Полный Java Junior Roadmap

📌 Здесь содержится подробный план того, что необходимо знать junior-middle разработчику на Java и подробный список вопросов для собеседований, к которым стоит готовиться

Всем привет! Это полный RoadMap на Junior-Middle Java разработчика. Если вы освоите все эти темы и качественно подготовитесь к собеседованиям, то сможете найти работу разработчиком достаточно легко.

А если хотите пройти быстрее и с обратной связью - обучение до оффера на Java разработчика - *sorokin.school*

Что здесь разбирается

- Что **не нужно учить**, если хотите как можно быстрее стать разработчиком
- Что стоит учить, какой **актуальный стек**

- Частые хард (**технические**) вопросы с собеседований
- Частые софт (**поведенческие**) вопросы с собеседований
- Большая **подборка материалов** для обучения и подготовки к собеседованиям

На что не тратить время точно

Важно не только изучать теоретические темы, но и применять знания на практике.Часто изучение множества технологий может сбивать с толку и затягивать процесс подготовки.

Советуем не тратить время на:

- Java Servlets, Java EE
- RabbitMQ (лучше Kafka)
- Thymeleaf или фронтенд (HTML, CSS) не стоит
- Лучше выбрать одно из Maven или Gradle
- увлекаться NoSQL БД (кроме Redis), S3, Minio
- Другие языки программирования, алгоритмами
- Вебсокеты, TCP, UDP, gRPC и прочие «ненужными» протоколами (редко используются)

Конечно, где-то это может пригодиться, но ваша задача — как можно быстрее войти в профессию, а не изучать всё подряд. Позже, когда вы будете искать работу или захотите углубиться в тему, сможете изучать дополнительные технологии с кайфом.

План изучения для Java Junior

Java Core

Учить в первую очередь. Это основа.

1. Переменные, циклы, условия, объекты, примитивы
2. Классы, Интерфейсы, принципы ООП
3. Работа с исключениями: try-catch-finally, try with resources
4. Коллекции в Java (List, Set, Map) и их основные реализации
5. Строки и их особенности в Java
6. Особенности Java 8+: Lambda выражения, Stream API
7. Memory Model, volatile, правила «happens-before» и атомарные операции (AtomicInteger)

Многопоточность

1. Поток, процесс, их отличия
2. Взаимодействие между потоками: wait-notify, notifyAll
3. Синхронизация: synchronized (методы, блоки, статические), монитор объекта
4. Executor Framework, пул потоков, разница между Thread и Runnable, а также Callable
5. Потокобезопасные коллекции (CopyOnWriteArrayList, ConcurrentHashMap и др.)
6. Race condition, deadlock, способы их предотвращения
7. Atomic типы и их применение для атомарных операций

Базы данных

1. Основы SQL: CREATE TABLE, ALTER TABLE
2. Запросы SQL: SELECT, INSERT, UPDATE, DELETE
3. Виды ограничений (constraints)

4. Нормализация баз данных (по желанию)
5. Уровни изоляции транзакций, принципы ACID
6. Индексы: типы (B-tree, hash), когда использовать, что они ускоряют и замедляют запросы
7. Обработка NULL, DISTINCT, GROUP BY, HAVING, LIMIT
8. Использование EXPLAIN (analyze) для оценки производительности

Spring Framework

1. Spring Core: инверсия контроля (IoC), внедрение зависимостей (DI), IoC контейнер, AOP
2. Spring WEB MVC: написание контроллеров, аннотации (@RequestMapping, @GetMapping, @PostMapping и др.)
3. Spring Data JPA / Hibernate: основы ORM, жизненный цикл сущности, кеширование, работа с EntityManager
4. Основы Spring Security: авторизация, аутентификация, конфигурация безопасности
5. Spring Boot: стартеры, автоматическая конфигурация
6. Отличия и особенности аннотаций (@Bean, @Component, @Autowired, @Configuration, @Qualifier)

