Section 1: Customizing Spring WebMVC

Now that you are done with the videos of section 1, let’s assess your learning. Here, are a few questions, followed by 4 options, out of which 1 is the correct option. Select the right option and validate your learning! The answers are provided in a separate sheet

Q1. You need to provide additional header fields in your RestTemplate. How do you achieve this?

a. Create a subclass of the rest template and provide a bean of that type
b. Provide a RestTemplateCustomizer that configure the rest template
c. Create a Bean programmatically that is configured appropriately
d. Add the correct property settings to your yams or property file

Q2. You want spring to be able to transform a Principal instance to your own class that contains user data and vice versa. How do have to do this?

a. Create a constructor and a toPrincipal method in your class and spring detects it
b. Implement and register a Formatter<Principal> and Formatter<YourClass>
c. Implement and register a Formatter<Principal, YourClass>
d. Implement and register a Converter<Principal, YourClass> and Converter<YourClass, Principal>
e. 5. Implement and register a Converter<Principal, YourClass>

Q3. You have a LocalDate field in your Result and you want to have only this as localized french String in your json result. How do you configure this?

a) Configure the app to use french as default language and configure Jackson to use the patter by setting the property: spring.jackson.date-format= dd. MMMM YYYY
b) Use annotation on the field: @JsonFormat(locale=“fr“, pattern=“dd. MMMM YYYY “)
c) Use annotation on the field: @DateFormat(locale=“fr“, pattern=“dd. MMMM YYYY “)

Q4. You have a controller where 4 out of 5 methods use the same RequestParameter, which you use to make a lookup for data in your view model. Can you use @Modelattribute and if you can, how?

a) You can’t. This is only possible when all methods have the same parameter.
b) Add @RequestParameter(optional=true) with the correct type and a name to the ModelAttribute annotated method.
c) Modify the Annotation instance to look like this: @ModelAttribute(optional=true)