

Java Learning is all about 3 things

- a. Java Language(Core java course)
- b. Java Technology(JDBC,Servlet,JSP,JSTL,EJB's,JMS,...)  
EJB-> Enterprise Java Bean  
JMS-> Java messaging Service
- c. Framework(Hibernate, Spring, SpringBoot, MicroServices, RestApi's,...)

Framework is not a new technology, rather it is an abstraction provided on top of technology.

Thirdparty team would give apis in the form of jars which would generate boiler plate code based on the inputs we give to the internal containers of the framework.  
eg: hibernate -----> based on configuration details supplied, it will create JDBC environment.

Spring -----> based on configuration details supplied, it will create an object and maintains the object and performs dependancy injection.

Different types of Framework to build application

- a. Web application based framework
  - b. ORM Framework
  - c. Application Framework
  - d. BigData Framework
  - e. Distrubuted Application Development framework
- etc....

Webapplication Framework

These frameworks provides abstraction on top of Servlet,JSP and simplifies MVC architecture based development.

M=> Model

V=> View

C=> Controller

eg: Struts(Apache foundation)

SpringMVC(Part of Spring)----> interface21(pivotal team)

JSF(Java Server Faces) -----> from SUNMS/OracleCorporation

WebWork -----> symphony

ORM Framework

These frameworks provides abstraction on top of JDBC and simplifies to develop object based DBS/w independent persitence logic

eg: Hibernate -----> redhat

TopLink -----> oracle

Ibatis -----> apache

Application Framework

It is an allrounder framework that provides abstraction on top of mulitple jee technologies and even on some frameworks to develop all kinds of logic and different type of app's.

eg: Distrubuted application

eg: myntra application

flipkart application

amazon application....

eg: facebook application(webapplications)

eg: Spring, SpringBoot

SpringFramework is not good in developing Distrubuted applications, so we prefer

using "WebServices".

## Distributed App development Framework

=====

It simplifies the process of developing Distributed App's/Remote Apps.  
SOAP(outdated), Rest/Restful Services/Restful WebServices(latest) ::  
jersey, RestEasy, ....

Based on the mode of development we do, we have 2 types of framework

- a. Invasive Framework
- b. Non-Invasive Framework

### Invasive Framework

=====

=> Developer class will extend or implement an interface given by framework api.  
=> Because of extends and implements the developer code would be tightly coupled with framework api.  
=> It won't promote portability(moving the classes to new framework would not execute).

eg: Servlet, Struts(1.X)

Note: working for a company with a bond.

### Non-Invasive Framework

=====

=> Developer class will not extend or implement any interface given by framework api.  
=> No extends and implements keyword, the developer code would be loosely coupled with framework api.  
=> It promotes portability(moving the classes to new framework would execute).  
eg: Spring, Hibernate, .....

Note: working for a company without a bond.

## Terminologies

- 
1. Middleware Services
  2. Java Bean
  3. Local Client vs Remote Client

### 1. Middleware Services

These are addition/option/secondary logics that can be enabled or disabled on primary logic of an application/project to make it more perfect and accurate.

Eg: banking project

primary logic(Mandatory logic) => deposit money, withdraw money, check balance, opening account.....

Secondary logic(Optional logic) => Logging(history), Auditing(Keeping track of user activities),

Security(Authentication+Authorization).....

### 2. Java Bean(Entity Class/Model class of hibernate)

It is a java class that is developed by following some standards

- a. It does not contain any Logic, rather it acts like a helper class to carry data from one class to another class in a project.
- b. It should implement Serializable interface
- c. All fields should be private and non-static.
- d. The members should have setters and getters

e. Their should be one zero argument constructor which is given by compiler/by programmer.  
f. It is a good practise to override toString() to print the field data.

### 3. LocalClient vs RemoteClient

=====

If App/Comp and its client are staying on the same jvm then that client is called as "LocalClient" to that App/Component.  
If App/Comp and its client are staying on two different jvm of same or different machine then that client is called as "RemoteClient" to that App/Component.  
Normal apps can have only local client, whereas Distrubuted apps like EJB/RMI/WebServices can have both local and remote clients.

