ЦЕНТРАЛЬНЫЙ УНИВЕРСИТЕТ

ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ

Сертификат: 025053D20009B2A5B043583E315FEF31AD
Владелец: Ивашкевич Евгений Васильевич
Действителен: с 14.10.2024 до 14.01.2026

УТВЕРЖДЕНА

Приказом Ректора АНО ВО
«Центральный университет»
Ивашкевич Е.В.
от «19» января 2024 г. № 0119.37

Рабочая программа дисциплины (модуля) «Разработка на языке программирования Kotlin» дополнительной профессиональной программы – программы профессиональной переподготовки «Академия data science»

Траектория: Backend-разработка

Москва
2024

Электронный документ

Содержание

1. Краткая характеристика дисциплины (модуля)

Изучение Kotlin как современного языка программирования открывает перед слушателями возможности создания высокопроизводительных и безопасных приложений, а также способствует улучшению качества кода благодаря лаконичности и выразительности синтаксиса. Кроме того, знание Kotlin расширяет профессиональные горизонты выпускников, так как этот язык активно используется в разработке на платформе Android и в серверной разработке, что делает их более конкурентоспособными на рынке труда.

Цель изучения дисциплины (модуля): формирование у слушателей глубоких знаний и практических навыков разработки серверных приложений с использованием языка Kotlin.

Задачи изучения дисциплины (модуля):

— формирование знаний синтаксиса и идиом Kotlin;

— формирование знаний асинхронности через корутины;

— формирование умения разрабатывать гибкую и надёжную бизнес-логику, используя инструменты Kotlin;

— формирование умения применять асинхронное и многопоточное программирование с корутинами;

— формирование умения реализовывать асинхронный транспорт с применением WebSocket, RabbitMQ и Kafka;

— формирование навыка использовать язык Kotlin для разработки серверных приложений;

— формирование навыка использовать Kotlin вместе со Spring и Ktor.

2. Тематический план

№ п/п	Наименование раздела дисциплины (модуля)	Трудоемкость, академические часы				ТКУ (текущий контроль успеваемости)
		Очная форма				
		Аудиторная работа		Контроль	Самостоятельная работа	
		Лекции	Семинары (практические занятия)			
1	Основы и идиоматический Kotlin	2	7		37	Домашнее задание
2	Серверные паттерны и асинхронность	2	7		37	Домашнее задание
3	Работа с данными и инфраструктурой	2			37	Домашнее задание Тест
4	Продвинутые темы	3	8		37	Домашнее задание Проект
	Зачет с оценкой			4		
	Итого:	***9***	***29***	***4***	***148***	
Объем дисциплины (модуля) (в ак. ч.)		***190***				

3. Содержание дисциплины (модуля)

№п/п	Наименование раздела дисциплины (модуля)	Содержание дисциплины (модуля) по темам
1	Основы и идиоматический Kotlin	Базовые конструкции и null-безопасность. Функциональные идиомы. Расширения и делегирование
2	Серверные паттерны и асинхронность	ООП и модели данных. Коллекции и производительность. Корутины и Flow
3	Работа с данными и инфраструктурой	Работа с БД. Веб-фреймворки (Ktor/Spring Boot). Тестирование и MockK. Безопасность и оптимизация
4	Продвинутые темы	Миграция Java → Kotlin. Kotlin в микросервисах. Финальный проект

4. Учебно-методическое обеспечение

Университет располагает полным набором лицензионного и свободно распространяемого программного обеспечения, включая продукты отечественного производства.

Каждый слушатель в течение всего периода обучения получает индивидуальный неограниченный доступ к электронно-библиотечной системе и электронной информационно-образовательной среде университета. Эти системы предоставляют возможность доступа к ресурсам из любой точки, где есть подключение к сети Интернет, как на территории университета, так и за его пределами.

Слушателям обеспечен удаленный доступ к современным профессиональным базам данных и информационным справочным системам.

