Иванников Дмитрий Владимирович
магистрант
Дёмин Степан Евгеньевич
магистрант
ФГБОУ ВО «Московский государственный
технологический университет «СТАНКИН»
г. Москва

ОРГАНИЗАЦИЯ ВЗАИМОДЕЙСТВИЯ С БАЗАМИ ДАННЫХ В РАМКАХ ПРОЕКТА АВТОМАТИЗИРОВАННОГО ТЕСТИРОВАНИЯ ВЕБ-СЕРВИСОВ

Аннотация: *в данной научной статье рассмотрена одна из задач автоматизированного тестирования веб-сервисов: организация взаимодействия работы с базами данных. Данный вопрос является актуальным, как при организации процесса тестирования пользовательских интерфейсов, так и программного интерфейса приложения.*

Ключевые слова*: тестирование, автоматизация тестирования, разработка программного продукта, архитектура программного обеспечения, пользовательские сценарии.*

Введение

С помощью автоматизации становится возможным сократить время на разработку и снизить затраты на стадии тестирования.

Для современного рынка программного обеспечения частота обновления продукта является основным показателем рентабельности и способности к конкурентному развитию компании. Ради этого все участники нацелены снизить затраты времени на каждом этапе разработки, особенно на этапе тестирования, так как тесты и, в частности, регрессионное тестирование требуют значительных временных затрат.

Во время проектирования нового продукта, разработчики взаимодействуют с большим объёмом кода, хорошей практикой является переиспользовать

Content is licensed under the Creative Commons Attribution 4.0 license (CC-BY 4.0)

готовый код, что сокращает время на разработку и позволяет использовать уже готовые решения. Но такой подход влечёт за собой ряд проблем, таких как: запутанность именования классов и переменных, возможность использования кода в определённом контексте. Таким образом, чем раньше будут выявлены все ошибки, тем быстрее компания сможет выпустить продукт на рынок.

Как правило, подобные ошибки выявляются на стадии функционального тестирования с помощью юнит-тестов. Главным преимуществом таких тестов является возможность многократного запуска и высокая скорость, что позволяет программистам использовать их каждый раз по выполнении своей задачи. Помимо этого, для каждого enterprise-приложения ключевую роль играют данные, что делает необходимым проверку информации непосредственно из базы данных. Таким образом каждый проект автоматизированного тестирования должен уметь взаимодействовать с базой данных приложения.

1. Организация взаимодействия с базами данных.

Для организации взаимодействия с базами данных был использован язык программирования Kotlin, главной его особенностью является работа поверх JVM, что делает его полностью совместимым с таким популярным языком программирования, как Java и среда разработки intellij idea. Оба инструмента принадлежат компании jetbrains. Перед реализацией необходимо унаследоваться от двух модулей: java.sql.DriverManager и java.sql.ResultSet, это можно сделать командой import. Затем необходимо определить базовый класс, который и будет осуществлять взаимодействие с базой данных, в данном примере класс назван DataBase. Данный класс должен иметь конструктор по умолчанию с полями определяющими подключение к базе данных. В данном примере это адрес базы данных (url), имя пользователя (user) и пароль (password).

Основные операции взаимодействия с базами данных при тестировании веб-сервисов это получение и обновление данных. Это значит, что в базовом классе необходимо определить два метода. При необходимости функционал класса может быть расширен и адаптирован для иных условный использования. Для получения данных был определен метод executeSqlQuery, который в

Содержимое доступно по лицензии Creative Commons Attribution 4.0 license (CC-BY 4.0)

качестве принимаемого значения ожидает строковую команду SQL запроса. При успешном выполнении операции метод возвращает ResultSet. Аналогичную конструкцию имеет метод обновления данных, который называется: executeSqlUpdate, только конструкция данного метода не подразумевает возвращаемого значения, а ответственность за успешность проведённой операции остаётся на компиляторе языка программирования, который всегда сообщит об ошибке в консоли, либо возможно передать данный участок кода встроенными механизмами обработок ошибок. Ниже представлен код реализации класса DataBase (рис. 1).

