Yesterday's session : Junit, Mocking overview

What is Unit testing?

-> Process of testing unit amount of work is called as Unit testing

-> To improve code quality and to identify bugs in earlier stage we will perform unit testing.

What is Junit?

-> Junit is an open source framework which is used to perform unit testing for java applications.

What is Mocking?

The process of creating substitute/alternate object for real object is called as Mocking.

Unit Testing For REST API

```
--------------------WelcomeRestController.java---------------
package com.ashokit.rest;

import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class WelcomeRestController {

	@GetMapping("/welcome")
	public ResponseEntity<String> welcomeMsg() {
		String msg = "Welcome to Ashok IT..!!";
		return new ResponseEntity<String>(msg, HttpStatus.OK);
	}

}

--------------------BookRestControllerTest.java----------------------
package com.ashokit.rest;

import static org.junit.jupiter.api.Assertions.assertEquals;

import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.autoconfigure.web.servlet.WebMvcTest;
import org.springframework.test.web.servlet.MockMvc;
import org.springframework.test.web.servlet.MvcResult;
import org.springframework.test.web.servlet.request.MockHttpServletRequestBuilder;
import org.springframework.test.web.servlet.request.MockMvcRequestBuilders;

@WebMvcTest(value = WelcomeRestController.class)
public class WelcomeRestControllerTest {
```

```
	@Autowired
	private MockMvc mockMvc;

	@Test
	public void test_welcomeMsg() throws Exception {

		MockHttpServletRequestBuilder reqBuilder =
MockMvcRequestBuilders.get("/welcome");
		MvcResult mvcResult = mockMvc.perform(reqBuilder).andReturn();

		// String responseStr = mvcResult.getResponse().getContentAsString();

		int status = mvcResult.getResponse().getStatus();

		assertEquals(200, status);

	}
}
```

----------Book.java----------

```
package com.ashokit.bindings;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data
@NoArgsConstructor
@AllArgsConstructor
public class Book {

	private Integer bookId;
	private String bookName;
	private Double bookPrice;

}
```

----------BookRestController.java----------

```
package com.ashokit.rest;

import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;

import com.ashokit.bindings.Book;

@RestController
public class BookRestController {

	@PostMapping(value = "/addbook", consumes = { "application/json" })
	public ResponseEntity<String> addBook(@RequestBody Book book) {
		// logic to store into db

		String body = "Book Saved";

		return new ResponseEntity<>(body, HttpStatus.CREATED);
	}

}
```

----------BookRestControllerTest.java----------

```
package com.ashokit.rest;

import static org.junit.jupiter.api.Assertions.assertEquals;

import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.autoconfigure.web.servlet.WebMvcTest;
import org.springframework.test.web.servlet.MockMvc;
import org.springframework.test.web.servlet.MvcResult;
import org.springframework.test.web.servlet.request.MockHttpServletRequestBuilder;
import org.springframework.test.web.servlet.request.MockMvcRequestBuilders;

import com.ashokit.bindings.Book;
import com.fasterxml.jackson.databind.ObjectMapper;

@WebMvcTest(value = BookRestController.class)
public class BookRestControllerTest {

	@Autowired
	private MockMvc mockMvc;

	@Test
	public void test_addBook() throws Exception {

		Book b = new Book(101, "Spring", 450.00);

		ObjectMapper mapper = new ObjectMapper();
		String bookJson = mapper.writeValueAsString(b);

		MockHttpServletRequestBuilder reqBuilder =
MockMvcRequestBuilders.post("/addbook")

.contentType("application/json")

.content(bookJson);

		MvcResult mvcResult = mockMvc.perform(reqBuilder).andReturn();

		int status = mvcResult.getResponse().getStatus();

		assertEquals(201, status);

	}

}
-------------------------------------------------------
```