

Tutorial 7 Writeup

A. Rangkuman

Pada tutorial minggu ini, saya mempelajari implementasi web service. Web service merupakan aplikasi yang diakses oleh program lain. Penggunaan web service ini dilakukan melalui interaksi service producer dan service consumer. User akan melakukan interaksi dengan service consumer, kemudian data yang diminta akan disediakan oleh service producer melalui web service. Service producer memiliki REST controller yang mengembalikan object JSON, bukan String seperti controller pada umumnya.

B. Tutorial

Membuat class baru yaitu StudentRestController.java pada package com.example.rest

```
@RestController
@RequestMapping("/rest")
public class StudentRestController {
    @Autowired
    StudentService studentService;

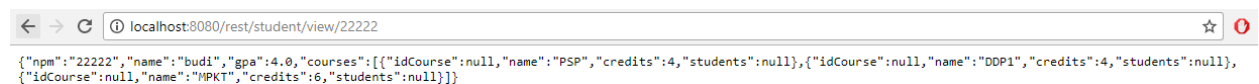
    @RequestMapping("/student/view/{npm}")
    public StudentModel view(@PathVariable(value = "npm") String npm) {
        StudentModel student = studentService.selectStudent(npm);
        return student;
    }

    @RequestMapping("/student/viewall")
    public List<StudentModel> viewall(){
        List<StudentModel> students = studentService.selectAllStudents();
        return students;
    }
}
```

Untuk mencobanya langsung, silakan jalankan program Anda dan buka halaman

"localhost:8080/rest/student/view/123" atau sesuai dengan NPM yang ada di data Anda.

Hasil tampilannya adalah sebagai berikut:



```
{
  "npm": "22222",
  "name": "budi",
  "gpa": 4.0,
  "courses": [
    {
      "idCourse": null,
      "name": "PSP",
      "credits": 4,
      "students": null
    },
    {
      "idCourse": null,
      "name": "DDP1",
      "credits": 4,
      "students": null
    },
    {
      "idCourse": null,
      "name": "MPKT",
      "credits": 6,
      "students": null
    }
  ]
}
```

Setelah JSON diparse

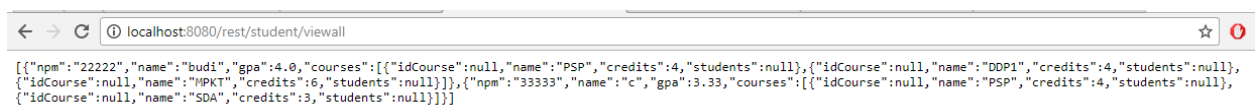
String parse

```
{
  "npm": "22222",
  "name": "budi",
  "gpa": 4.0,
  "courses": [
    {
      "idCourse": null,
      "name": "PSP",
      "credits": 4,
      "students": null
    },
    {
      "idCourse": null,
      "name": "DDP1",
      "credits": 4,
      "students": null
    },
    {
      "idCourse": null,
      "name": "MPKT",
      "credits": 6,
      "students": null
    }
  ]
}
```

Latihan 1: Method viewall

```
@RequestMapping("/student/viewall")
public List<StudentModel> viewall(){
    List<StudentModel> students = studentService.selectAllStudents();
    return students;
}
```

Method viewall pada rest controller akan membuat object students berupa List of StudentModel yang diisi oleh kumpulan student pada database. Kemudian rest controller akan mengembalikan objek students berupa JSON ke web browser.



```
{
  "npm": "22222", "name": "budi", "gpa": 4.0, "courses": [
    { "idCourse": null, "name": "PSP", "credits": 4, "students": null },
    { "idCourse": null, "name": "DDP1", "credits": 4, "students": null },
    { "idCourse": null, "name": "MPKT", "credits": 6, "students": null }
  ]
},
{
  "npm": "33333", "name": "c", "gpa": 3.33, "courses": [
    { "idCourse": null, "name": "PSP", "credits": 4, "students": null },
    { "idCourse": null, "name": "SDA", "credits": 3, "students": null }
  ]
}
```

```
{
  {
    "npm": "22222",
    "name": "budi",
    "gpa": 4.0,
    "courses": [
      {
        "idCourse": null,
        "name": "PSP",
        "credits": 4,
        "students": null
      },
      {
        "idCourse": null,
        "name": "DDP1",
        "credits": 4,
        "students": null
      },
      {
        "idCourse": null,
        "name": "MPKT",
        "credits": 6,
        "students": null
      }
    ]
  },
  {
    "npm": "33333",
    "name": "c",
    "gpa": 3.33,
    "courses": [
      {
        "idCourse": null,
        "name": "PSP",
        "credits": 4,
        "students": null
      },
      {
        "idCourse": null,
        "name": "SDA",
        "credits": 3,
        "students": null
      }
    ]
  }
]
```

Latihan 2: Membuat service dan controller baru untuk kelas Course

```
@RequestMapping("/course/view/{id}")
public CourseModel view(@PathVariable(value="id") String id) {
    CourseModel course = courseService.selectCourse(id);
    return course;
}

@RequestMapping("/course/viewall")
public List<CourseModel> viewall(){
    List<CourseModel> courses = courseService.selectAllCourses();
    return courses;
}
```

Mirip seperti pada latihan 1, pertama-tama saya membuat kelas dan interface baru untuk CourseMapper, CourseService dan CourseServiceDatabase. Setelah itu, saya melengkapi RequestMapping untuk method view course by ID dan viewall pada RestCourseController.

