Spring вопросы к собеседованию

Этот небольшой список вопросов даст вам понимание самых важных концепций Spring, а так же поможет подготовится к собеседованию

1 Для чего нужен Component Scan?

Если вы понимаете как работает Component Scan, то вы понимаете Spring

Первый шаг для описания Spring Beans это добавление аннотации — @Component, или @Service, или @Repository.

Однако, Spring ничего не знает об этих бинах, если он не знает где искать их. То, что скажет Spring где искать эти бины и называется Component Scan. В @ComponentScan вы указываете пакеты, которые должны сканироваться.

Spring будет искать бины не только в пакетах для сканирования, но и в их подпакетах.

2 Как вы добавите Component Scan в Spring Boot?

@SpringBootApplication

public class Application {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

}

}

@SpringBootApplication определяет автоматическое сканирование пакета, где находится класс Application

Всё будет в порядке, ваш код целиком находится в указанном пакете или его подпакетах.

Однако, если необходимый вам компонент находится в другом пакете, вы должны использовать дополнительно аннотацию @ComponentScan, где перечислите все дополнительные пакеты для сканирования

3 В чём отличие между @Component и @ComponentScan?

@Component и @ComponentScan предназначены для разных целей

@Component помечает класс в качестве кандидата для создания Spring бина.  
@ComponentScan указывает где Spring искать классы, помеченные аннотацией @Component или его производной

4 Для чего используется аннотация @Bean?

В классах конфигурации Spring, @Bean используется для определения компонентов с кастомной логикой.

5 В чём разница между @Bean и @Component?

@Bean используется в конфигурационных классах Spring. Он используется для непосредственного создания бина.

@Component используется со всеми классами, которыми должен управлять Spring. Когда Spring видит класс с @Component, Spring определяет этот класс как кандидата для создания bean.

6 В чём разница между @Component, @Service и @Repository аннотациями?

Все они определяют бины Spring. Однако между ними всё же есть разница.

@Component — универсальный компонент  
@Repository — компонент, который предназначен для хранения, извлечения и поиска. Как правило, используется для работы с базами данных.  
@Service — фасад для некоторой бизнес логики

Пользовательские аннотации, производные от @Component, могут добавлять специальную логику в бинах.

Например, бины, получившиеся при помощи @Repository, дополнительно имеют обработку для JDBC Exception

7 Можем ли мы использовать @Component вместо @Service для бизнес логики?

Да. конечно.

Если @Component является универсальным стереотипом для любого Spring компонента, то @Service в настоящее время является его псевдонимом. Однако, в официальной документации Spring рекомендуется использовать именно @Service для бизнес логики. Вполне возможно, что в будущих версиях фреймворка, для данного стереотипа добавится дополнительная семантика, и его бины станут обладать дополнительной логикой.

8 В чем различие между web.xml и the Spring Context - servlet.xml?

web.xml — Метаданные и конфигурация любого веб-приложения, совместимого с Java EE. Java EE стандарт для веб-приложений.  
servlet.xml — файл конфигурации, специфичный для Spring Framework.

9 Что предпочитаете использовать для конфигурации Spring - xml или аннотирование?

Предпочитаю аннотации, если кодовая база хорошо описывается такими элементами, как @Service, @Component, @Autowired

Однако когда дело доходит до конфигурации, у меня нет каких-либо предпочтений. Я бы оставил этот вопрос команде.

10 Можем ли мы применить @Autowired с не сеттерами и не конструкторами методами?

Да, конечно.

@Autowired может использоваться вместе с конструкторами, сеттерами или любым другими методами. Когда Spring находит @Autowired на методе, Spring автоматически вызовет этот метод, после создания экземпляра бина. В качестве аргументов, будут подобраны подходящие объекты из контекста Spring.

11 В чем разница между Сквозной Функциональностью (Cross Cutting Concerns) и АОП (аспектно оринтированное программирование)?