Проектирование

1. Принципы SOLID
2. Паттерны проектирования: знание ключевых паттернов (Singleton, Proxy, Facade, Builder, Adapter, Strategy)
3. Микросервисы: что такое, зачем нужны, основные проблемы и типы масштабирования
4. Паттерны микросервисной архитектуры: Saga, API Gateway, Rate Limiter, Transactional Outbox

---

Kafka

1. Основы Kafka: назначение, принципы работы, архитектура (брокер, продюсер, консюмер)
2. Топики и партиционирование: структура топиков, разделение данных на партиции
3. Consumer Group: механизм работы групп потребителей и балансировка нагрузки
4. Producer API: основные методы, особенности отправки сообщений
5. Consumer API: чтение сообщений, управление offset, механизмы commit
6. Интеграция с Spring: использование Spring Kafka, базовые настройки и примеры использования

Тестирование

1. Виды тестирования, когда и зачем используются (интеграционное, unit)
2. JUnit, Mockito

Инструменты разработчика

1. IntelliJ IDEA
2. Git
3. Docker
4. Сборщики: выбор между Maven и Gradle

DevOps

1. Базовое понимание DevOps, проблемы, которые решает CI/CD
2. Jenkins или TeamCity – базовое понимание (прочитать 1-2 статьи, посмотреть видео)

Процессы разработки

1. Роли и обязанности в IT командах
2. Процессы разработки, доставки кода, CI/CD
3. Как происходит разработка, планирование, встречи команды, SCRUM

Популярные вопросы на собеседованиях

Вопросы по Java Core

ООП

1. Что такое ООП?
2. Что такое инкапсуляция, наследование, полиморфизм?
3. Что такое SOLID?

Типы данных

1. Какие примитивные типы данных есть в Java?
2. Что такое автоупаковка и автораспаковка?
3. Почему не рекомендуется изменять строки в цикле? Что рекомендуется использовать?
4. Что делает метод intern() в классе String?
5. Каким образом переменные передаются в методы, по значению или по ссылке?

ООП в Java

1. Какие виды классов есть в java?
2. Что такое «локальный класс»? Каковы его особенности?
3. Каким образом из вложенного класса получить доступ к полю внешнего класса?
4. Как проблема ромбовидного наследования решена в java?
5. Какие модификаторы доступа есть в Java? Какие применимы к классам?

6. Может ли статический метод быть переопределён или перегружен?
7. Можно ли сузить уровень доступа/тип возвращаемого значения при переопределении метода?
8. Могут ли классы быть статическими?
9. Что такое абстрактные классы? Чем они отличаются от обычных?
10. Могут ли быть конструкторы у абстрактных классов? Для чего они нужны?
11. Чем интерфейсы отличаются от абстрактных классов? В каких случаях следует использовать абстрактный класс, а в каких интерфейс?
12. Что такое дефолтные методы интерфейсов? Для чего они нужны?
13. Что такое класс Class?
14. Расскажите про equals и hashcode.
15. Зачем нужен equals(). Чем он отличается от операции ==?
16. Что будет, если переопределить equals() не переопределяя hashCode()? Какие могут возникнуть проблемы?
17. Для чего нужен метод hashCode()?
18. Есть ли какие-либо рекомендации о том, какие поля следует использовать при подсчете hashCode()?
19. Почему нельзя реализовать hashcode() который будет гарантированно уникальным для каждого объекта?

Исключения

1. Что такое исключения?
2. Расскажите про обрабатываемые и необрабатываемые исключения.
3. Какой оператор позволяет выбросить исключение?
4. Как создать собственное («пользовательское») исключение?
5. Расскажите про механизм обработки исключений в java (Try-catch-finally).
6. Может ли один блок catch отлавливать сразу несколько исключений?

7. Что такое механизм try-with-resources?
8. Что произойдет если исключение будет выброшено из блока catch после чего другое исключение будет выброшено из метода close() при использовании try-with-resources?