View course by ID:

```
localhost:8080/rest/course/view/csc123
{"idCourse": "CSC123", "name": "PSP", "credits": 4, "students": [{"npm": "22222", "name": "budi", "gpa": 4.0, "courses": null}, {"npm": "33333", "name": "c", "gpa": 3.33, "courses": null}]}
```

```
{
  "idCourse": "CSC123",
  "name": "PSP",
  "credits": 4,
  "students": [
    {
      "npm": "22222",
      "name": "budi",
      "gpa": 0.0,
      "courses": null
    },
    {
      "npm": "33333",
      "name": "c",
      "gpa": 0.0,
      "courses": null
    }
  ]
}
```

View all courses:

```
← → ↻ localhost8080/rest/course/viewall ☆ 🔍
[{"idCourse": "CSC123", "name": "PSP", "credits": 4, "students": [{"npm": "22222", "name": "budi", "gpa": 0.0, "courses": null}, {"npm": "33333", "name": "c", "gpa": 0.0, "courses": null}], {"idCourse": "CSC124", "name": "SDA", "credits": 3, "students": [{"npm": "33333", "name": "c", "gpa": 0.0, "courses": null}], {"idCourse": "CSC125", "name": "DDP1", "credits": 4, "students": [{"npm": "22222", "name": "budi", "gpa": 0.0, "courses": null}], {"idCourse": "CSC126", "name": "MPKT", "credits": 6, "students": [{"npm": "22222", "name": "budi", "gpa": 0.0, "courses": null}]}
```

```
{
  "idCourse": "CSC123",
  "name": "PSP",
  "credits": 4,
  "students": [
    {
      "npm": "22222",
      "name": "budi",
      "gpa": 0.0,
      "courses": null
    },
    {
      "npm": "33333",
      "name": "c",
      "gpa": 0.0,
      "courses": null
    }
  ],
  "idCourse": "CSC124",
  "name": "SDA",
  "credits": 3,
  "students": [
    {
      "npm": "33333",
      "name": "c",
      "gpa": 0.0,
      "courses": null
    }
  ],
  "idCourse": "CSC125",
  "name": "DDP1",
  "credits": 4,
  "students": [
    {
      "npm": "22222",
      "name": "budi",
      "gpa": 0.0,
      "courses": null
    }
  ],
  "idCourse": "CSC126",
  "name": "MPKT",
  "credits": 6,
  "students": [
    {
      "npm": "22222",
      "name": "budi",
      "gpa": 0.0,
      "courses": null
    }
  ]
}
```

Membuat service consumer

Membuat interface StudentDAO

```
package com.example.dao;

import java.util.List;

public interface StudentDAO {
    StudentModel selectStudent (String npm);
    List <StudentModel> selectAllStudents ();
}
```

Implementasikan interface tersebut dalam kelas StudentDAOImpl

```
import java.util.List;

@Service
public class StudentDAOImpl implements StudentDAO {
    @Autowired
    private RestTemplate restTemplate;

    @Override
    public StudentModel selectStudent(String npm) {
        StudentModel student = restTemplate.getForObject("http://localhost:8080/rest/student/view/" + npm,
            StudentModel.class);
        return student;
    }

    @Override
    public List<StudentModel> selectAllStudents() {
        List<StudentModel> students = restTemplate.getForObject("http://localhost:8080/rest/student/viewall", List.class);

        return students;
    }
}
```

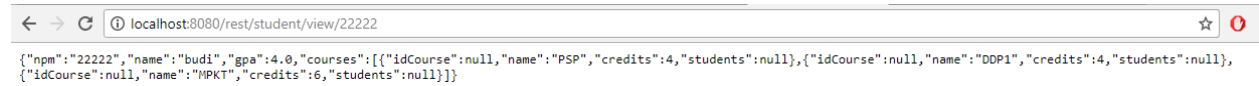
Lalu, buat kelas StudentServiceRest untuk mengambil data dari web service

```
@Slf4j
@Service
@Primary
public class StudentServiceRest implements StudentService {
    @Autowired
    private StudentDAO studentDAO;

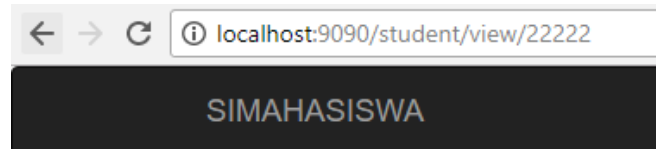
    @Override
    public StudentModel selectStudent(String npm) {
        log.info("REST - select student with npm {}", npm);
        return studentDAO.selectStudent(npm);
    }

    @Override
    public List<StudentModel> selectAllStudents() {
        log.info("REST - select all students");
        return studentDAO.selectAllStudents();
    }
}
```

