**Spring**

5 layers in any enterprise application:

1. Presentation Layer
2. Controller Layer
3. Business Layer
4. Service Layer
5. Data Layer

By using struts, we can prepare only presentation and controller layer, but by using spring we can prepare all layers using different modules.

We can combine spring with combination any other frame work.

* By Rod Johnson

**Spring 5: In 2017**

**A screenshot of a cell phone

Description generated with very high confidence**

**Documentation:** [**www.spring.io**](http://www.spring.io)

* Light weight development with java Pojo’s.
* Dependency injection to promote loose coupling.
* Declarative programming with **Aspect-oriented programming(AOP**).
* Minimize boilerplate java code.

**Inversion of control:**

* It is design process for construction and management of object.
* Outsource to object factory

**Spring Container:**

* Create and manage objects(IOC)
* Dependency Injection: Inject objects dependencies

**Spring bean:**

* In spring, the objects in your application that are managed (Instantiated and assembled) by spring IOC container called beans.

**Dependency Injection:**

* The client delegates call to another object for responsibility of providing its dependencies.
* There are many types of injections.
  + Constructor Injection
  + Setter Injection
  + Literal values Injection

**Bean Scope:**

* Singleton: by default
  + One instance of the bean in cache
  + All requests for the bean will return a shared reference to same bean.
  + Can specify by using **scope=” singleton”**
  + Stateless data
* Prototype:
  + New instance for each request.
  + Can specify by using **scope=” prototype”**
  + can keep track of stateful data.

**Bean Life Cycle:**

**A close up of a sign

Description generated with very high confidence**

**A screenshot of a cell phone

Description generated with very high confidence**

* We can provide additional details in XML configuration like **init-method and destroy-method**
* **Access modifier**  
  The method can have any access modifier (public, protected, private)
* **Return type**  
  The method can have any return type. However, "void' is most commonly used. If you give a return type just note that you will not be able to capture the return value. As a result, "void" is commonly used.
* **Method name**  
  The method can have any method name.
* **Arguments**  
  The method cannot accept any arguments. The method should be no-arg.

**Note**: for prototype scoped beans, spring does not call destroy methods. Spring does not manage the complete life cycle of a prototype bean, just the container instantiates, configures and hands it to client with no further records of the instance. This means client code must clean up prototype scoped bean object and release the resources.

**Configuration with Annotations:**

* Special labels or markers that are added to java classes which provide meta-data about the class.
* They can be processed at compile time or run time.
* Xml configuration is verbose.
* Minimizes the Xml configuration
* **Spring will scan java classes for special annotations, if it finds register those beans in spring container.**

**Spring Auto Wiring:**

* Spring looks for the class that matches the property by type and will inject it automatically.
* Three types of injections:
  + Constructor Injection
  + Setter Injection
  + Field Injections: sets values using java technology called reflection.

@Autowired Annotation:

**It is optional if the target bean defines one constructo**r to begin with. If there are several constructors at least one must be annotated to tell the constructor which one to use.

@Qualifier Annotation:

We use this annotation to tell spring which implementation it should use for the dependency injection.

* Spring uses the **Java Beans Introspector** to generate the default bean name.

If the annotation's value doesn't indicate a bean name, an appropriate name will be built based on the short name of the class (with the first letter lower-cased).

For example:

HappyFortuneService --> happyFortuneService

However, for the **special case of when BOTH the first and second characters of the class name are upper case**, then the **name is NOT converted**.

For the case of RESTFortuneService

RESTFortuneService --> RESTFortuneService

*No conversion* since the first two characters are upper case.

**@PostConstruct and @PreDestroy Method Signatures:**

* postConstruct annotation is used for init method. This method will be executed after the general process of the bean class.
* When Spring creates the beans it follows this general process
  + 1. Construct an instance of class  
    2. Inject dependencies  
    3. Set properties etc (@Value)
* preDestroy is used for destroy method and it is called when we close the context object.