Основная литература:

1. Чернышев, С. А. Принципы, паттерны и методологии разработки программного обеспечения : учебник для вузов / С. А. Чернышев. — Москва : Издательство Юрайт, 2025. — 176 с. — (Высшее образование). — ISBN 978-5-534-14383-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/567946.

Дополнительная литература:

1. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. — СПб.: Питер, 2022. — 1120 с. — ISBN 978-5-4461-1155-8.

5. Материально-техническое обеспечение

Университет располагает материально-технической базой, соответствующей действующим противопожарным правилам и нормам и обеспечивающей проведение всех видов дисциплинарной и междисциплинарной подготовки, практической и научно-исследовательской работ обучающихся, предусмотренных учебным планом.

Помещения, которые представляют собой учебные аудитории для проведения занятий лекционного типа, занятий семинарского (практического) типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, а также помещения для самостоятельной работы и помещения для хранения и профилактического обслуживания учебного оборудования. Помещения укомплектованы специализированной мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории.

Изучение дисциплины (модуля) обеспечивается в учебных аудиториях, оснащенных:

— столами и стульями;

— компьютерной техникой;

— механическими калькуляторами;

— специализированным оборудованием, включая демонстрационное оборудование.

Помещения для самостоятельной работы обучающихся, в том числе приспособленные для использования инвалидами и лицами с ограниченными возможностями здоровья, оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду Университета.

Обучающимся предоставляется доступ (в том числе удаленный) к ресурсам информационно-телекоммуникационной сети «Интернет», электронным ресурсам (в том числе электронным библиотечным системам, современным профессиональным базам данных и информационным справочным системам):

№	Наименование портала (издания, курса, документа)	Ссылка
1.	Научная электронная библиотека elibrary.ru библиотека	https://elibrary.ru/defaultx.asp
2.	База данных для IT-специалистов	https://habr.com
3.	База данных ScienceDirect	https://www.sciencedirect.com
4.	Официальный сайт Министерства науки и высшего образования Российской Федерации	https://minobrnauki.gov.ru/
5.	Федеральный портал «Российское образование"	https://www.edu.ru/
6.	Информационная система "Единое окно доступа к образовательным ресурсам"	http://window.edu.ru/
7.	Единая коллекция цифровых образовательных ресурсов	http://school-collection.edu.ru/
8.	Федеральный центр информационно - образовательных ресурсов	http://fcior.edu.ru/

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), в том числе комплект лицензионного программного обеспечения, современные профессиональные базы данных и информационные справочные системы:

Наименование ПО	Производство	Лицензионное / свободно распространяемое
Операционные системы:		
Microsoft Imagine (Windows Client, Server)	зарубежное	лицензионное
Браузеры:		
Яндекс.Браузер	отечественное	свободно распространяемое
Google Chrome	зарубежное	свободно распространяемое
Офисные приложения:		
Microsoft Imagine (Visio, OneNote)	зарубежное	лицензионное
TeXstudio	зарубежное	свободно распространяемое
Adobe Acrobat Reader	зарубежное	свободно распространяемое
Программное обеспечение для планирования и учета времени:		
Toggle app	зарубежное	свободно распространяемое
Системы управления проектами:		
Microsoft Imagine (Project)	зарубежное	лицензионное
Системы управления базами данных:		
Microsoft Imagine (SQL Server)	зарубежное	лицензионное
Системы резервного копирования (backup):		
Acronis Backup Advanced for HyperV	зарубежное	лицензионное
Справочно-правовые системы:		
КонсультантПлюс: справочно-правовая система	отечественное	лицензионное
Средства антивирусной защиты:		
Kaspersky Endpoint Security для бизнеса Стандартный Russian Edition	отечественное	лицензионное
Среды разработки:		
Visual Studio Code	зарубежное	свободно распространяемое
Bash (Unix shell)	зарубежное	свободно распространяемое
Anaconda	зарубежное	свободно распространяемое
Robotic Operating System	зарубежное	свободно распространяемое
CopelliaSim	зарубежное	свободно распространяемое

