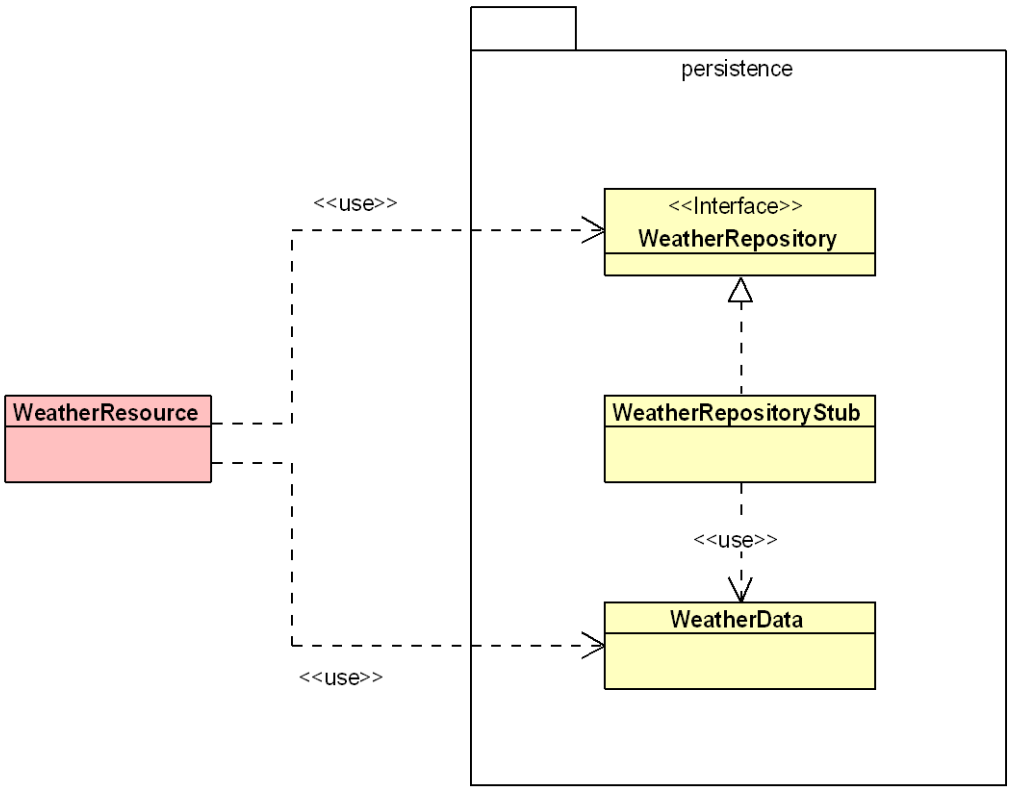
Aufgabe "Wetterdaten"

Fertigstellen eines Wetterdaten-REST-Service bei den Wetterdaten eingereicht, angerufen, verändert und gelöscht werden kann. Die Schnittstelle soll eine REST-Schnittstelle sein.

Ausgangslage

Im vorliegenden Maven-Projekt sind folgende Klassen sind vorhanden:

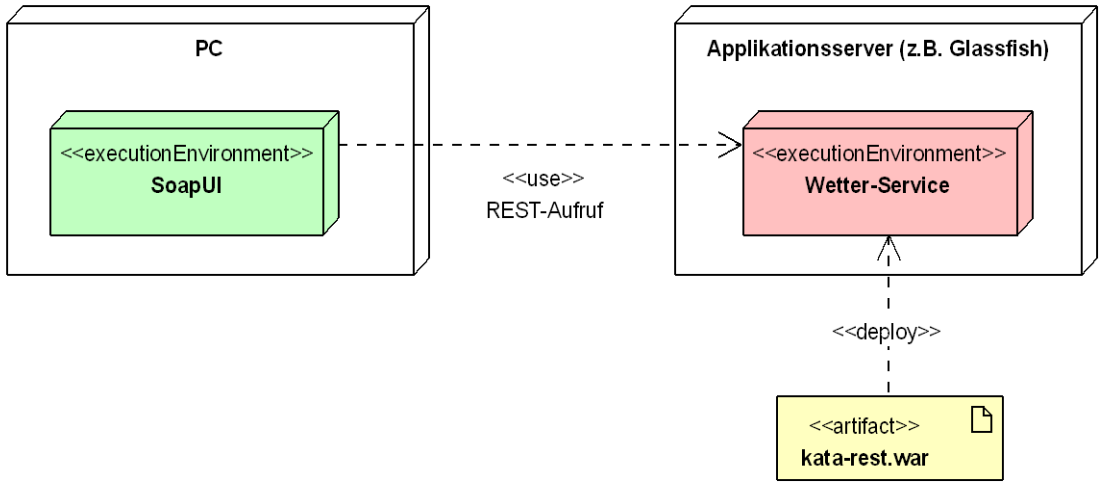


Die gelb markierten Klassen sind vollständig vorhanden und dürfen nicht verändert werden. Die rot markierte Klasse ist unvollständig vorhanden und muss vervollständigt werden. Weiter Klassen die notwendig sind dürfen selbstverständlich erstellt werden.

