

# Angular Dependency Injection - Deep Dive

## 1. What is Dependency Injection (DI)?

- A design pattern where a class receives its dependencies from an external source.
- Angular uses a built-in DI framework for efficient service management.

## 2. Benefits of DI:

- Modularity, Testability, Reusability, Scalability

## 3. Key Terminology:

- Injector, Provider, Token, Service, InjectionToken

## 4. DI Flow in Angular:

- Component/Service --> Injector --> Provider --> Instance --> Injected

## 5. Basic Usage Example:

```
@Injectable({ providedIn: 'root' })
export class LoggerService { log(msg: string) { console.log(msg); } }
constructor(private logger: LoggerService) { logger.log('Init'); }
```

## 6. Provider Strategies:

- providedIn: 'root', NgModule-level, Component-level, Factory, Value, Alias, Existing

## 7. Advanced Topics:

- Multi Providers, InjectionToken usage, Scoped Services

## 8. Hierarchical Injectors:

- Parent and Child injectors manage service scopes and instances hierarchically.

## 9. Common Pitfalls:

- Forgetting @Injectable, misusing tokens, conflicting providers, not unsubscribing

## 10. Best Practices:

- Use root for shared services, prefer InjectionTokens for values, limit use of 'any'