* Spring: It's a Dependency Injection framework to make java application loosely coupled. (i.e. removing tied coupling to make things easier).

→ It makes the easy development of JavaEE application.
→ Spring framework provides many modules like Spring MVC, Spring Jdbc, spring core, spring security etc. it makes all these makes the application development easy.

* Dependency Injection:

→ It's a design pattern
→ ~~We can inject a dependency dependency of~~

→ Dependency is not hard-coded in the class it's actually injected by an entity outside the class

e.g:

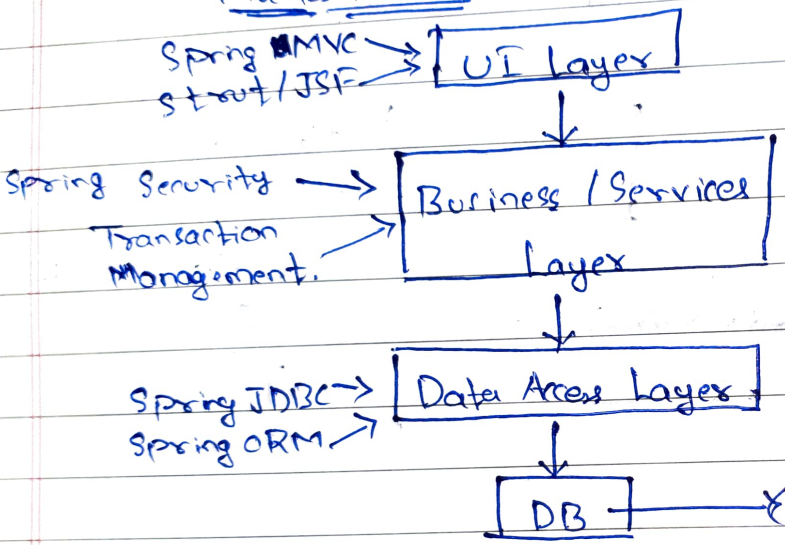
```
class R                          class g
{                                {
    g obj;                           public void doWork()
    public void doWork()             {
    {                                }
    }                            }
}
```

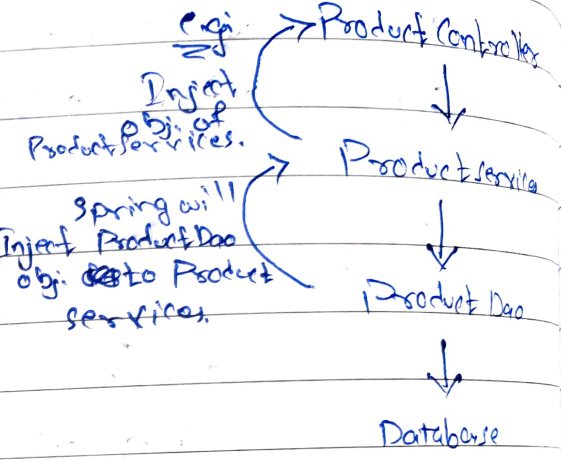
Here, R is completely ~~dependent~~ depended on the g class for doing work (i.e. known as dependency). ~~Whenever the object is given to the~~

→ Spring can create object of another class & inject it in the class, We won't create object manually, it will be done by spring only then we can use the object of (g class) into (R class).

* Spring & JEE :

Java application Road-Map:
Modules to work with

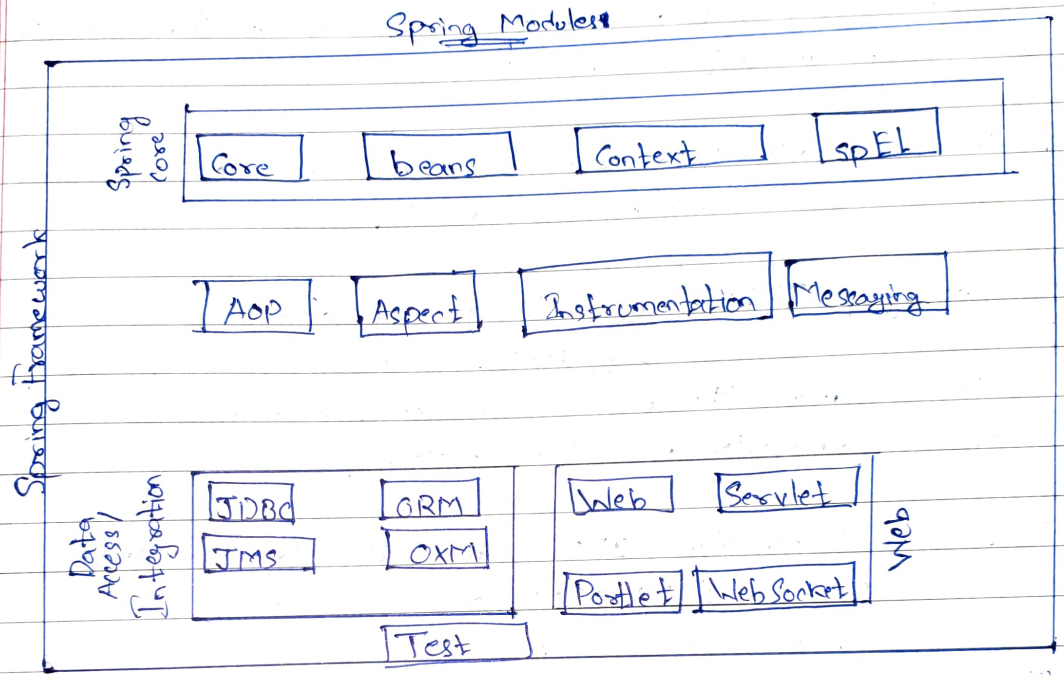




→ Spring JBDC : It's a module to access / to contact the database easily.

→ To work with UI layer (i.e. for frontend) we can use spring MVC.

* Spring-Modules:



→ Spring core & spring beans provide fundamental parts of the framework like IOC & Dependency injection.

→ Context inherits the features from the bean module. (it also adds some more features like event propagation, resource loading & transparent creation of context).

→ SpEL: It's a powerful expression language that supports querying & manipulating an object graph at runtime. The language syntax is similar to Unified EL but offers additional features, most notably method invocation & basic string templating functionality.

→ AOP (Aspect-oriented programming): It helps in breaking down the Logic of the program into several distinct parts called as concern. The cross-cutting concerns helps in increasing the modularity & separate it from the business logic of an application.

→ Instrumentation: It provides us class.instrumentation supports & class loader instrumentation.

→ Messaging: This is used to serve as a foundation for messaging based application (used for messaging app. developing) We can map the messages with the help of method by various annotation present in messaging.

→ JDBC: It provides JDBC abstraction layer that removes the need of JTDS JDBC code & parsing database vender specific error code.

→ ORM: For using any ORM Tools like Hybernet, etc we have to use these module. It provides integration layer. through which we can integrate another ORM tool in our project.

→ Spring OXM: It provides an abstraction layer that supports object xml mapping. It ~~makes~~ use to ease the mapping between java objects & XML documents. OXM (Object XML mapping) It is extensible & hence it provides integration with various popular frameworks like Castor, JAXB, XmlBeans. & xStream.