Das WeatherRepository bietet eine Schnittstelle zur Persistierung der Daten, welche für den REST-Service verwendet werden muss. Beachte, bei der WeatherRepositoryStub Implementierung der Schnittstelle es handelt sich dabei um eine ganz einfache Peristierung mittels Speicherung ins Memory.

Für das WeatherRepository existiert ein JUnit-Test, welcher alle Funktionen des Repository testet. Zugleicht dient der Test als Dokumentation der Schnittstelle. Durch Analyse des Tests wird klar, wie das Repository zu verwenden ist.

Verteilungsdiagramm für den Testaufbau

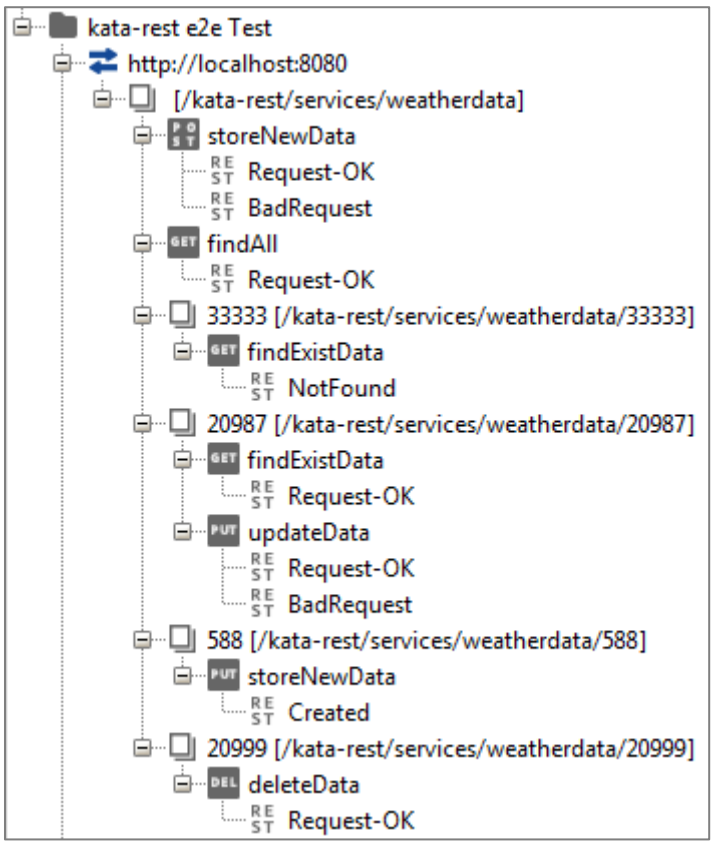


Testing mittels SoapUI

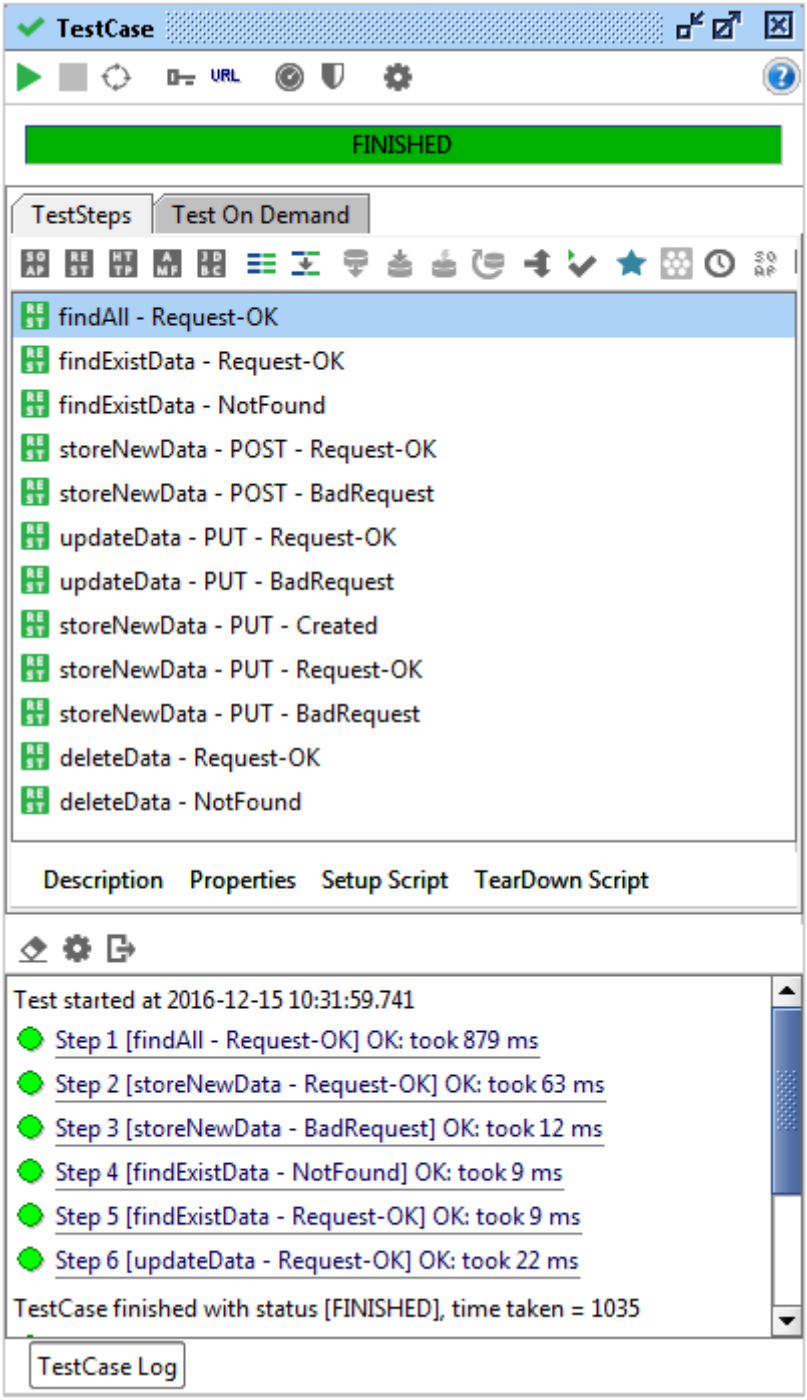
Im Verzeichnis 'SoapUI' des Maven-Projekts, befindet sich ein XML-Datei welche ein SoapUI Projekt darstellt. Im SoapUI kann dieses Projekt importiert werden. Dieses SoapUI Projekt dient als Test für die REST-Schnittstelle. Mittels Analyse der Test-Methoden und dem TestCase mit allen Schritten, wird klar mit welchem Request welche Antworten bzw. http-Status-Code zu erwarten ist.

Basis REST-URL: http://localhost:8080/kata-rest/services/weatherdata

Folgende Testmethoden sind vorhanden:



Folgender TestCase ist vorhanden:



Der TestCase muss bei erfolgreicher und richtiger Implementation der Schnittstelle mit allen Schritten erfolgreich sein. Der TestCase entspricht den funktionalen Anforderungen.