Сквозная Функциональность — функциональность, которая может потребоваться вам на нескольких различных уровнях — логирование, управление производительностью, безопасность и т.д.  
АОП — один из подходов к реализации данной проблемы

12 В чем разница между IOC (Inversion of Control) и Application Context?

IOC — инверсия управления. Вместо ручного внедрения зависимостей, фреймворк забирает ответственность за это.  
ApplicationContext — реализация IOC спрингом.

Bean Factory — это базовая версия IOC контейнера

Application Context также включает дополнительные функции, которые обычно нужны для разработки корпоративных приложений

13 В чем разница между classPathXmlApplicationContext и annotationConfigApplicationContext?

classPathXmlApplicationContext — если вы хотите инициализировать контекст Spring при помощи xml  
annotationConfigApplicationContext — если вы хотите инициализировать контекст Spring при помощи конфигурационного класса java

14 Почему возвращаемое значение при применении аспекта @Around может потеряться? Назовите причины.

Метод, помеченный аннотацией @Around, должен возвращать значение, которое он (метод) получил из joinpoint.proceed()

@Around("trackTimeAnnotation()")

public Object around(ProceedingJoinPoint joinPoint) throws Throwable{

long startTime = System.currentTimeMillis();

Object retVal = joinPoint.proceed();

long timeTaken = System.currentTimeMillis() - startTime;

logger.info("Time taken by {} is equal to {}",joinPoint, timeTaken);

return retVal;

}

15 Как вы решаете какой бин инжектить, если у вас несколько подходящих бинов. Расскажите о @Primary и @Qualifier?

Если есть бин, который вы предпочитаете большую часть времени по сравнению с другими, то используйте @Primary, и используйте @Qualifier для нестандартных сценариев.

Если все бины имеют одинаковый приоритет, мы всегда будем использовать @Qualifier

Если бин надо выбрать во время исполнения программы, то эти аннотации вам не подойдут. Вам надо в конфигурационном классе создать метод, пометить его аннотацией @Bean, и вернуть им требуемый бин.

16 Что нового в Spring Framework 5.0?

На мой взгляд это Functional Web Framework, Kotlin и и поддержка реактивного программирования.

17 Сравните Application Context, IOC Container, vs Web Container и EJB Container. Нужен ли Web Container для запуска Spring Boot приложения?

Web Container и EJB Containers являются частью приложения/веб-сервера, таких как Tomcat, Websphere, Weblogic. Они добавляют свою дополнительную функциональность к ним. Java EE определяет контракт для веб-приложений, эти контейнеры являются реализацией этих контрактов.

Spring контейнер может являться частью любого приложения, которое вы делаете на java. Spring может работать внутри веб-контейнера, ejb контейнера или даже без них.

18 Как мы можем выбрать подходящий бин при помощи application.properties?

Рассмотрим пример:

interface GreetingService {

public String sayHello();

}

И два компонента

@Component(value="real")

class RealGreetingService implements GreetingService {

public String sayHello() {

return "I'm real";

}

}

@Component(value="mock")

class MockGreetingService implements GreetingService {

public String sayHello() {

return "I'm mock";

}

}

Тогда в application.properties добавим свойство  
application.greeting: real

Воспользуемся данным решением:

@RestController

public class WelcomeController {

@Resource(name="${application.greeting}")

private GreeterService service1;

}

19 Какая минимальная версия Java поддерживается в Spring Boot 2 и Spring 5?

Spring 5.0 и Spring Boot 2.0 поддерживают Java 8 и более поздней версии.

20 В чём разница между @Controller и @RestController?

@RestController = @Controller + @ResponseBody

@RestController превращает помеченный класс в Spring-бин. Этот бин для конвертации входящих/исходящих данных использует Jackson message converter. Как правило целевые данные представлены в json или xml.

21 Почему иногда мы используем @ResponseBody, а иногда ResponseEntity?

