# How Many Ways we can Collabrate the Object

There are two ways of collabrating the objects.

1.Dependency pull

2.Dependency Injection

# 1.Dependency Pull

There are 2 types of Dependecy Pull

1. Dependecy Lookup.
2. Contextual Dependency Lookup.

* This is the old way of collabrating the obj.If we go for Dependency pull comp of my App will become tightly coupled.
* Ex: - If class A wants to talk to class B,either Class A can pull the obj of class B or obj of class B can be injected to Class A.If we go for Dependecy pull then comps of my App will become tightly coupled.So don’t use Dependecy Pull.

# (i).Dependecy Lookup

My class requires another class obj as dependency,so my class should perform lookup or searching operation for getting dependency.This process is called Dependency Lookup.

Example:- As we need DataSource obj as dependent obj to my class.So my class should not create the DataSource obj rather than DataSource obj is being created by App Server.How can my class get the DataSource obj so that my App get the connection obj unless app server provides the DataSource obj we can not access the DataSource obj.So that App server bind the DataSource obj with Jndi Registry.So that my App should connect to Jndi Registry and to perform Lookup operation and to pull the depend obj as DataSouce obj.Since I am pulling dependent obj by performing lookup operation ,this process is called Dependency Lookup.

* Jndi Registry is public

# (ii).Contextual Dependency Lookup

My class requires another class object as dependency, so my class get contractual agreement in order to acquire dependent object by looking up an object. This process is called Dependency Lookup.

Example:-My servlet wants to access the environment info in which it is running, for which we need get access of ServletContext object ,for ServletContext Object we need ServletConfig obj.So we can not create ServletConfig obj rather than ServletConfig obj is created by Servlet Container and ServletConfig obj is specific to a Servlet class.my class requires ServletConfig object as dependent object that is there in the Servlet container,so my class get contractual agreement in order to acquire dependent obj,it should implement container provided interface and should override the init()method.Servlet container is called init() method automatically at the time of creating the servlet class obj by passing ServletConfig obj as parameter value.So my class get ServletConfig obj as dependent obj as an contextual agreement by looking up an obj.So this process is called Contextual Dependency Lookup.

# 2.Dependency Injection

There are 2 types of Dependency injection

1. Setter injection.
2. Constructor injection.

* This is the modern way of collaborating the object .If we go for Dependency injection , comp of my App will become loosely coupled.

# What is Inversion Of Control ( IOC )

* There is no difference b/w IOC container and Inversion of Control.
* The way it is managing the dependency is called Inversion of Control.
* There are 2 ways of Managing the dependency

1.Dependency pull.

2.Dependency injection.

* As IOC container supports both dependency pull and Dependency injection, so it is also called Inversion of control.
* One who wants to inject some one is called target.
* One who is being injected is called dependent.

**In traditional way of collabrating the obj (Dependency pull)**

# A B

* If class A wants to talk with class B ,either class A can create the obj of class B or A can pull the obj of class B.
* Then class A is being called as dependent , bcz class A is dependent on class B, without B class A can not work.
* Class B is called target class bczz class A is pulling other dependent obj

**In Dependency Injection**

**A B**

* If class A wants to talk with class B,then class B is injected to class A.
* Then class A is called Target and Class B is called dependent.
* The roles of the comp is inverse while managing the dependency of objs ,So it is called Inverse Of Control.

# (i)Setter Injection.

My dependent obj is being injected into target classes by calling setter method on target class so it is called setter injection.

# (ii)constructor Injection.

My dependent obj is being injected into target classes by calling constructor of target class so it is called constructor injection.

# If we go for Dependency pull

If class A wants to talk to class B as dependent object, Even our class is design based on the design principles to manage the dependency via Dependecy pull so that logic of class A that is refering other class is still left in class A ,to switch from one class to other class the piece of logic has to be modified. So dependency pull has always tightly coupled. So class A is called dependent class and class B is called target class.

# If we go for Dependency injection

If class A wants to talk to class B as dependent obj ,class A will not write the logic to get dependent rather than the underlaying container is going to inject the dependent obj to my class ie. B .As the obj is being injected to class A , class A becomes target class and the obj that is being injected to Class A is called dependent.

**As the roles of our classes becomes inverse so it is called** Inversion of Control

# Difference B/w setter injection & constructor injection

**(1)**

* In case of a constructor injection the dependent obj gets injected into target class while creating the obj of target class.hence dependent obj is instantiated first.
* **So that we can access the functionality of dependent obj with in the constructor of target class.**
* In case of setter injection the dependent obj gets injected into target class after creating the obj of target class.
* **So that we can not access the functionality of dependent obj with in the constructor of target class.**
* **If** we want to access the functionality of dependent class by passing some addition input within the constructor of target class then go for Constructor injection not go for setter injection.

**(2)**

* If our class is having circular depenecy b/w each other those dependecy is not managed by constructor injection.
* If our class is having circular depenecy b/w each other those dependecy is managed by setter injection.