Beachte: Da die Daten im Speicher (Memory) "gespeichert" werden, läuft der TestCase nur einmal, direkt nach dem Neustart des Applikationsservers, korrekt durch. Für einen erneuten Test-Ablauf muss der Applikationsserver neu gestartet werden.

Funktionale Anforderungen

Der TestCase bildet praktische alle funktionalen Anforderungen ab.
Im Code ist ersichtlich, dass die Verwendung des 'WeatherRepository' eine 'RepositoryException' auslösen kann. Alle diese Exceptiones sollen auf den HTTP-Error-Code 900 gemappt werden.

Nicht funktionale Anforderungen

- Maven 3.x
- SubVersion (SourceForge)
- Java JDK 1.8 oder höher
- JAX-RS (Java REST API, Teil von JEE 7)
- JUnit 4.1x
- Code-Konvention
    - Sun/Oracle Code Conventions
    - http://www.oracle.com/technetwork/java/javase/documentation/codeconvtoc-136057.html
    - Ausnahme Zeilenlänge max. 150 Zeichen
- Mittels Maven soll ein WAR-File gebildet werden können welches auf einem JEE-Application-Server deployed werden kann.

Aufgabe

Umsetzen der funktionalen und nicht funktionalen Anforderungen. Dazu muss das vorliegend Projekt erweitert werden.

Abgabe

Abgegeben werden soll ein ZIP-File mit dem entsprechenden Maven-Projekt. Dieses ZIP muss alle notwendigen Dateien enthalten. Im POM soll zusätzlich die URL zum Projekt bei SourceForge hinterlegt sein.