**Konfigurationsmanagement Lab Report**

Abgabe 02 „Maven“

David Riegler

Islam Hemida

Lukas Weber

Repository: https://github.com/iamWebLuk/20Abgabe02-Hemida-Riegler-Weber

Vorgehen

(Einleitung, Bugbehebung und Testen)

Zu Beginn des Projektes wurde ein neues Repository auf github erstellt. Jeder Beteiligte hat das Repository kopiert um remote Zugriff auf die verteilte Versionierung zu haben. Dies ist wichtig, um ein gemeinsames Arbeiten zu ermöglichen und Fehler schnell und zielorientiert beheben zu können. Da in einem Unternehmen stehts neue Mitarbeiter einsteigen können und von dem ganzen Code „Überfallen“ werden, ist es sehr zu empfehlen immer gut lesbare Commits erstellen und auch den Code entweder gut zu kommentieren, oder im optimalfall, den Code so schreiben, dass man es erst gar nicht kommentieren muss, da der Code selbst erklärend ist. Das vorhandene Queue Template wurde zu diesem Zweck verwendet. Hierbei mussten einige Bugs entfernt werden um eine Testung zu ermöglichen. Es wurden alle Dokumentationsvorgaben eingehalten.

Um die vorhandenen Bugs im vorgegebenen Programm zu beheben wurde zunächst eine umfangreiche Sichtung des Codes vorgenommen. Dabei wurden auch die Debugging Möglichkeiten der IntelliJ IDE verwendet. Zunächst wurden die Klassenvariablen und der Konstruktor auf eine sinnvolle und funktionale Weise abgeändert. Die inhärente maxSize Variable wurde in den Konstruktor integriert (this.maxSize = maxSize). Die Elementliste wurde ebenfalls hinzugefügt und das neu Erstellen einer ArrayList bei jeder neuErstellung der String Queue Klasse wurde berichtigt (this.element = element). Danach stellte sich heraus, dass die offer Methode nicht die gewünschte Funktionalität besaß und das Ungleichzeichen wurde mit einem Kleiner-als-zeichen ersetzt. Ebenso wurde die Bedingung der poll Methode angepasst (if (element != null), um nur wirklich zu löschen, wenn auch ein Element vorhanden ist). Die remove Methode besaß eine Stringzuweisung, die eine darauffolgende If- Bedingung nutzlos machte und wurde entfernt. Schlussendlich wurde die peek Methode lediglich syntaktisch umgeschrieben um eine effektivere Zeilenanzahl zu erreichen.

Nachdem die Funktionalität der Queue Klasse hergestellt wurde, ist begonnen worden einen JUnit Test für die verschiedenen Methoden der Queue Klasse zu erstellen. Dabei ist die selbe Vorgehensweise wie in der Übung 1 Konfigurationsmanagement angewandt worden. Zunächst sind für die String Queue Test Klasse zwei String Queue Elemente sowie zwei Elementlisten erstellt worden. Vor jeder Testung werden Elemente in die Liste eingefügt und mithilfe der Konstruktors in der String Queue Klasse Objekte mit den Elementen und einer maxSize erstellt. Die Testung aller weiteren Methoden wurde daraufhin durchgeführt. Eine 100% Testabdeckung ist ohne Vorfälle erreicht worden.

Um eine Generics Queue Klasse zu ermöglichen wurde zunächst das IQueue Interface angepasst um mit alle Generics Objekte und nicht nur Strings arbeiten zu können (interface IQueue<Object>). Die String Queue Klasse wurde dementsprechend abgeändert (class StringQueue implements IQueue <String>). Daraufhin wurde lediglich eine Generics Queue Klasse erstellt und alle Strings mit Objects ersetzt. Schlussendlich wurde eine Generics Queue Test Klasse erstellt und die gleiche Herangehensweise wie für die String Queue Test Klasse angewandt. Um die Funktionsweise zu testen wurden noch weitere Generics wie zum Beispiel Integer in die Elementliste eingefügt.

Es wurde auf eine ganzheitliche Dokumentierung geachtet und es fanden regelmäßige und umfangreich kommentierte git commits statt.

Als nächsten erstellten wir die Maven Pom Datei. Pom steht für Project Object Model. Die Syntax ähnelt HTML. Eine pom.xml Datei wird verwendet, um Hintergrundinformation rund um das Projekt zu generieren. Als aller erstes haben wir den Link eingefügt, das zu unserem GitHub Repository führt. Als nächsten Schritt haben wir jeden einzelnen Developer angegeben der an dem Projekt mitgearbeitet hat. Hier wurden Informationen angegeben wie etwa der GitHub Benutzer Name, der echte Name des Developer, dessen E-Mail Adresse, ein Link die zur seiner eigenen Homepage führt, in welcher Firma bzw. Universität wir tätig sind, welche Rolle er in dem Betrieb bzw. Universität einnimmt, einen Link zur GitHub Seite des Developer und ein Avatar Bild. Weiters haben wir über Intellij die einzelnen Targets ausgeführt und alle bis auf Maven Site kompilieren ohne Warnings. Nachdem ausführen wurde ein Target Folder erstellt wo man die kompilierten Targets auffindet.

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

(Maven site) ->

(Weitere Beschreibungen siehe Vorgabe) ->?