Spring Introduction

CODEMIND
Your Success Partner

By CODEMIND Technology

Contact us 7758094241

Spring Basic

- ★ What is Spring Framework?
- ★ Spring Modules.
- ★ Advantages of Spring Framework.
- ★ What is Tight Coupling and Loose Coupling Between Java Objects?

★ What is Spring Framework?

- Spring is a light weight and open source framework created by Rod Johnson in 2003.
- ★ Spring is a complete and a modular framework means spring can be used to develop all layers or particular layer of real time application unlike struts (Only for front end) and Hibernate (only for Database).
- ★ Spring is light weight framework because of its POJO model.
- ★ Spring Framework made J2EE application development little easier, by introducing POJO model
- ★ Spring framework is said to be a non-invasive means it doesn’t force a programmer to extend or implement their class from any predefined class or interface given by Spring API.
- ★ Spring having this much of demand because of the following 3 reasons….

  1] Simplicity
  2] Testability
  3] Loose Coupling

Simplicity

Simplicity

★ Spring framework is simple because as it is non-invasive, POJO (Plain Old Java Object) and POJI (Plain Old Java Interface) model.

Testability

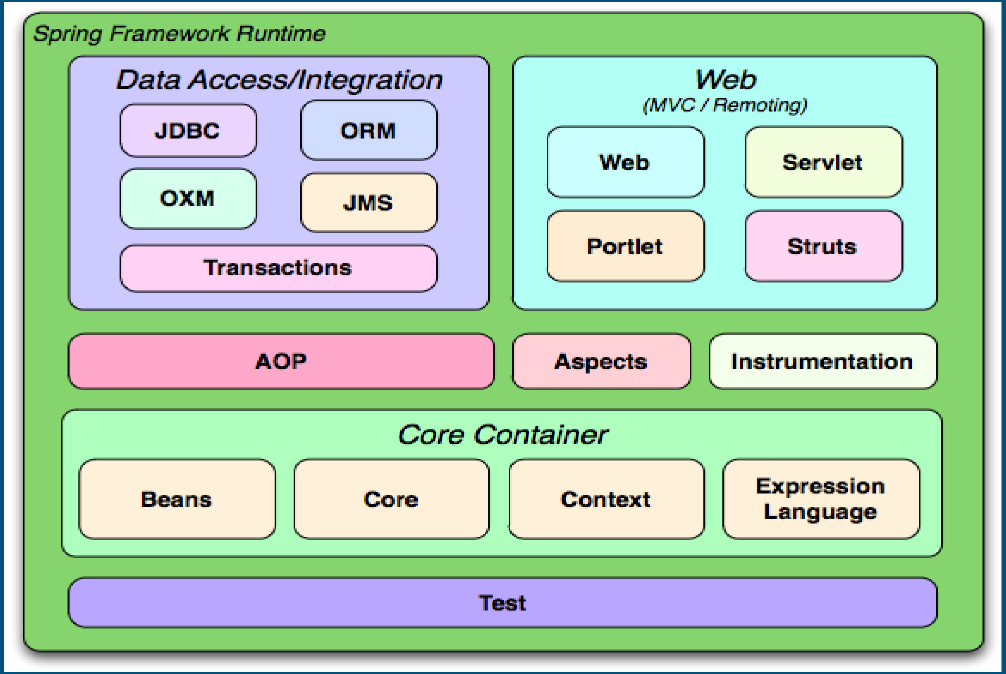
★ Actually for writing the spring application, server [Container] is not mandatory, but for struts applications we need a server, and for EJB too. Each time we must restart the server to view the changes, which is little tedious and time taking but we can over come this in Spring, for testing spring application server is not mandatory spring has it own container to run the applications.

Loose Coupling

★ In spring objects are loosely coupled, this is the core concept of spring framework we will see in depth about this loose coupling and how its differ from tight coupling

★ Spring Modules?

All spring modules are grouped into Test, Core Container, AOP, Aspects, Instrumentation, Data Access / Integration, Web (MVC / Remoting) as displayed in the following diagram.



★ Advantages of Spring Framework.

1] Predefined Templates :- Spring framework provides templates for JDBC, Hibernate, JPA etc. technologies. So there is no need to write too much code. It hides the basic steps of these technologies.

2] Loose Coupling :- The Spring applications are loosely coupled because of dependency injection.

3] Easy to test :- The Dependency Injection makes easier to test the application. The EJB or Struts application require server to run the application but Spring framework doesn't require server.

