Verinite Technologies Pvt Ltd

Concept - Spring Constructs

1. Controllers

Definition: In Spring MVC, controllers handle HTTP requests and return responses. They act as the link between the view and the service layer.

2.DAOs (Data Access Objects)

Definition: DAOs provide an abstract interface to some type of database or other persistence mechanism. They encapsulate data access logic.

3. Repositories

Definition: In Spring Data JPA, repositories are a type of DAO that use JPA to manage database operations. They are often used to abstract and simplify data access.

Example:

```
@Repository

public interface UserRepository extends JpaRepository<User, Long> {

    // JpaRepository provides CRUD operations and more

}
```

4. Services

Definition: Services contain business logic. They interact with DAOs or repositories and perform operations that are crucial to the business.

5. Entities

Definition: Entities are classes that represent the data structure of a table in the database. They are managed by JPA.

Example:

```
@Entity

public class User {

    @Id

    @GeneratedValue(strategy = GenerationType.IDENTITY)

    private Long id;

  private String username;
```

```
    private String password;

  // getters and setters

}
```

6. Models

Definition: Models refer to the classes used to transfer data between layers of the application, often between the controller and view.

Example:

```
public class UserModel {

    private Long id;

    private String username;

    // getters and setters

}
```

7. Dependency Injection

Definition: A design pattern used to implement Inversion of Control (IoC), where an object's dependencies are injected by the Spring container rather than being created by the object itself.

Example:

```
@Component

public class MyService {

    private final Dependency dependency;


    @Autowired

    public MyService(Dependency dependency) {

        this.dependency = dependency;

    }

}
```

8. Spring Data JPA

Definition: A Spring project that provides a high-level abstraction for working with JPA and databases, offering a repository pattern to simplify data access.

Example:

```
@Entity
public class Product {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
     private String name;
     // getters and setters
}
@Repository
public interface ProductRepository extends JpaRepository<Product, Long> {
}
```

9. Beans vs POJO

Definition:

Beans: Objects managed by the Spring container. They are created, initialized, and managed by the container.

POJO (Plain Old Java Object): Simple Java objects without special restrictions or requirements.

Example:

```
@Component
public class MyBean {
    // Managed by Spring container
}
public class MyPojo {
    // Regular Java class, not managed by Spring
}
```

10. Bean Scopes

Definition: Bean scopes define the lifecycle and visibility of Spring beans. Common scopes are:

Singleton: One instance per Spring container (default).

Prototype: A new instance each time requested.

Request: One instance per HTTP request.

Session: One instance per HTTP session.

11. Log Levels

Definition: Levels used to control the granularity of logging output. Common levels are:

ERROR: For serious issues.

WARN: For potentially problematic situations.

INFO: For general informational messages.

DEBUG: For detailed diagnostic information.

TRACE: For highly detailed tracing information.

Example:

```
private static final Logger logger = LoggerFactory.getLogger(MyClass.class);


logger.info("This is an info message.");
logger.debug("This is a debug message.");
```

12. Exception Handling, Controller Advice

Definition:

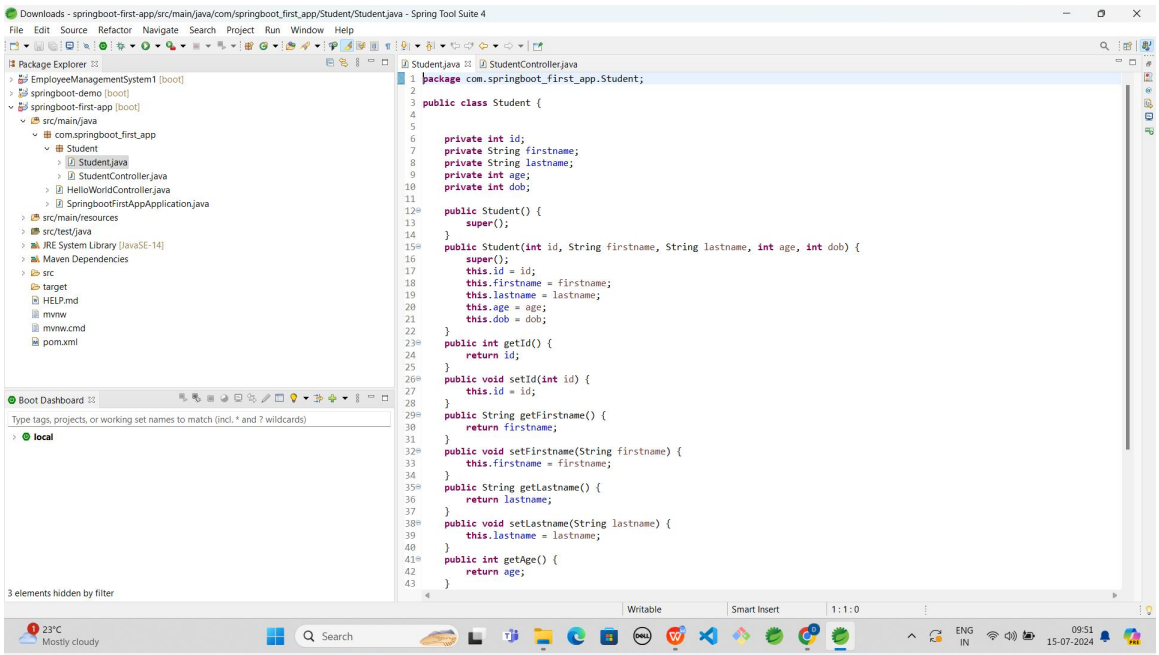
Exception Handling: Mechanism to manage and respond to errors in a controlled manner.

