Лабораторная работа 4.

Приложение akka предназначенное для удаленного тестирования JS приложений.

1. Задача

Требуется разработать приложение использующее технологию akka и позволяющее с помощью http запроса отправить задачу на тестирование указав текст функции на JS и набор тестов.

Тесты состоят из входных данных функции и ожидаемого результата.

Также пользователь может отправить запрос на получение результатов работы приложеня

Примеры запросов :

Запрос на запуск теста

POST

```
{
"packageId":"11",
"jsScript":"var divideFn = function(a,b) { return a/b}  ",
"functionName":"divideFn",
"tests": [
        {"testName":"test1",
         "expectedResult":"2.0",
         "params":[2,1]
        },
        {"testName":"test2",
         "expectedResult":"2.0",
         "params":[4,2]
        }
]
}
```

Запрос на получение результатов

GET http://localhost:8080/<url>?packageId=11

3. Подсказки.

а. Создаем actor system

б. В приложении будем использовать следующие акторы :

- актор который хранит результаты тестов.

Обрабатывает следующие сообщения :

сообщение с результатом одного теста -> кладет его в локальное хранилище.

Сообщение с запросом результата теста → отвечает сообщением с результатом всех тестов для заданного ***packageId***

- актор который исполняет один тест из пакета.

Для исполнения JS кода можно воспользоваться следующим примером

```
ScriptEngine engine = new
ScriptEngineManager().getEngineByName("nashorn");
engine.eval(jscript);
Invocable invocable = (Invocable) engine;
return invocable.invokeFunction(functionName, params).toString();
```

После исполнения теста результат передается актору хранилищу

- актор роутер

инициализирует актор хранилище а также пул акторов исполнителей тестов

в. После инициализации actor system — создаем актор роутер который в свою очередь создает все дочерние акторы

г. Создаем `ActorMaterializer` и инициализируем `http` систему с помощью `high level api`

д. Строим дерево `route` и пишем обработчики запросов.

е. Когда приходит запрос на запуск теста – запускаем тест и сразу овтечаем константным ответом.

ё. В случае запроса на получение информции о тесте – используем `Putterns.ask` и возвращаем `Future` с ответом