### How Spring evolved?

1995 --->Applet(Good for gaming)  
1996 --->Java Bean[Technology used earlier]  
(Started developing by using java classes + java beans)  
Limitations  
a. Doesn't allow remote clients,it works only with local clients.  
b. Not suitable for large scale application  
c. Programmer should handle Middelware webservices along with primarylogic of the application.

1998 ---> EJB(Enterprise java Bean) for building Distrubuted application.  
Advantage  
a. It can handle both remote and local clients.  
b. Gives built in middleware services  
Disadvantage  
a. It runs only in server mode(heavy weight containers)  
b. It is very complex to learn and use.

Note: What is Transaction Management?

The process of combining related operations into single unit and executing them by applying do everything or nothing principle is called "Transaction Management".

Transaction Management is a MiddleWare service.

In Distrubuted applications from 1998 to 2003 the companies have used EJB's to take the benefit of TransactionManagement.

2002/2003---> Spring Framework -----> RodJhonson(interface21)

|==> Provides abstraction over technology/frameworks

Advantage  
a. All kinds of development is possible  
b. Application development is non-invasive programming.  
c. Built in middle-ware services[TransactionService,connectionpoolService,....)  
d. Light weight application development  
e. Easy to learn and easy to use.

Is Spring alternative to EJB,Struts,Hibernate,JEE technology?

Spring vs EJB

Answer. No , Spring framework is used to develop all kinds of app.  
WebServices are alternative to EJB's.

### Spring Vs Struts

Answer. No, Struts will be used to build only webbased application  
Spring can be used to build any type of application.  
SpringMVC is an alternative to Struts.

### Spring vs Hibernate

Answer. NO, hibernate is orm framework to build peristence logic  
Spring has its own orm module through which it promotes  
abstraction  
SpringORM, SpringDataJPA is an alternative to hibernate.

### Spring vs JEE(Java Enterprise Edition)

Answer. No, JEE is a technology which gives api for persistence logic  
and building webapps  
Where as Spring provides an abstraction on top of JEE  
api's  
SpringJDBC-> JDBC, SpringMVC-> Servlet, JSP

### SpringCore

=====

=> It is base module for other modules.

=> This module is given to supply Spring containers to perform Dependency  
management.

=> This module gives 2 spring containers/IOC[Inversion Of Control] containers  
called

a. BeanFactory Container

b. ApplicationContext Container(Latest one)

=> These 2 containers perform the following operations

a. It manages the SpringBean/Object life cycle

b. It performs Dependency Management

a. Dependency LookUp[JNDI]

b. Dependency Injection[commonly used]

### SpringApp can be developed in 4 approaches

a. XML approach(only used in maintainence project).

b. using Annotation driven configuration(XML + Java).

c. using java code configuration.(no XML)

d. using Spring boot autodriven configuration.[supported by SpringBoot]

### Different modes of DependencyInjection

=====

1. Setter Injection.

2. Constructor injection.

3. Field injection.

4. MethodInjection/Method replacer.

5. LookUp Method Injection.

6. Dependency LookUp Injection.

### What is SpringBean?

Any Java class(PreDefined/userDefined/ThirdParty api) whose object is created and  
managed by Spring container is called "SpringBean".

### Dependency Management

=====

=> It is the process of assigning dependant object to Target object by loading

both the classes and by creating the objects for both the classes.

=> The classes/objects which uses the other class services is called "Target class".

=> The classes that acts like helper class to main/target class is called "Dependant class".

eg:: Target class => Flipkart, Vehicle, Student, Mobile

Dependant class => DTDC , Engine , CourseMaterial, SIM

### 1. Setter Injection.

=> It is the process of assigning dependant object to Target object by loading both the classes and by creating the objects for both the classes and injecting the Dependent object to Target Object using Setter is called "Setter Injection".

refer:SetterInjection-01