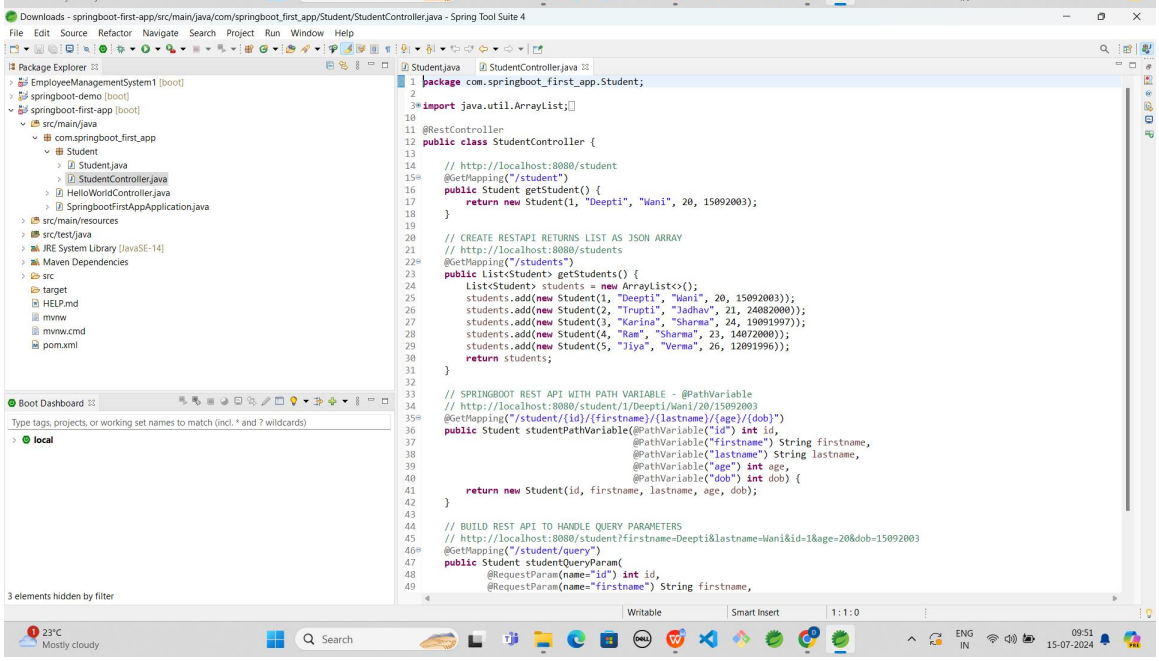
Controller Advice: A way to handle exceptions globally across multiple controllers using @ControllerAdvice.

