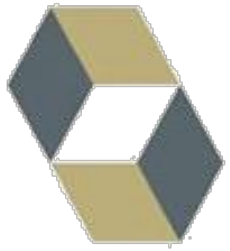


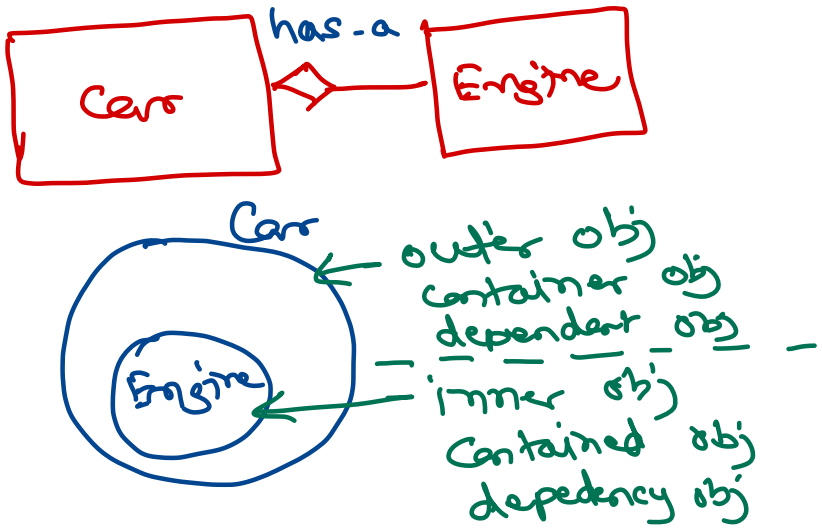
HIBERNATE

Trainer: Nilesh Ghule

Wake up from Hibernate, Spring up!!!

Agenda

- Spring Introduction
- Dependency injection
- XML config DI
- Bean factory
- Application context
- XML based DI
  - properties
  - beans
  - collections
- Polymorphism with Spring beans
- Annotation config
- Mixed config
- Auto-wiring



