Matheus Braschi Haliski

Bryan Strey

Victor Luís Soares Brito

Bacharelado em Sistemas de Informação

PUCPR

Trabalho Discente Efetivo sobre conceituação e aplicação de frameworks de Dependency Injection como Dagger e Hilt em aplicativos Android

Trabalho apresentado como requisito parcial para conclusão da Disciplina [“Mobile Development: Android”, sob orientação do Prof. Mark Eirik Scortegagna Joselli.](https://pucpr.instructure.com/courses/41860)

CURITIBA

07/09/2024

**Sumário**

1. **Introdução**3

1.1 Objetivo do Trabalho3

1.2 Conceituação – Aplicativo Android e Frameworks3

**2. Análise de Frameworks de Dependency Injection para Android**3

2.1 Frameworks Disponíveis3

2.2 Vantagens e Desvantagens4

2.3 Análise de funcionalidades4

2.4 Escolha do Framework para o atual projeto4

**3. Implementação de um projeto prático com Dependency Injection**..............5

3.1 Introdução ao projeto prático.............................................................................5

3.2 Desenvolvimento do projeto prático..................................................................5

**4.Conclusão**...........................................................................................................6

1. Introdução
   1. Objetivo do Trabalho

Este trabalho visa analisar e demonstrar os conceitos e aplicação prática dos frameworks de Dependency Injection(DI), com foco específico no Dagger e Hilt, que são frameworks de desenvolvimento de App Android que promovem a Injeção de Dependência promovendo a melhor legibilidade e manutenção do código. Este estudo busca proporcionar uma compreensão detalhada dos princípios fundamentais da injeção de dependência, explicando as características, vantagens e desvantagens do Dagger e do Hilt, assim como aplicar estas ferramentas em um projeto prático no **Item 3.** para mostrar como elas podem beneficiar a modularidade, escalabilidade e testabilidade do código.

* 1. Conceituação – Aplicativo Android e Frameworks

Um aplicativo Android é definido como um produto de software que deve ser executado em disposivitos que utilizam o SO Android tais como smartphones, tablets e smartwatches. Estes apps são tipicamente escritos em linguagens de programação como Java e Kotlin e são compostos por vários componentes (frameworks) tais como Activities, Fragments, Services e Broadcast Receivers. Cada componente desempenha um papel específico no gerenciamento e comunicação da interface do usuário.

A arquitetura de um app Android pode ser de dois tipos: Model-View-ViewModel(MVVM) ou Model-View-Presenter(MVP).Esses padrões ajudam a separar as responsabilidades e facilitam a evolução e manutenção do código.

1. Análise de Frameworks de Dependency Injection para Android
   1. Frameworks disponíveis

Os dois frameworks disponíveis para Dependency Injection em app Android são o Dagger e o Hilt.

* 1. Vantagens e Desvantagens

Dagger – Vantagens: é um framework que se destaca por sua performance e por gerar código altamente eficiente em tempo de compilação. Ele usa anotações para declarar dependências e fornece um conjunto robusto de ferramentas para injetar dependências de forma eficiente.

Dagger – Desvantagens: a configuração do Dagger pode ser mais complexa devido a necessidade de definir módulos e componentes.

Hilt – Vantagens: É uma extensão do Dagger, sendo então projetado para simplificar a configuração e uso do Dagger, reduzindo sua complexidade através da integração com o Android Jetpack.

Hilt – Desvantagens: Abstração limitada e dependência do Dagger.

* 1. Análise de funcionalidades

A injeção de dependência em Android é um padrão de design que visa separar a criação das dependências de suas classes consumidoras, promovendo o desacoplamento e facilitando os testes unitários. As principais funcionalidades são:

* Desacomplamento
* Facilidade nos testes unitários
* Redução de boilerplate
* Gerenciamento do ciclo de vida
* Modularidade e reutilização do código
* Melhor controle sobre as dependências
  1. Escolha do framework para o atual projeto

Este projeto irá fazer uso do Dagger e de suas funcionalidades, sendo que o **Item 3.** Mostra um projeto prático com uso deste framework.

1. Implementação de um projeto prático com Dependency Injection
   1. Introdução ao projeto prático

Foi desenvolvido um projeto prático desenvolvido no Android Studio através de linguagem Kotlin e a criação de classes e objetos. O nome do projeto (aplicativo) é “MyIntentApplication” e foi desenvolvido para ser composto por duas activities principais que cadastram e exibem nomes de cidades e seus respectivos países: Main Activity e Second Activity.

A Main Activity exibe em formato de lista, criada com o uso de Recycler Views, os dados das cidades inseridas na Second Activity, a qual possui um formulário para serem inseridos o nome da cidade e seu respectivo país.

* 1. Desenvolvimento do projeto prático

O projeto prático foi desenvolvido declarando as activities no AndroidManifest.xml.

Foi desenvolvido e publicado um vídeo no Youtube explicando o projeto prático, **disponível no seguinte link**: <https://www.youtube.com/watch?v=dgM5JTs829M>

Foram publicados os códigos no GitHub, disponível no seguinte link:

<https://github.com/MatheusHaliski/TDE_ANDROID_DI>

Em seguida, foram criadas várias classes e objetos importantes tais como a MainActivity, a SecondActivity, as quais cadastram e exibem o nome das cidades, sendo que a SecondActivity retorna os dados para a MainActivity através dos Intents.

O layout das listas das cidades e seus países, e os formulários de cadastro, foram desenvolvidos nos arquivos “activity\_main.xml” e “activity\_second.xml”.

Também existem arquivos importantes para armazenar os dados através de Data Bases do tipo SQLlite com uso do Room, tais como o CityDatabase e o objeto CityDatabaseModule.

Existem também arquivos para o controle, gerenciamento e injeção de dependências tais como o AppModule e o MyApplication.

1. Conclusão

Tanto o uso do Hilt quando do Dagger são vantajosos para realizar Injeção de Dependência e deixar o código mais limpo e modular, porém é necessário tomar cuidados ao manipular estes frameworks para não configurar incorretamente o projeto, visto que os dois dependem da correta declaração e instalação em gradle.build (configuração do projeto).

No meu projeto, realizei o uso de DI com a injeção da variável CityDao, a qual livrou a necessidade de inicializar esta variável manualmente. Porém, apesar de ter suas vantagens, a Injeção de Dependência pode ser custosa e trabalhosa caso seja utilizado sem conhecimento prévio e cuidado na sua instalação.