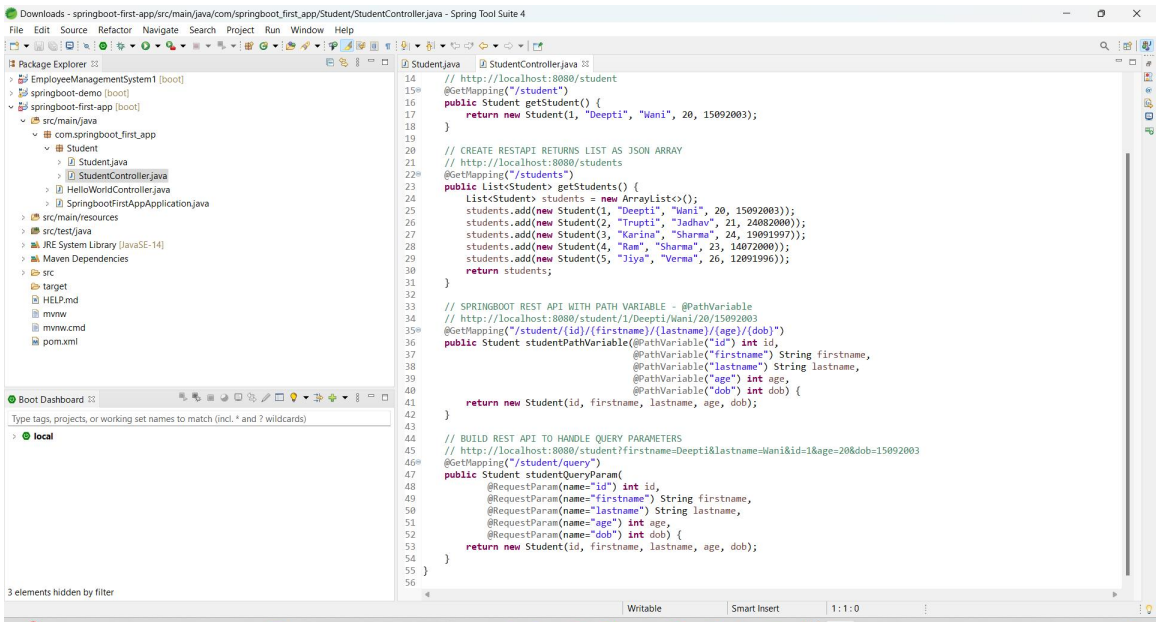
Example:

```
@ControllerAdvice
public class GlobalExceptionHandler {
    @ExceptionHandler(Exception.class)
    public ResponseEntity<String> handleException(Exception ex) {
        return new ResponseEntity<>("Error: " + ex.getMessage(),
HttpStatus.INTERNAL_SERVER_ERROR);
```

```
    }
}
```


Downloads - springboot-first-app/src/main/java/com/springboot_first_app/Student/Student.java - Spring Tool Suite 4
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
EmployeeManagementSystem1 [boot]
springboot-demo [boot]
springboot-first-app [boot]
src/main/java
com.springboot_first_app
Student
Student.java
StudentController.java
HelloWorldController.java
SpringbootFirstAppApplication.java
src/main/resources
src/test/java
JRE System Library [JavaSE-14]
Maven Dependencies
src
target
HELP.md
mvnw
mvnw.cmd
pom.xml
Boot Dashboard
Type tags, projects, or working set names to match (incl. * and ? wildcards)
local
3 elements hidden by filter
Student.java
StudentController.java
package com.springboot_first_app.Student;
public class Student {
private int id;
private String firstname;
private String lastname;
private int age;
private int dob;
public Student() {
super();
}
public Student(int id, String firstname, String lastname, int age, int dob) {
super();
this.id = id;
this.firstname = firstname;
this.lastname = lastname;
this.age = age;
this.dob = dob;
}
public int getId() {
return id;
}
public void setId(int id) {
this.id = id;
}
public String getFirstname() {
return firstname;
}
public void setFirstname(String firstname) {
this.firstname = firstname;
}
public String getLastname() {
return lastname;
}
public void setLastname(String lastname) {
this.lastname = lastname;
}
public int getAge() {
return age;
}
Writable
Smart Insert
1 : 1 : 0
23°C
Mostly cloudy
Search
ENG
IN
09:51
15-07-2024


Downloads - springboot-first-app/src/main/java/com/springboot_first_app/Student/StudentController.java - Spring Tool Suite 4
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
EmployeeManagementSystem1 [boot]
springboot-demo [boot]
springboot-first-app [boot]
src/main/java
com.springboot_first_app
Student
Student.java
StudentController.java
HelloWorldController.java
SpringbootFirstAppApplication.java
src/main/resources
src/test/java
JRE System Library [JavaSE-14]
Maven Dependencies
src
target
HELP.md
mvnw
mvnw.cmd
pom.xml
Boot Dashboard
Type tags, projects, or working set names to match (incl. * and ? wildcards)
local
3 elements hidden by filter
Student.java
StudentController.java
package com.springboot_first_app.Student;
import java.util.ArrayList;
@RestController
public class StudentController {
// http://localhost:8080/student
@GetMapping("/student")
public Student getStudent() {
return new Student(1, "Deepti", "Wani", 20, 15092003);
}
// CREATE RESTAPI RETURNS LIST AS JSON ARRAY
// http://localhost:8080/students
@GetMapping("/students")
public List<Student> getStudents() {
List<Student> students = new ArrayList<>();
students.add(new Student(1, "Deepti", "Wani", 20, 15092003));
students.add(new Student(2, "Trupti", "Jadhav", 21, 24082000));
students.add(new Student(3, "Karina", "Sharma", 24, 19091997));
students.add(new Student(4, "Ram", "Sharma", 23, 14072000));
students.add(new Student(5, "Jiya", "Verma", 26, 12091996));
return students;
}
// SPRINGBOOT REST API WITH PATH VARIABLE - @PathVariable
// http://localhost:8080/student/1/Deepti/Wani/20/15092003
@GetMapping("/student/{id}/{firstname}/{lastname}/{age}/{dob}")
public Student studentPathVariable(@PathVariable("id") int id,
@PathVariable("firstname") String firstname,
@PathVariable("lastname") String lastname,
@PathVariable("age") int age,
@PathVariable("dob") int dob) {
return new Student(id, firstname, lastname, age, dob);
}
// BUILD REST API TO HANDLE QUERY PARAMETERS
// http://localhost:8080/student?firstname=Deepti&lastname=Wani&id=1&age=20&dob=15092003
@GetMapping("/student/query")
public Student studentQueryParam(
@RequestParam(name="id") int id,
@RequestParam(name="firstname") String firstname,
Writable
Smart Insert
1 : 1 : 0
23°C
Mostly cloudy
Search
ENG
IN
09:51
15-07-2024


