Вопросы и ответы на собеседование по Spring

(61-75)

61). В чем разница между ModelMap и ModelAndView?

ModelMap и ModelAndView - это два разных способа передачи модели (данных) из контроллера в представление в Spring MVC.

ModelMap:
ModelMap - это простая структура данных, используемая для хранения модели (данных), которые должны быть переданы в представление. ModelMap представляет собой обычный словарь, где ключи - это имена атрибутов, а значения - это сами данные. Вы можете добавлять и удалять атрибуты в ModelMap и передавать её в представление.
Пример использования ModelMap:

```
@GetMapping("/example")
public String example(ModelMap model) {
    model.addAttribute("message", "Hello, World!");
    return "exampleView";
}
```

Здесь мы добавляем атрибут "message" в ModelMap, который будет доступен в представлении "exampleView".

61). Продолжение

ModelAndView:
ModelAndView - это объект, который сочетает в себе модель (данные) и информацию о представлении (название представления). В отличие от ModelMap, ModelAndView позволяет явно указать, какое представление должно быть использовано для отображения данных.
Пример использования ModelAndView:

```
@GetMapping("/example")
public ModelAndView example() {
    ModelAndView modelAndView = new ModelAndView("exampleView");
    modelAndView.addObject("message", "Hello, World!");
    return modelAndView;
}
```

Здесь мы создаем ModelAndView, указываем название представления "exampleView" и добавляем атрибут "message" к модели. В результате, Spring MVC знает, какое представление использовать для отображения данных.

Разница между ними заключается в том, что ModelMap предоставляет только данные (**модель**), но не указывает, как их отображать, в то время как ModelAndView позволяет объединить модель и представление в один объект, что может быть удобно в некоторых случаях. Выбор между ними зависит от ваших предпочтений и требований для конкретного контроллера и представления.

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

model.**put**() - это метод, который предоставляется Java Map, и его можно использовать для добавления атрибутов в модель.

model.**addAttribute**() - это специфичный метод, предоставляемый Spring MVC для добавления атрибутов в модель.

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”);

63). 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.

64). 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

Hibernate Validator является сертифицированным.

65). 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.

66). 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.

67). Можно ли передать в запросе один и тот же параметр несколько раз?

Пример:
http://localhost:8080/login?name=Ranga&name=John&name=Smith

Да, можно принять все значения, используя массив в методе контроллера.

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

68). В чем разница между Spring Boot и Spring MVC? Или между Spring Boot и Spring Framework? Можете ли вы использовать их вместе в одном проекте?

Spring Boot построен поверх Spring Framework.
Пример: Spring Framework предлагает вам возможность читать файлы свойств .properties из различных мест, например, с помощью аннотаций @PropertySource. Он также предлагает вам возможность писать JSON REST контроллеры с помощью инфраструктуры Web MVC.
Проблема в том, что вы должны указать Spring откуда читать свойства приложения и правильно настроить веб-фреймворк, например, для поддержки JSON. Spring Boot, с другой стороны, берет эти отдельные части и предварительно настраивает их для вас.
Например:

- Он всегда автоматически ищет файлы application.properties в различных заранее определенных местах и считывает их.
- Он всегда загружает **встроенный** Tomcat, поэтому вы можете сразу увидеть результаты написания ваших @RestControllers и начать писать веб-приложения.
- Он автоматически настраивает все для отправки / получения JSON, не беспокоясь о конкретных зависимостях Maven / Gradle. Всё, путем запуска основного метода в классе Java, который аннотируется аннотацией @SpringBootApplication.

69). Почему вам не нужно указывать версии зависимостей в файле pom.xml при включении сторонних библиотек? Верно ли это для всех сторонних библиотек или только для некоторых?