Память в Java

1. Что такое Heap и Stack память в Java? Чем они отличаются?
2. Что такое сборщик мусора? (Garbage collector)
3. Какие нюансы у строк в Java?

Дженерики

1. Что такое дженерики?
2. Для чего нужны дженерики? Параметр vs Аргумент.
3. Что такое wildcard?

Коллекции

1. Что такое «коллекция»?
2. Почему Map — это не Collection, в то время как List и Set являются Collection?
3. Как между собой связаны Iterable, Iterator и «for-each»?
4. Как удалить элемент из ArrayList при итерации?
5. Чем Set отличается от List?
6. Расскажите про реализации интерфейса Set.
7. В чем отличия TreeSet и HashSet?
8. Что будет, если добавлять элементы в TreeSet по возрастанию?
9. Как устроен HashSet, сложность основных операций.
10. Как устроен LinkedHashSet, сложность основных операций.
11. Как устроен ArrayList, сложность основных операций.

12. Почему LinkedList реализует и List, и Deque?
13. Что такое Queue?
14. Что такое Deque? Чем отличается от Queue?
15. Приведите пример реализации Deque.
16. Как работает HashMap при попытке сохранить в него два элемента по ключам с одинаковым hashCode(), но для которых equals() == false?

Функциональные интерфейсы

1. Что такое функциональный интерфейс?
2. Что такое лямбда-выражение? Чем его можно заменить? Как можно и как нельзя?

Stream API

1. Что такое Stream API? Для чего нужны стримы?
2. Какие существуют способы создания стрима?
3. Какие промежуточные методы в стримах вы знаете?
4. Методы терминальные.
5. Расскажите про класс Collectors и его методы.
6. Что такое IntStream и DoubleStream?

Вопросы по многопоточности

Основа

1. Что такое поток, процесс, их отличия?
2. Для чего используется synchronized? Как он работает?
3. Для чего нужен volatile и как он работает? Что гарантирует? Почему не гарантирует атомарность операции инкремента?
4. Чем Thread отличается от Runnable? Когда нужно использовать Thread, а когда Runnable?

5. Что такое синхронизация? Какие способы синхронизации существуют в java?
6. Расскажи про правила «happens-before» и Java Memory Model (JMM).
7. Чем Runnable отличается от Callable?
8. Что является монитором у статического синхронизированного класса?

Проблемы многопоточности

1. Что такое race condition?
2. Что такое deadlock и какие есть способы решения?

Продвинутые функции

1. Потокобезопасные коллекции: какую решают проблему и зачем использовать?
2. Что такое пул потоков, виды и их отличия?
3. Completable Future для чего используется?
4. Расскажи про устройство и алгоритм работы ConcurrentHashMap

Вопросы Spring

Spring Core

1. Что такое инверсия контроля (IoC) и внедрение зависимостей (DI)?
2. Что такое IoC контейнер?
3. Что такое Bean в спринге?
4. Чем отличаются аннотации @Bean и @Component?
5. Расскажите про аннотацию @Autowired
6. Расскажите про аннотации @Primary и @Qualifier
7. Расскажите про скоупы бинов? Какой скоуп используется по умолчанию?
8. Что такое ApplicationContext в Spring и для чего он используется?

9. Какую проблему решает Spring с помощью внедрения зависимостей (Dependency Injection)?
10. За что отвечают аннотации @Component, @Service, и @Repository в Spring?
11. За что отвечает аннотация @Configuration?
12. В чем разница между созданием бинов с помощью аннотации @Bean внутри класса с аннотацией @Configuration и автоматическим сканированием компонентов с помощью @ComponentScan?
13. В чем разница между scope Singleton и Prototype для Spring бинов?
14. Как можно задать приоритет бинов, если на одно и то же autowired-поле подходит несколько бинов?
15. Что такое AOP? Как реализовано в спринге?

Spring Boot

1. Для чего нужен Spring Boot?
2. Что такое стартеры Spring Boot?
3. Как Spring Boot упрощает работу?

