Kotlin Developer. Basic

Kotlin Developer. Basic

Длительность курса: 86 академических часов

1 Введение в Kotlin

1 Вводное занятие

Цели занятия:

- о курсе: цели, кем станете по окончании, как будем заниматься, о домашних заданиях
- о Kotlin: общие сведения о языке, обзор функциональности Kotlin, экосистемы, место место на рынке языков, его применимость и ограничения, конкурентные преимущества и недостатки, критерии выбора для конкретных проектов;
- инициализация программы на Kotlin, создание проекта с Gradle и Kotlin, первая программа и автотест.

Краткое содержание:

- обсудить порядок работы на курсе, роадмап, взаимодействие;
- место Kotlin на рынке языков, его применимость и его ограничения;
- конкуренты Kotlin;
- процесс инициализации первого проекта на Gradle, создание первой программы на Kotlin и написание первых тестов на Kotlin.

Домашние задания

1 Реализовать Sealed Class

Цель: Цель ДЗ - сделать первые шаги в программировании на Kotlin.
В результате его выполнения вы научитесь создавать классы, методы и свойства на языке Kotlin.

Для успешной сдачи ДЗ вам необходимо выполнить следующие требования.
1. Написать Sealed Class для одной из ниже указанных (либо другой по вашему выбору) сущностей.
2. Sealed Class должен включать не менее трех других классов.
3. Классы должны содержать по крайней мере один метод и не менее одного свойства.
4. Все классы должны быть покрыты автоматическими тестами

Возможные сущности для описания Sealed-классов.

- Animals (Dog, Monkey, Horse, etc)
- Professions (Cooker, Driver, Scientist, etc)
- Vehicles (Car, Truck, Tram, etc)
- Plants (Palm, Grass, Bush)
- Frameworks (Spring, Ktor, Micronaut, etc)
- Languages (Php, Perl, Python, Kotlin, etc)

1. Создайте проект в Intellij Idea, укажите тип проекта Gradle, Kotlin DSL, тип Kotlin JVM.
2. Создайте Sealed Class
3. Добавьте входящие в Sealed Class классы, их свойства и методы
4. Добавьте класс для автоматического теста, подключите его в build.gradle.kts.
5. Напишите методы для тестирования классов, входящих в Sealed Class.
6. Создайте проект в github.com в своем аккаунте
7. Перенесите в него код, который вы сделали выше в ветку m1-sealed, закомитьте его и запушьте в гит-репозиторий.
8. Создайте Pull Request, назначьте в качестве ревьюера пользователя с ником svok

---

2 **Базовые элементы Kotlin**

Цели занятия:

- знать типы данных, используемых в Kotlin
- операторы
- создание переменных
- управлять инициализацией переменных
- пользоваться Null
- управлять потоком исполнения с помощью циклов и условий

Краткое содержание:

1. Базовые типы
2. Операторы
3. Null-safety
4. var/val
5. Приведение типов
6. Управление потоком исполнения
6.1 if
6.2 for
6.3 while
6.4 when

---

3 **Функции в Kotlin**

Цели занятия:

- научиться создавать функции в Kotlin
- знать различные виды функций
- создавать функции высшего порядка

Краткое содержание:

- Функции
- Расширения
- infix
- Лямбды
- Scope functions

---

4 **ООП-начало**

Цели занятия:

- основные принципы ООП
- создавать классы, объекты, интерфейсы
- определять свойства классов различными способами
- декларировать собственные геттеры и сеттеры свойств

Краткое содержание:

- что такое ООП, основные принципы
- интерфейсы, классы, объекты
- структура класса в Котлин
- свойства и методы, геттеры и сеттеры

---

5 **ООП-продолжение**

Цели занятия:

- пользоваться различными типами классов
- создавать интерфейсы, абстрактные классы и понимать отличия между ними
- планировать классы с использованием подхода SOLID

Краткое содержание:

- типы классов: data, enum, sealed, value, abstract
- разница между интерфейсами и абстрактными классами
- (опционально) основы SOLID, создание качественных классов

6	**Практика по базовым элементам Kotlin**	**Цели занятия:** - прояснить все вопросы по подготовке и сдаче ДЗ; - задать вопросы по пройденному материалу; - порешать практически задачки на Kotlin; **Краткое содержание:** - формат сдачи ДЗ; - создание репозитория на github; - решение прикладных задач программирования на Kotlin; - разбор трудностей.

