Gestione dei controller – Parte II

Cosa imparerai

- Come definire ed utilizzare i controller in una web app Spring MVC





Gestione dei controller

Creiamo un controller Spring MVC

I passi per creare un controller:

1. Creare una classe ed annotarla con *@Controller*
2. Creare almeno un metodo nella classe ed annotarlo con *@RequestMapping*, specificando il path (o i path) a cui dovrà rispondere. Nel caso più semplice il metodo ritorna una stringa.

*In questo caso, grazie all'annotation **@ResponseBody**, la stringa di ritorno costituirà l'output inserito nella response*

```
@Controller
public class HelloController {
    @ResponseBody
    @RequestMapping("/")
    public String showForm() {
        return "hello";
    }
}
```

```
@Controller
public class HelloController {
    @RequestMapping("/")
    public String showForm() {
        return "hello";
    }
}
```

In questo caso la stringa di ritorno costituirà il nome della view che conterrà l'HTML da inserire come output della response

3. Creare una classe ed annotarla con *@Configuration* e *@ComponentScan*, specificando il package dove si trova il controller.

```
@EnableWebMvc
@Configuration
@ComponentScan(basePackages = { "it.test.web.controller" })
public class WebConfig  {
```

Gestione dei controller

Mapping delle URL

Per mappare uno o più path di una URL ad un controller è possibile utilizzare l'annotation **@RequestMapping**.

L'annotation può essere utilizzata:

- **A livello di classe**: utilizzata se vogliamo mappare dei path condiviso tra più metodi
- **A livello di metodo**: utilizzata per mappare dei path su un metodo

```
@RestController
@RequestMapping(path = "/rest")
public class HelloRestController {
    @RequestMapping(path = "/1")
    public String showForm() {
        return "CIAO!!!!";
    }

    @RequestMapping(path = "/2")
    public String showForm2() {
        return "CIAO 2!!!!";
    }
}
```

http://mysite.com/rest/1

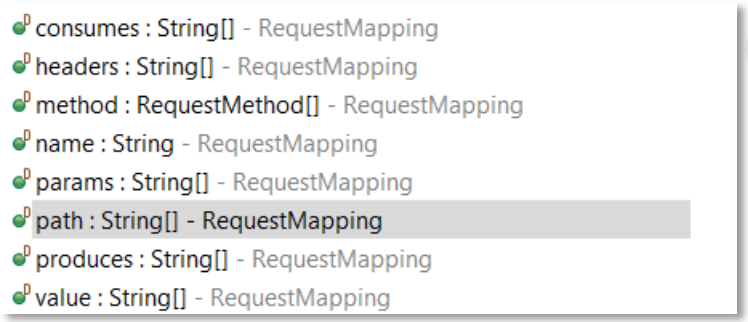
http://mysite.com/rest/2

Gestione dei controller

Mapping delle URL

L'annotation ha vari attributi che consentono di specificare:

- **path**: i path a cui sarà associato il metodo o la classe
- **method**: il metodo HTTP (GET, POST, PUT,…)
- **params**: i parametri di richiesta
- **headers**: le intestazioni della richiesta
- **consumes** e **produces**: i tipi di media



Gestione dei controller

Mapping delle URL – l'attributo «path»

Consente di specificare il path (o i path) a cui sarà associato il metodo o la classe.

esempio 1

```
@Controller
@RequestMapping("/")
public class HelloController {
    @RequestMapping("/hello")
    @ResponseBody
    public String hello() {
        return "hello";
    }

    @RequestMapping("/bye")
    @ResponseBody
    public String bye() {
        return "bye";
    }
}
```

esempio 2

```
@Controller
@RequestMapping(path = {"/", "v1"})
public class HelloController {
    @RequestMapping(path = "/hello")
    @ResponseBody
    public String hello() {
        return "hello";
    }

    @RequestMapping(path = "/bye")
    @ResponseBody
    public String bye() {
        return "bye";
    }
}
```

esempio 3

```
@RestController
@RequestMapping(path = {"/rest", "/rest2"})
public class HelloRestController {
    @RequestMapping()
    public String showForm() {
        return "CIAO!!!!";
    }

    @RequestMapping(path = {"/2", "/2b"})
    public String showForm2() {
        return "CIAO 2!!!!";
    }
}
```

NOTA: se non vogliamo utilizzare altri parametri, possiamo inserire direttamente il path (o i path) senza specificare l'attributo (vedi *esempio 1*)

Di cosa abbiamo parlato in questa lezione

- Come definire ed utilizzare i controller in una web app Spring MVC