Spring Web

1. Как написать контроллер-обработчик http запросов с помощью spring mvc?
2. Как сделать обработку различных типов запросов (GET, POST, PUT ...)?
3. За что отвечают аннотации @RequestBody, @PathVariable, @RequestMapping, @GetMapping, @PostMapping...
4. Как обработать исключение, которое было выборшено за пределы контроллера?
5. Что такое REST и RESTfull сервисы?

Data JPA

1. Что за аннотация @Transactional в Spring?

2. Какие есть аргументы у @Transactional и для чего они используются?
3. Опишите различия и особенности типов отношений между сущностями: Many-To-One, Many-To-Many, One-To-Many.

Spring Security

1. Что такое авторизация и аутентификация?
2. Как работает Spring Security? Как сконфигурировать? Какие интерфейсы используются?
3. Как работает и что такое JWT?

Вопросы по паттернам

1. Что такое SOLID? За что отвечает каждая буква?
2. Объясните принцип работы Proxy, какими способами можно сделать Proxy?
3. Назовите три основные группы паттернов (классификация).
4. Для чего используется паттерн Strategy?
5. Для чего используются паттерны: Prototype, Decorator, Chain of Responsibility, iterator, template method?

Вопросы по hibernate

1. Что такое ORM?
2. Что появилось раньше JPA или Hibernate?
3. Какие ключевые интерфейсы использует Hibernate?
4. Как работать с кэшем второго уровня?
5. Что такое EntityManager?
6. Какие функции он выполняет?
7. Расскажите про жизненный цикл сущности Entity, перечисли четыре статуса Entity объекта (Entity Instance’s Life Cycle).

8. Может ли абстрактный класс быть Entity?
9. Может ли Entity класс наследоваться от других Entity классов?
10. Какие стратегии маппинга иерархии наследования (Inheritance Mapping Strategies) описаны в JPA?
11. Что такое каскадные операции?
12. Для чего нужна аннотация Column?
13. Какие @GeneratedValue вы знаете?
14. Для чего нужна аннотация Transient?
15. Что такое Criteria API и для чего он используется?
16. Расскажите про проблему N+1 Select и путях ее решения.

Вопросы по SQL и БД

Основа SQL

1. Что такое SQL?
2. Какие бывают связи таблиц SQL?
3. Что такое DDL? Какие операции в него входят? Рассказать про них.
4. Нюансы работы с NULL в SQL. Как проверить поле на NULL?
5. Виды Join’ов, left, right, inner.
6. Что делает UNION?
7. Что такое GROUP BY?
8. Что такое DISTINCT? Что такое LIMIT?
9. Расскажите про операторы IN, BETWEEN, LIKE.
10. Какие агрегатные/агрегирующие функции вы знаете?

Транзакции и ACID

1. Что такое транзакции? Расскажите про принципы ACID.
2. Какие аномалии встречаются и на каких уровнях они решаются?

3. Какие есть уровни изоляций транзакций?

Оптимизация запросов

1. Анализ выполнения запросов при помощи Explain (analyze).
2. Какие типы индексов существуют и в каких случаях каждый из них эффективен?
3. Когда использовать индексы? Что они ускоряют, что замедляют?

Доп. темы

1. Какие есть виды масштабирования БД?
2. Что такое партиции и для чего они используются?
3. Что такое репликация и шардирование БД, для чего используются?

Вопросы по архитектуре

Архитектура

1. Зачем нужны микросервисы?
2. Какие есть проблемы микросервисов?
3. Какие плюсы и минусы монолитов?
4. Паттерны микросервисной архитектуры: Saga, Api Gateway, Rate Limiter, Transactional Outbox.
5. Какие есть типы масштабирования? Когда какой использовать?

