Spring Framework:

Framework for building Java Applications

Simpler and lightweight alt to J2EE

Early version of J2EE complex object management

Multiple deployment descriptors

Multiple interface

Poor Performance

Rod Johnson

Lightweight Object management tool : Object Factory/Bean Factory/Application Context

Spring :

=> highly Modular

=> Independent to be used : loose coupling among modules

=> Java POJOs : Lightweight dev process

Relationship among resources of Spring application

1. IoC : Inversion of control
2. DI : Dependency Injection
3. AOP : Aspect Oriented Programming (proxy)

Modules

- Core Container : Bean Factory
- Web Layer : MVC Framework
- Data Access Module : JDBC/ORM/Transaction
- Infra : AOP/Messaging
- Testing : JUnit/Mocking

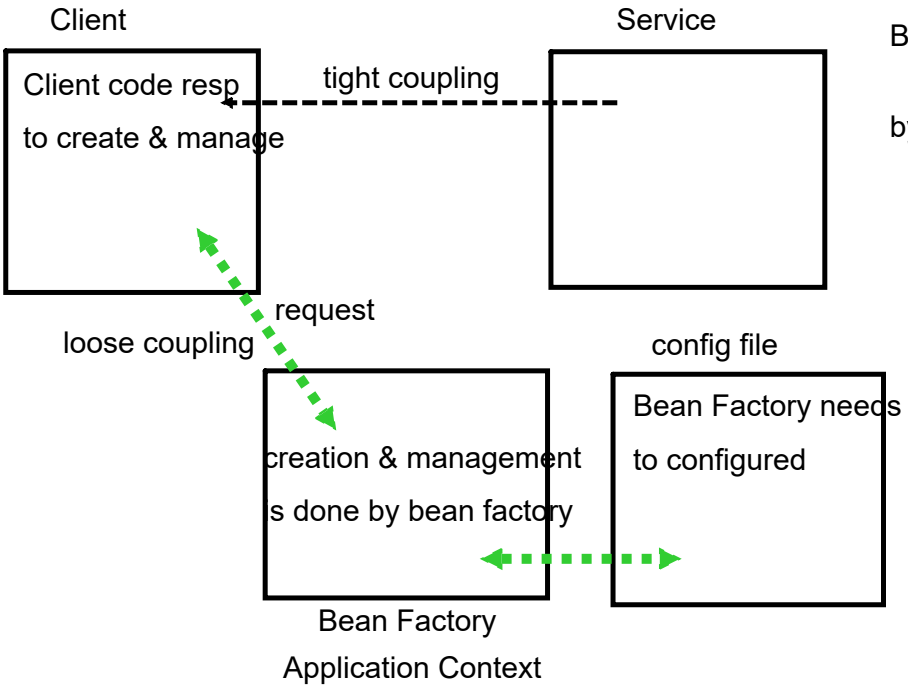
Spring Projects:

- Spring Data
- Spring Cloud
- Spring security
- WebFlux

1. IoC : Inversion of Control

Outsourcing the Creation and Management of Object :

Object Factory/Bean Factory



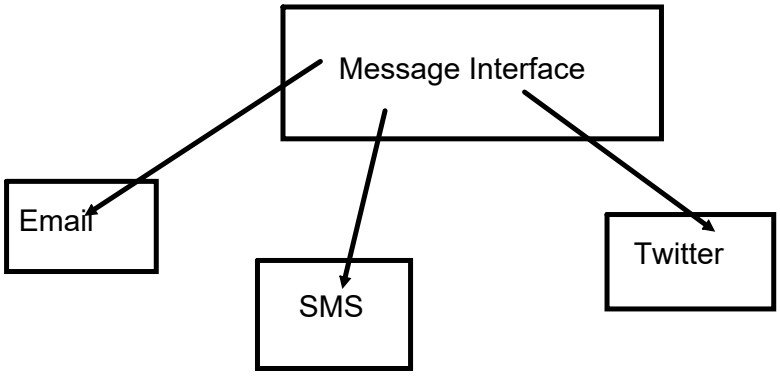
Beans :

Objects managed by spring containers

Spring Jar files : https://repo.spring.io/release/org/springframework/spring/

Configuration of Bean Factory

1. XML based config (legacy)
2. Annotations
3. Pure Java Code



```
<bean id="mservice" class="com.wf.training.spring.service.EmailService">
```

key: by which bean is exposed

class: whose bean we want to create

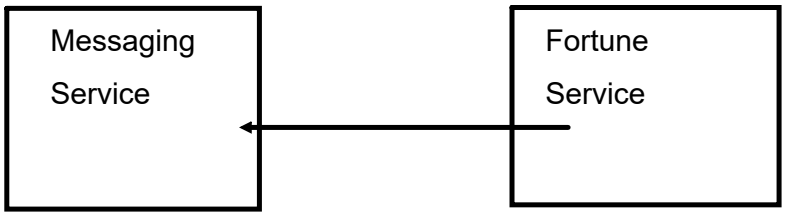
DI : Dependency Injection

Creation of complex object might have dependency on other object :

injecting those dependency with the help of Spring Container

XML based config:

