**Dependency Injection**

Dependency Injection (DI) is a design pattern used to implement IoC. It allows the creation of dependent objects outside of a class and provides those objects to a class through different ways. Using DI, we move the creation and binding of the dependent objects outside of the class that depends on them. Dependency Injection reduces the hard-coded dependencies among classes by injecting those dependencies at run time instead of design time technically.

It is helpful to understand the Dependency Inversion Principle, which gives us the guidelines for writing loosely-coupled classes. Here is the definition:

* High-level modules should not depend on low-level modules. Both should depend on abstractions.
* Abstractions should not depend upon details. Details should depend upon abstractions.

**Types of Dependency Injection**

1. **Constructor Injection:**

In the constructor injection, the injector supplies the service (dependency) through the client class constructor.

1. **Property Injection:**

In the property injection (aka the Setter Injection), the injector supplies the dependency through a public property of the client class.

1. **Method Injection:**

In this type of injection, the client class implements an interface which declares the method(s) to supply the dependency and the injector uses this interface to supply the dependency to the client class.

**Advantages of Dependency Injection**

1. Reduces class coupling
2. Increases code reusability
3. Improves code maintainability
4. Make unit testing possible