Spring Boot - und gut?

Ein kurzer Überblick über das Spring Framework und Spring Boot

Inhalt

- Motivation
- Was ist das Spring Framework
- Was kann das Spring Framework?
  - Module
- Spring Boot

Motivation

- Einen **Überblick** über Spring Framework liefern
- **Bewusstsein** für Anwendungsgebiete schaffen
- Grundlage zur Abgrenzung zu **CoreFrame** liefern


Was ist das Spring Framework?
- Und was ist es nicht?

Die Entwicklung von Spring

- Basiert auf **Rod Johnsons** Buch *Expert One-On-One J2EE Design and Development (2002)*

- Hauptentwickler **Pivotal** ist (inzwischen) ein Joint Venture von **VMWare** und **General Electric**

- Version 0.9: Juni 2003
- Version 5.0: September 2017
- Version 5.0.7: Juni 2018

Was ist das Spring Framework?

- *„Framework of Frameworks“*
- Unterstützt Entwicklung in **Java**, **Groovy**, **Kotlin**
- Ziele
  - Entwicklung von Enterprise-Anwendungen **vereinfachen**
  - Gute **Programmierpraktiken** fördern
  - **Entkopplung** der Applikationskomponenten
  - Reduzierung von **Gluecode** und **Redundanz**

Was ist das Spring Framework nicht?

- Keine Konkurrenz zu **Jakarta EE**

- **Jakarta EE** (aka **Java EE** oder **J2EE**)
  - Neuer Name seit Umzug zur **Eclipse Foundation**
  - Sammlung von **API-Spezifikationen**
  - **Referenzimplementierungen** zu APIs
  - Häufig mehrere **konkurrierende** Implementierungen

Was ist das Spring Framework nicht?

- Spring Framework
  - Enterprise-taugliche **Frameworks** statt API-Spezifikationen
  - **Basiert** in Teilen auf Jakarta EE-Spezifikationen
  - Konkurrenz zu Jakarta EE-**Implementierungen**


Was kann das Spring Framework?

Features

- Aspektorientierte Programmierung
  - Dependency Injection durch **IoC Container** (*Spring Context*)
    - Deckt nicht alle Anforderungen an AOP ab
    - „…addresses the 80% sweet spot of AOP requirements in Java enterprise programming.“
  - Wenn nötig umfangreiche **AspectJ-Integration**

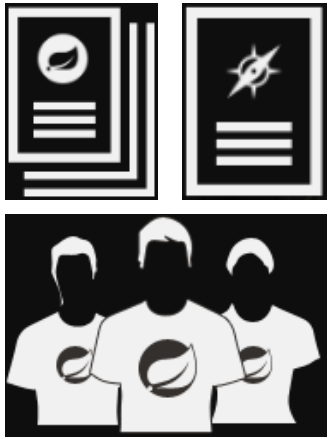
Features

- Integration von Frameworks
  - **Keine eigenen Lösungen, wenn es schon gute Frameworks gibt**
  - Hibernate, Flyway, JUnit, Mockito, Thymeleaf, Jackson, log4j, …
- Abstraktionsschichten durch Module
  - Persistierung, Datenzugriff, Sicherheit, MVC, Messaging

Allgemeines

- Entwicklungsumgebung: **Spring Tool Suite** (STS)
  - Angepasste **Eclipse**-Umgebung
  - Unterstützt bei der Entwicklung mit **Spring Framework** und **Spring Modulen**

- Dokumentation
  - Detaillierte **Dokumentation** zu allen Modulen
  - Viele **Guides** (in *STS* integriert)
  - Große Community
    - **Aber**: In **15 Jahren** kann man viel diskutieren und beschreiben…



Allgemeines

- Konfiguration
  - XML
  - **Annotationen**
  - Java-Code

- Module
  - Essenziell für Spring
  - 204 Repositories auf GitHub
  - Ca. 50 aktiv
  - Weitere Community Projekte


