```
Client     => String ====> VO(all variable datatype is String)
Controller => String to specific datatype ===> DTO
Service    => Buisness Object(BO/Model/Entity) gets Generated which will be used in
DAO layer.


Collection Injection
--------------------
 => It is all about injecting values to array,collection type bean properties
through Dependancy injection.

      Property type                tag/attribute
=================================================
   simple/primitive  ====>   <value>
      object          ====>   <ref>
      array           ====>   <array>/<list>
    List            ====>   <list>
    Set             ====>   <set>
    Map             ====>   <map>
    Properties      ====>   <props>

                        refer:: IOCProj7-CollectionInjection

Collection injection in realtime
================================
DriverManagerDataSource
      |-> driverClassName
      |-> url
      |-> connectionProperties(java.util.Properties object with fixed key called
"user" and "password")
                  |-> user
                  |-> password

applicationcontex.xml
---------------------
<bean id="mysqlDataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
            <property name="driverClassName" value='com.mysql.cj.jdbc.Driver' />
            <property name="url" value='jdbc:mysql:///enterprisejavabatch' />

            <property name="connectionProperties"><!-- java.util.Properties-->
                  <props>
                        <prop key="user">root</prop>
                        <prop key="password">root123</prop>
                  </props>
            </property>
</bean>
                        refer: IOCProj8-CollectionRealTimeDependancyInjection

NullInjection
------------
In constructor injection, all params must participate in injection process
otherwise it would result in "Exception".
If constructor param type is object/reference type and we are not ready with value
then we can go for null injection.
This is very handy(useful) when we are working with predeifned classes as a spring
bean, that is a spring bean will have limited no of
Overloaded constructors and no setter injection support is available.
                        syntax: <constructor-arg name=''><null/></constructor-arg>
```

```
applicationcontext.xml
----------------------
<bean id="per2" class='in.ineuron.bean.PersonInfo'>
            <constructor-arg value='45' />
            <constructor-arg value='rohit' />
            <constructor-arg name='dob'><null/></constructor-arg>
            <constructor-arg name='doj'><null /></constructor-arg>
            <constructor-arg ref="dom" />
</bean>

PersonInfo personInfo2= factory.getBean("per2", PersonInfo.class);
System.out.println(personInfo2);

output
PersonInfo [pno=45, pname=rohit, dob=null, doj=null, dom=Wed Apr 05 12:02:21 IST
2017]
```

refer:: IOCProj9-NullInjectionApp

```
Bean inheritance
---------------------
<bean id='baseCar' class="in.ineuron.bean.Car" abstract="true">
            <constructor-arg name="engineCC" value='1500' />
            <constructor-arg name="model" value='swift' />
            <constructor-arg name="company" value='suziki' />
            <constructor-arg name="fuelType" value='diesel' />
            <constructor-arg name="type" value='hatchback' />
</bean>

<bean id='car1' class='in.ineuron.bean.Car' parent="baseCar">
            <constructor-arg name="owner" value='sachin' />
            <constructor-arg name="regNo" value='KA4567' />
            <constructor-arg name="color" value='red' />
            <constructor-arg name="engineNo" value='12345' />
</bean>

<bean id='car2' class='in.ineuron.bean.Car' parent="baseCar">
            <constructor-arg name="owner" value='dhoni' />
            <constructor-arg name="regNo" value='JH5647' />
            <constructor-arg name="color" value='white' />
            <constructor-arg name="engineNo" value='56789' />
</bean>

Car car3 = factory.getBean("baseCar", Car.class);
System.out.println(car3);

Output
      org.springframework.beans.factory.BeanIsAbstractException(class is abstract
so)
```

Important points of inheritance in bean configuration file
----
1. This is not a class level inheritance, it is spring bean cfg file level bean properties inheritance across multiple spring bean cfgs.
2. <bean abstrat='true'> will neve make the class as abstract,but it makes spring

bean cfg as abstract.
3. One spring bean can inherit and resuse the spring bean properties only from one spring bean..

Bean inheritance in realtime

```
      DataSource(I)===> javax.sql.*
            |
    DriverManagerDataSource(C)===> org.spring.**
            |
     HikariDatasource(C)  ======> com.zaxxer.**
```

applicationContext.xml

```
<bean id="mysqlDataSource"
class="org.springframework.jdbc.datasource.DriverManagerDataSource">
            <property name="driverClassName" value='com.mysql.cj.jdbc.Driver' />
            <property name="username" value='root' />
            <property name="password" value='root123' />
</bean>

<bean id='hikariDataSource' class='com.zaxxer.hikari.HikariDataSource'
parent="mysqlDataSource">
      <property name="jdbcUrl" value='jdbc:mysql:///enterprisejavabatch' />
      <property name="minimumIdle" value='10' /><!-- min pool size -->
      <property name="maximumPoolSize" value='20' />
      <property name="connectionTimeout" value='2000' />
</bean>

<!-- DAO Configuration -->
<bean id='mysqlDaoImpl' class='in.ineuron.dao.CustomerMySQLDAOImp'>
            <constructor-arg name='dataSource' ref='hikariDataSource' />
</bean>
```

Default Beanid

  If we don't provide any beadid to the bean, then IOC container will give the default bean id with the following syntax.

                  Syntax: <pkg>.<className>#<n> ===> 0 based index

```
<bean class='in.ineuron.bean.EngCourse' parent="baseYear"> ====>
in.ineuron.bean.EngCourse#0
            <property name="subjects">
                  <set>
                        <value>GTC</value>
                        <value>ADA</value>
                        <value>M4</value>
                  </set>
            </property>
</bean>

<bean class='in.ineuron.bean.EngCourse' parent="baseYear"> ====>
in.ineuron.bean.EngCourse#1
            <property name="subjects">
                  <set>
                        <value>DMS</value>
                        <value>DS</value>
                        <value>M3</value>
                  </set>
            </property>
</bean>

<bean class='in.ineuron.bean.EngCourse' parent="baseYear"> ====>
in.ineuron.bean.EngCourse#2
            <property name="subjects">
                  <set>
                        <value>OS</value>
                        <value>FLAT</value>
                        <value>CompilerDesign</value>
                  </set>
            </property>
</bean>
```