1. Constructor based
2. Setter based



literal values should not be integrated in config file

==> property file to keep the literal values

KEY - VALUE PAIRS

Referred by SpEL

Scope of Beans

Default scope : Singleton

Only one instance which shared among all calls

Scope possibility :

singleton

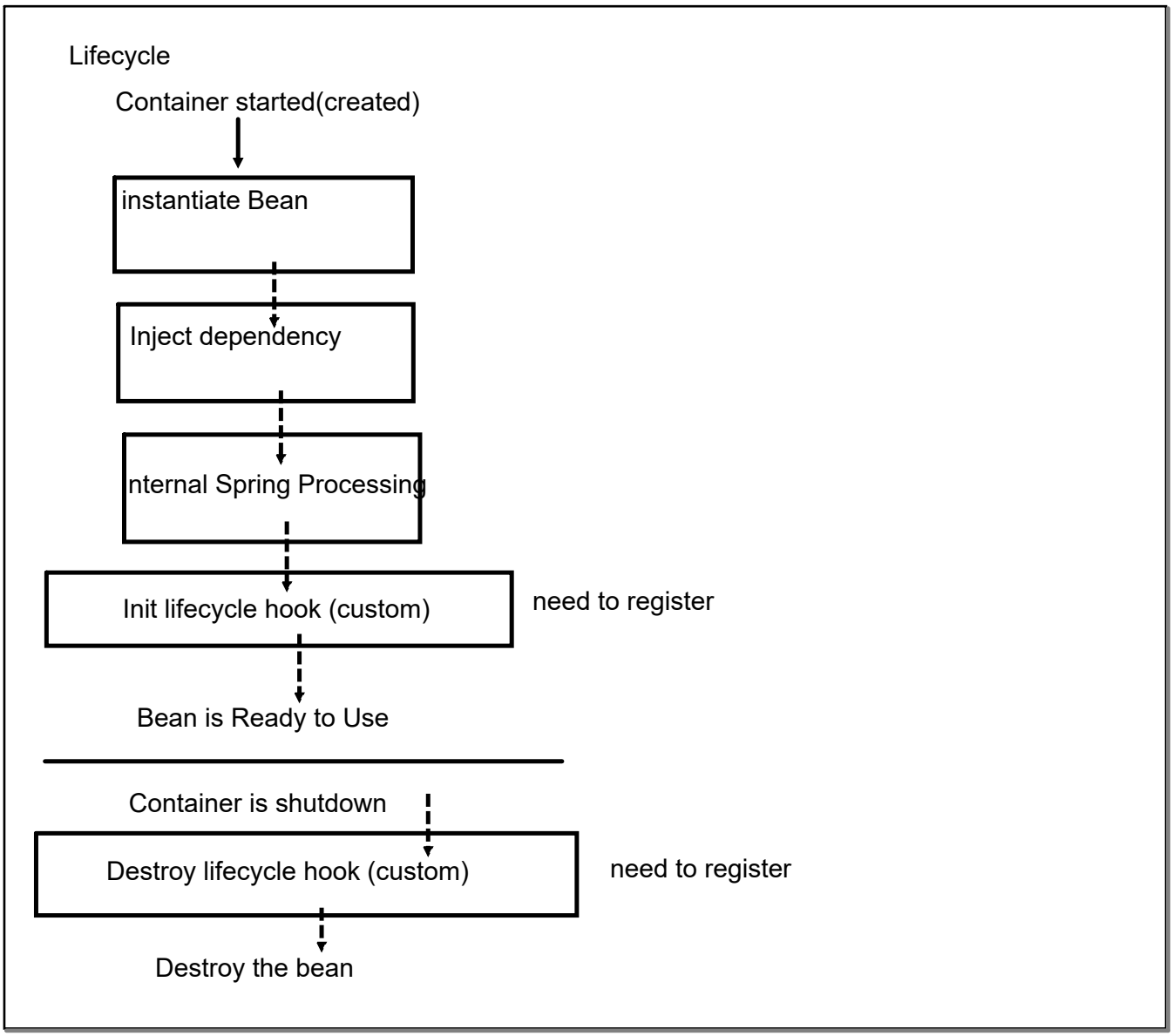
prototype : a new bean would be created

Web Context :

request

session

global-session

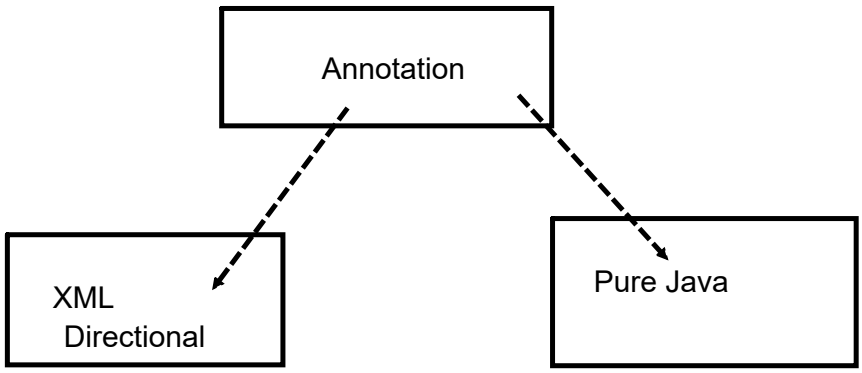

Lifecycle
Container started(created)
instantiate Bean
Inject dependency
nternal Spring Processing
Init lifecycle hook (custom)
need to register
Bean is Ready to Use
Container is shutdown
Destroy lifecycle hook (custom)
need to register
Destroy the bean

Life cycle hook method

1. any name
2. any access modifier
3. not static
4. they may return values but can't capture
5. No parameter

Prototype : Spring container does not maintain lifecycle

Annotation based config



@Component : informing spring to create and manage bean of that class

Every bean must be exposed by an id (Class name (with first char small) is default id)

Annotation based DI

1. Constructor
2. Setter
3. Field