Google Colaboratory	зарубежное	свободно распространяемое
Пакеты программных средств и библиотек:		
AutoPsy	зарубежное	свободно распространяемое
Interactive Disassembler (IDA)	зарубежное	свободно распространяемое
Системы управления библиографической информацией:		
Zotero	зарубежное	свободно распространяемое
Сервисы и службы:		
Bind	зарубежное	свободно распространяемое
Docker	зарубежное	свободно распространяемое

6. Методические и оценочные материалы

Методические указания для обучающихся по освоению дисциплины (модуля)

В процессе изучения дисциплины (модуля) «Разработка на языке программирования Kotlin» в рамках текущего контроля успеваемости используются такие виды учебной работы, как лекции, семинары, тест, проект, домашние задания, а также различные виды самостоятельной работы обучающихся по заданию преподавателя, направленные на развитие навыков профессиональной лексики, закрепление практических профессиональных компетенций, поощрение инициатив.

Лекция – систематическое, последовательное, монологическое изложение преподавателем учебного материала, как правило, теоретического характера.

В процессе лекций рекомендуется вести конспект лекций: кратко и схематично фиксировать основные идеи, выводы и обобщения лекции; выделять важные мысли, ключевые слова и термины. Необходимо отметить вопросы или материалы, которые вызывают затруднения, и попытаться найти ответы в рекомендованной литературе. Если разобраться в материале не удается, следует сформулировать вопрос и задать его преподавателю на консультации или во время семинарского (практического) занятия.

Участие в семинаре (аудиторная работа) – активная работа слушателя на семинаре, его ответы на вопросы преподавателя и участие в дискуссии.

Для успешного участия в семинаре слушателям рекомендуется заранее ознакомиться с темой обсуждения, прочитать необходимые материалы и подготовить вопросы. Важно активно слушать и вовлекаться в дискуссию, высказывая свои мнения и аргументируя их. При ответах на вопросы преподавателя стоит быть уверенным, четким и логичным, опираясь на изученный материал. Также полезно поддерживать диалог с однокурсниками, чтобы обогатить обсуждение и расширить свои знания.

Проект – исследовательская работа по курсу и презентация результатов.

Для успешной подготовки к проекту: четко определите цели и задачи проекта, распределите роли и обязанности между участниками, а также установите сроки выполнения каждой части работы. Регулярно проводите встречи для обсуждения прогресса и решения возникающих вопросов.

Тест – особая форма проверки знаний. Проводится после освоения одной или нескольких тем и свидетельствует о качестве понимания основных понятий изучаемого материала. Тестовые задания составлены к ключевым понятиям, основным разделам, важным терминологическим категориям изучаемой дисциплины (модуля).

Для подготовки к тесту необходимо знать терминологический аппарат дисциплины (модуля), понимать смысл научных категорий и уметь их использовать в профессиональной лексике. Владение понятийным аппаратом, включённым в тестовые задания, позволяет преподавателю быстро проверить уровень понимания слушателями важных методологических категорий.

Домашнее задание – набор задач по темам недели.

При работе над домашними заданиями важно внимательно ознакомиться с требованиями и сроками выполнения. Рекомендуется разбивать задания на этапы, чтобы избежать перегрузки и лучше усвоить материал. Использовать различные источники информации, включая учебники и онлайн-ресурсы, для более глубокого понимания темы.

Самостоятельная работа – работа слушателей, направленная на углубленное изучение отдельных тем и вопросов учебной дисциплины (модуля).

В процессе самостоятельной работы слушатели взаимодействуют с рекомендованными материалами при минимальном участии преподавателя. Задачи слушателя включают работу с конспектами лекций (обработка текста), повторное изучение учебных материалов планов и тезисов ответов, изучение дополнительных тем, выполнение учебно-исследовательских заданий и другое.

Система оценивания результатов обучения по дисциплине (модулю)

Оценивание уровня учебных достижений обучающихся по дисциплине (модулю) осуществляется в виде текущего контроля успеваемости и промежуточной аттестации.

