Spring Framework

Ground rules

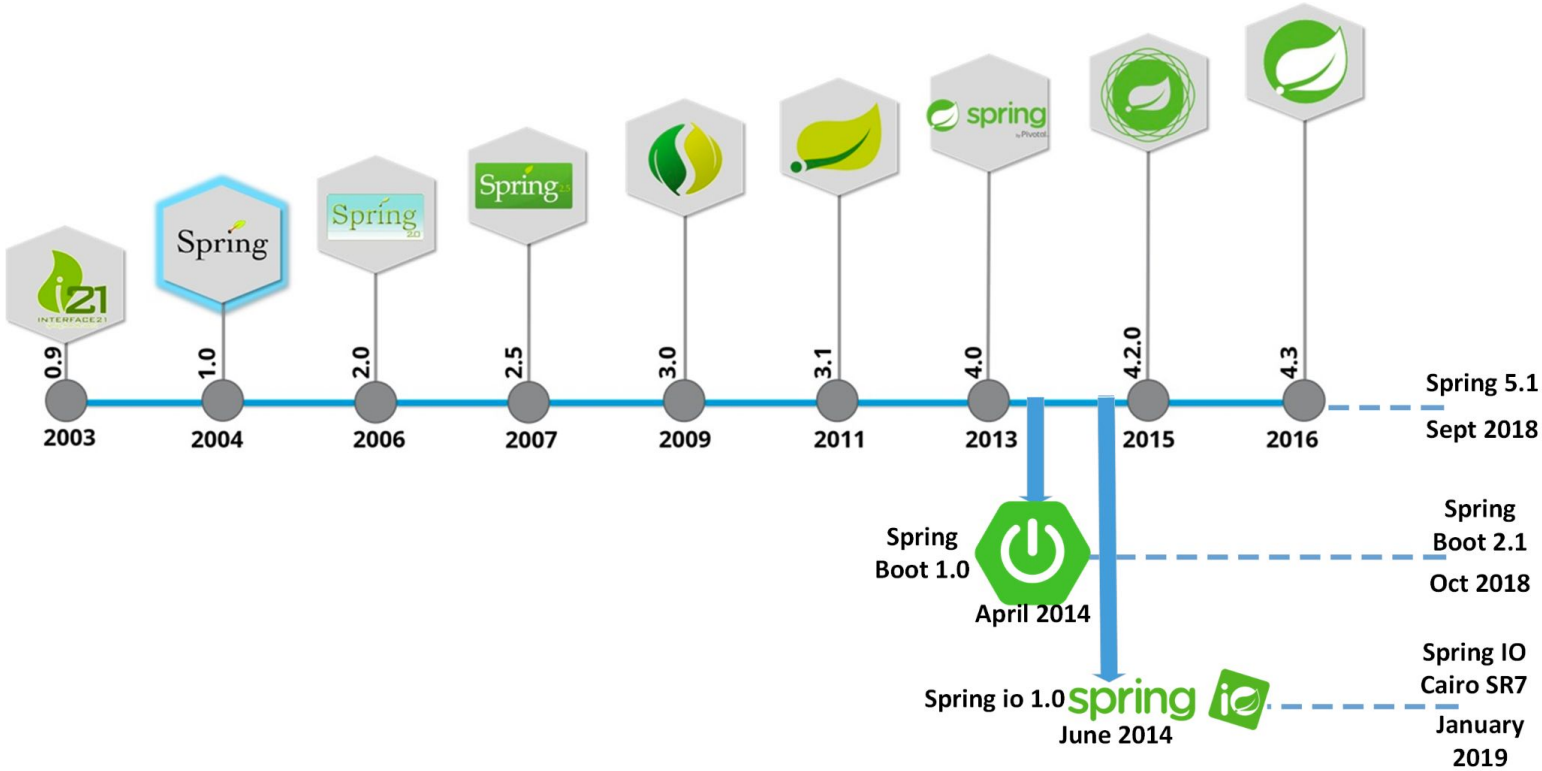
- Ask anytime when you a question
- Allow to ask in Thai
- I’m not a native English speaker, so please ask for repeat if it not clear
- Will temporary not present in the meeting > 5m please drop message, brb->b
- Take group photo screenshot before Lunch break and before end of the day class

What’s Spring framework

Spring Framework is a **Java platform** that provides **comprehensive infrastructure** support for developing Java applications. Spring handles the infrastructure so you can focus on your application.

