

Technische Dokumentation für die Software **we**Factor

Als Projektarbeit für die Vorlesung
Labor für Softwareentwicklung und Project Skills
im Studiengang Software Engineering
der Hochschule Heilbronn

Autoren:

Florens Hückstädt
Florian Wohlgemuth
Patrick Wohlgemuth
Samuel Morchner
Cagdas Bektas

Heilbronn, 18. Januar 2015

Änderungshistorie

Version	Name	Kapitel	Datum
0.1	Florens Hückstädt	2.1	17.01.2015
0.2	Florens Hückstädt	4, 2.2, 1.1, 3.2	18.01.2015

Inhaltsverzeichnis

Änderungshistorie	I
Inhaltsverzeichnis	II
Abkürzungsverzeichnis	III
Abbildungsverzeichnis	IV
1 Verwendete Technologien	1
1.1 Gradle	1
2 Installationsanleitung	2
2.1 Erforderliche Hard- und Software	2
2.2 Starten des Servers	3
3 Konfiguration der Anwendung	5
3.1 Environments	5
3.2 Konfiguration der Datenbank	6
3.3 Konfiguration der Social-Login-Provider	6
4 Entwicklung mit Eclipse	7
4.1 Initialisierung und Import des Projekts	7
4.2 Starten der Anwendung	7
Literaturverzeichnis	9

Abkürzungsverzeichnis

JRE Java Runtime Environment

JDK Java Development Kit

JVM Java Virtual Machine

IDE Integrated Development Environment

Abbildungsverzeichnis

1 Profil über die VM-Arguments aktivieren	6
---	---

1 Verwendete Technologien

sfdsfd

1.1 Gradle

2 Installationsanleitung

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

2.1 Erforderliche Hard- und Software

Zum Betreiben der Anwendung ist ein Java Runtime Environment (JRE) bzw zum Entwickeln ein Java Development Kit (JDK) in der Version 8 erforderlich. Dafür gelten folgende Systemvoraussetzungen laut (Oracle, o. J.)

Windows

- Windows 8 (Desktop)
- Windows 7
- Windows Vista SP2
- Windows Server 2008 R2 SP1 (64-Bit)
- Windows Server 2012 (64 Bit)
- RAM: 128 MB
- Datenträgerkapazität: 124 MB für JRE; 2 MB für Java Update
- Prozessor: Mindestens Pentium 2 266 MHz-Prozessor
- Browser: Internet Explorer 9 und höher, Firefox, Chrome

Mac OS X

- Intel-basierter Mac unter Mac OS X 10.8.3+, 10.9+
- Administratorberechtigungen für die Installation
- 64-Bit-Browser

Ein 64-Bit-Browser (Beispiele: Safari, Firefox oder Chrome) ist zur Ausführung von Oracle Java auf Mac OS X erforderlich.

Linux

- Oracle Linux 5.5+1
- Oracle Linux 6.x (32-Bit), 6.x (64-Bit)2

-
- Oracle Linux 7.x (64-Bit)²
 - Red Hat Enterprise Linux 5.5+1, 6.x (32-Bit), 6.x (64-Bit)²
 - Ubuntu Linux 12.04 LTS, 13.x
 - Suse Linux Enterprise Server 10 SP2+, 11.x
 - Browser: Firefox

2.2 Starten des Servers

Das Starten des Servers kann grundsätzlich über einen direkten java-Aufruf geschehen (siehe 2.2.1). Eine weitere Möglichkeit ist der Start des Servers über das Application-Plugin von gradle (siehe 2.2.2). Im Auslieferungszustand wird eine interne HSQLDB-Datenbank verwendet. Daher ist ein direktes Starten des Servers möglich. Zur Verwendung einer anderen Datenbank siehe Abschnitt 3.2.

2.2.1 Starten des Servers über die Kommandozeile

Zum Starten des Servers über einen direkten Java-Aufruf mit der Kommandozeile sind folgende Schritte durchzuführen:

1. Navigation mit der Kommandozeile in das Root-Verzeichnis von weFactor und Ausführen folgenden Befehls:

```
gradlew assemble
```

Dadurch wird im Verzeichnis /build/libs die Datei wefactor.jar erzeugt. Gradle ist im Projekt integriert und muss nicht separat installiert werden (siehe 1.1).

2. Ausführen des jars über den Befehl:

```
java -jar wefactor.jar
```

2.2.2 Starten des Servers über Gradle

Das Gradle Application-Plugin erweitert die Sprachen-Plugins um allgemeine anwendungsbezogene Tasks. Es erlaubt das Ausführen und Bündeln von Anwendungen für die Java Virtual Machine (JVM).

Für weitere Informationen zur Verwendung von Gradle innerhalb des weFactor-Projekts siehe 1.1.

Für weitere Informationen zum Application-Plugin siehe Gradle Dokumentation¹.

¹http://www.gradle.org/docs/current/userguide/application_plugin.html

Gradle run

Zum Starten des Servers über den Gradle-Task run sind folgende Schritte durchzuführen:

1. Navigation mit der Kommandozeile in das Root-Verzeichnis von weFactor
2. Ausführen des Gradle-Befehls:

```
gradlew run
```

Gradle distZip

Der Task distZip erzeugt ein Distributionsarchiv mit den erforderlichen Bibliotheken und entsprechenden Startskripten. Zur Erzeugung der Distribution sind folgende Schritte notwendig:

1. Navigation mit der Kommandozeile in das Root-Verzeichnis von weFactor und Ausführen folgenden Befehls:

```
gradlew distZip
```

Dadurch wird im Verzeichnis /build/distributions die Datei wefactor.zip erzeugt.