Промежуточная аттестация по дисциплине (модулю) осуществляется в форме ***зачета с оценкой***.

Для оценивания текущего контроля успеваемости и промежуточной аттестации используется десятибалльная шкала оценивания, которая соотносится с традиционной пятибалльной шкалой следующим образом:

Десятибалльная оценка	**Пятибалльная оценка**	**Оценка за зачет**	**Общая характеристика результата обучения по дисциплине (модулю)**
10	Отлично	Зачтено	Слушатель полностью владеет знаниями, изложенными в рабочей программе, и глубоко осмысляет дисциплину (модуль). Он самостоятельно и логически последовательно отвечает на все вопросы, акцентируя внимание на наиболее важном. Умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя ключевые моменты и устанавливая причинно-следственные связи. Четко формулирует ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные задачи. Слушатель хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты дисциплины (модуля) с практическими задачами.
9	Отлично	Зачтено	
8	Отлично	Зачтено	
7	Хорошо	Зачтено	Слушатель обладает знаниями предмета почти в полном объеме рабочей программы и самостоятельно, логически последовательно и всесторонне отвечает
6	Хорошо	Зачтено	

Десятибалльная оценка	Пятибалльная оценка	Оценка за зачет	Общая характеристика результата обучения по дисциплине (модулю)
			на все вопросы, акцентируя внимание на наиболее значимых моментах. Он умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя его ключевые аспекты и устанавливая причинно-следственные связи. Формулирует свои ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные ситуационные задачи. Слушатель хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты предмета с практическими задачами.
5	Удовлетворительно	Зачтено	Слушатель обладает базовыми знаниями по дисциплине (модулю), но испытывает трудности при самостоятельных ответах и использует неточные формулировки. В ходе ответов он допускает ошибки, касающиеся сути вопросов. Слушатель способен решать только самые простые задачи и владеет лишь минимальным набором методов исследования.
4	Удовлетворительно	Зачтено	
3	Не сдан	Не зачтено	Слушатель не овладел обязательным минимумом знаний по предмету и не может ответить на вопросы, даже если преподаватель задает дополнительные наводящие вопросы.
2	Не сдан	Не зачтено	
1	Не сдан	Не зачтено	

Дисциплина (модуль) «Разработка на языке программирования Kotlin» оценивается следующим образом:

Активность	Вес	Описание
Домашние задания	50%	За каждое из заданий можно набрать 10 баллов
Тест	10%	Ответы на вопросы, по изученным темам
Проект	40%	Исследовательская работа по дисциплине (модулю) и презентация результатов

Формула расчёта итоговой оценки по дисциплине (модулю) «Разработка на языке программирования Kotlin»: «0,5 × среднее за домашние задания + 0,1 × тест + 0,4 × проект».

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные домашние задания

Домашнее задание: Основы и идиоматический Kotlin

1. Напишите функцию, которая принимает строку и возвращает ее длину. Используйте null-безопасность, чтобы обработать случай, когда строка может быть null.

- **Пример:** `fun getStringLength(input: String?): Int`

2. Создайте класс **Person** с полями **name** и **age**. Реализуйте метод, который возвращает строку в формате "Имя: [name], Возраст: [age]".

- **Пример:** `class Person(val name: String, val age: Int)`

3. Используйте функциональные идиомы Kotlin, чтобы написать функцию, которая принимает список чисел и возвращает их сумму, используя **fold**.

- **Пример:** `fun sumNumbers(numbers: List<Int>): Int`

4. Реализуйте расширение для класса **String**, которое будет возвращать строку в верхнем регистре, если она не пустая, и "Пустая строка" в противном случае.

- **Пример:** `fun String.toUpperOrEmpty(): String`

5. Создайте класс **DelegateExample**, который будет использовать делегирование для хранения значения. Реализуйте делегат, который будет выводить значение при его изменении.

- **Пример:** `var delegatedProperty: String by Delegates.observable("initial") { _, old, new -> println("Changed from $old to $new") }`

Домашнее задание: Серверные паттерны и асинхронность

1. Определите интерфейс **DataModel**, который будет содержать методы для получения и сохранения данных. Реализуйте класс **User** , который будет реализовывать этот интерфейс.