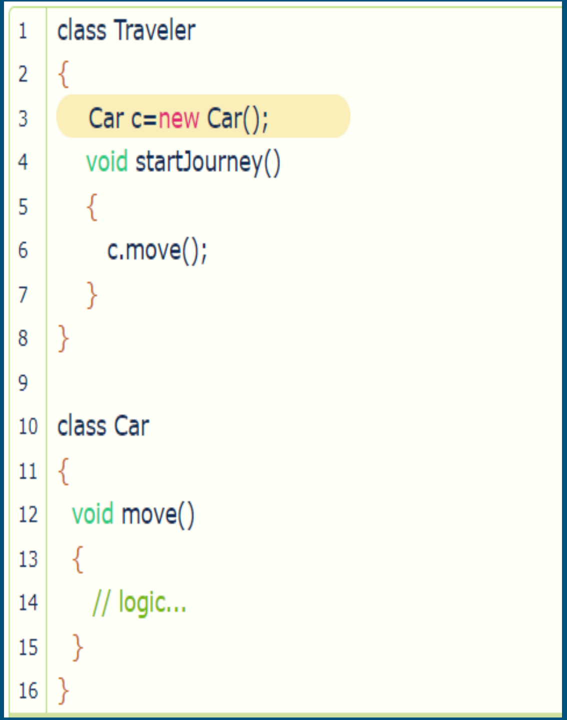
4] Lightweight :- Spring framework is lightweight because of its POJO implementation. The Spring Framework doesn't force the programmer to inherit any class or implement any interface. That is why it is said non-invasive.

5] Fast Development :- The Dependency Injection feature of Spring Framework and it support to various frameworks makes the easy development of JavaEE application.

★ What is Tight Coupling and Loose Coupling Between Java Objects?

Tight Coupling :-

- In the above example, Traveler object is depends on car object.
  So traveler class creating an object of Car class inside it [line number 3]
- If the other class object is created in the dependent class [like Car class object in Traveler class], there exist tight coupling, i mean if method in car object is changed then we need to do the changes in the Traveler class too so its the tight coupling between Traveler and Car class objects

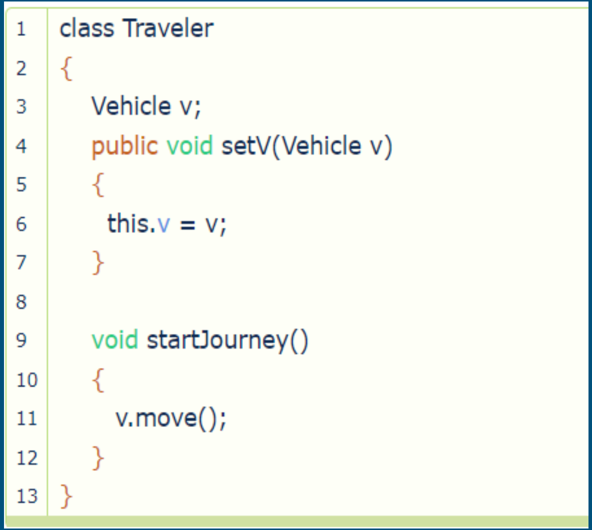


```
1  class Traveler
2  {
3     Car c=new Car();
4     void startJourney()
5     {
6        c.move();
7     }
8  }
9
10 class Car
11 {
12   void move()
13   {
14      // logic...
15   }
16 }
```

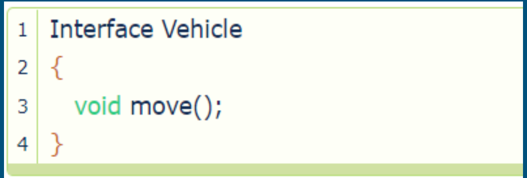
★ What is Tight Coupling and Loose Coupling Between Java Objects?

- In order to over come tight coupling between objects, spring framework uses dependency injection mechanism with the help of POJO/POJI model and through dependency injection its possible to achieve loose coupling.

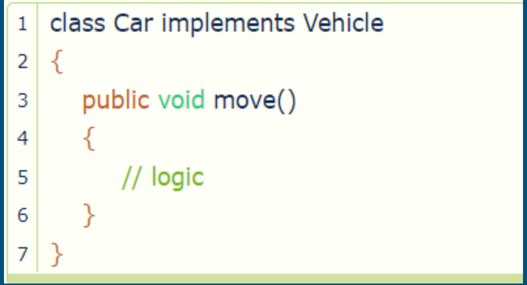
- In the above example Traveler , Car are tightly coupled. If we want to achieve loose coupling between the objects Traveler and Car, we need to re-write the application like….



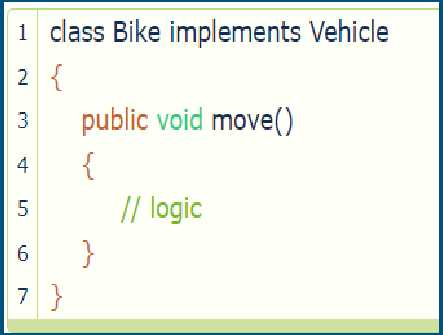
```java
class Traveler
{
    Vehicle v;
    public void setV(Vehicle v)
    {
      this.v = v;
    }

    void startJourney()
    {
       v.move();
    }
}
```



```java
Interface Vehicle
{
   void move();
}
```



```java
class Car implements Vehicle
{
    public void move()
    {
         // logic
    }
}
```



```java
class Bike implements Vehicle
{
    public void move()
    {
         // logic
    }
}
```

- In above example, spring container will inject either Car object or Bike object into the Traveler by calling setter method, So if Car object is replaced with Bike then no changes are required in Traveler class, this means there is loose coupling between Traveler and Vehicle object. Actually setter method will be activated through xml file, will see this later for now we are good to go 😉

Thank you