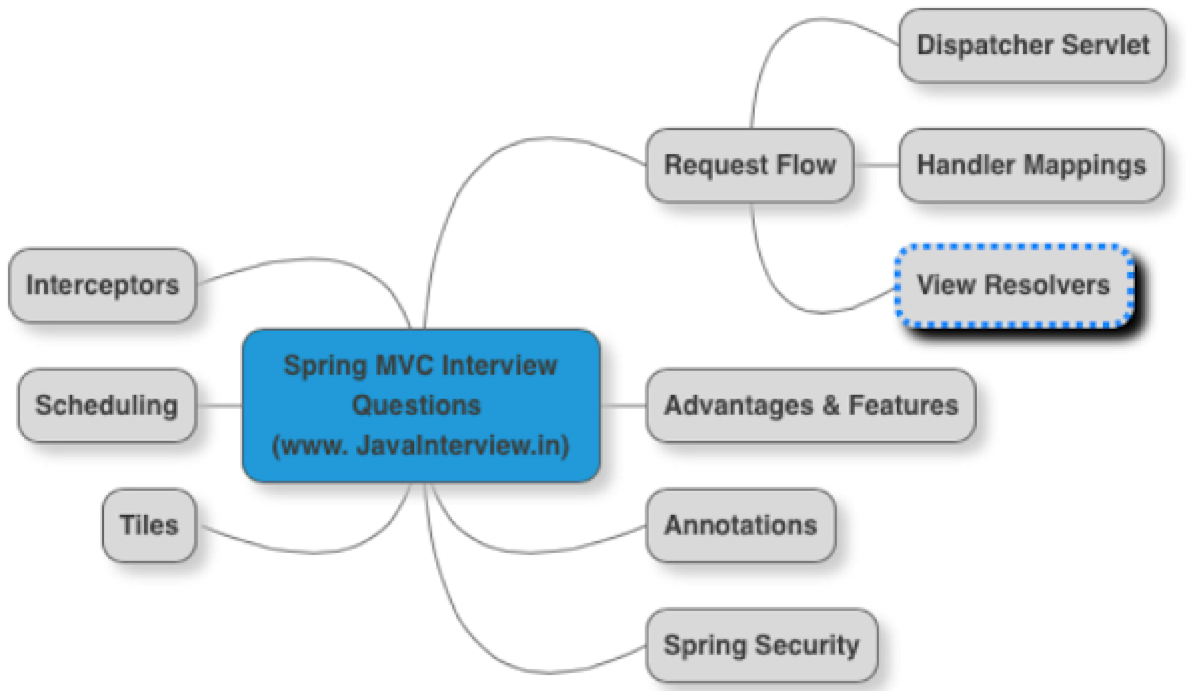
Spring MVC Interview Questions and Answers

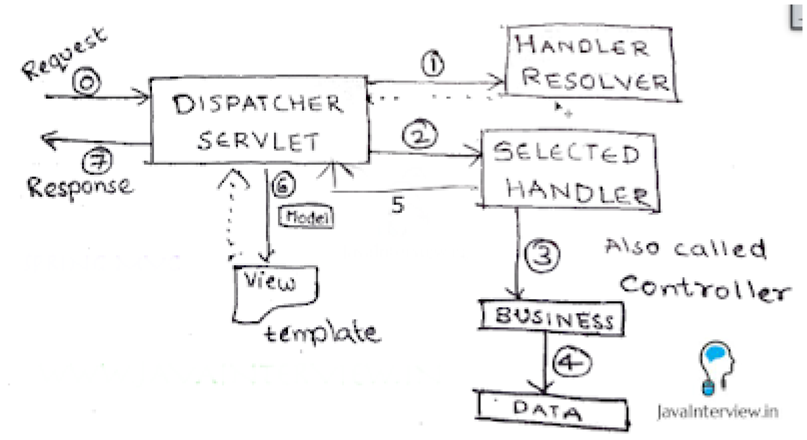


(http://4.bp.blogspot.com/-6W72sFW3v4I/VVDVvI_L_EI/AAAAAAAAAIs/FXHW_NoOR5I/s1600/SpringMVCInterviewQuestions.png)

1. How does request flow happen in Spring MVC?
2. Can you list a few advantages of using Spring MVC framework?
3. Give examples of important Spring MVC annotations?
4. Can you explain the concept of Interceptors in Spring MVC?
5. Interceptors can be configured using the interceptors property.

6. How do you schedule tasks with Spring?
7. How do you integrate Spring MVC with tiles?
8. How do you configure Spring MVC web application to use UTF-8 encoding for handling forms?
9. How do you enable spring security for a web application?

How does request flow happen in Spring MVC?



(http://4.bp.blogspot.com/-iz_lkHWqkNM/VVDHgFUkLsI/AAAAAAAAAHY/i9l_PNDFe3U/s1600/Spring%2BMVC%2BRequest%2BFlow.png)
Shown in the picture below. DispatcherServlet acts as the front controller. Simplified actions taken by DispatcherServlet are listed below.

- All requests arrive at the DispatcherServlet (Front Controller) - STEP 0 in Figure
- DispatcherServlet resolves theme and locale as configured.
- Find's appropriate Controller (Handler) to handle the request. (pre-processors and post-processors, if configured) (STEP 1)
- Redirect to the Controller (Handler) - STEP 2. Controller executes the request and returns a view name and a view model object. (STEP 3,4,5)
- DispatcherServlet resolves the view name and redirects to the view template. The response html is returned to DispatcherServlet. (STEP 6)
- DispatcherServlet send the response back to the browser. (STEP 7)

Can you list a few advantages of using Spring MVC framework?