Взаимодействие сервисов

1. Типы взаимодействий сервисов: асинхронное и синхронное. Когда какой предпочтительнее?
2. HTTP протокол для чего используется? Какие есть методы в HTTP протоколе?
3. Что такое REST и RESTfull? PUT/POST/PATCH - в чем отличие?
4. Как устроен топик в kafka? На какие части внутри он делится?

5. Как сделать запросы идемпотентными?

Софт вопросы (поведенческие)

Их на самом деле могут задать очень много разных. Но тут прям база, на которую надо уметь ответить всегда.

1. Расскажите про себя, опыт работы
2. Почему уходите из компании текущей?
3. Что ищете на новом месте работы?
4. В какой компании работал? Почему решил уходить?
5. Кто был в команде?
6. Что за проект вы реализовывали, стек проекта?
7. Процессы разработки в команде?
8. Как выпускались релизы?
9. Как происходит процесс планирования и назначения задач?
10. Пример задачи, которую ты решил хорошо?
11. Пример задачи, где у тебя случился провал?
12. Что делать если коллега на ПР не исправляет и не делает так, как ты ему говоришь, не воспринимает критику?
13. Как решаешь конфликты на работе?
14. С кем взаимодействовал внутри команды?

Материалы для обучения и подготовки

✅ Здесь содержатся дополнительные материалы для подготовки к реальным собеседованиям

Java Core и Многопоточность

1. видео - разбор всех частых вопросов по Java Core и Многопоточности
2. доклад - Разбираем Garbage Collector в Java
3. статья - Избавляемся от мусора в Java

Многопоточность

- статья - основные концепции параллелизма
- Лекция №1 - рассматриваются основы многопоточности. Как работать с потоками, средства синхронизации между потоками, ключевое слово synchronized, семантика happens-before.
- Лекция №2 - рассматриваются продвинутые средства синхронизации в Java: atomic типы для безопасных операций с переменными без блокировок, ExecutorService для управления пулами потоков и эффективного выполнения асинхронных задач, ReentrantLock для гибкой блокировки с возможностью повторного входа.
- Многопоточность глазами разработчика - для закрепления материала советую после двух лекций выше посмотреть видео на 1.5 часа по всем важным темам многопоточности
- серия лекций по многопоточности- лекция-1, лекция-2, лекция-3
- статья - Synchronized Keyword Guide
- статья - Проблема Deadlock и методы борьбы с ней

Базы данных

- видео - ACID и транзакции в БД
- статья - Подробная шпаргалка по SQL
- Видео обзор нормализации БД - небольшое видео про основы нормализации (15 мин)
- Статья о нормализации на Хабр - разбираются все формы нормализации. Полезно знать первые 3-4, дальше уже может быть тяжело и на практике/ на собеседованиях редко спрашиваютф
- Курс по основам SQL.
- Статья про основы изоляций транзакций
- Подробная шпаргалка по SQL
- Устройство PostgreSQL внутри - изоляция и многоверсионность
- Статья про индексы - глубокое погружение в индексы PostgreSQL
- Очень подробная статья про уровни изоляции транзакций с примерами
- Подробное практическое видео про @Transactional
- Подробно о том, как работает propogation @Transactional
- Продвинутая лекция об индексах - подробный разбор основных ошибок, проблем при их использовании.
- Колоночные СУБД основы - как работают и зачем нужны

Оптимизация SQL запросов

- Практические примеры оптимизации запросов доклад
- видео - индексы в БД разбор с примерами
- производительность в postgeSQL

Spring

Для собесов:

- вопросы по Spring на собеседованиях - видео на 1 час о всех частых вопросах https://www.youtube.com/watch?v=_7riaBlOxbM
- 75 вопросов и ответов по Spring https://www.youtube.com/watch?v=WsXECfqVaBo
- Жизненный цикл статья https://habr.com/ru/articles/720794/
- Жизненный цикл видео https://youtu.be/MR_h8kljt-w?si=NLHw9JedlD0hhkgo

Очень советую все доклады Евгения Борисова. Он очень глубоко копает про Spring, объясняет понятно и с примерами. Это уже для более продвинутых ребят, кто в спринге разбирается. С самого нуля лучше не смотреть.