1 **Коллекции и последовательности**

Краткое содержание:

- структура классов: Iterable, Collection, List, MutableList, SortedSet, NavigableSet и т.д.
- типы коллекций:
-- list
--- array list
--- linked list
--- array linked list
-- map
--- hash map
--- tree map
--- linked hash map
-- set
--- hash set
--- tree set
- мутабельные и иммутабельные коллекции
- Sequences
- Операции над коллекциями и последовательностями

Домашние задания

1 Разработать DSL для класса

Цель: - освоить работу с лямбда-функциями;
- научиться писать билдеры для классов;
- потренироваться в создании DSL для классов.

1. Создайте автоматический тест в своем проекте.
2. В тесте напишите как должен выглядеть ваш DSL. Реализацию до завершения работы над тестом не выполняйте!
3. Сделайте реализацию вашего DSL в этом же файле.
4. Оптимизируйте код вашего DSL
5. Разнесите получившиеся классы по пакетам.

Предметные области для DSL:
- Animals (Dog, Monkey, Horse, etc)
- Professions (Cooker, Driver, Scientist, etc)
- Vehicles (Car, Truck, Tram, etc)
- Pants (Palm, Grass, Bush)

- Frameworks (Spring, Ktor, Micronaut, etc)
- Languages (Php, Perl, Python, Kotlin, etc)

---

2 **Обобщенные типы**

Краткое содержание:

- что это такое
- обобщенные типы в функциях
- inline, reified функции
- обобщенные типы в классах
- in и out модификаторы
- примеры использования

---

3 **Предментно-ориентированные языки (DSL)**

Краткое содержание:

- элементы DSL: лямбда-функции, переопределение операторов, инфиксы, инлайн-функции и т.д.
- тест для DSL
- реализация DSL-билдеров

---

4 **Практика по Kotlin DSL**

Цели занятия:

- пользоваться основными компонентами Kotlin DSL:
-- лямбда-функциями;
-- функциями высшего порядка;
-- переопределением операторов;
-- инфиксными функциями;
- создавать собственные DSL.

Краткое содержание:

- ответы на вопросы;
- подготовка к сдаче ДЗ;
- практические задачи по DSL.

3 Асинхронное и многопоточное программирование

1 Основы конкуретного программирования

Цели занятия:

- знать основы многопоточности
- создавать потоки в JVM
- выполнять синхронизацию данных между разными потоками
- пользоваться разными шаблонами многопоточности

Краткое содержание:

- потоки, процессы, сегменты кода/данных/стека
- как микропроцессор обрабатывает параллельные потоки
- Java Memory Model
- Thread, запуск потоков в функциональном и ООП-стилях
- синхронизация данных между потоками: syncronized, locks, atomic
- шаблоны синхронизации: семафор, циклический барьер, замок с обратным отсчетом и т.д.

Домашние задания

1 Программа с параллельным выполнением

Цель: - научиться пользоваться инструментами для параллельных вычислений в Kotlin
- освоить корутины
- разобраться в работе многопоточных программ
- понять особенности работы квазипараллельного функционирования корутин

Требуется реализовать класс-обертку для коллекции классов. В нем реализовать suspend-методы, выполняющие различные обработки этой самой коллекции, включая map, filter, flow и пр. Среди обработок необходимо использовать delay и Thread.sleep(), демонстрируя разницу в подхода к работе с этими функциями.
Работоспособность кода продемонстрировать с помощью автоматического теста.