Pastikan service producer sudah berjalan dengan menjalankan
localhost:8080/rest/student/view/22222



Untuk menguji service consumer buka localhost:9090/student/view/22222



NPM = 22222

Name = budi

GPA = 4.0

Kuliah yang diambil

- PSP-4 sks
- DDP1-4 sks
- MPKT-6 sks

Tampilan pada console

```
2017-11-04 00:29:01.022 INFO 1184 --- [nio-9090-exec-1] com.example.service.StudentServiceRest : REST - select student with npm 22222
```

Latihan 3: Implementasi service consumer untuk view all Students

Untuk mengimplementasikan method viewall Students, saya membuat method pada service StudentDAOImpl untuk consume object REST web service. Method getForObject digunakan dengan menerima URL producer web service <http://localhost:8080/rest/student/viewall> dan tipe class List.

```
@Override
public List<StudentModel> selectAllStudents() {
    List<StudentModel> students = restTemplate.getForObject("http://localhost:8080/rest/student/viewall", List.class);

    return students;
}
```

Kemudian saya melengkapi implementasi `selectAllStudents()` pada kelas `StudentServiceRest`.

```
@Override
public List<StudentModel> selectAllStudents() {
    log.info("REST - select all students");
    return studentDAO.selectAllStudents();
}
```

Tampilan viewall students:

Showing 1 to 2 of 2 entries

Latihan 4: Implementasi service consumer untuk kelas `CourseModel`

Pertama-tama, saya membuat interface `CourseDAO`

```
public interface CourseDAO {
    CourseModel selectCourse (String idCourse);
    List<CourseModel> selectAllCourses ();
}
```

Setelah itu, saya mengimplementasikan interface tersebut pada kelas `CourseDAOImpl` dan membuat method yang sesuai

```
@Override
public CourseModel selectCourse(String idCourse) {
    CourseModel course = restTemplate.getForObject("http://localhost:8080/rest/course/view/" + idCourse,
        CourseModel.class);
    return course;
}

@Override
public List<CourseModel> selectAllCourses() {
    List<CourseModel> courses = restTemplate.getForObject("http://localhost:8080/rest/course/viewall", List.class);
    return courses;
}
```

Setelah itu, saya membuat interface `CourseService`

```
public interface CourseService {
    CourseModel selectCourse(String idCourse);
    List<CourseModel> selectAllCourses();
}
```

Untuk mengambil data dari web service, maka perlu dibuat kelas CourseServiceRest yang mengimplementasi CourseService.

```
@Slf4j
@Service
@Primary
public class CourseServiceRest implements CourseService{
    @Autowired
    private CourseDAO courseDAO;

    @Override
    public CourseModel selectCourse(String idCourse) {
        log.info("REST - select course with id {}", idCourse);
        return courseDAO.selectCourse(idCourse);
    }

    @Override
    public List<CourseModel> selectAllCourses(){
        log.info("REST - select all courses");
        return courseDAO.selectAllCourses();
    }
}
```

Terakhir, saya membuat CourseController untuk mengatur RequestMapping path yang sesuai

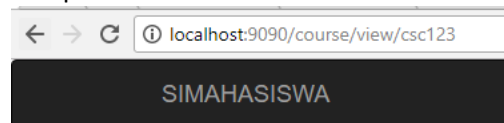
```
@RequestMapping(value="/course/view/{idCourse}")
public String view(Model model, @PathVariable(value="idCourse") String idCourse) {
    CourseModel course = courseDAO.selectCourse(idCourse);
    model.addAttribute("course", course);
    model.addAttribute("page_title", "View Course Detail");

    return "course";
}

@RequestMapping("/course/viewall")
public String viewAll(Model model) {
    List<CourseModel> courses = courseDAO.selectAllCourses();
    model.addAttribute("courses", courses);
    model.addAttribute("page_title", "View All Courses");

    return "viewall-courses";
}
```

Tampilan view course



ID = CSC123

Nama = PSP

SKS = 4

Mahasiswa yang mengambil

- 22222 - budi
- 33333 - c

Tampilan view all courses

← → ↻

localhost:9090/course/viewall

☆

🔍

🌐

SIMAHASISWA

HomeDaftar MahasiswaTambah Mahasiswa

All Courses

Show 10 entries

Search:

No	Course ID	Name	Credits
1	CSC123	PSP	4
2	CSC124	SDA	3
3	CSC125	DDP1	4
4	CSC126	MPKT	6

Showing 1 to 4 of 4 entries

Previous1Next