Это потому, что Spring Boot выполняет за вас некоторое управление зависимостями.
На верхнем уровне стартеры Spring Boot закачивают родительский файл pom.xml (или файл build.gradle), в котором определены все зависимости и соответствующие версии, которые поддерживает конкретная версия Spring Boot - так называемый **BOM** (**Bill Of Materials**). Затем вы можете просто использовать эти предопределенные версии или переопределить номера версий в своих собственных сценариях сборки.

70). Расскажите о Spring Security.

Проект Spring Security предоставляет широкие возможности для защиты приложения. Кроме стандартных настроек для аутентификации, авторизации и распределения ролей и маппинга доступных страниц, ссылок и т.п., предоставляет защиту от различных вариантов атак (например CSRF). Имеет множество различных настроек, но остается легким в использовании.

Чтобы использовать Spring Security в веб-приложениях, мы можем начать с простой аннотации **@EnableWebSecurity** .

71). Как работает прототип Scope?

Prototype Scope означает, что каждый раз, когда мы вызываем экземпляр Bean, Spring будет создавать новый экземпляр и возвращать его. Это отличается от **singleton** по умолчанию, где один экземпляр объекта создается один раз для контейнера Spring **IoC**.

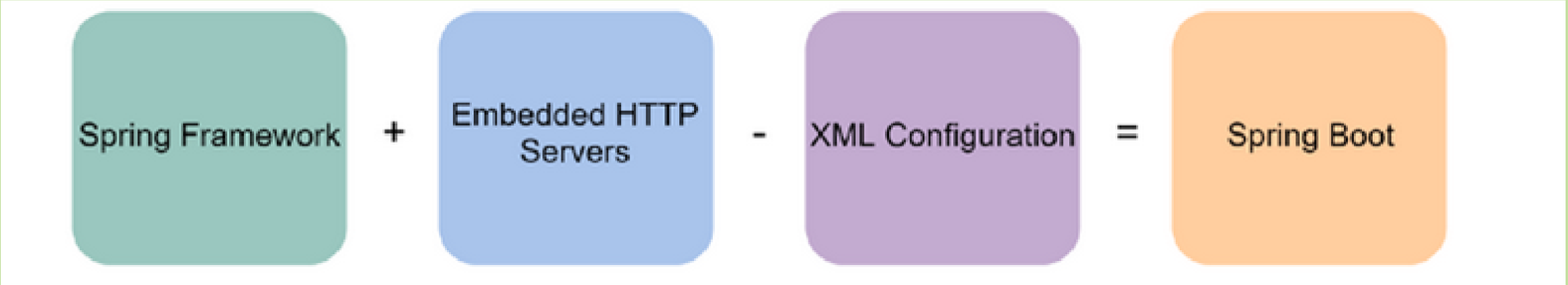
72). Что такое Spring Boot?

Spring Boot – это Java-фреймворк с открытым исходным кодом, используемый для создания микросервисов. Это проект, построенный на основе Spring, чтобы упростить задачу развертывания приложений Java. Его двумя основными компонентами являются Spring Framework и встроенные HTTP-серверы.

Spring Boot используется для:

- Упрощение процесса разработки готовых к производству пружинных приложений
- Избегания конфигурации XML Spring
- Сокращения времени разработки за счет уменьшения количества необходимых инструкций по импорту
- Обеспечения взвешенного подхода к развитию

Часто используются для быстрого запуска приложений Spring.



