1. **Перечислите Spring модули и их назначение.**

Модули разбиты на контейнеры. Контейнер данных:orm(бд), oxm – объект/xml, jms – сообщения, transactions, web (wev-mvc,web-socket,servlet)

основной контейнер:(включает aop – аспекто-ориентированный язык, aspect – для интеграции сaspectJ, instrumentation – спец классы для серверных приложений, messaging – поддержка stomp.(протокол обменна данных)

1. **Что такое Spring Boot? В чем его преимущества и для чего он используется?**

Связующее звено, объединяющее набор компонентов. Всё через аннотации. Автоматическая конфигурация приложения.

**Spring Boot** - дает вам инструмент командной строки для создания приложения. Позволяет избежать XML конфигурации

**4. Для чего используется аннотация @SpringBootApplication?**

Этот класс аннотирован через @**SpringBootApplication.** Он выполняет автоматическую конфигурацию Spring, и автоматически сканирует (scan) весь проект, чтобы найти компоненты Spring (Controller, Bean, Service, ...)

. **5. Объясните принцип IoC (Inversion of Control)? Какие формы используются в Spring для внедрения?**

Ядро Spring Framework основано на принципе *инверсии управления* (lnversion of Control - IoC), когда создание и управление зависимостями между компонентами становятся внешними.(внедрение и поиск зависимостей)

**6. В чем суть понятия lnversion of Control (IoC)?**

 это некий абстрактный принцип, набор рекомендаций для написания слабо связанного кода. Суть которого в том, что каждый компонент системы должен быть как можно более изолированным от других, не полагаясь в своей работе на детали конкретной реализации других компонентов.

**7. В чем различие внедрение зависимостей (Dependency lnjection) и поиска зависимостей (Dependency Lookup)?**

* Внедрение через конструктор
* Внедрение через set-метод
* Внедрение через интерфейс

В случае с поиском зависимостей класс должен самостоятельно реализовывать логику получения зависимостей извне. Для этого он должен иметь доступ к некоему источнику зависимостей.

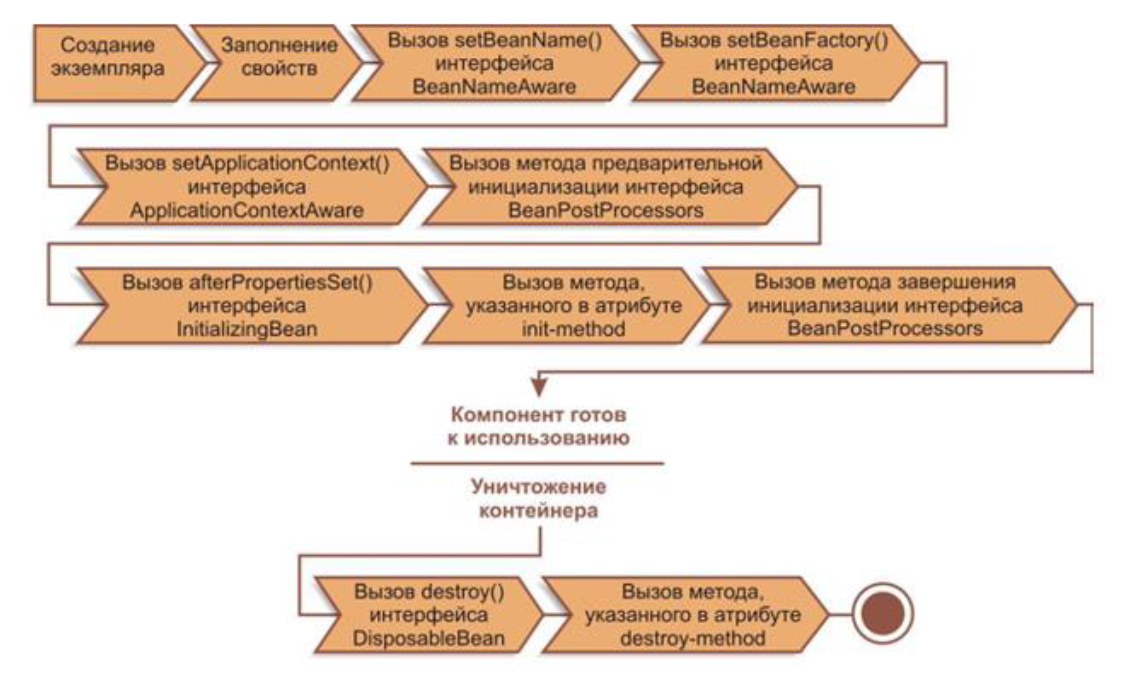
**8. Что такое JavaBean? Какие есть правила описания и использования?**

**JavaBean** – **это** одноуровневые объекты, использующиеся для того, чтобы инкапсулировать в одном объекте код, данные или и то и другое. Компонент **JavaBean** может иметь свойства, методы и события, открытые для удаленного доступа.