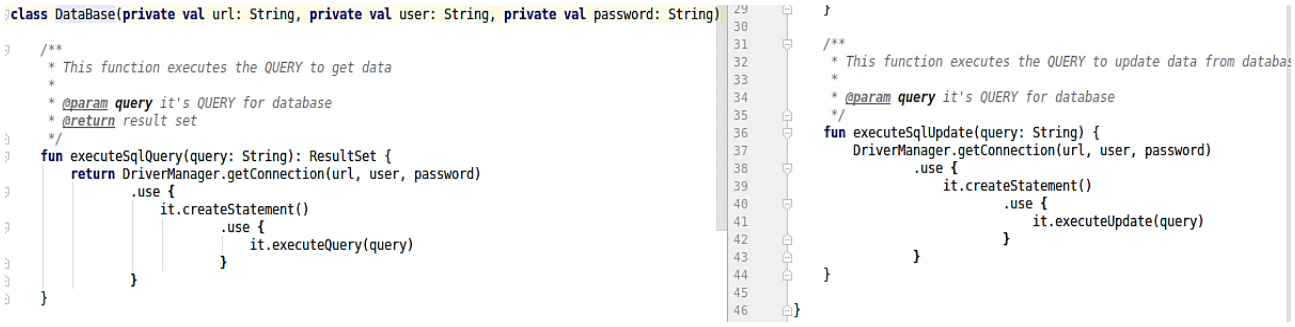

Рис. 1. Код реализации класса DataBase

2. Обращение к базе данных во время выполнения теста.

Использовав такой известный фреймворк для тестирования как TestNg, определяем архитектуру проекта автоматизированного тестирования, и базовые классы.

Смоделируем сценарий назначения врача на вызов пациента (рис. 2).

```
class SetDoctorToOrder {

    @Test
    @Description( value: "Назначение заказа на врача")
    fun execute() {
        println(orderId)

        val nowDate = Date()
        val formatForDateNow = SimpleDateFormat( pattern: "yyyy-MM-dd HH:mm:ss")
        val date = "\"" + formatForDateNow.format(nowDate) + "\""
        DataBase(URL, USER, PASSWORD)
                .executeSqlUpdate(
                        query: "UPDATE doc_telemed_card SET DOCTOR_ID = ${DoctorInfo.doctorId}
        DataBase(URL, USER, PASSWORD).executeSqlUpdate(
                query: "UPDATE doc_calling_card\n" +
                        "SET DATE_CONFIRMATION = $date,\n" +
                        " DATE_DESTINATION = $date,\n" +
                        " DOCTOR_ID = ${DoctorInfo.doctorId}\n" +
                        " WHERE ORDER_ID = ${orderId}")
    }
}
```

Рис. 2 Пример использования класса DataBase во время выполнения теста

Content is licensed under the Creative Commons Attribution 4.0 license (CC-BY 4.0)

Как видно из фрагмента кода для вызова команды обращения к базе данных необходимо объявить класс DataBase, передать ему параметры подключения к базе данных и вызвать метод взаимодействия передав в качестве параметра SQL запрос.

Список литературы

1. Алиев Р. С. о. Информационная система электротехнических расчётов WebЭнерго (тезисы доклада). Семинар «Компьютерные средства подготовки персонала», ВВЦ, павильон «Электрификация», 16–20 октября 2000.

2. Алиев Р. С. о. Математическая модель оценки сложности разработки программного обеспечения // Вестник МГТУ Станкин. – М., 2015. – №2 (33). – С. 93–97.

3. Документация программы SoapUI [Электронный ресурс]. – Режим доступа: https://www.soapui.org/soapui-projects/soapui-projects.html (дата обращения: 20.04.18).

4. Документация библиотеки Junit [Электронный ресурс]. – Режим доступа: https://junit.org/junit5/docs/current/user-guide/ (дата обращения: 20.04.18).

5. Кузнецов В.Е. Представление в ЭВМ неформальных процедур. – М.: Наука, 1989. – 160 с.

6. Официальный сайт языка программирования Kotlin [Электронный ресурс]. – Режим доступа: https://jetbrains.ru/products/kotlin/ (дата обращения: 01.05.18).

7. Тестирование. Фундаментальная теория [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/post/279535/ (дата обращения: 20.03.18).

8. Charitha Kankanamge. Web Services Testing with soapUI. Birmingham: Packt Publishing Ltd, 2012.

9. McConnell S. Best Practices. IEEE Software, Vol. 13, No. 6, December 1996.

10. SOAP UI и тестирование сервисов [Электронный ресурс]. – Режим доступа: https://www.dataart.ru/news/soap-ui-i-testirovanie-servisov/ (дата обращения: 20.04.18).

Содержимое доступно по лицензии Creative Commons Attribution 4.0 license (CC-BY 4.0)