Android Advanced Development

Duración: 50 Hrs

Objectives:

- Develop Android Applications with Kotlin
- Using Dagger to manage dependencies
- Learn MVVM architecture concepts
- Use LiveData as an observable data holder
- Use Retrofit library to access a backend
- Use Room as an object-mapping library

Prerequisite:

- Experience with Android Development using Kotlin

Contenido del curso

1 Dependency injection using Dagger

- What is dependency management
- Design classes with dependencies
- Design based on inversion of control
- Design based on injection
- Designing own dependency management framework
- Singleton & Introduction to Dagger
  History of Dagger 2
- Create your own custom Annotation
- Understanding Dagger Framework
- Project introduction, @Module, @Inject, and @Provides
- Create and use a Component
- Singleton and Scope
- Use Dagger in an Activity
- Share instances between components
- Scope and Component Methods
- Constructor Injection
- Qualifier and Named

2 Architecture I

- Introduction to Lifecycle
- Challenges of lifecycle handling
- Activity rotation problem
- Lifecycle States and Events
- Create a Lifecycle Aware Component
- Create a TimerToast in an Activity
- Make the TimerToast lifecycle aware
- What is a ViewModel?
- How ViewModel solves screen rotation problems
- What is a LiveData?
- Types of LiveData
- Sharing a ViewModel
- Using ViewModel and LiveData
- Create ViewModel and LiveData based TimerToast
- Use ViewModel and LiveData in an Activity

ExecuTrain
Líder en capacitación empresarial
Querétaro

Android Advanced Development

Duración: 50 Hrs

3 RxJava

- How does threading work in Android?
- What is RxJava?
- Components and basic examples
- Schedulers
- AsyncTask vs RxJava
- Operators Examples - Map, Filter, Zip & FlatMap
- Disposable & CompositeDisposable
- Types of observables and Create your own observables
- Solving Search problem with RxJava Operators - Debounce,
- DistinctUntilChanged, SwitchMap
- Advantages of RxJava

4 Database

- Relational database concepts
- Tables and Schema
- Problems in a bad Schema design
- Types of Relationships and Foreign Keys
- Normalization and many-to-many relationships
- Introduction to Room Database
- CRUD operations in Room Database
- Project setup and User Entity
- Create User DAO and queries
- Create Room Database instance
- Using Room Database
- Dagger setup for Room
- Making Room queries using RxJava in ViewModel
- Show Room data in UI using LiveData
- Create relations in Room Database
- Embedded
- Relation and Foreignkey
- DAO and queries across tables
- Test queries using UI
- Advanced Concepts
- TypeConvertors
- Migration

5 Network

- Concepts: HTTP, OKHttp, and Retrofit
- Introduction to Networking
- What is Retrofit?
- Network Caching
- Interceptors
- Read and Write Timeout
- Parse data with Gson
- Retrofit with RxJava
- Implementing Network APIs through code
- Configure Retrofit
- Create Request and Response Model
- Create POST request
- Configure Dagger for Networking
- Make Network call in a ViewModel
- Create GET request and complex data Model
- Add Query parameters and Headers
- Delete query

6 Architecture II

- Different types of Architectures
- An Architecture use case?
- Feature addition problem
- Why tests are important
- Some suggestions for adopting an Architecture
- Separation of concern
- No hard dependency principle
- What is MVC architecture?
- What is MVP architecture?
- What is MVVM architecture?
- MVVM architecture blueprint
- MVVM package overview
- Getting started with MVVM
- Base classes overview

- Introduction to Generics
- ViewModel overview
- Build the Base classes for MVVM

7 Unit Testing

- What is testing and its advantages
- Types of Unit testing and packaging
- Implementation
- Writing Unit Test
- Libraries used in unit test
- UI Test

8 Kotlin Coroutines

- What exactly are Coroutines?
- Need for the solution which Kotlin Coroutines provide
- Dispatchers, suspend, launch, async
- What are scopes in Kotlin Coroutines?
- Exception handling in Kotlin Coroutines