2. Entpacken des zip-Archivs.
3. Starten der Anwendung über das Startskript

```
wefactor.bat
```

Das Startscript ist im bin-Verzeichnis zu finden.

3 Konfiguration der Anwendung

Die Konfiguration der Anwendung findet hauptsächlich über die *application.properties* Datei statt. Hier können sämtliche Werte wie zum Beispiel Datenbank-URL, Logging-Einstellung usw. festgelegt werden. Die Konfigurationsdateien befinden sich im Verzeichnis *resources*. Siehe Listing 3.1 als Beispiel.

Listing 3.1 application.properties

```
app.name=weFactor

# IDENTITY (ContextIdApplicationContextInitializer)
spring.application.name=weFactor

logging.level.org.springframework.web: DEBUG
logging.level.org.hibernate: ERROR

# THYMELEAF (ThymeleafAutoConfiguration)
spring.thymeleaf.encoding=UTF-8
spring.thymeleaf.content-type=text/html

# EMBEDDED SERVER CONFIGURATION (ServerProperties)
server.tomcat.uri-encoding = UTF-8

spring.mvc.locale=en_UK
spring.mvc.date-format= dd/MM/yyyy

# INTERNATIONALIZATION (MessageSourceAutoConfiguration)
spring.messages.basename=messages
spring.messages.cacheSeconds=-1
spring.messages.encoding=UTF-8
```

Eine ausführliche Liste der Einstellmöglichkeiten ist in der Spring Dokumentation¹ zu finden.

3.1 Environments

Als Ergänzung zur *application.properties* Datei können umgebungsspezifische Einstellungen festgelegt werden. Dafür muss folgende Namenskonvention eingehalten werden: *application-profile.properties*.

So lassen sich zum unterschiedliche Datenbank-Einstellungen für die verschiedenen Zielumgebungen festlegen. Möchte man spezielle Properties für ein Profil mit dem Namen *dev* anlegen.

¹<http://docs.spring.io/spring-boot/docs/current/reference/html/common-application-properties.html>

So haben sich diese in einer Datei mit dem Namen *application-dev.properties* zu befinden. Profile können nun zum Beispiel über die VM-Arguments aktiviert werden (siehe Abbildung 1).

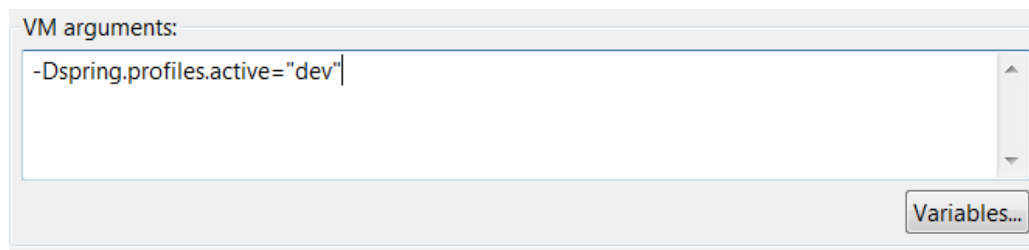


Abbildung 1: Profil über die VM-Arguments aktivieren

3.2 Konfiguration der Datenbank

3.3 Konfiguration der Social-Login-Provider

4 Entwicklung mit Eclipse

In diesem Kapitel werden die Schritte beschrieben, die notwendig sind den Quellcode in Eclipse zu bearbeiten und die Anwendung darüber zu starten. Die Anleitung liegt zwar ausschließlich für Eclipse vor, es kann jedoch auch ein alternatives Integrated Development Environment (IDE) verwendet werden.

4.1 Initialisierung und Import des Projekts

Zur Initialisierung des Projekts für Eclipse wird das Gradle-Plugin `eclipse` verwendet. Folgende Schritte sind durchzuführen:

1. Navigation mit der Kommandozeile in das Root-Verzeichnis von `weFactor` und Ausführen folgenden Befehls:

```
gradlew eclipse
```

Dadurch werden folgende Dateien erstellt:

- `.project`
- `.classpath`

Außerdem werden alle notwendigen Bibliotheken von Gradle aus dem Internet geladen.

2. Import des Projekts in Eclipse
 - a) Über `File/Import/Existing Project into Workspace` Import-Dialog aufrufen.
 - b) Über die Schaltfläche `Browse` neben der Auswahlliste zu `Select Root Directory` das Wurzelverzeichnis des `weFactor`-Projekts auswählen.
 - c) Bestätigen über die Schaltfläche `Finish`.

4.2 Starten der Anwendung

Die Anwendung wird über die `main`-Methode der Klasse `de.hhn.labswps.wefactor.Application` gestartet (siehe Listing 4.1).

Listing 4.1 Main-Methode

```
public static void main(final String[] args) {  
    SpringApplication.run(Application.class, args);  
}
```

Durch die Methode `SpringApplication.run()` wird der integrierte Tomcat-Server gestartet und anschließend die SpringMVC-Anwendung initialisiert. Innerhalb der Launch Configuration

können Parameter für die JVM eingegeben werden wie zum Beispiel das Setzen des aktiven Environments (siehe dazu Kapitel 3).

Für weitere Informationen zum Thema Starten einer Spring Boot Anwendung siehe Spring Dokumentation¹.

¹<https://spring.io/guides/gs/spring-boot/>

Literaturverzeichnis

Oracle. (o. J.). *Welche systemvoraussetzungen gelten für java?* Zugriff am 17.01.2015 auf <https://www.java.com/de/download/help/sysreq.xml>