[Bài đọc] Annotation @RequestMapping

Trong lớp controller bạn viết mỗi phương thức xử lý request để xử một action. Để báo cho Spring biết phương thức nào xử lý cho action nào thì bạn cần ánh xạ URI với phương thức bằng cách sử dụng loại annotation theorg.springframework.web.bind.annotation.RequestMapping.

Annotation RequestMapping dùng để ánh xạ một request tới một phương thức. Bạn có thể sử dụng @RequestMapping chú thích cho một phương thức hoặc một lớp. Một phương thức chú thích bởi annotation  @RequestMapping sẽ trở thành phương thức xử lý request và sẽ được gọi khi dispatcher servlet nhận được một request với URL phù hợp.

Dưới đây là một lớp controller có chứa một phương thức sử dụng annotation RequestMapping

@Controller  
public class ProductController {  
  
 @Autowired  
 private ProductService productService;  
  
 @RequestMapping("/products")  
 public ModelAndView listProducts() {  
 List<Product> products = productService.findAll();  
 ModelAndView modelAndView = new ModelAndView("/product/list", "products", products);  
 return modelAndView;  
 }  
}

Để ánh xạ URL với phương thức cần sử dụng thuộc tính “value” trong annotation RequestMapping. Trong ví dụ trên, URI “/products” ánh xạ với phương thức listProducts(). Có nghĩa là phương thức listProduct có thể được gọi bởi mẫu URL như dưới:

[http](https://james.codegym.vn/mod/page/view.php?id=6839)://domain/context/products

Vì thuộc tính “value” là thuộc tính mặc định của annotation RequestMapping nên bạn có thể bỏ qua tên của thuộc tính nếu nó là thuộc tính duy nhất trong annotation RequestMapping. Nói cách khác hai annotation sau có cùng ý nghĩa

@RequestMapping(value = **"/products"**)  
@RequestMapping(**"/products"**)

Tuy nhiên nếu có nhiều hơn một thuộc tính trong @RequestMapping bạn phải viết thêm tên của thuộc tính “value”.

Thuộc tính “value” của một request mapping có thể trống. Trong trường hợp này phương thức sẽ ánh xạ với URL như dưới :

[http](https://james.codegym.vn/mod/page/view.php?id=6839)://domain/context

RequestMapping có các thuộc tính khác ngoài “value”. Ví dụ như thuộc tính “method” sẽ lấy một tập các phương thức “HTTP” để xử lý bằng phương thức tương ứng.

Ví dụ, phương thức “saveProduct” ở dưới chỉ có thể được gọi bằng phương thức  [HTTP](https://james.codegym.vn/mod/page/view.php?id=6839) POST

@RequestMapping(value = "/save-product", method = {RequestMethod.*POST, RequestMethod.PUT}*)  
public ModelAndView saveProduct(@ModelAttribute("productform") ProductForm productform, BindingResult result) {  
  
 // thong bao neu xay ra loi  
 if (result.hasErrors()) {  
 System.*out*.println("Result Error Occured" + result.getAllErrors());  
 }  
  
 // lay ten file  
 MultipartFile multipartFile = productform.getImage();  
 String fileName = multipartFile.getOriginalFilename();  
...  
}

Nếu có duy nhất một phương thức [HTTP request](https://james.codegym.vn/mod/page/view.php?id=6840) gán cho thuộc tính “method” thì dấu ngoặc “{}” là không bắt buộc.

@RequestMapping(value=**"/save-product"**, method=RequestMethod.POST)

Nếu thuộc tính method không được sử dụng thì phương thức request sẽ xử lý bất kì [HTTP](https://james.codegym.vn/mod/page/view.php?id=6839) method nào.

Annotation @RequestMapping cũng có thể sử dụng để chú thích cho một lớp controller như dưới.

