# Dependency Injection

Dependency Injection is a design pattern used to implement **Inversion of Control** in software development. It allows the creation of dependency objects outside of a class and provides those objects to a class that depends on it in three different was (i.e., using Constructor, Method, and Property).

In simpler terms, DI is a technique used to achieve **loose coupling** between classes and their dependencies. It is a design pattern that allows for the separation of concerns in an application, making it more maintainable, testable, and flexible.

The Dependency Injection Design Pattern in C# is divided into three parts: **Constructor Injection**, **Property Injection**, and **Method Injection**.

* **Constructor Injection**: When the Injector Injects the Dependency Object (i.e. Service Object) into the Client Class.
* **Property Injection**: When the Injector Injects the Dependency Object (i.e. Service Object) into the Client Class through a Property.
* **Method Injection**: When the Injector Injects the Dependency Object (i.e. Service Object) into the Client Class through a Method.



In this example, the ClientClass has a dependency on the IService interface. The IService interface is injected into the ClientClass constructor, which allows the ClientClass to use the PerformAction() method of the IService interface