Отчет по лабораторной работе 5

1 Постановка задачи

В процессе выполнения лабораторной работы необходимо выполнить следующие задачи:

1. Реализовать интерфейс, для которого предусмотрено внедрение зависимостей.
2. Сделать несколько реализаций этого интерфейса в рамках контейнера.
3. Показать все доступные реализации этого интерфейса.

2 Выполнение

2.1 Структура проекта

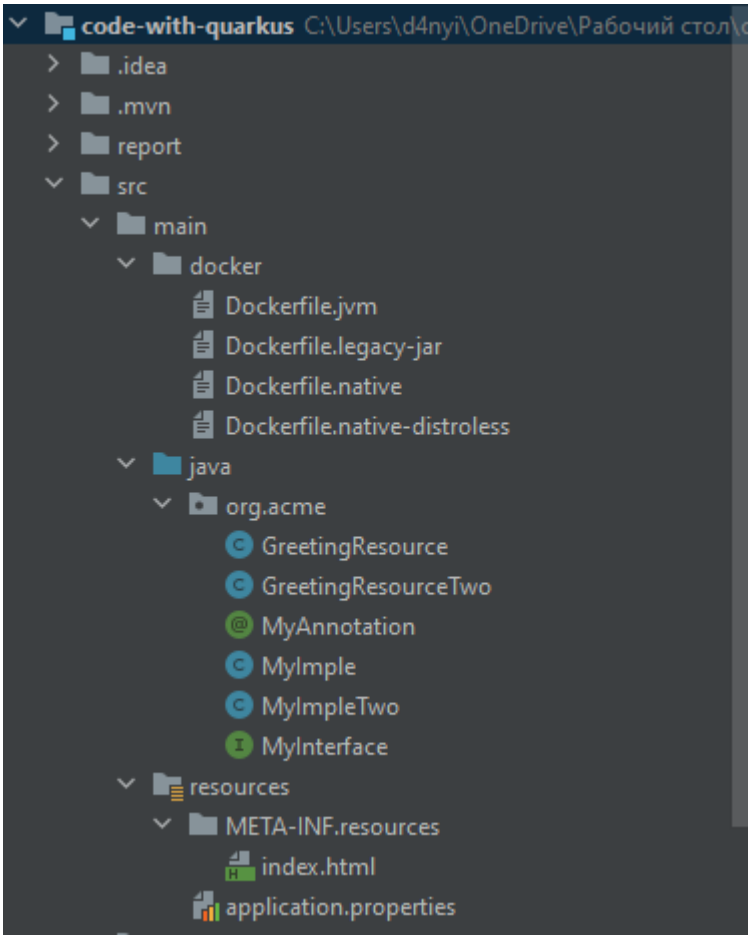

Рисунок 1. Структура проекта

2.2 Задание

Интерфейс MyInterface

Листинг 1. Листинг интерфейса MyInterface.

```
package org.acme;

public interface MyInterface {
    void draw();
}
```

Листинг 2. Первая реализация интерфейса MyInterface.

```
package org.acme;

import javax.enterprise.context.ApplicationScoped;

@ApplicationScoped
public class MyImple implements MyInterface {

    @Override
    public void draw() {
        System.out.println("Работает!");
    }
}
```

Листинг 3. Вторая реализация интерфейса MyInterface.

```
package org.acme;

import javax.enterprise.context.ApplicationScoped;

@ApplicationScoped
@MyAnnotation
public class MyImpleTwo implements MyInterface {

    @Override
    public void draw() {
        System.out.println("Работает дважды!");
    }
}
```

Листинг 4. Вторая реализация интерфейса MyInterface.

```
package org.acme;

import javax.enterprise.context.ApplicationScoped;

@ApplicationScoped
@MyAnnotation
public class MyImpleTwo implements MyInterface {

    @Override
    public void draw() {
        System.out.println("Работает дважды!");
    }
}
```

В нашем случае присутствует 2 реализации интерфейса. Используем аннотацию MyAnnotation. Данный способ поможет нам устранить неоднозначность использования реализации MyInterface.

Листинг 5. Аннотация MyAnnotation.

```
package org.acme;

import javax.inject.Qualifier;
import java.lang.annotation.Retention;
import java.lang.annotation.Target;

import static java.lang.annotation.ElementType.*;
import static java.lang.annotation.RetentionPolicy.RUNTIME;

@Qualifier
@Retention(RUNTIME)
@Target({METHOD, FIELD, PARAMETER, TYPE})
public @interface MyAnnotation {}
```

Листинг 6. Аннотация MyAnnotation.

```
package org.acme;

import javax.inject.Qualifier;
import java.lang.annotation.Retention;
import java.lang.annotation.Target;

import static java.lang.annotation.ElementType.*;
import static java.lang.annotation.RetentionPolicy.RUNTIME;

@Qualifier
@Retention(RUNTIME)
@Target({METHOD, FIELD, PARAMETER, TYPE})
public @interface MyAnnotation {}
```

Листинг 7. Класс GreetingResourceTwo.

```
@Path("/hello")
public class GreetingResourceTwo {

    @Inject
    @MyAnnotation
    MyInterface myInterface;

    @Inject
    @Any
    Instance<MyInterface> instanceNew;

    @GET
    @Produces(MediaType.TEXT_PLAIN)
    public String hello() {
        for (MyInterface m : instanceNew) {
            m.draw();
        }
        return "Hello RESTEasy";
    }

    @GET
    @Path("1")
    @Produces(MediaType.TEXT_PLAIN)
    public String hello1() {
        for (MyInterface m : instanceNew) {
            if (m instanceof MyImple)
                m.draw();
        }
        return "Привет";
    }

    @GET
    @Path("2")
    @Produces(MediaType.TEXT_PLAIN)
    public String hello2() {
        for (MyInterface m : instanceNew) {
            if (m instanceof MyImpleTwo)
                m.draw();
        }
        return "Пока";
    }
}
```

3 Результаты выполнения

В качестве результата работы приложены соответствующие скриншоты.

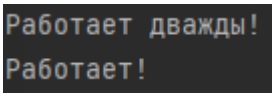

Рисунок 2. Перейдем на 'http://localhost:8080/hello' и посмотрим на результат в консоли

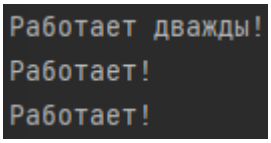

Рисунок 3. Перейдем на 'http://localhost:8080/hello/1' и увидим вывод слова 'Работает!'

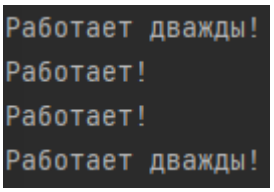

Рисунок 4. Перейдем на 'http://localhost:8080/hello/2' и увидим вывод слова 'Работает дважды!'

Вывод

В результате выполнения лабораторной работы, мы попрактиковались в работе с quarkus. В итоге, был реализован интерфейс MyInterface и несколько его реализации, которые в итоге были продемонстрированны.