@Controller  
@RequestMapping("/product")  
public class ProductController {  
  
 @Autowired  
 private ProductService productService;  
  
 @RequestMapping(value = "/save-product", method = RequestMethod.*POST*)  
 public ModelAndView saveProduct(@ModelAttribute("productform") ProductForm productform, BindingResult result) {  
  
 // thong bao neu xay ra loi  
 if (result.hasErrors()) {  
 System.*out*.println("Result Error Occured" + result.getAllErrors());  
 }  
  
 // lay ten file  
 MultipartFile multipartFile = productform.getImage();  
 String fileName = multipartFile.getOriginalFilename();   
 ...  
  
 }  
}

Vì ánh xạ của lớp controller là “/product” và  ánh xạ của phương thức là “/save-product”, nên phương thức sẽ được gọi khi sử dụng mẫu URL như dưới:

[http](https://james.codegym.vn/mod/page/view.php?id=6839)://domain/context/product/**save-product**

Viết phương thức xử lý request

Một phương thức xử lý request có thể có nhiều loại đối số đầu vào và có thể trả về các kiểu khác nhau. Ví dụ nếu bạn cần truy câp đến đối tượng HttpSession trong phương thức của bạn, bạn có thể thêm HttpSession như là một đối số và Spring sẽ truyền đối số chính xác cho bạn:

@RequestMapping("/uri")  
 public String myMethod(HttpSession session) {  
...  
 session.addAttribute(key, value);  
...  
 }

Hoặc nếu bạn cần “locale” của client  và HttpServletRequest, bạn có thể thêm cả hai như là đối số của phương thức như dưới:

@RequestMapping("/uri")  
 public String myOtherMethod(HttpServletRequest request,  
 Locale locale) {  
...  
// access Locale and HttpServletRequest here  
...  
 }

Dưới đây là danh sách các loại đố số có thể thêm vào như một đối số của phương thức xử lý request.

* javax.servlet.ServletRequest or javax.servlet.[http](https://james.codegym.vn/mod/page/view.php?id=6839).HttpServletRequest
* javax.servlet.ServletResponse or javax.servlet.[http](https://james.codegym.vn/mod/page/view.php?id=6839).HttpServletResponse
* javax.servlet.[http](https://james.codegym.vn/mod/page/view.php?id=6839).HttpSession
* org.springframework.web.context.request.WebRequest or org.springframework.web.context.request.NativeWebRequest
* java.util.Locale
* java.io.InputStream or java.io.Reader
* java.io.OutputStream or java.io.Writer
* java.security.Principal
* HttpEntity<?> parameters
* java.util.Map / org.springframework.ui.Model / org.springframework.ui.ModelMap
* org.springframework.web.servlet.mvc.support.RedirectAttributes
* org.springframework.validation.Errors / org.springframework.validation.BindingResult
* Command hoặc đối tượng form
* org.springframework.web.bind.support.SessionStatus
* org.springframework.web.util.UriComponentsBuilder
* Các loại annotation @PathVariable, @MatrixVariable, @RequestParam, @RequestHeader, @RequestBody, hoặc @RequestPart.

Đối số org.springframework.ui.Model là một loại đặc biệt quan trọng. Đối số này không thuộc thư viện của Servlet,  mà một loại Spring MVC có chứa một kiểu Map. Mỗi khi có một phương thức xử lý request được gọi, Spring MVC sẽ tạo ra một đối tượng Model và điền các đối tượng khác nhau vào Map.

Một phương thức xử lý request có thể trả về một trong những đối tượng sau:

* Một đối tượng ModelAndView
* Một đối tượng Model object
* Một Map chứa các thuộc tính của model
* Một đối tượng View
* Một String biểu diễn cho tên logic của view
* void
* Một đối tượng HttpEntity hoặc ResponseEntity để truy cập tới Servlet response, [HTTP](https://james.codegym.vn/mod/page/view.php?id=6839) headers và contents
* Một Callable
* Một DeferredResult
* Bất kỳ kiểu gì. Trong trường hợp bày giá trị trả về sẽ được xem xét như một thuộc tính của model để hiển thị ở view.

Last modified: Thursday, 21 February 2019, 3:25 PM