Харківський національний університет ім. В. Н. Каразіна

Факультет комп'ютерних наук

Лабораторна робота №12

З дисципліни

«Математичні методи і технології тестування та верифікації програмного забезпечення»

Тема: **«Тестування API (Application programming interface)**»

Виконав:

Студенте групи КС-21

Зоренко Я. С.

Перевірив:

Ст. в. Мелкозьорова О. М.

Харків – 2019

Метою виконання поточної лабораторної роботи було вивчити тестування API та написати запити до веб-ресурсу. Програма містить класи GETRequest, POSTRequest та тест-клас GETRequestTest, залежності Maven та файл web.xml. (Рис. 1).

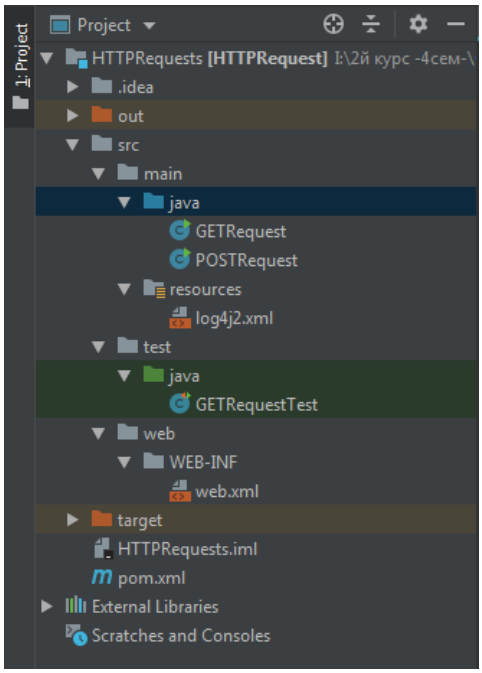

Рисунок 1 – Структура програми

Зміст класу GETRequest:

```
public class GETRequest {
    private final static Logger LOG =
LogManager.getLogger(GETRequest.class);
    private HttpURLConnection connection;
    public static void main(String[] args) throws IOException {
        GETRequest request = new GETRequest();
        String query = "http://localhost:8080/EE_war_exploded/";
        LOG.info(request.createGETRequest(query));
    }
    public String createGETRequest(String query) throws IOException {
        StringBuilder builder = new StringBuilder();
        connection = (HttpURLConnection) new URL(query).openConnection();
        connection.setRequestMethod("GET");
        connection.setUseCaches(false);
        connection.setConnectTimeout(2500);
        connection.setReadTimeout(2500);
        connection.connect();
        if (checkResponseCode()) {
            BufferedReader reader = new BufferedReader(new
InputStreamReader(connection.getInputStream()));
            String line;
            while ((line = reader.readLine()) != null) {
                builder.append(line);
                builder.append("\n");
            }
        } else {
            builder = new StringBuilder();
        }
```

```
        return builder.toString();
    }
    public boolean checkResponseCode() throws IOException {
        LOG.info(HttpURLConnection.HTTP_OK == connection.getResponseCode()
? "OK" : "ERROR.. code = " + connection.getResponseCode());
        return HttpURLConnection.HTTP_OK == connection.getResponseCode();
    }
}
```

Зміст класу POSTRequest:

```
public class POSTRequest {
    private final static Logger LOG =
LogManager.getLogger(POSTRequest.class);
    public static void main(String[] args) {
        String query = "http://localhost:8080/index.jsp";
        HashMap<String, String> postDataParams = new HashMap<>();
        postDataParams.put("6", "6");
        postDataParams.put("-", "-");
        postDataParams.put("7", "7");
        postDataParams.put("%3D", "%3D");
        LOG.info(performPostCall(query, postDataParams));
    }
    private static String performPostCall(String requestURL,
HashMap<String, String> postDataParams) {
        URL url;
        StringBuilder response = new StringBuilder();
        try {
            url = new URL(requestURL);
            HttpURLConnection connection = (HttpURLConnection)
url.openConnection();
            connection.setReadTimeout(15000);
            connection.setConnectTimeout(15000);
            connection.setRequestMethod("POST");
            connection.setDoInput(true);
            connection.setDoOutput(true);
            OutputStream outputStream = connection.getOutputStream();
            BufferedWriter writer = new BufferedWriter(new
OutputStreamWriter(outputStream, StandardCharsets.UTF_8));
            writer.write(getPostDataString(postDataParams));
            writer.flush();
            writer.close();
            outputStream.close();
            int responseCode = connection.getResponseCode();
            LOG.info((responseCode == 200) ? "OK" : "ERROR.. code = " +
responseCode);
            if (responseCode == HttpURLConnection.HTTP_OK) {
                String line;
                BufferedReader reader = new BufferedReader(new
InputStreamReader(connection.getInputStream()));
                while ((line = reader.readLine()) != null) {
                    response.append(line);
                    response.append("\n");
                }
            } else {
                response = new StringBuilder();
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return response.toString();
    }
```

Maven залежності:

```
<dependencies>
    <dependency>
        <groupId>org.junit.jupiter</groupId>
        <artifactId>junit-jupiter-api</artifactId>
        <version>5.4.0</version>
        <scope>test</scope>
    </dependency>
        <dependency>
        <groupId>org.apache.logging.log4j</groupId>
        <artifactId>log4j-core</artifactId>
        <version>2.11.2</version>
    </dependency>
    <dependency>
        <groupId>org.apache.logging.log4j</groupId>
        <artifactId>log4j-api</artifactId>
        <version>2.11.2</version>
    </dependency>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.13-beta-2</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.apache.httpcomponents</groupId>
        <artifactId>httpclient</artifactId>
        <version>4.5.3</version>
    </dependency>
</dependencies>
```

Висновок: в цій лабораторній роботі ми ознайомилися з способом тестування, використовуючи API (Application programming interface), яке являє собою опис способів, котрими одна комп'ютерна програма може взаємодіяти з іншою програмою. Як правило, процедура обміну інформацією і формат даних, які ми передаємо, структуровані. Також при зверненні до веб API зазвичай використовуються запити HTTP. До таких відносять:

GET – використовують для отримання або читання даних;

PUT – використовують для поновлення ресурсу;

POST – використовують для створення нового ресурсу;

DELETE – використовують для видалення даних.

Під час виконання лабораторної роботи було створено два запиту: GET і POST, які були протестовані за допомогою тестів. Добре освоїв матеріал та зробив висновки.