- **Пример:** `interface DataModel { fun save(): Boolean fun load(): DataModel }`

2. Создайте класс **User Repository**, который будет использовать коллекцию для хранения объектов **User** . Реализуйте методы для добавления и получения пользователей.

- **Пример:** `class UserRepository { private val users = mutableListOf<User>() }`

3. Напишите функцию, которая использует корутины для асинхронного получения данных из сети. Используйте **withContext** для переключения контекста.

- **Пример:** `suspend fun fetchData(): String`

4. Создайте класс **FlowExample**, который будет использовать **Flow** для генерации последовательности чисел от 1 до 10. Реализуйте метод, который будет выводить эти числа.

- **Пример:** `fun numberFlow(): Flow<Int>`

5. Опишите, как реализовать паттерн "Singleton" в Kotlin, и создайте класс **DatabaseConnection**, который будет являться синглтоном.

- **Пример:** `object DatabaseConnection { /* реализация */ }`

Домашнее задание: Работа с данными и инфраструктурой

1. Создайте класс **Database**, который будет содержать методы для подключения к базе данных и выполнения SQL-запросов. Реализуйте метод для получения всех пользователей.

- **Пример:** `class Database { fun connect() { /* реализация */ } }`

2. Напишите простое приложение на Ktor, которое будет обрабатывать HTTP GET запросы и возвращать список пользователей в формате JSON.

- **Пример:** `fun Application.module() { routing { get("/users") { /* реализация */ } } }`

3. Создайте тест для класса **Database**, используя MockK для имитации работы с базой данных. Реализуйте тест на получение данных.

- **Пример:** `@Test fun testGetUsers() { /* реализация */ }`

4. Опишите, как реализовать безопасность в приложении на Spring Boot, используя аннотации для защиты эндпоинтов.

- **Пример:** `@PreAuthorize("hasRole('ADMIN')")`

5. Напишите функцию, которая будет оптимизировать запросы к базе данных, используя индексы. Объясните, как индексы влияют на производительность.

- **Пример:** `fun optimizeQuery(query: String) { /* реализация */ }`

Примерные тестовые задания

Тест: Работа с БД и веб-фреймворками

1. **Какой из следующих фреймворков используется для создания веб-приложений на Kotlin?**

A) Django
B) Spring Boot
C) Laravel
D) Express

2. **Какой аннотацией в Spring Boot можно защитить метод, чтобы только пользователи с ролью "ADMIN" могли его вызывать?**

A) @Secured
B) @RolesAllowed
C) @PreAuthorize
D) @PermitAll

3. **Что такое ORM?**

A) Object-Relational Mapping
B) Online Resource Management
C) Object-Resource Management
D) Open Resource Model

4. **Какой из следующих методов используется для выполнения SQL-запросов в Spring Data JPA?**

A) executeQuery()
B) findAll()
C) getResult()
D) selectAll()

5. **Какой из следующих компонентов Ktor отвечает за маршрутизацию запросов?**

A) Application
B) Routing
C) Pipeline
D) Engine

6. **Какой метод используется для создания теста с использованием MockK?**

A) mock()
B) createMock()
C) newMock()
D) buildMock()

7. **Что такое "Dependency Injection"?**

A) Метод управления состоянием приложения
B) Метод для создания тестов

C) Метод, позволяющий внедрять зависимости в классы
D) Метод для работы с базами данных

8. **Какой из следующих подходов используется для тестирования контроллеров в Spring Boot?**
A) Unit Testing
B) Integration Testing
C) Functional Testing
D) All of the above

9. **Какой из следующих методов используется для настройки соединения с базой данных в Spring Boot?**
A) application.properties
B) config.xml
C) database.yml
D) settings.json

10. **Какой из следующих инструментов используется для миграции схемы базы данных в Spring Boot?**
A) Flyway
B) Liquibase
C) Hibernate
D) A и B

