basierten Systemen vorbehalten, sondern wird aus Kostengründen zunehmend auch auf java-basierten Plattformen durchgeführt. Mit Java EE 7 ist nun erstmals eine Standard-API für Batchverarbeitung veröffentlicht worden.

Beschreiben Sie zunächst die Grundlagen der Batchverarbeitung:

- Massendatenverarbeitung
- Jobs / Steps
- Checkpoint / Restart

Erläutern Sie, wie sich mit Java Batch Processing Massendatenverarbeitung umsetzen lässt.

Einstiegsliteratur und Internetquellen:

The Java EE Tutorial Kapitel 59
https://javaee.github.io/tutorial/batch-processing.html
Oracle September 2017

Consumer Contract Driven Testing mit Pact

Der zunehmende Einsatz von Microservices zwingt uns auch zu einem Umdenken beim Testen unserer Java-Applikationen:

- Wie können wir sicherstellen, dass wir die API unserer Services so implementiert haben wie mit dem Kunden vereinbart?
- Wie kann ein Kunde sicherstellen, dass er problemlos eine neue Version der von uns bereitgestellten API einsetzen kann?

Hier verspricht der Testansatz des Consumer Contract Driven Testings Abhilfe. Ein bekanntes Tools für diese Art des Testens ist Pact.

⊙ Erläutern Sie zunächst die Methodik hinter dem Consumer Contract Driven Test.

⊙ Zeigen Sie, wie sich mit Pact Consumer Contract Driven Testing im Java-Umfeld umsetzen lässt.

⊙ Bewerten Sie die Stärken und Schwächen des Consumer Driven Contract Testens.

Einstiegsliteratur und Internetquellen:

Pact HomePage
https://swagger.io/docs/specification/about/

Ian Robinson, Thoughtworks
Consumer-Driven Contracts: A Service Evolution Pattern
https://martinfowler.com/articles/consumerDrivenContracts.html

Testen von JEE Applikationen mit Arquillian

Arquillian ist eine innovative und erweiterbare Test-Plattform auf Java-Basis, die es Entwicklern ermöglicht, einfach automatisierte Integrationstests, funktionale Tests und Akzeptanz-Tests für server-seitige Java-Komponenten zu erstellen.

- Beschreiben Sie zunächst, wo bei der Entwicklung server-seitiger Java-Anwendungen die Herausforderungen bezüglich automatisiertem Testen liegen.
- Schildern Sie anschließend die wesentlichen Konzepte und Eigenschaften von Arquillian und wie sich diese von denen klassischer Test-methoden unterscheiden.
- Demonstrieren Sie den konkreten Einsatz von Arquillian mit einem Beispiel.

Einstiegsliteratur und Internetquellen:

Arquillian Homepage
http://arquillian.org/

Verteilte Domänenereignisse mit CDI und RabbitMQ

Die aus dem Domain Driven Design stammenden Domänenereignisse (Domain Events) stellen ein wirksames Mittel zu asynchronen Kommunikation zwischen lose gekoppelten Services dar. Innerhalb einer Java EE Applikationen lassen sich diese Domänen-ereignisse direkt mit CDI Events umsetzten. Zwischen Applikationen können diese Ereignisse dann über leicht-gewichtige AMQP-basierte Messagingsystem wie RabbitMQ ausgetauscht werden.

- Beschreiben Sie zunächst kurz, was die Motivation für den Einsatz von Domänenereignissen ist.
- Zeigen Sie dann, wie sich innerhalb einer Anwendung Domänenevents mit CDI umsetzten lassen.
- Schildern Sie anschließend, wie sich diese internen Ereignisse an andere Applikationen über RabbitMQ weiterleiten lassen. Gehen Sie dabei insbesondere auf die Integration von RabbitMQ Java EE Applikationen ein, die als Messagingtechnologie nur JMS kennen.

Einstiegsliteratur und Internetquellen:

Martin Fowler
Domain Event
https://martinfowler.com/eaaDev/DomainEvent.html

Java EE 8 Tutorial Kapitel 27: Contexts and Dependency Injection for Java EE: Advanced Topics Using Events in CDI Applications
https://javaee.github.io/tutorial/cdi-adv005.html#GKHIC

Java EE 8 Tutorial Part IX: Messaging
https://javaee.github.io/tutorial/partmessaging.html#GFIRP3

Advanced Message Queuing Protocol (AMQP) Homepage
https://www.amqp.org/

Rabbit MQ Homepage
https://www.rabbitmq.com

Monitoring von Java-Applikationen mit Prometheus

Ein wesentlicher Erfolgsfaktor für eine Applikation in der Cloud ist die permanente Sichtbarkeit des Zustandes der Applikation. Das Problem dabei ist nur, dass eine in der Cloud betriebene Applikation sich wie eine Sonde im Weltraum anfühlt: man kann nicht mehr auf die Applikation direkt zugreifen, sondern nur auf die Telemetriedaten hören, die die Applikation sendet. Daher ist sowohl der Einsatz von Monitoring-Tools als auch ein genauer Plan, was sinnvollerweise überwacht werden soll, essenziell für den erfolgreichen Betrieb der Applikation. Als relevantes Monitoring-Tool hat sich inzwischen Prometheus etabliert.

- Erstellen Sie zunächst einen Katalog von Metriken, nach denen eine Applikation sinnvollerweise überwacht werden sollte.
- Erläutern Sie dann, wie ein Monitoring-Tool am Beispiel von Prometheus funktioniert
- Beschreiben Sie dann, wie sich Prometheus prinzipiell in eine Java-Applikation integrieren lässt.
- Zeigen Sie auf, wie sich die gesammelten Daten in einem Dashboard anzeigen lassen.
- Bewerten Sie Prometheus nach seinen Stärken und Schwächen.

Einstiegsliteratur und Internetquellen:

Richard Rodger: *The Tao of Microservices*
Manning, 28. September 2016
ISBN: 978-1617293146

Prometheus Homepage
https://prometheus.io/

Einsatz von Docker bei der Entwicklung von Java-Applikationen

Docker als Container-Technologie hat die IT-Welt wie wir sie kennen grundliegend verändert. Cloud-Dienstmodelle wie Container as a Service (CaaS) sind auf dem Vormarsch und haben häufig herkömmliche Betriebsmodelle für Softwaresysteme komplett verdrängt. Doch wie sieht es mit Docker am eigenen Entwicklerarbeitsplatz aus?

- Zeigen Sie zunächst, wie sich Applikationsserver und Datenbanken am eigenen Rechner ohne aufwändige Installation über Docker einfach starten. Die Wahl eines Applikationsservers und einer Datenbank steht Ihnen frei.
- Erläutern Sie anschließend, wie sich dockerisierte Applikationsserver in die Entwicklungsumgebung Ihrer Wahl integrieren lassen. Zeigen Sie dabei insbesondere, wie Sie am geschicktesten die App, an der Sie gerade arbeiten, in den dockerisierten AppServer deployen können.
- Vergleichen Sie die lokale Entwicklung mit Docker mit der herkömmlichen Entwicklung mit installiertem AppServer und Datenbank mit ihren Vor- und Nachteilen.

Einstiegsliteratur und Internetquellen:

Fragen?

Kontakt

Michael Theis
Lehrbeauftragter Hochschule München

email michael.theis@hm.edu
mobile + 49 170 5403805
web https://tschutschu.de/Lehrauftrag.html