- In Spring Web MVC, any POJO can be used as a command or form-backing object.

- Highly flexible databinding – If there is a type mismatch, it is shown as a validation error on the screen. Business POJO’s can directly be used as form-backing objects.
- Flexible view resolution: Controller can either select a view name and prepare model map for it or write directly to response stream.
- Supports JSP, Velocity and Freemarker view technologies.
- Can directly generate XML, JSON, Atom, and many other types of content.
- Highly convenient tag library.

Give examples of important Spring MVC annotations?

Important Spring MVC annotations are listed below.

- @Controller : This class would serve as a controller.
- @RequestMapping : Can be used on a class or a method. Maps an url on the class (or method).
- @PathVariable : Used to map a dynamic value in the url to a method argument.

Example 1 : Maps a url “/players “ for the controller method.

```
@RequestMapping(value="/players", method=RequestMethod.GET)
public String findAllPlayers(Model model) {
```

Example 2 : If a url /players/15 is keyed in, playerId is populated with value 15.

```
@RequestMapping(value="/players/{playerid}", method=RequestMethod.GET)
public String findPlayer(@PathVariable String playerId, Model model) {
```

Can you explain the concept of Interceptors in Spring MVC?

Handler interceptors are used when you want to apply specific functionality to certain requests. Handler Interceptors should implement the interface HandlerInterceptor.

Three methods are defined:

- preHandle(..) is called before the actual handler is executed;
- postHandle(..) is called after the handler is executed;
- afterCompletion(..) is called after the complete request has finished.

Interceptors can be configured using the interceptors property.

```
<bean id="handlerMapping" class="org.springframework.web.servlet.mvc.method.annotation.RequestMa
ppingHandlerMapping">
      <property name="interceptors">
            <list>
                  <ref bean="yourCustomHandlerInterceptor"/>
            </list>
        </property>
</bean>
```

How do you schedule tasks with Spring?

Spring 3.0 introduced TaskScheduler abstract to deal with scheduling jobs. Spring has support for Timer (Jdk) and Quartz. Sample methods in the interface TaskScheduler are shown below:

```
ScheduledFuture scheduleAtFixedRate(Runnable task, long period);
ScheduledFuture scheduleWithFixedDelay(Runnable task, long delay);
```

Scheduling can also be done using an annotation

```
@Scheduled(fixedDelay=5000)
public void doSomething() {
    // something that should execute periodically
}
```

Below example shows scheduling with xml configuration

```
<task:scheduler id="customScheduler" pool-size="30"/>
<task:scheduled-tasks scheduler=" customScheduler ">
    <task:scheduled ref="someBean" method="someOtherMethod" fixed-delay="5000" initial-delay="10
00"/>
    <task:scheduled ref="someOtherBean" method="someMethod" cron="*/5 * * * * MON-FRI"/>
</task:scheduled-tasks>
```

How do you integrate Spring MVC with tiles?

Tiles helps us to define the layout for a web page. We can integrate Spring MVC with tiles by configuring TilesConfigurer and setting up appropriate view resolver.

```
    <bean id="tilesConfigurer"
    class="org.springframework.web.servlet.view.tiles2.TilesConfigurer"
    p:definitions="/WEB-INF/tiles-defs/templates.xml" />
    <bean id="tilesViewResolver"
    class="org.springframework.web.servlet.view.UrlBasedViewResolver"
    p:viewClass="org.springframework.web.servlet.view.tiles2.TilesView" />
```

How do you configure Spring MVC web application to use UTF-8 encoding for handling forms?

Using org.springframework.web.filter.CharacterEncodingFilter. Shown below.

```
<filter>
    <filter-name>encoding-filter</filter-name>
    <filter-class>
        org.springframework.web.filter.CharacterEncodingFilter
    </filter-class>
    <init-param>
        <param-name>encoding</param-name>
        <param-value>UTF-8</param-value>
    </init-param>
</filter>
<filter-mapping>
    <filter-name>encoding-filter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

How do you enable spring security for a web application?

Spring Security is used to implement Authentication and Authorization for a web application. We can enable spring security by configuring an appropriae security filter. Example shown below. We can create a separate security-context.xml to define the authentication and authorization roles and accesses.

```
<filter>
    <filter-name>springSecurityFilterChain</filter-name>
    <filter-class>
  org.springframework.web.filter.DelegatingFilterProxy
    </filter-class>
</filter>
<filter-mapping>
    <filter-name>springSecurityFilterChain</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

3. Design : Design (http://www.javainterview.in/p/design-interview-questions.html), Design Patterns (http://www.javainterview.in/p/design-patterns-interview-questions.html), Code Review (http://www.javainterview.in/p/code-review-interview-questions.html)
4. Architecture : Architecture (http://www.javainterview.in/p/architect-interview-questions.html), Performance & Load Testing (http://www.javainterview.in/p/performance-and-load-testing-interview.html), Web Services (http://www.javainterview.in/p/web-services-interview-questions.html), REST Web Services (http://www.javainterview.in/p/rest-web-services-interview-questions.html),Security (http://www.javainterview.in/p/security-interview-questions.html), Continuous Integration (http://www.javainterview.in/p/continuous-integration-interview.html)