

 무료 전자 책

배우기

spring-data

Free unaffiliated eBook created from
Stack Overflow contributors.

#spring-
data

.....	1
1:	2
.....	2
Examples.....	2
.....	2
2:	3
.....	3
Examples.....	3
Spring JPA	3
.....	6

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [spring-data](#)

It is an unofficial and free spring-data ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official spring-data.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1:

. NoSQL Cloud . , . API API .

Spring Data API . Spring , .

Spring Data . Spring Data .

Examples

Spring Data . [Spring Data JPA](#) , [Spring Data MongoDB](#) , [Spring Data Elasticsearch](#) , [Spring Data Neo4J](#) , [Spring Data Cassandra](#) [Spring Data Redis](#) .

Spring . . Spring .

```
<dependencies>
  <dependency>
    <groupId>org.springframework.data</groupId>
    <artifactId>spring-data-commons</artifactId>
    <version>[version-number]</version>
  </dependency>
</dependencies>
```

Gradle

```
dependencies {
  compile 'org.springframework.data:spring-data-commons:[version-number]'
}
```

[] .

: <https://riptutorial.com/ko/spring-data/topic/5440/-->

2:

Spring JPA

Examples

Spring JPA

Spring Boot 1.4.4.RELEASE MySQL Spring Data JPA MySQL ., JPA .

// {id} . classroomId (1) .

```
@Entity
@Table(name = "student")
public class Student {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(name = "name")
    private String name;

    @NotNull
    @Column(name = "rollnumber", nullable = false)
    private Integer rollnumber;

    @Column(name = "date_of_birth")
    private LocalDate dateOfBirth;

    @Column(name = "address")
    private String address;

    @ManyToOne(optional = false)
    @NotNull
    private Classroom classroom;

    //getter and setter

}
```

```
@Entity
@Table(name = "classroom")
public class Classroom {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(name = "standard")
    private String standard;
```

```

    @Column(name = "section")
    private String section;

    @Column(name = "year")
    private String year;

    //getter && setter

}

```

RestController .

```

@RestController
@RequestMapping("/api")
public class StudentResource {

    private final StudentService studentService;

    public StudentResource(StudentService studentService) {
        this.studentService = studentService;
    }

    @GetMapping("/students/classroom/{id}")
    public ResponseEntity<Page<StudentDTO>> getAllStudentsBasedOnClassroom(@ApiParam Pageable
pageable,@PathVariable Long id)
        throws URISyntaxException {
        Page<StudentDTO> page = studentService.findByClassroomId(id, pageable);
        HttpHeaders headers = PaginationUtil.generatePaginationHttpHeaders (page,
"/api/students/classroom");
        return new ResponseEntity<Page<StudentDTO>>(page, headers, HttpStatus.OK);
    }

}

```

RequestParams . endpoint / students / classroom / 1? page = 0 & size = 3 Spring
Pageable . Pageable . .

```

public interface StudentService {

    Page<StudentDTO> findByClassroomId(Long id,Pageable pageable);

}

```

impl (StudentMapper DTOby mapStruct)

```

@Service
@Transactional
public class StudentServiceImpl implements StudentService{

    private final StudentRepository studentRepository;

    private final StudentMapper studentMapper;

    public StudentServiceImpl(StudentRepository studentRepository, StudentMapper
studentMapper) {
        this.studentRepository = studentRepository;
        this.studentMapper = studentMapper;
    }
}

```

```

    }
    @Override
    public Page<StudentDTO> findByClassroomId(Long id, Pageable pageable) {
        log.debug("Request to get Students based on classroom : {}", id);
        Page<Student> result = studentRepository.findByClassroomId(id, pageable);
        return result.map(student -> studentMapper.studentToStudentDTO(student));
    }
}

```

```

@Mapper(componentModel = "spring", uses = {})
public interface StudentMapper{

    StudentDTO    studentToStudentDTO(Student student);
}

```

StudentRepository

```

public interface StudentRepository extends JpaRepository<Student,Long> {

    Page<Student> findByClassroomId(Long id, Pageable pageable);

}

```

```

"last": false,
"totalElements": 20,
"totalPages": 7,
"size": 3,
"number": 0,
"sort": null,
"first": true,
"numberOfElements": 3

```

: <https://riptutorial.com/ko/spring-data/topic/9142/--->

S. No	Contributors
1	Community , manish , sunku02
2	Aman Tuladhar , VISHWANATH N P