1. Создать автоматический тест, в котором продемонстрировать создание и вызов класса-обертки для коллекции.
2. Когда вид теста вас устроит, сделать реализацию классов прямо в файле теста. По ходу реализации допустимо корректировать тест. Необходимо реализовать:
2.1. Класс-обертку для коллекции
2.2. Класс-элемент коллекции
2.3. suspend-метод для обработки с использованием sequence
2.4. suspend-метод для обработки с использованием flow
2.5. suspend-метод для обработки с использованием flow и Thread.sleep внутри операций над flow
3. Оптимизировать полученный код
4. Разнести полученные программные элементы из одного файла по пакетам в отдельные файлы

---

2 **Асинхронное и многопоточное программирование с корутинами**

Цели занятия:

- понимать устройство корутин и принципы из работы
- создавать код с асинхронным выполнением
- запускать корутины в нескольких потоках
- тестировать асинхронный код

Краткое содержание:

- suspend-функции;
- связь с JVM и JS;
- корутины;
- асинхронные исполнение;
- многопоточность в рамках корутин;
- тестирование корутин.

---

3 **Асинхронное и многопоточное программирование с каналами**

Цели занятия:

- пользоваться подходом функционального программирования с Flow
- знать устройство Flow и принципы их работы

Краткое содержание:

- Flow
- операции с Flow
- Смена контекста
- перехват исключений
- SharedFlow
- StateFlow

---

4 **Практика по конкурентному программированию**

Цели занятия:

- писать асинхронный и многопоточный код с использованием корутин
- пользоваться классами Flow, SharedFlow, StateFlow
- обеспечивать синхронизацию данных в разных потоках

Краткое содержание:

- ответы на вопросы;
- подготовка к сдаче ДЗ;
- практические задачи по корутинам

4 Мультиплатформенная разработка

1 Kotlin Multiplatform (KMP)

Цели занятия:

- знать устройство мультиплатформенного проекта
- самостоятельно создавать проекты для JVM, JS, Native
- тестировать мультиплатформенный код

Краткое содержание:

- настройка KMP в Gradle
- создание разделяемого кода
- создание платформно-зависимого кода
-- JVM
-- JS
-- Native (Си)
- Тестирование KMP-логики

Домашние задания

1 Реализовать мультиплатформенную бибилиотеку с KMP

Цель: - освоить мультиплатформенную разработку с KMP
- создать разделяемый код для разных платформ
- создать специфический для разных платформ код
- внедрить корутины в мультиплатформенную библиотеку
- научиться тестировать мультиплатформенный код

Необходимо разработать мультиплатформенную бибилиотеку на базе KMP. Выполнить тестирование мультиплатформенного кода

1. Создать в проекте МОДУЛЬ с Gradle Kotlin DSL / Kotlin Multiplatform
2. Настроить build.gradle.kts для всех целевых платформ
3. Реализовать автоматические тесты для синхронных функций и методов библиотеки
4. Реализовать автоматические тесты для suspend-функций и методов библиотеки
5. Написать реализацию библиотеки,

		отрефакторить ее и разместить в разных файлах в пакетах
2	**Интероперабельность с Java**	**Цели занятия:** - выполнять интеграцию Kotlin-проектов в Java-проекты **Краткое содержание:** - аннотации для Java - обертки для Java - практические аспекты миграции Java-кода в Kotlin
3	**Интероперабельность с JavaScript**	**Цели занятия:** - подключать JS и TS библиотеки в Kotlin-код - использовать Kotlin в проектах на JS и TS **Краткое содержание:** - подключение npm-зависимостей - импорт JS-кода в Kotlin-проект - экспорт Kotlin-кода в JS и TS - корутины в Kotlin-JS - тестирование в Kotlin-JS
4	**Интероперабельность с Си**	**Цели занятия:** - использовать dll и so библиотеки в Kotlin Native проектах - экспортировать и импортировать в C-код Kotlin переменные и наоборот - управлять памятью C-библиотек **Краткое содержание:** - настройка Kotlin-Native проекта в Gradle - cinterop - импорт native-библиотеки (dll, so) - компиляция Kotlin-кода в native библиотеку (dll, so) - работа с Си-кодом в Kotlin - управление памятью Си-кода в Kotlin

5 Прикладная разработка на Kotlin

1 Работа с Gradle Kotlin DSL

Цели занятия:

