Spring Professional Exam Tutorial v5.0

Question 16

Question 16 - If you saw example Controller code, would you understand what it is doing? Could you tell if it was annotated correctly?

Controller can be defined in one of following ways:

- `@Controller` - Spring MVC Controller, should return view name and model
- `@RestController` - REST API Controller, `@RestController = @Controller + @ResponseBody`

Controller mapping can be defined with usage of one of following annotations:

- `@RequestMapping`
- Composed annotation:
  - `@GetMapping`
  - `@PostMappping`
  - `@PutMapping`
  - `@PatchMapping`
  - `@DeleteMapping`

Question 16 - If you saw example Controller code, would you understand what it is doing? Could you tell if it was annotated correctly?

Request parameter body can be mapped with usage of:

- `@RequstBody`
  - Additionally `@Valid` annotation can be used to trigger Bean Validation

Response can be bound to web response by:

- Usage of `@ResponseBody` annotation on top of `@Controller` or `@Controller` method
- Usage of `@RestController` annotation

Custom HTTP status can be provided for controller methods and exception with usage of `@ResponseStatus` annotation.

Question 16 - If you saw example Controller code, would you understand what it is doing? Could you tell if it was annotated correctly?

Request and URI parameters can be accessed with:

- `@RequestParam` - Servlet request parameters
- `@PathVariable` - access to URI template variables
- `@MatrixVariable` - access to name-value pairs in URI path segments, allows mapping variables from requests like `/employees/id=1;name=John`
- `@CookieValue` - bind the value of an HTTP cookie to a method argument in a controller
- `@RequestHeader` - access request header values or all header key and values when binding against a Map

Question 16 - If you saw example Controller code, would you understand what it is doing? Could you tell if it was annotated correctly?

Calls to controller can be intercepted, and custom exception handling can be implemented with one of:

- `@ExceptionHandler` - when applied at controller level method, acts as controller level exception handler
- `@ControllerAdvice` used together with `@ExceptionHandler` - global exception handler for all controllers, acts as global annotation driven call interceptor

Question 16 - If you saw example Controller code, would you understand what it is doing? Could you tell if it was annotated correctly?

```
@RestController
@RequestMapping("/api")
public class ApiController {


    @Autowired
    private CustomersDao customersDao;


    @GetMapping("/customers")
    public Iterable<Customer> list() {
        ...
    }


    @PostMapping("/customers")
    public ResponseEntity<Customer> create(@RequestBody @Valid Customer customer, ...) {
        ...
    }
}
```

@RestController annotation makes class discoverable Spring Bean that will handle API calls, also it informs spring that each value returned from method needs to be serialized

@RequestMapping sets URI root for this REST Controller

Controller should not contain any heavy logic, most of work should be delegated to components/services/dao, Controllers usually have collaborators injected.

Handler methods that handles calls to specific URIs registered with one of request mapping annotation.