Module

Spring Data

- Zugriff auf **relationale** und **NoSQL**-Datenbanken
- Basis: **JPA** (Jakarta EEs ***J**ava **P**ersistence **A**PI*)
- Features und Integrationen
  - Zugriff auf Relationale Datenbanken via teilweise **selbstimplementierender** *Repositories* (DAOs) => *declarative queries*
  - **Reaktiver** Zugriff auf **NoSQL**-Datenbanken
  - JPA-Provider: **Hibernate**, OpenJPA, EclipseLink, …
  - DBMS: MySQL, MSSQL, **H2**, MongoDB, …
  - Pooling: **HikariCP**, Tomcat pooling, Commons DBCP2

Spring Security

- Authentifizierung
  - *AuthenticationManager*
    - Erhält Benutzerdaten (*Authentication*) und **entscheidet** über Gültigkeit
    - Implementierungen
      - *ProviderManager*
      - AD / LDAP
      - JAAS (**J**ava **A**uthentication and **A**uthorization **S**ervice)
      - OpenID
      - …

Spring Security

- *UserDetailsService*
  - Liefert **Benutzerdaten** und wird (u. a.) vom *AuthenticationManager* verwendet
  - Implementierungen
    - *InMemoryUserDetailsManager*
    - JDBC
    - LDAP
    - …

Spring Security

- Autorisierung
  - *AccessDecisionVoter*
    - Stimmt **für** oder **gegen** Zugriff auf Ressource
    - Implementierungen
      - Auf ACL-Basis
      - Ausdrücke
      - Rollenbasiert
      - …
  - *AccessDecisionManager*
    - Verknüpft eine Reihe von *AccessDecisionVotern* mit bestimmter Logik

Spring Security

- Schutz vor Angriffen
  - *Session fixation*
  - *Clickjacking*
  - *Cross site request forgery*
  - …

Web UI Module

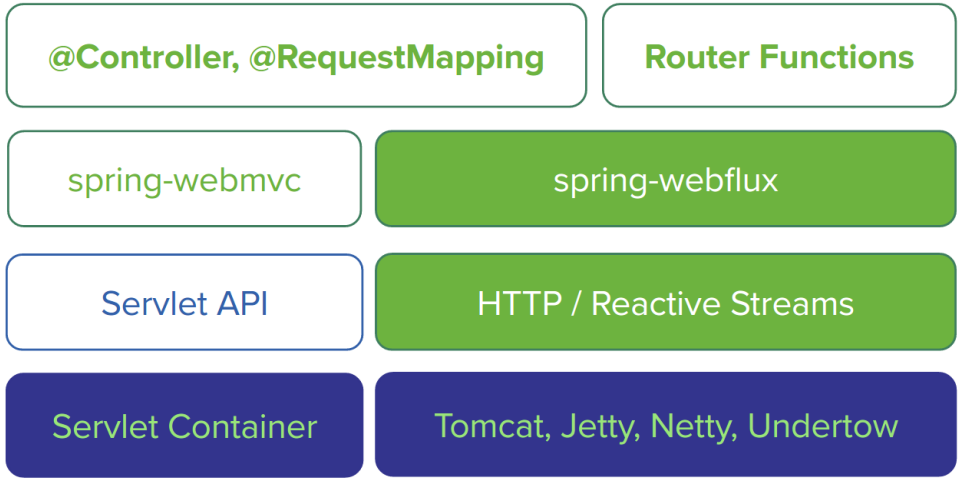
- Spring MVC
  - Basiert auf *ServletEngine*
  - *Controller* +
    - *ViewTemplates*
      - Thymeleaf, Velocity, JSP
  - REST-Controller +
    - dedizierte GUI (z. B. Angular)

Vaadin

- **UI-Framework** das mit Spring zusammen funktioniert
- Für **single-page** web UIs
- Logik und Layout werden in **Java-Klassen** erstellt (Swing-Ähnlich)
- Stark **beworben** jedoch relativ **unflexibel**
- Möglichkeiten ähneln **CoreFrame**