Доклады Евгения Борисова

- Spring Patterns https://www.youtube.com/watch?v=61duchvKl6o&
- Spring Patterns для взрослых https://www.youtube.com/watch?v=GL1txFxswHA
- Spring – Глубоко и не очень https://www.youtube.com/watch?v=nGfeSo52_8A
- Тут Евгений пишет за 2 часа свой Spring - видос топ для тех, кто хочет понять как же спринг работает внутри

Spring data JPA

- Основы @Transactional - видео про работу с транзакциями с помощью аннотации @Transactional
- Статья о @Transactional - детальное рассмотрение работы аннотации @Transactional в контексте Spring Data JPA (аккуратно читаем, может быть

тяжело)

- Статья про связи между сущностями - рассматриваются все типы связей между сущностями: `@ManyToOne`, `@OneToMany`, `@ManyToMany`

Spring Boot

- Введение в Spring Boot MVC (Видео) - обзор основных концепций и практическое создание простого веб-приложения
- Валидация входящих запросов - статья по валидации данных, которые принимает контроллер. В этой статье можно (нужно) пропустить последние темы про: написание собственного валидатора и групп валидации (в нашем случае это пока излишне)
- Статья по обработке ошибок приложения - о том, как обрабатывать ошибки и возвращать пользователю понятный ответ, который мы хотим

Взаимодействие между сервисами

- Что такое REST? (Видео) - концепции и принципы RESTful веб-сервисов.
- Статья о различии методов PUT, POST, PATCH
- Подробная статья о REST - советую вам обратиться к этой статье только в крайнем случае, достаточно глубокая
- Kafka Лекция бизарованная - покрывает почти все, что нужно. С первого раза будет тяжело все осознать, пересматривайте

Security

- Статья про JWT токен - что такое JWT токены и как они устроены, для чего используются. Разбираются также access и refresh токены
- Еще про JWT статья
- Еще про JWT статья
- Видео-лекция JWT на Spring Boot 3 [на английском]

SOLID

SOLID - набор принципов для написания качественного кода, их обязательно нужно знать каждому

1. Статья про SOLID - основы о SOLID
2. Доклад SOLIDный код - рассматриваются примеры написания кода

Паттерны

- Статья что такое паттерны - основы про паттерны и для чего они нужны
- Strategy - позволяет менять поведение объекта за счет разных стратегий поведения
- Facade - оборачивает сложную логику работы нескольких классов за простым интерфейсов взаимодействия
- Builder - используется для построения сложных объектов по частям
- Proxy - является оберткой для объекта, которая добавляет какую-то промежуточную дополнительную логику
- Adapter - помогает привести один интерфейс взаимодействия к другому, который нужен нам

Мок-собесы

Полезно посмотреть для подготовки к вопросам:

- видео - Java Middle собес
- видео - Java mock-interview
- видео - Senior mock-interview
- видео - mock собеседование

Архитектура

Микросервисы

- видео - Что спрашивают на собеседованиях
- видео - Работа с данными в микросервисах
- Объяснение микросервисов - видео базовое про основные принципы и паттерны
- Преимущества и недостатки микросервисов - небольшая статья про плюсы и минусы микросервисного подхода
- Паттерны проектирования микросервисов через статьи: часть 1 и часть 2
- Паттерн Saga - лекция как бороться с отсутствием консистентности в микросервисной архитектуре

System Design

- Основы за 30 минут (лекция) - базово о том что такое system design
- Теория (доклад) - методы и приемы

System design interview

- Как подготовиться и пройти (доклад)
- Доклад как построить распределенную систему
- Интервью для примера - как проходит system design interview

Senior собесы

Вопросы с senior собесов

- часть 1
- часть 2
- часть 3

Мок-собес - На грани middle/senior

🔥 **Полное обучение до ОФФЕРА на Java разработчика - *sorokin.school***