Date : 19/03/2021
Spring Boot 9AM
Mr. RAGHU

SpringBoot ReactJS:-
https://www.youtube.com/c/NareshIT/search?query=reactjs%20raghu

Parameters to RestController#methods

a. Request Parameter : URL?key=val&key=val&key=val....

=> Data is sent to App using URL in key=val format
=> We can pass multiple key=val, need not to follow order
=> To Read Data:
   @RequestParam("key")DataType localVariable
       (or)
   @RequestParam DataType key
=> Supports even DataType conversion.
=> Still, so many web apps are using RequestParam,
   even we can use same in webservices also.

```
https://www.google.com/search
? q=india
& oq=india
& aqs=chrome..69i57j46i275i433j0i131i433l3j0j69i60j69i65.2086j0j7
& sourceid=chrome
& ie=UTF-8
```

=> It internally uses servlets concept,

```
@RequestParam("sid")Integer id
--Servlets equals code---
String sid = request.getParameter("sid");
int id = Integer.parseInt(sid);
```

---

*)RestController:

```
package in.nareshit.raghu.rest;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class ProductRestController {

	@GetMapping("/data")
	public String showData(
			@RequestParam Integer pid,
			@RequestParam String pcode,
			@RequestParam Double pcost
			)
	{
		return "HELLO=>"+pid+","+pcode+","+pcost;
	}
}
```

```
URL:
http://localhost:8080/data?pid=10&pcode=PEN&pcost=300
http://localhost:8080/data?pcode=PEN&pcost=400&pid=101
-----------------------------------------------------------------
b. ** PathVariable

static path : It indicates location(Path) given to
resource(Controller#method)
                ex: /product/export, /employee/save ..etc
dynamic path: It indicates sending data along with URL as PathType
                without any key(DirectValue)
                ex: /product/{id}  , /employee/find/{code}
                Here /{ }  indicates dynamic path(ie send data at
runtime)


=> PathVariables are called as clean URL
  (No additional Symbols ?, & as they are overloaded)
=> Must follows order. It avoid confusion for other devs/users.
=> Execution is bit faster compared to RequestParam.
=> URL size also reduced

With Request Param
http://localhost:8080/data?pid=10&pcode=PEN&pcost=300
With Path Variable
http://localhost:8080/data/10/PEN/300

=> To read data in App, syntax is
   @PathVariable("key")DataType localVar
                (or)
   @PathVariable DataType key

=> Supports even type conversion, (String->int) based on DataType
   that you have provided.
-----------------------------------------------------------------
*)Note:
=> While adding Path in code, we must specify static and dynamic path
   details using @___Mapping("_") annotation

3 Steps,
S#1 Define Dynamic Path at Method Level
ex: @GetMapping("/employee/find/{eid}")

S#2 Read Path Variable data at Method Param
Ex:   @PathVariable Integer eid

S#3 Pass data using Request URL
Ex: http://localhost:8080/employee/find/10


*) While Making request we must maintain
  no.of level count in code is equals
  to URL Path Levels.
Ex:
  code: @GetMapping("/emp/find/data/{code}")
  ReqURL:
    http://localhost:8080/emp/find/data/A    (200-OK)
```

```
    http://localhost:8080/emp/find/data/A/B  (404-Not Found)
    http://localhost:8080/emp/find/data      (404-Not Found)

*) If we compare above cases with request param
    http://localhost:8080/emp/find/data?code=A        (200-OK)
    http://localhost:8080/emp/find/data?code=A&dec=B  (Additional
Param)
    http://localhost:8080/emp/find/data               (400-Bad
Request)


Q) How can we make PathVariable from required to optional param?
A)
  By default PathVariable is required(ie required=true)
  In required=true case if we did not send data using Path
  FC returns 404-Not Found.

  After making it optional as
    @PathVariable(required = false) DataType localVarible
  then holds default value as null in this case for same URL.

** There is a bug in Spring Boot , which is not working as
   expected for required=false for PathVariable.

-------CaseStudy------------------------
Case#1
  @GetMapping("/emp/find/{a}")--m1()
  @GetMapping("/emp/find/{b}")--m2()

URL:
  http://localhost:8080/emp/find/ABC
Output: 500 - Internal Server Error
 IllegalStateException: Ambiguous handler methods

*) Just bcoz of name/dataptye change in dynamic path ,
  they are never going to be different URLs.

-Example RestController---
package in.nareshit.raghu.rest;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class ProductRestController {

        @GetMapping("/format/{pcode}")
        public String findDataA(
                        @PathVariable String pcode
                        )
        {
                return "PATH DATA-A=>"+pcode;
        }

        @GetMapping("/format/{pcost}")
```

```
        public String findDataB(
                        @PathVariable Double pcost
                        )
        {
                return "PATH DATA-B=>"+pcost;
        }


}
============================================================
*) If a Request URL is matched with multiple methods
   which are defined using PathVariables,
   then first priority is given to more static count levels.
   (static means letter to letter should be compared including case
     --full path must be same)
   If not then come to next level dynamic.

=> We can have even path with all dynamics(valid)
=>Hint: Dynamic means anything can store.

Ex
m1() -- /abc/xyz/mno
m2() -- /abc/xyz/{mno}
m3() -- /abc/{xyz}/{mno}
m4() -- /{abc}/{xyz}/{mno}

Request URLs
d. /80/90/100
Matching methods: m4()
Selected Method : m4()

c. /abc/xzy/mno
Matching methods: m3(), m4()
Selected Method : m3

a. /abc/xyz/500

Matching methods: m2(), m3(), m4()
Selected Method : m2()

b. /abc/XYZ/Mno

Matching methods: m3(), m4()  [conside case-sensitivity also]
Selected Method : m3()

-------RestController code--------------
package in.nareshit.raghu.rest;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class ProductRestController {

        //http://localhost:8080/Emp/Mode/code --which one is matched?
```

```
	@GetMapping("/emp/mode/code")
	public String showA() {
		return "FROM#A";
	}

	@GetMapping("/emp/mode/{code}")
	public String showB(
			@PathVariable String code
			)
	{
		return "FROM#B "+ code;
	}

	@GetMapping("/emp/{mode}/{code}")
	public String showC(
			@PathVariable String mode,
			@PathVariable String code
			)
	{
		return "FROM#C "+ mode +"," +code;
	}

	@GetMapping("/{emp}/{mode}/{code}")
	public String showD(
			@PathVariable String emp,
			@PathVariable String mode,
			@PathVariable String code
			)
	{
		return "FROM#D " +emp +", "+ mode +"," +code;
	}

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