Q1.What is the Spring MVC framework?

Ans : Spring MVC is a Java framework that is used to develop web applications. It is built on a Model-View-Controller (MVC) pattern and possesses all the basic features of a spring framework, such as Dependency Injection, Inversion of Control.

Q2.What are the benefits of Spring MVC framework over other MVC frameworks?

Ans : Advantages of Spring MVC

1. Different roles: The Spring MVC has separate roles which can be fulfilled with the help of a specialized object. The roles involved in this are command object, model object, controller, dispatcher servlet, view resolver, validator, and more.
2. Lightweight: The container used for the development and deployment of applications uses a lightweight servlet.
3. Fast development: The MVC spring framework enables rapid and parallel development. It helps the developer to complete the project on time.
4. Strong and powerful configuration: It provides powerful configuration for application and framework classes. These configurations include easy simple referencing like business objects and web controllers.
5. Business code: It provides reusable business codes that allow the developer to use existing business objects instead of creating new objects.
6. Easy testing: Spring generally uses Java Beans that allows the developer to inject data using easy methods.
7. Mapping: It provides flexible mapping that allows the page to redirect easily.
8. Dependency Injection: Inversion of Control or dependency injection allows the developer to not create a complete environment for the dependencies.

Q3.What is DispatcherServlet in Spring MVC? In other words, can you explain the Spring MVC architecture?

Ans : The DispatcherServlet is the front controller in Spring web applications. It's used to create web applications and REST services in Spring MVC. In a traditional Spring web application, this servlet is defined in the web.

Spring MVC is a Java framework that is used to develop web applications. It is built on a Model-View-Controller (MVC) pattern and possesses all the basic features of a spring framework, such as Dependency Injection, Inversion of Control.



* Model - A model contains the data of the application. A data can be a single object or a collection of objects.
* Controller - A controller contains the business logic of an application. Here, the @Controller annotation is used to mark the class as the controller.
* View - A view represents the provided information in a particular format. Generally, JSP+JSTL is used to create a view page. Although spring also supports other view technologies such as Apache Velocity, Thymeleaf and FreeMarker.
* Front Controller - In Spring Web MVC, the DispatcherServlet class works as the front controller. It is responsible to manage the flow of the Spring MVC application.

Q4.What is a View Resolver pattern and explain its significance in Spring MVC?

Ans : All MVC frameworks for web applications provide a way to address views. Spring provides view resolvers, which enable you to render models in a browser without tying you to a specific view technology.

Q5.What are the differences between @RequestParam and @PathVariable annotations?

Ans : @RequestParam is used to capture query parameters or form parameters from the URL, while @PathVariable is used to capture values from the URL path.

Q6.What is the Model in Spring MVC?

Ans : In Spring MVC, the model works a container that contains the data of the application. Here, a data can be in any form such as objects, strings, information from the database, etc.

It is required to place the Model interface in the controller part of the application.

Q7.What is the role of @ModelAttribute annotation?

Ans : @ModelAttribute is an annotation that binds a method parameter or method return value to a named model attribute, and then exposes it to a web view.

Q8.What is the significance of @Repository annotation?

Ans : @Repository Annotation is a specialization of @Component annotation which is used to indicate that the class provides the mechanism for storage, retrieval, update, delete and search operation on objects.

Q9.What does REST stand for? and what is RESTful web services?

Ans : REST or Representational State Transfer is an architectural style that can be applied to web services to create and enhance properties like performance, scalability, and modifiability. RESTful web services are generally highly scalable, light, and maintainable and are used to create APIs for web-based applications.

Q10.What is differences between RESTful web services and SOAP web services?

Ans :

|  |  |  |
| --- | --- | --- |
| No. | SOAP | REST |
| 1) | SOAP is a protocol. | REST is an architectural style. |
| 2) | SOAP stands for Simple Object Access Protocol. | REST stands for REpresentational State Transfer. |
| 3) | SOAP can't use REST because it is a protocol. | REST can use SOAP web services because it is a concept and can use any protocol like HTTP, SOAP. |
| 4) | SOAP uses services interfaces to expose the business logic. | REST uses URI to expose business logic. |
| 5) | JAX-WS is the java API for SOAP web services. | JAX-RS is the java API for RESTful web services. |
| 6) | SOAP defines standards to be strictly followed. | REST does not define too much standards like SOAP. |
| 7) | SOAP requires more bandwidth and resource than REST. | REST requires less bandwidth and resource than SOAP. |
| 8) | SOAP defines its own security. | RESTful web services inherits security measures from the underlying transport. |
| 9) | SOAP permits XML data format only. | REST permits different data format such as Plain text, HTML, XML, JSON etc. |
| 10) | SOAP is less preferred than REST. | REST more preferred than SOAP. |