- создавать проекты с Gradle
- пользоваться инфраструктурой Gradle, включая таски, конфигурации и пр.
- генерировать "толстые" jar-файлы и запускать их
- генерировать Docker-образы

Краткое содержание:

- обзор стандартных блоков Gradle
-- plugins
-- repositories
-- dependencies
-- tasks
- создание и связывание таск
- создание application
- генерация "толстых" Jar-файлов
- публикация в maven-репозиторий
- генерация Docker-образов
- создание своего плагина

Домашние задания

1 Разработка прикладного приложения на Kotlin

Цель: Цель ДЗ - определиться с дальнейшей специализацией в разработке (фронтенд или бэкенд).
Результат выполнения ДЗ - работающее приложение на одном из фреймворков, рассмотренных на вэбинарах.

ДЗ: Выбрать один из пройденных фреймворков - Jetpack Compose (frontend), Spring, Ktor (backend) - и реализовать приложение на его основе.

1. Выберите один из фреймворков для ДЗ: JetPack Compose, Spring или Ktor
2. Создайте соответствующий модуль в проекте, настройте build.gradle.kts
3. Для JetPack Compose:
3.1. Создайте страничку с одним полем ввода "name" и одним полем вывода "greeting"
3.2. Реализуйте логику заполнения приветствия в поля greeting при вводе имени в поле name
4. Для Spring и Ktor:

4.1. Создайте REST-контроллер
4.2. Добавьте в контроллер метод для эндпоинта get, принимающего в качестве аргумента параметр id и возвращающего Json с полученным id
4.3. Добавьте в контроллер метод для эндпоинта post, принимающего Json-запрос с полем name и возвращающего Json-ответ с полем greeting, содержащим приветствие для name.
4.4. Реализуйте два автоматических теста для обоих методов

---

2 **Разработка frontend на Compose Multiplatform**

Цели занятия:

- создавать несложные фронтендд-приложения
- делать формы ввода и логику отображения

Краткое содержание:

- настройка в Gradle
- работа в Intellij Idea
- разработка Hello World для Desktop, Android и Web
- верстка (layouts) приложения в Compose
- работа с состояниями в Compose
- добавление дизайн-элементов
- работа со стилями
- формы в Compose
- отправка данных в backend

3	**Разработка backend на Spring**	**Цели занятия:** - создавать Web-приложения на Spring - формировать контроллеры и эндпоинты - обрабатывать REST-запросы - тестировать Spring-приложения **Краткое содержание:** - о Spring и Spring Boot, AOP - создание Spring-приложения в Gradle - особенности Spring с Kotlin - компоненты: Component, Controller, Configuration, Service - Spring DI, Bean - репозиторий - краткий обзор библиотек - тестирование в Spring
4	**Разработка backend/frontend на Ktor**	**Цели занятия:** - создавать Web-приложения на Ktor - формировать контроллеры и эндпоинты - обрабатывать REST-запросы - тестировать Ktor-приложения **Краткое содержание:** - запуск Hello World приложения на Ktor - Ktor-server и Ktor-client - REST в Ktor - тестирование в Ktor

5 **Практика по прикладной разработке**

Цели занятия:

- создавать простые приложения без фреймворков в Gradle
- создавать фронтенд-приложения с Jetpack Compose
- создавать бэкенд-приложения с Ktor и Spring

Краткое содержание:

- ответы на вопросы;
- подготовка к сдаче ДЗ;
- практические задачи по JetPack Compose, Ktor, Spring

6 Выпускной проект

1 Консультация по проектам и домашним заданиям

Цели занятия:

- выбрать тему выпускного проекта
- обсудить порядок сдачи выпускного проекта
- задать вопросы по пройденному курсу

Краткое содержание:

- раздача тем выпускных проектов
- обсуждение порядка защиты проектных работ
- обсуждение вопросов по курсу

Домашние задания

1 Проектная работа

Цель: (описание задания проектной работы указано в д/з к занятию "Работа с Gradle Kotlin DSL")

---

2 Защита проектных работ

Цели занятия:

- продемонстрировать результаты выполнения выпускной работы

Краткое содержание:

- демонстрация студентами своих выпускных работ