ResponseEntity необходим, только если мы хотим кастомизировать ответ, добавив к нему статус ответа. Во всех остальных случаях будем использовать @ResponseBody.

@GetMapping(value=”/resource”)

@ResponseBody

public Resource sayHello() { return resource; }

@PostMapping(value=”/resource”)

public ResponseEntity createResource() {

….

return ResponseEntity.created(resource).build();

}

Стандартные HTTP коды статусов ответов, которые можно использовать.  
200 — SUCCESS  
201 — CREATED  
404 — RESOURCE NOT FOUND  
400 — BAD REQUEST  
401 — UNAUTHORIZED  
500 — SERVER ERROR

Для @ResponseBody единственные состояния статуса это SUCCESS(200), если всё ок и SERVER ERROR(500), если произошла какая-либо ошибка.

Допустим мы что-то создали и хотим отправить статус CREATED(201). В этом случае мы используем ResponseEntity.

22 В чем разница между Filters, Listeners and Interceptors?

Концептуально всё просто, фильтры сервлетов могут перехватывать только HTTPServlets. Listeners могут перехватывать специфические события. Как перехватить события которые относятся ни к тем не другим?

Фильтры и перехватчики делают по сути одно и тоже: они перехватывают какое-то событие, и делают что-то до или после.

Java EE использует термин Filter, Spring называет их Interceptors.

Именно здесь AOP используется в полную силу, благодаря чему возможно перехватывание вызовов любых объектов

23 В чем разница между ModelMap и ModelAndView?

Model — интерфейс, ModelMap его реализация..

ModelAndView является контейнером для пары, как ModelMap и View.

Обычно я люблю использовать ModelAndView. Однако есть так же способ когда мы задаем необходимые атрибуты в ModelMap, и возвращаем название View обычной строкой из метода контроллера.

24 В чем разница между model.put() и model.addAttribute()?

Метод addAttribute отделяет нас от работы с базовой структурой hashmap. По сути addAttribute это обертка над put, где делается дополнительная проверка на null. Метод addAttribute в отличии от put возвращает modelmap.  
model.addAttribute(“attribute1”,”value1”).addAttribute(“attribute2”,”value2”);

25 Что можете рассказать про Form Binding?

Нам это может понадобиться, если мы, например, захотим взять некоторое значение с HTML страницы и сохранить его в БД. Для этого нам надо это значение переместить в контроллер Спринга.

Если мы будем использовать Spring MVC form tags, Spring автоматически свяжет переменные на HTML странице с Бином Спринга.

Если мне придется с этим работать, я обязательно буду смотреть официальную документацию Spring MVC Form Tags.

26 Почему мы используем Hibernate Validator?

Hibernate Validator никак не связан с БД. Это просто библиотека для валидации.

Hibernate Validator версии 5.x является эталонной реализацией Bean Validation 1.1

Так же если взглянуть по адресу <http://beanvalidation.org/2.0>, то Hibernate Validator является единственным, который сертифицирован.

27 Где должны располагаться статические (css, js, html) ресурсы в Spring MVC приложении?

Расположение статических ресурсов можно настроить. В документации Spring Boot рекомендуется использовать /static, или /public, или /resources, или /META-INF/resources

28 Почему для конфиденциальных данных рекомендуется использовать POST, а не GET запросы?

В случае GET запроса передаваемые параметры являются частью url, и все маршрутизаторы, через которые пройдет наш GET запрос, смогут их прочитать.

В случае POST запроса передаваемые параметры являются частью тела запроса. При использовании HTTPs, тело запроса шифруется. Следовательно, использование POST запросов является более безопасным

29 Можно ли передать в запросе один и тот же параметр несколько раз?

Пример:  
http://localhost:8080/login?name=Ranga&name=Ravi&name=Sathish  
Да, можно принять все значения, используя массив в методе контроллера

public String method(@RequestParam(value="name") String[] names){

}