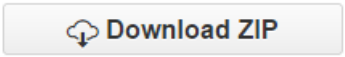
Übungsblatt zur
Vorlesung Cloud Computing

Übungsblatt:
Programmiermodelle für die Cloud
Functional Reactive Programming

Vorbereitung:

Holen sie sich die Vorlage zur Übung aus dem github Repository der Vorlesung (https://github.com/adersberger/cloudcomputing). Der einfachste Weg dafür ist, den Inhalt des gesamten Repositories als ZIP herunterzuladen (Button rechts unten im github Projekt) und zu entpacken.

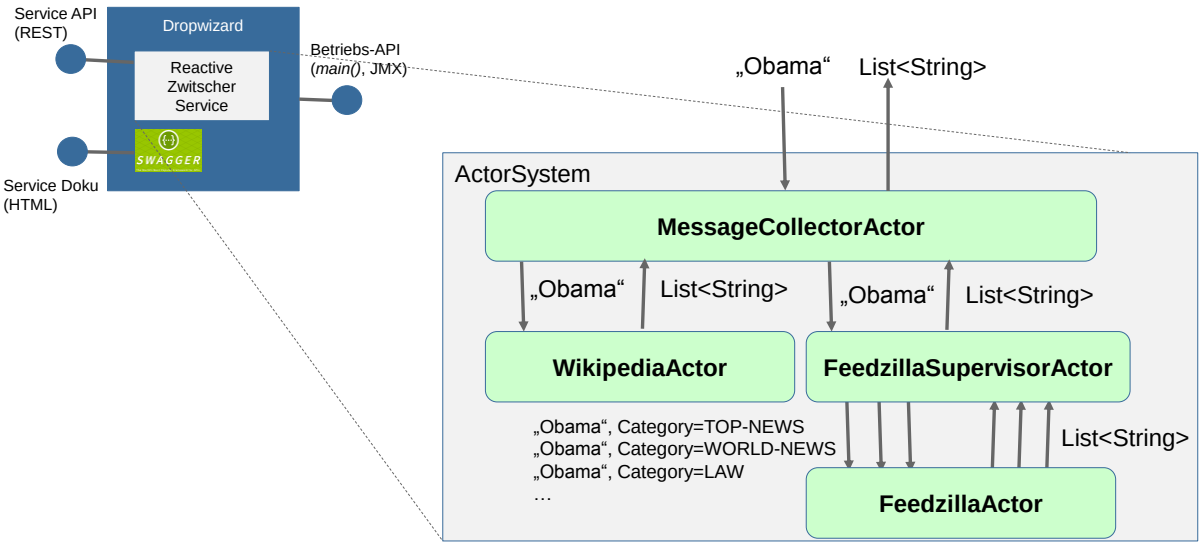


Erstellen Sie innerhalb von Netbeans ein neues Projekt (*Datei* ➔ *Neues Projekt*). Wählen Sie dabei eine *Maven / Java Application* als Projektart aus. Kopieren sie sämtliche Dateien aus der Vorlage in das Netbeans-Projekt und überschreiben sie dabei bereits existierende Dateien (z.B. die *pom.xml*) und Verzeichnisse (z.B. *src*).

Führen Sie das Maven Goal *clean package* aus.

Das Ziel:

Das Ziel der heutigen Übung ist es, einen Service für Zwitscher zu erstellen, mit dem Nachrichten sowohl aus Feedzilla (News-Feed-Aggregationsdienst) und Wikipedia (Wikipedia-Artikel) zu einem Suchwort extrahiert werden können.



Übungsblatt zur
Vorlesung Cloud Computing

Aufgaben:

1. Die Code aus der Vorlage sichten

Die folgende Abbildung zeigt den Inhalt der Vorlage.

Dort sind die Connectoren zu finden (*connectors*), mit denen Wikipedia und Feedzilla aufgerufen werden kann (synchron). Ferner existieren Vorlagen zu allen notwendigen Aktoren (*actors*). Dabei stehen zwei Klassen mit *main()*-Methoden zur Verfügung: Main nutzt die Connectoren direkt und sequenziell. *MainWithActors* nutzt das Aktorensystem, um die Ergebnisse zu ermitteln. Die jeweiligen Aktoren nutzen dann intern jeweils auch wieder die Connectoren. Ferner ist noch eine Konfigurationsdatei für akka hinterlegt (*application.conf*), die die Log-Ausgaben konfiguriert.

Machen sie ausgehend von den Main-Klassen einen Rundgang durch den Code und erschließen sie sich das definierte Aktorensystem dadurch. Nutzen sie bei Bedarf die akka Online-Dokumentation bei Verständnisproblemen. Lassen sie die sequentielle Variante (*Main*) einmal laufen und notieren sie sich die dabei gemessene Laufzeit (siehe Konsolenausgabe).

2. Das Aktor-System lauffähig machen

Füllen sie die Lücken im Aktor-System. Gehen sie dazu durch den Code und füllen sie alle Code-Stellen, an denen ein TODO als Kommentar hinterlegt ist.

Führen sie das Aktorensystem aus (*MainWithActors*) und vergleichen sie die Laufzeit mit der sequenziellen Variante.

3. Kür: Den Code in das Dropwizard Projekt integrieren

Falls sie schnell durch die vorherigen Übungen gekommen sind, so können sie noch einen Schritt weiter gehen: Den Code, der bisher nur aus den Main-Klassen angestoßen wird in den Code der Übung zu REST auf Basis von Dropwizard integrieren. Sie müssen dazu das Aktorensystem bei der Initialisierung der Dropwizard-Applikation erzeugen und dann eine

Übungsblatt zur
Vorlesung Cloud Computing

REST-Schnittstelle entwerfen, die einen Request an das Aktorensystem delegiert und die Ergebnisse nach Außen reicht. Hinweise dazu finden sie im folgenden Blog-Artikel: http://www.hascode.com/2013/12/jax-rs-2-0-rest-client-features-by-example.

Viel Spaß!

Übungsblatt zur
Vorlesung Cloud Computing

Quellen:

Diese Übung soll auch eine eigenständige Problemlösung auf Basis von Informationen aus dem Internet vermitteln. Sie können dazu für die eingesetzten Technologien z.B. die folgenden Quellen nutzen:

Netbeans

- http://netbeanside61.blogspot.de/2008/04/top-10-netbeans-ide-keyboard-shortcuts.html

Maven

- http://maven.apache.org/guides/getting-started

JAX-RS

- https://github.com/wordnik/swagger-core/wiki/Java-JAXRS-Quickstart
- https://jersey.java.net/documentation/latest/jaxrs-resources.html

Dropwizard

- http://dropwizard.io/manual/core.html
- http://kielczewski.eu/2013/04/developing-restful-web-services-using-dropwizard

Swagger

- http://swagger.io
- https://github.com/wordnik/swagger-core/tree/master/modules/swagger-annotations/src/main/java/com/wordnik/swagger/annotations
- http://java.dzone.com/articles/swagger-make-developers-love

Beispiele für REST APIs

- https://dev.twitter.com/rest/tools/console
- http://www.programmableweb.com/apis/directory

akka

- https://github.com/akollegger/akka-jersey-samples
- http://doc.akka.io/docs/akka/2.3.6/general/actor-systems.html#actor-systems