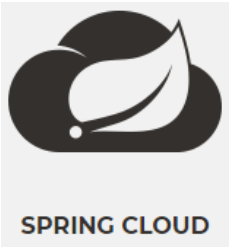
Spring WebFlux

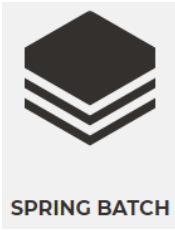
- Modul für **reaktive** Web-UIs
- **Controller-basierte** API mit reaktiven Rückgabewerten (Flux- / Mono-Streams)
- **Threads** werden nicht blockiert
- Handling ähnlich zu **Spring MVC**
- Passende **reaktive Client-Technologie** nötig

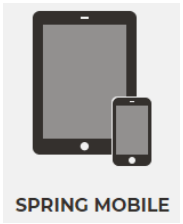


Weitere Module

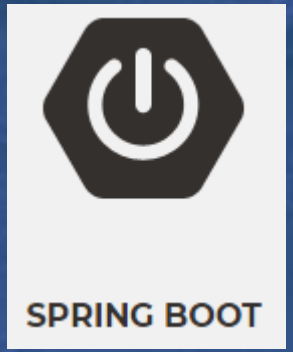
- **Spring Roo**: Rapid Application Development / Low code Generator (nicht aus einem Modell, sondern per Kommandozeile)
- **Spring Testing**: Unit-Tests, Integrationstests, …
- **Spring Cloud**: Tools für die Entwicklung verteilter Anwendungen (configuration management, service discovery, leadership election, distributed sessions, …)
- **Lombok**: Automatische, **implizite** Implementierung von Gettern, Settern, Konstruktoren, …
- **Spring Batch**
- **Statemachine**
- **Spring Mobile**
- **…**







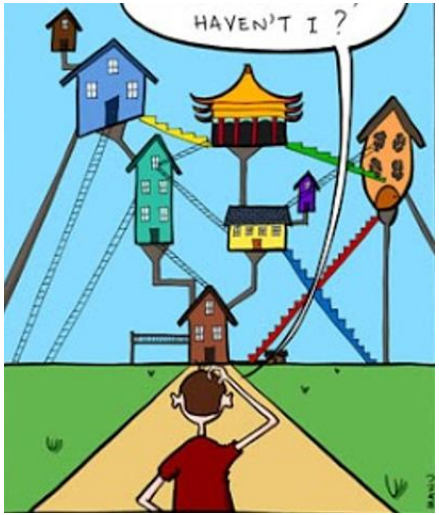
Spring Boot
Die Lösung für Springs größtes Problem



Welches Problem löst Spring Boot?

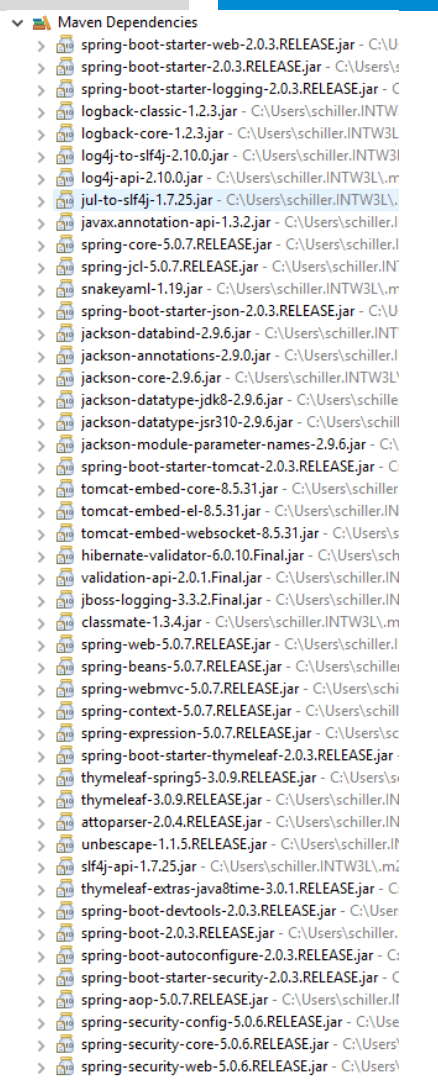
- Spring besteht aus sehr **vielen Modulen** und integriert sehr **viele Frameworks**
- Es müssen die richtigen **Abhängigkeiten** in der passenden **Version** ausgewählt werden
- Jede Komponente muss **konfiguriert** werden