Introduction

Java is simple — 1995 - Java 2 | Java 5 i Java 8

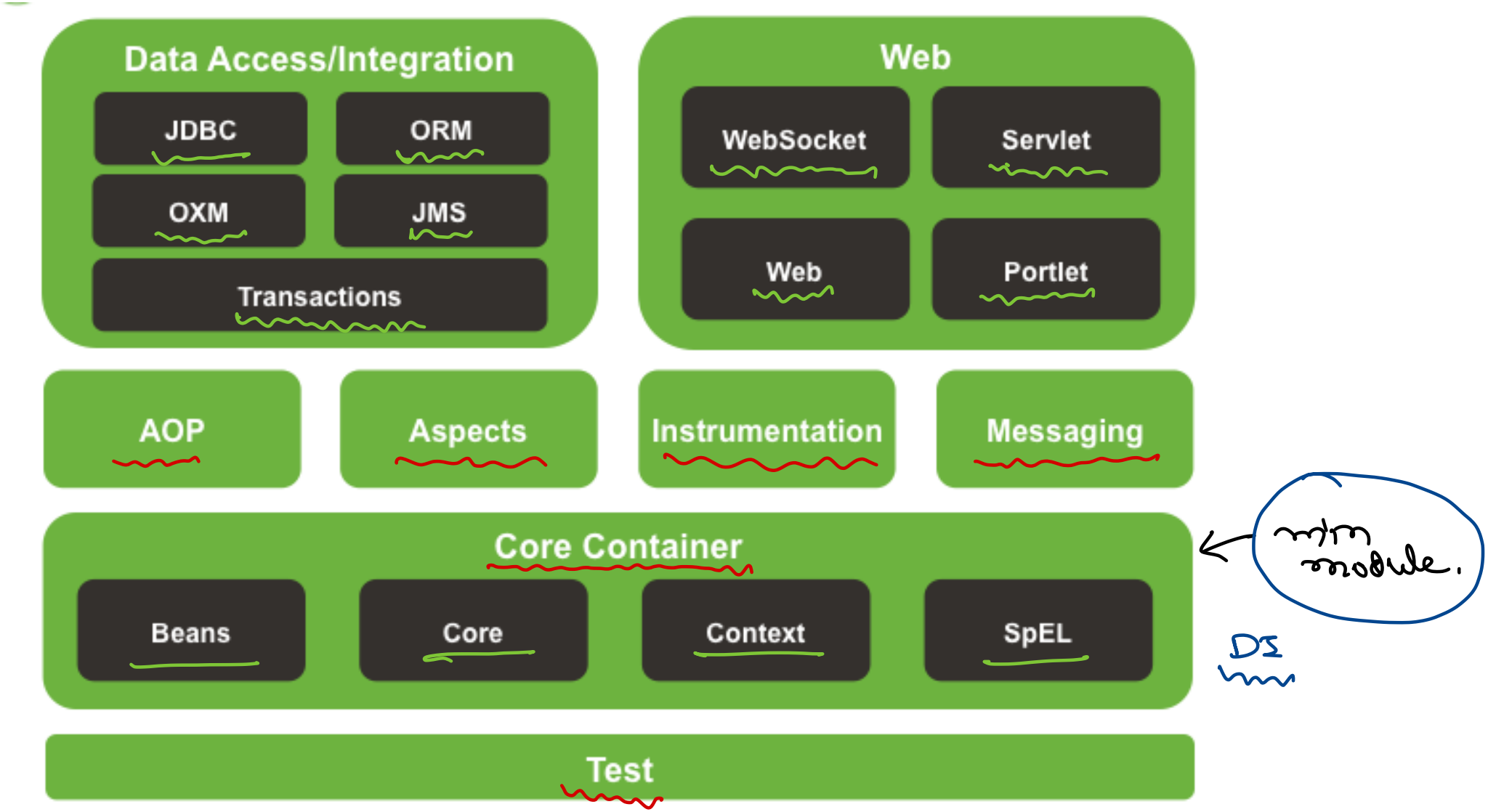
- Spring is light-weight comprehensive framework to simplify Java development.
  - light-weight → minimal overheads
  - comprehensive → small framework - can develop many app. web, db, batch, jms, ...
  - simplify → than plain java devel.
- Spring framework is originally developed by Rod Johnson. 2003
- Spring 3 version added support of annotation configuration. 2009
- Spring pros & cons
  - Flexible ✓ NO all or nothing.
  - Modular ✓ core, orm, web, jms, ...
  - Test driven ✓ built in unit test.
  - Maintainable ✓ n-tier applns, aop, ...
  - Extendable ✓ design with interface.
  - Tedious configuration ✗ heavy config.

① coding
② testing ←
③ frameworks + libraries
④ build & deploy.

Spring Boot

Simplified Configuration.

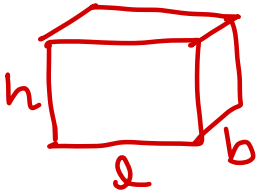
Spring architecture



Spring Dependency Injection

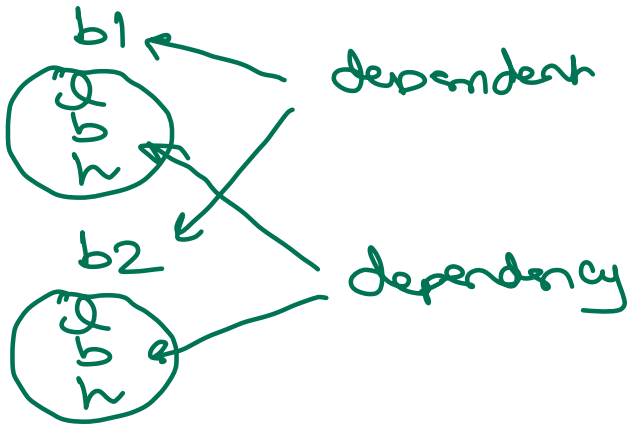
initialization

- Spring container injects dependency beans into dependent beans.
- Spring container is also called as IoC container.




```
class Box {
    int len, br, ht;
    Box() { -- }
    Box(l, b, h) { -- }
    get/set() { -- }
    toString() { -- }
    double calcVolume() {
        return len * br * ht;
    }
};
```

```
main() {
    Box b1 = new Box();
    b1.setLen(5);
    b1.setBr(4);
    b1.setHt(2);
    sys1 = b1.calcVolume();
    Box b2 = new Box(5, 4, 3);
    sys2 = b2.calcVolume();
}
```