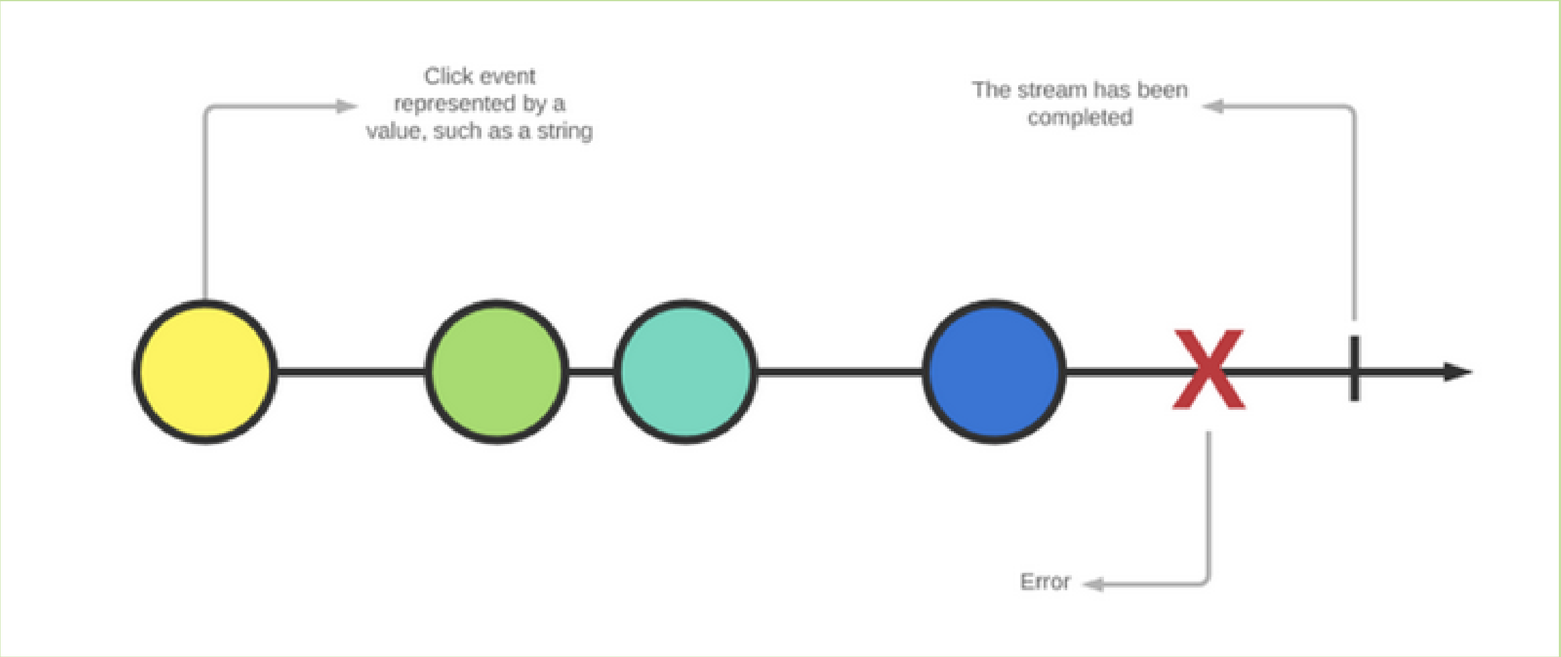
73). Как проще всего развернуть приложение Spring Boot в рабочей среде? Какие еще есть варианты?

Самый простой способ развернуть приложение Spring Boot — это поместить **.jar** файл со встроенным контейнером сервлетов на любой сервер или платформу, на которой установлена **JRE**.
По организационным и историческим причинам вы также можете развернуть приложение Spring Boot как файл **.war** в существующем контейнере сервлетов или сервере приложений.

И последнее, но не менее важное: вы, конечно, также можете поместить свой **.jar** файл в образ **Docker** и даже развернуть его с помощью **Kubernetes**.

74). Что такое Реактивное программирование?

Реактивное программирование – это парадигма программирования, которая основывается на запрограммированных действиях, запускаемых в связи с событиями, а не на хронологическом порядке кода. Реактивные программы эффективно используют компьютерные ресурсы и хорошо масштабируются всего несколькими потоками. Его непоследовательная форма позволяет избежать блокировки стека и поддерживать оперативность реагирования.



75). Что такое Spring webflux?

Webflux – это реактивный веб-фреймворк, который служит альтернативой MVC. Webflux обеспечивает лучшую масштабируемость и предотвращает блокировку стека.