- Spring Boot wählt **kontextabhängig** die sinnvollsten **Abhängigkeiten** und **konfiguriert** diese automatisch
- **Convention over Configuration**



Ich sage: „Sichere MVC-Anwendung mit Spring Boot 2“
Spring Boot sagt:

- Runtime: **Spring Boot** 2.0.3 und **Spring Core** 5.0.7
- Autorisierung / Authentifizierung: **Spring Security** 5.0.6
- Container: **Tomcat** 8.5.31 (embedded) auf Port 80
- Logging: **Logback** 1.2.3 (Log4J-Nachfolger) via slf4j (Logging-Facade)
- JSON-Binding: **Jackson** 2.9.6
- Datenhaltung: **JPA** in einer **H2** Datenbank
- View-Engine: **Thymeleaf** 3.0.9
- YAML-Verarbeitung: **SnakeYAML** 1.19
- Validierung: Validation API / **Hibernate Validator** 6.0.10
- …
- **Alles konfiguriert und „production ready“**



Features

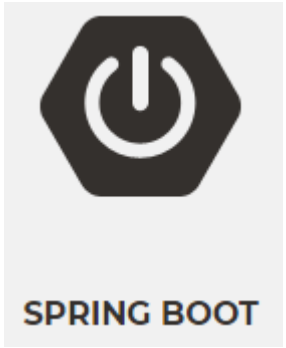
- Spring Boot Starter
  - JAR (per **Maven-** oder **Gradle-Abhängigkeit**)
  - Enthält
    - **Abhängigkeiten** zu benötigten Frameworks
    - Daten zur kontextabhängigen **Standardkonfiguration**

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
</dependency>
```

- Auto Configuration
  - Abhängig von
    - Frameworks im **Classpath**
    - existierender **Konfigurationsdatei**

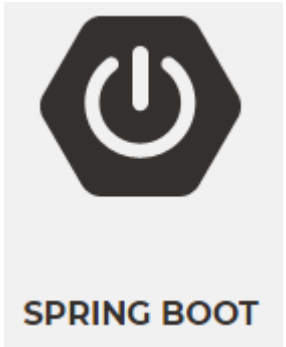
Features

- Versionsmanagement via Maven ***BOM*** (**B**ill **o**f **M**aterial)
  - Für die **wichtigsten** Abhängigkeiten wird die Version angegeben
  - Die Version aller anderen Abhängigkeiten wird entsprechend **abgeleitet**
- Auslieferung
  - Als *JAR* inkl. embedded Container (Tomcat, Jetty, …)
  - oder *WAR*
- Und (natürlich) **Module**…



Module

- Actuator
  - **HTTP-Endpunkte** für das Applicationmanagement
    - *health*, *info*, *metrics*, *sessions*, … (insg. 21)
    - Erweiterbar
- Developer Tools
  - Caches (auch von versch. Modulen) werden deaktiviert
  - Auto reload / auto restart
- Rest Repositories
  - **Spring Data Repositiories** per REST zugreifbar machen
  - HAL Browser
    - Per **Browser** durch Repositories navigieren



Fazit

- Extrem **umfangreich**
- Moderne **Architekturen** und **Techniken**
- Für viele **Anwendungszwecke** geeignet
- Gut **dokumentiert** und strake **Community**
- Alleine **Spring + Spring Boot** stellt oft schon eine Verbesserung zum Start auf der grünen Wiese dar

- Steile **Lernkurve**
- **Eignung** für **Projekte** und Auswahl der **Module** muss mit viel Hintergrundwissen geprüft werden

Vielen Dank!

Schiller, Nils
Software-Ingenieur

Tel. +49 (231) 61 804 - 142
Mail: nils.schiller@w3l.de