Design Principle : Inversion of Control

Design Pattern : Dependency Injection

XML based DI

SJS 3.x

- Spring beans are declared into *Spring bean configuration file*.
- Bean factory or application context reads the file. It instantiate and initialize bean objects at runtime.
- Dependency Injection
  - Setter based DI
  - Constructor based DI
  - Field based DI

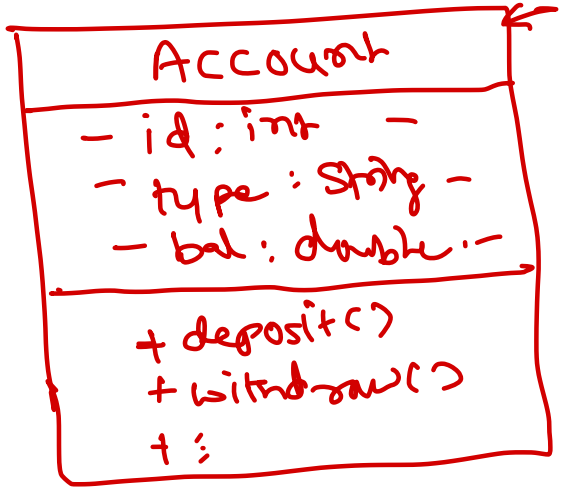
DI

```
main() {
    Box b1 = new Box();
    b1.setLen(5);
    b1.setBr(4);
    b1.setHt(2);
    vol1 = b1.calcVolume();

    Box b2 = new Box(5, 4, 3);
    vol2 = b2.calcVolume();
}
```