11. **Что такое "SQL Injection"?**
A) Метод оптимизации запросов
B) Уязвимость, позволяющая злоумышленнику выполнять произвольные SQL-запросы
C) Способ защиты базы данных
D) Метод работы с транзакциями

12. **Какой из следующих методов Ktor используется для обработки HTTP POST запросов?**
A) post()
B) get()
C) request()
D) handle()

13. **Какой аннотацией в Spring Boot можно пометить метод, который будет обрабатывать все исключения?**
A) @ControllerAdvice
B) @ExceptionHandler
C) @RestController
D) @GlobalExceptionHandler

14. **Какой из следующих инструментов используется для создания моков в тестах на Kotlin?**
A) Mockito
B) JUnit
C) MockK
D) EasyMock

15. **Что такое "Rate Limiting"?**
A) Метод оптимизации производительности базы данных

B) Метод ограничения числа запросов к API в единицу времени
C) Метод защиты от SQL Injection
D) Метод управления сессиями пользователей

Примерное задание для проекта

Тема проекта:
Миграция существующего Java-приложения на Kotlin с использованием микросервисной архитектуры.

Требования к проекту:

1. **Исходный код:** Необходимо выбрать существующее Java-приложение (например, простую систему управления задачами или интернет-магазин) и предоставить его исходный код.
2. **Микросервисы:** Приложение должно быть переработано в микросервисную архитектуру, где каждый сервис отвечает за отдельную функциональность (например, пользовательский сервис, сервис задач, сервис оплаты).
3. **Kotlin:** Все микросервисы должны быть переписаны на Kotlin, включая использование Kotlin Coroutines для асинхронного программирования.
4. **Документация:** Проект должен содержать документацию, описывающую архитектуру, использованные технологии и процесс миграции.
5. **Тестирование:** Каждый микросервис должен иметь набор юнит-тестов и интеграционных тестов, написанных с использованием Kotlin и подходящих библиотек (например, JUnit, MockK).
6. **CI/CD:** Настройка непрерывной интеграции и развертывания (CI/CD) для автоматизации тестирования и развертывания микросервисов.

Задание:

1. Выберите существующее Java-приложение, которое вы будете мигрировать на Kotlin.
2. Проанализируйте архитектуру приложения и определите, как его можно разбить на микросервисы.
3. Перепишите каждый сервис на Kotlin, используя соответствующие библиотеки и фреймворки (например, Ktor, Spring Boot).
4. Реализуйте асинхронное программирование с использованием Kotlin Coroutines.
5. Напишите тесты для каждого микросервиса и настройте CI/CD для автоматизации процессов.
6. Подготовьте документацию о процессе миграции и архитектуре нового приложения.

Этапы выполнения:

1. **Этап 1: Исследование и планирование**
   - Выбор приложения для миграции.
   - Анализ текущей архитектуры и определение микросервисов.

2. **Этап 2: Миграция первого микросервиса**
   - Переписывание первого микросервиса на Kotlin.

- Реализация тестов и настройка CI/CD.

3. **Этап 3: Миграция остальных микросервисов**
   - Переписывание оставшихся микросервисов.
   - Написание тестов и настройка CI/CD для каждого сервиса.

4. **Этап 4: Документация и финальная проверка**
   - Подготовка документации.
   - Защита проекта перед аудиторией.

Критерии оценивания:

1. **Качество кода (30%)**
   - Чистота и читаемость кода.
   - Следование стандартам Kotlin и принципам SOLID.

2. **Функциональность (30%)**
   - Полнота функционала, реализованного в микросервисах.
   - Корректность работы всех сервисов.

3. **Тестирование (20%)**
   - Наличие и полнота тестов.
   - Уровень покрытия кода тестами.

4. **Документация (10%)**
   - Полнота и ясность документации.
   - Описание архитектуры и процесса миграции.

5. **CI/CD (10%)**
   - Настройка автоматизации тестирования и развертывания.
   - Корректность работы CI/CD пайплайна.