* **Spring** : spring framework is a **Dependency Injection framework** to make java application loosely coupled. Spring was **develoved by Rod Johnson in 2003** and it is the most used and popular framework of java for J2EE or application development.

Spring provides alot of modules such as **Spring MVC, Spring Security, Spring Core**. With the help of these our application development becomes more easy.

1. **What is loosely coupled?**

Loosely coupled means We can make changes easily in our application.

1. **Why spring is called as dependency injection framework?**

Because it injects dependencies or objects itself.

Spring provides IOC with the help of which we perform dependency injection.

1. **What is dependency injection?**

It is the main functionality provided by Spring IOC. Dependency injection is a design pattern and a core part of IOC, by following which we can develop applications.

Dependency means one class is dependent on another class to do the work. In java we create object using new keyword. And if we do this then our application will become tightly coupled.

So what does Spring do is, the object we were creating using new keyword. Now it will be done by dependency injection. It will automatically create the object at runtime and will inject that object in another class.

Dependency Injection is a specific implementation of the IoC principle. It refers to the technique of passing (injecting) dependencies (objects) into a class at runtime rather than the class creating them itself. This can be done through constructor injection, setter injection, or method injection.

**Advantages:**

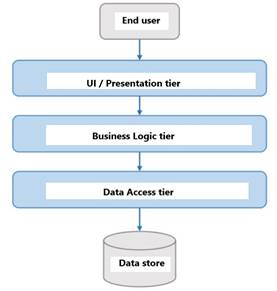
1. Clean code
2. Decoupling is more effective when objects are provided with their dependencies.
3. Classes become easier to test.
4. **What is IOC?**

Inversion of control is a design principle in which the control of object creation and management is transferred from the application code to a container or framework.

1. **Where this design pattern (Dependency Injection) required?**

It is particularly useful in scenarios where decoupling components, managing configurations, and improving code quality are priorities. By leveraging DI, developers can create more flexible and robust applications that are easier to manage and evolve over time.

1. **Design pattern / N-tier pattern / Layered architecture pattern**



we always follow design patterns while developing a application. This process happens on server side.

1. **UI Layer (ProductController) :** It is a simple java class. This class needs to access some services or some business logics. It has the capabilities to accept requests. It will simply use the services of other class (ProductService).We do not write logic in this class.
2. **Business/Service Layer (Product Service) :** Business logics are written there and the ProductController will simply use these logics.

Business layer does not directly communicate with database. It only provides business services.

1. **Data Access Layer (ProductDao) :** It communicates with database and send that to service layer.

**Spring Dependency Injection** will create the object of ProductDao and will inject it to ProductService.

And will create the object of ProductService and will inject it to ProductController. This process will take place by IOC container.