Downloads - springboot-first-app/src/main/java/com/springboot_first_app/Student/StudentController.java - Spring Tool Suite 4
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
EmployeeManagementSystem1 [boot]
springboot-demo [boot]
springboot-first-app [boot]
src/main/java
com.springboot_first_app
Student
Student.java
StudentController.java
HelloWorldController.java
SpringbootFirstAppApplication.java
src/main/resources
src/test/java
JRE System Library [JavaSE-14]
Maven Dependencies
src
target
HELP.md
mvnw
mvnw.cmd
pom.xml
Boot Dashboard
Type tags, projects, or working set names to match (incl. * and ? wildcards)
local
3 elements hidden by filter
Student.java
StudentController.java
// http://localhost:8080/student
@GetMapping("/student")
public Student getStudent() {
return new Student(1, "Deepti", "Wani", 20, 15092003);
}
// CREATE RESTAPI RETURNS LIST AS JSON ARRAY
// http://localhost:8080/students
@GetMapping("/students")
public List<Student> getStudents() {
List<Student> students = new ArrayList<>();
students.add(new Student(1, "Deepti", "Wani", 20, 15092003));
students.add(new Student(2, "Trupti", "Jadhav", 21, 24082000));
students.add(new Student(3, "Karina", "Sharma", 24, 19091997));
students.add(new Student(4, "Ram", "Sharma", 23, 14072000));
students.add(new Student(5, "Jiya", "Verma", 26, 12091996));
return students;
}
// SPRINGBOOT REST API WITH PATH VARIABLE - @PathVariable
// http://localhost:8080/student/1/Deepti/Wani/20/15092003
@GetMapping("/student/{id}/{firstname}/{lastname}/{age}/{dob}")
public Student studentPathVariable(@PathVariable("id") int id,
@PathVariable("firstname") String firstname,
@PathVariable("lastname") String lastname,
@PathVariable("age") int age,
@PathVariable("dob") int dob) {
return new Student(id, firstname, lastname, age, dob);
}
// BUILD REST API TO HANDLE QUERY PARAMETERS
// http://localhost:8080/student?firstname=Deepti&lastname=Wani&id=1&age=20&dob=15092003
@GetMapping("/student/query")
public Student studentQueryParam(
@RequestParam(name="id") int id,
@RequestParam(name="firstname") String firstname,
@RequestParam(name="lastname") String lastname,
@RequestParam(name="age") int age,
@RequestParam(name="dob") int dob) {
return new Student(id, firstname, lastname, age, dob);
}
}
Writable
Smart Insert
1 : 1 : 0
23°C
Mostly cloudy
Search
ENG
IN
09:51
15-07-2024
localhost:8080/student
Pretty-print
{
"id": 1,
"firstname": "Deepti",
"lastname": "Wani",
"age": 20,
"dob": 15092003
}


localhost:8080/students
Pretty-print
[
{
"id": 1,
"firstname": "Deepti",
"lastname": "Wani",
"age": 20,
"dob": 15092003
},
{
"id": 2,
"firstname": "Trupti",
"lastname": "Jadhav",
"age": 21,
"dob": 24082000
},
{
"id": 3,
"firstname": "Karina",
"lastname": "Sharma",
"age": 24,
"dob": 19091997
},
{
"id": 4,
"firstname": "Ram",
"lastname": "Sharma",
"age": 23,
"dob": 14072000
},
{
"id": 5,
"firstname": "Jiya",
"lastname": "Verma",
"age": 26,
"dob": 12091996
}
]


localhost:8080/student/1/Deepti/Wani/20/15092003
Pretty-print
{
"id": 1,
"firstname": "Deepti",
"lastname": "Wani",
"age": 20,
"dob": 15092003
}


localhost:8080/student?firstname=Deepti&lastname=Wani&id=1&age=20&dob=15092003
Pretty-print
{
"id": 1,
"firstname": "Deepti",
"lastname": "Wani",
"age": 20,
"dob": 15092003
}