**9. Перечислите области видимости bean.**

**Singleton** Определяет один единственный бин для каждого контейнера Spring IoC (используется по умолчанию).  
**Prototype** Позволяет иметь любое количество экземпляров бина.  
**request** Создаётся один экземпляр бина на каждый HTTP запрос. Касается исключительно ApplicationContext.  
**session** Создаётся один экземпляр бина на каждую HTTP сессию. Касается исключительно ApplicationContext.  
**global-session** Создаётся один экземпляр бина на каждую глобальную HTTP сессию. Касается исключительно ApplicationContext.

**10. Опишите ЖЦ бина.**



**11. Поясните значения аннотаций: @Configuration, @Bean, @Component, @Service, @Repository, @Controller.**

@Configuration сообщает Spring что данный класс является конфигурационным, содержит определения и зависимости bean-компонентов.

Для определения бина используется аннотация @Bean.

**Component –** класс как компонент спринга

@service – сервис(тоже самое что компонент, просто для вида)

@repository

@controller – в mvc

**12. Spring Expression Language (SpEL): расскажите об особенности и области использования.**

является мощным языком выражений, который поддерживает запросы и динамическую манипуляцию графа объекта

**13. Охарактеризуйте основные Core Container Spring.**

Core содержит несколько базовых классов и инструментов. Вся Spring Framework основан на Core контейнере.

Модуль Core содержит базовые классы Spring Framework, включая внедрение зависимостей (DI) и инверсию управления (IOC).

А так же бины, контекст и spel

**14. Как в Spring происходит разрешение зависимостей?**

порядке:

1) Создается и инициализируется ApplicationContext с метаданными конфигурации, которые описывают все бины. Эти метаданные могут быть описаны через XML, Java-код или аннотации

2) Для каждого бина и его зависимостей вычисляются свойства, аргументы конструктора. Эти зависимости предоставляются бину.

3) Каждому свойству или аргументу конструктора устанавливается значение или ссылка на другой бин в контейнере

4) Для каждого свойства или аргумента конструктора подставляемое значение конвертируется в тот формат, который указан для свойства или аргумента. По умолчанию Spring может конвертировать значения из строкового формата во все встроенные типы, такие как int, long, String, boolean и др.

**15. Поясните как работает DispatcherServlet, HandlerMapping, ViewResolver?**

Когда мы пишем в строке браузера запрос, его принимает ***Dispatcher Servlet***, далее он находит для обработки этого запроса подходящий контроллер с помощью ***HandlerMapping*** , затем диспетчер находит нужное представления по имени при помощи ***ViewResolver***'а, после чего на это представление передаются данные модели и на выход мы получаем нашу страничку.

**16. Как используется паттерн «Front Controller» в Spring?**

Паттерн Front Controller используется для обеспечения централизованного механизма обработки запросов, так что все запросы обрабатываются одним обработчиком. Этот обработчик может выполнить аутентификацию, авторизацию, регистрацию или отслеживание запроса, а затем передать запрос соответствующему контроллеру.

**17. Как происходит адресация в контроллере?**

В RequestMapping передаем тип метода и ссылку.

**18. Расскажите про Spring MVC архитектуру.**

Фреймворк **Spring MVC** обеспечивает архитектуру паттерна Model — View — Controller (Модель — Отображение (далее — Вид) — Контроллер) при помощи слабо связанных готовых компонентов. Паттерн MVC разделяет аспекты приложения (логику ввода, бизнес-логику и логику UI), обеспечивая при этом свободную связь между ними

* **Model** (Модель) инкапсулирует (объединяет) данные приложения, в целом они будут состоять из POJO («Старых добрых Java-объектов», или бинов).
* **View** (Отображение, Вид) отвечает за отображение данных Модели, — как правило, генерируя HTML, которые мы видим в своём браузере.
* **Controller** (Контроллер) обрабатывает запрос пользователя, создаёт соответствующую Модель и передаёт её для отображения в Вид.

**SpringBook application. Аннотирован через @SpringBootApplication. Тут автоматическая конфигурация spring.(+ищет все компоненты)**

**Application.properties строки приветствия и ошибки**

**Класс book – модель с двумя полями.**

**Класс bookForm представляет данные когда создастся новый бук**

**Контроллер – класс, обрабатывающий запросы.**

**@value – самый легкий способ добавления значений из конфигурации в код**