

Dagger-Hilt

vs.

Koin

vs.

Manual Dependency Injection

A Practical Comparison for Android Development

SWIPE



Esmaeeil Moradi



DI

Dependency Injection (DI) is a fundamental design pattern that helps decouple your code, making it more modular, testable, and maintainable. In Android development, the choice of DI approach can significantly impact your project's structure and scalability. Here, I'll provide a concise comparison of the three most common DI strategies: Dagger-Hilt, Koin, and Manual Dependency Injection.

KEEP SWIPING



What is Dependency Injection?

Dependency Injection means providing a class's dependencies from the outside, rather than having the class construct them itself.

This enables:

- Flexible swapping of implementations (great for testing)
- Easier management of complex dependency graphs
- Reduced coupling between components

KEEP SWIPING



1. Dagger-Hilt

- Type: Compile-time, annotation-based DI framework developed by Google.
- How it works: Uses annotations to define modules and dependencies. Hilt generates code behind the scenes to wire everything together.
- Pros:
 - Type-safe and highly efficient
 - Officially supported and integrated with Android architecture components
 - Excellent for large, complex projects
- Cons:
 - Requires code generation (KAPT/KSP)
 - Steeper learning curve for beginners

KEEP SWIPING



2. Koin

- Type: Lightweight, Kotlin-first DI framework; uses a DSL for configuration.
- How it works: Dependencies are defined in modules using Kotlin code. No code generation or annotations are required.
- Pros:
 - Simple to set up and use
 - No build-time code generation
 - Great for small to medium-sized projects or rapid prototyping
- Cons:
 - Slightly less performant than Dagger-Hilt
 - May become harder to maintain as complexity grows

KEEP SWIPING



3. Manual Dependency Injection

06/08

- Type: No framework; dependencies are constructed and passed manually.
- How it works: You instantiate dependencies and inject them directly via constructors or setters.
- Pros:
- Full control over dependency creation and lifecycle
- No external dependencies or magic
- Cons:
- Can lead to boilerplate and tightly coupled code
- Difficult to scale and maintain in large projects

KEEP SWIPING

Esmaeel Moradi



Comparison Table

07/08

Feature	Dagger-Hilt	Koin	Manual DI
Type Safety	✓ Compile-time	Runtime (improved in new versions)	✓ Manual, but error-prone
Performance	✓ High	Moderate	Depends on implementation
Setup Complexity	Moderate/High	✓ Low	✓ Low (for small projects)
Scalability	✓ Excellent	Good for small/medium	✗ Poor for large projects
Learning Curve	Steep	✓ Gentle	✓ Simple
Android Integration	✓ Deep	Decent	✗ None

KEEP SWIPING

Key Takeaways

08/08

- Dagger-Hilt is ideal for large, complex Android projects where performance and compile-time safety are priorities.
- Koin is perfect for smaller projects or when you need a fast, flexible setup without build-time code generation.
- Manual DI offers maximum control but is only practical for very simple projects or learning purposes.

Choose the approach that fits your project's scale, your team's experience, and your long-term maintenance needs.

Which DI approach has worked best for you in your Android projects? Share your experiences below!

COMMENT BELOW

Esmaeeil Moradi