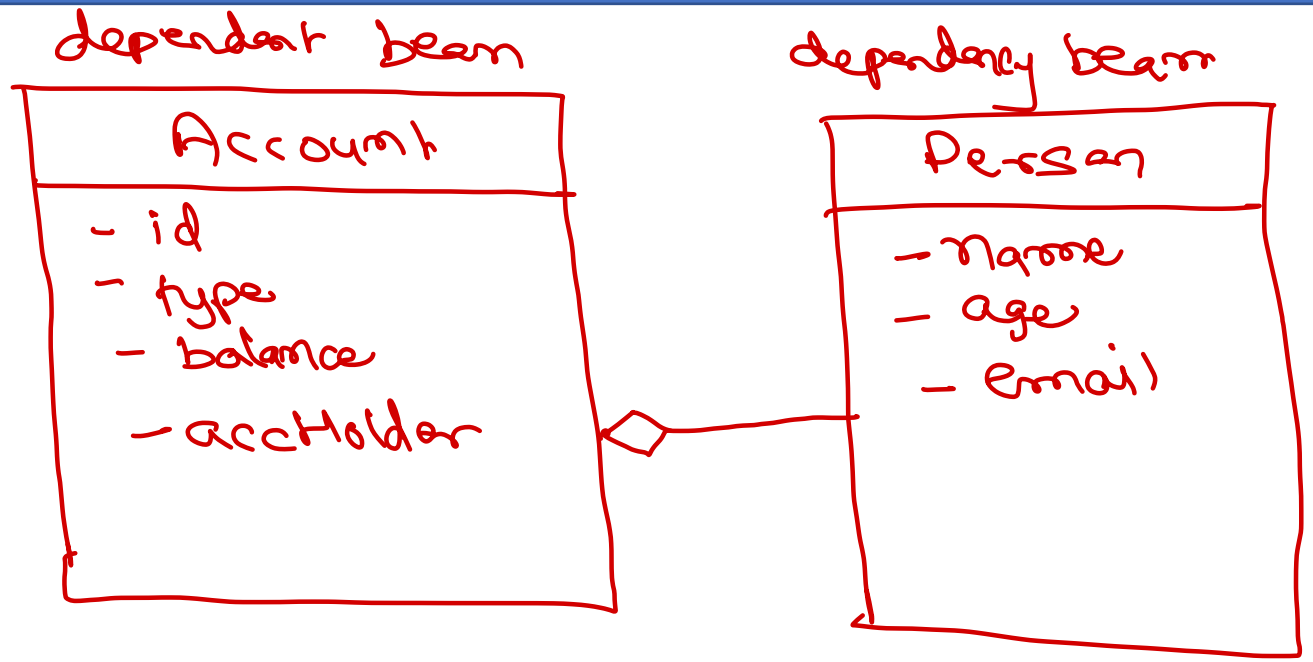
← automated setter based DI

← automated ctor based DI



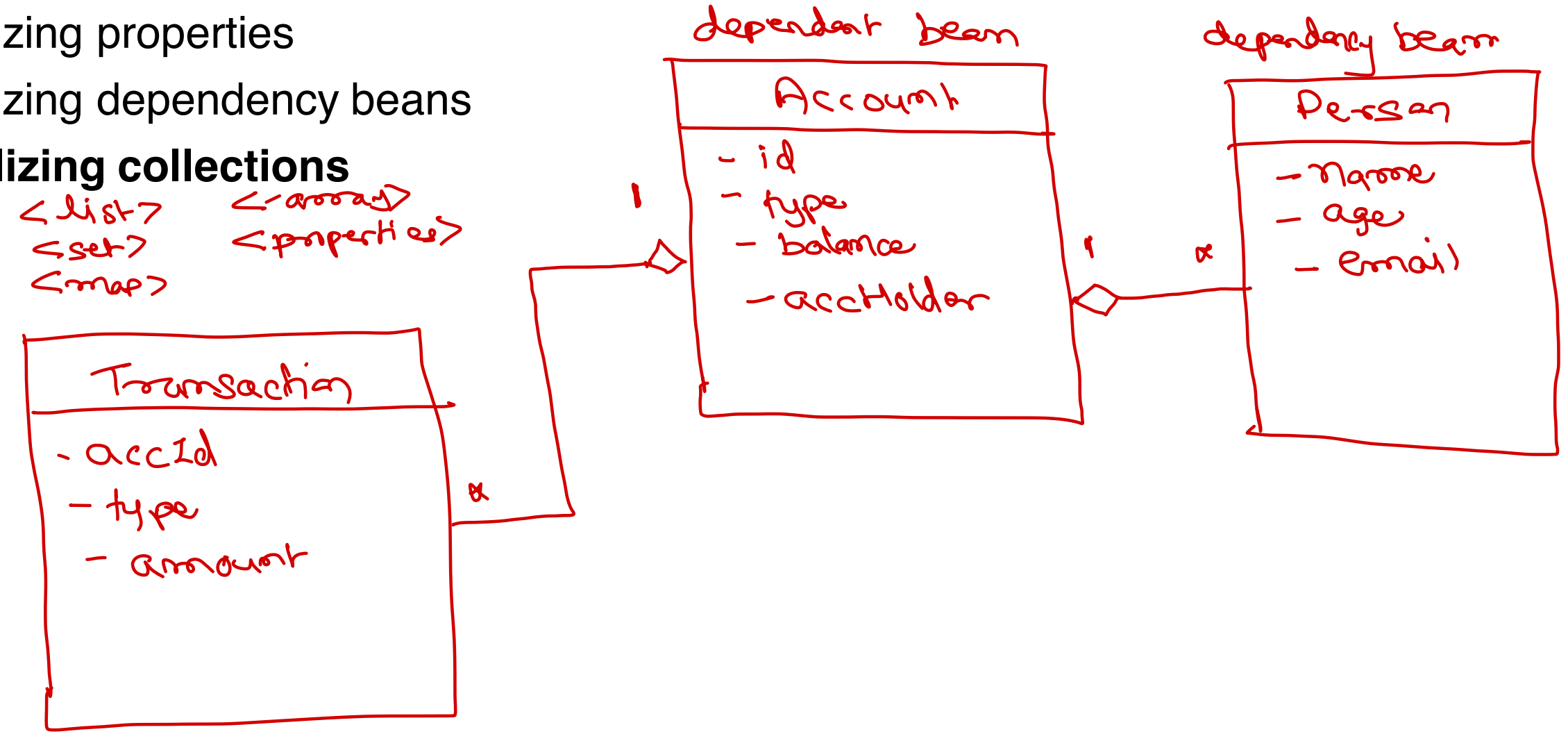
Dependency Injection

- Initializing properties
- **Initializing dependency beans**
- Initializing collections



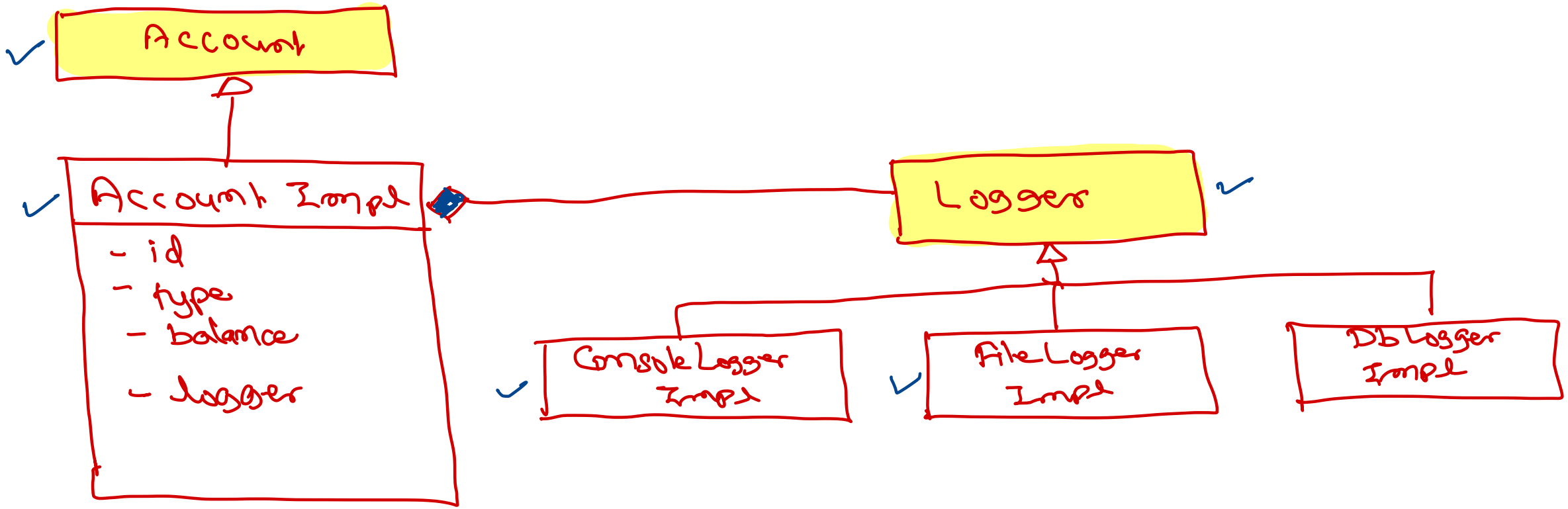
Dependency Injection

- Initializing properties
- Initializing dependency beans
- **Initializing collections**
  - <list> <array>
    <set> <properties>
    <map>



Dependency Injection

- One interface may have multiple implementations.
- This makes application more extendable and maintainable.
- Desired implementation can be plugged in using DI.



Annotation config or Java Config

- Does spring configuration without using any XML file.
- The bean creation is encapsulated in @Configuration class and beans are represented as @Bean methods.
- Annotation configuration is processed using AnnotationConfigApplicationContext.

Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>