Spring enables you to build applications from “**plain old Java objects**” (POJOs) and to apply enterprise services non-invasively to POJOs. This capability **applies to the Java SE** programming model and **to full and partial Java EE.**

History of Spring framework



Spring

Spring Boot

BUILD ANYTHING

Spring Boot is designed to get you up and running as quickly as possible, with minimal upfront configuration of Spring. Spring Boot takes an opinionated view of building production-ready applications.

Spring Cloud

COORDINATE ANYTHING

Built directly on Spring Boot's innovative approach to enterprise Java, Spring Cloud simplifies distributed, microservice-style architecture by implementing proven patterns to bring resilience, reliability, and coordination to your microservices.

Spring Cloud Data Flow

CONNECT ANYTHING

Connect the Enterprise to the Internet of Anything—mobile devices, sensors, wearables, automobiles, and more. Spring Cloud Data Flow provides a unified service for creating composable data microservices that

Spring Projects

Modular by design

Spring Framework	Spring Boot	Spring Data Flow
Spring Cloud	Spring Data	Spring Integration
Spring Batch	Spring Security	Spring AMQP
Spring LDAP	Spring WebFlow	Spring REST Doc

https://spring.io/projects

Why Spring is popular ?

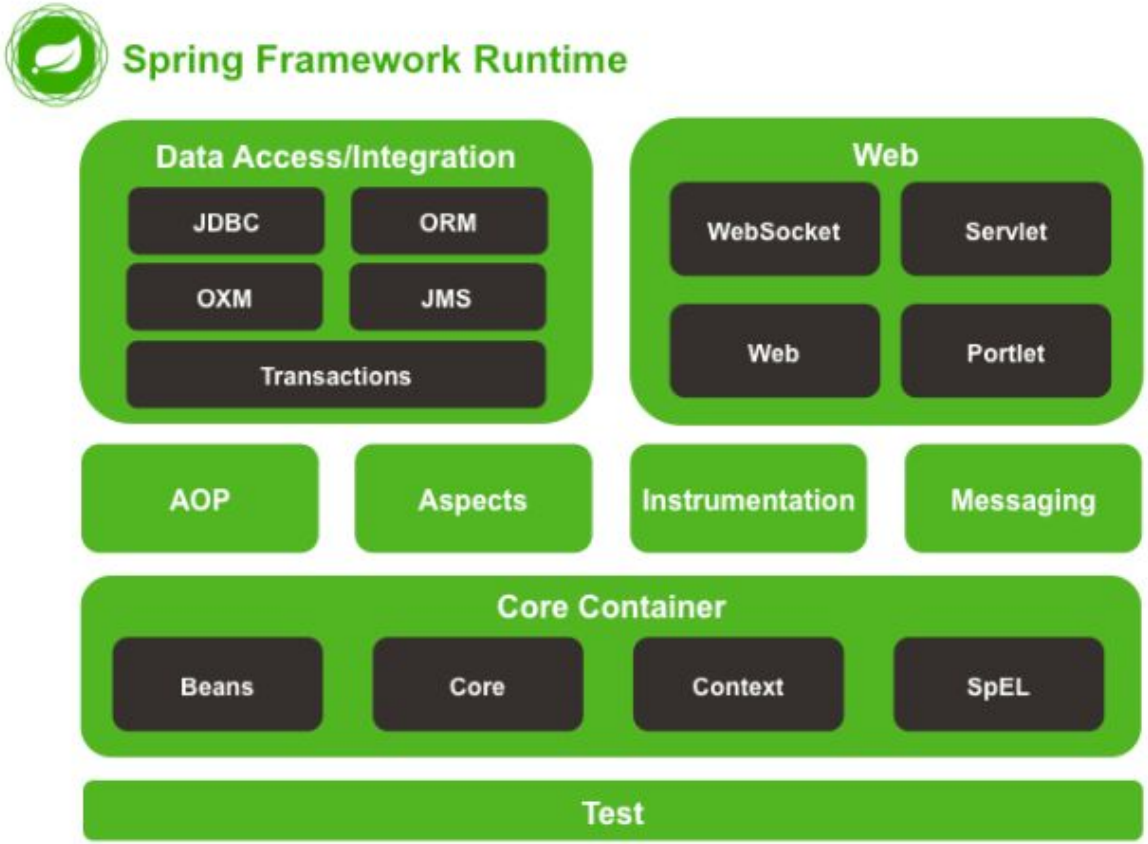
Enable testable code

No plumbing code

Flexible architecture

Staying current

Overview of Spring Framework

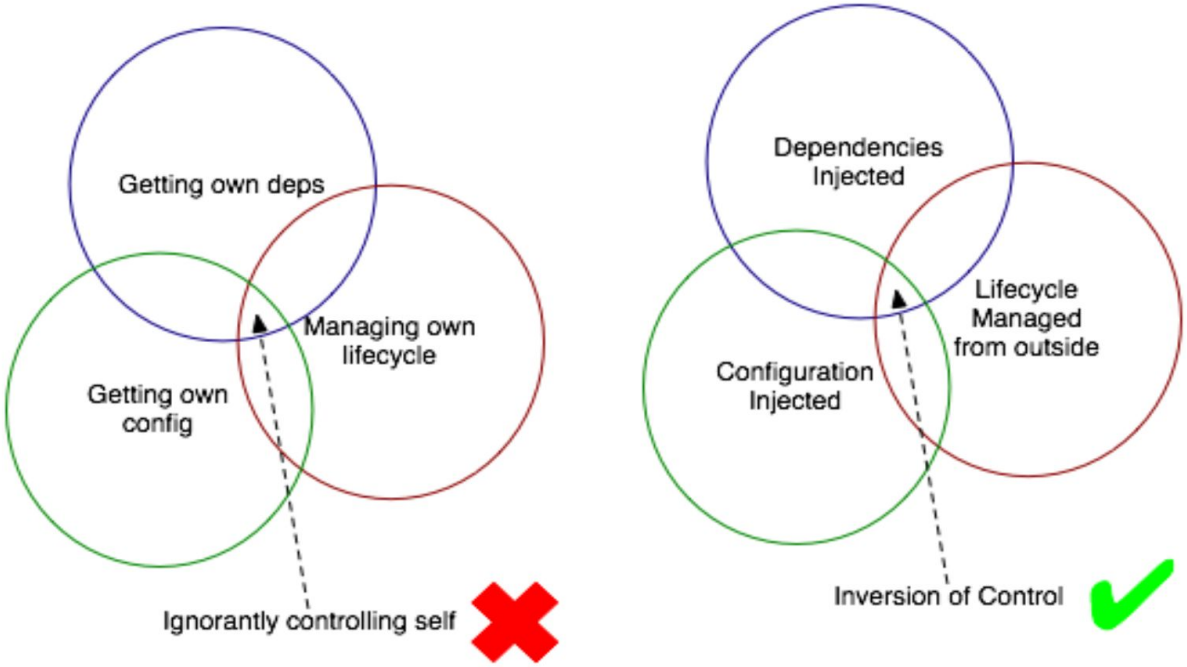


Core Container

Core and Beans
Context
Expression language

Core and Beans

Provide the fundamental parts of framework Including **IoC** and **Dependency Injection**



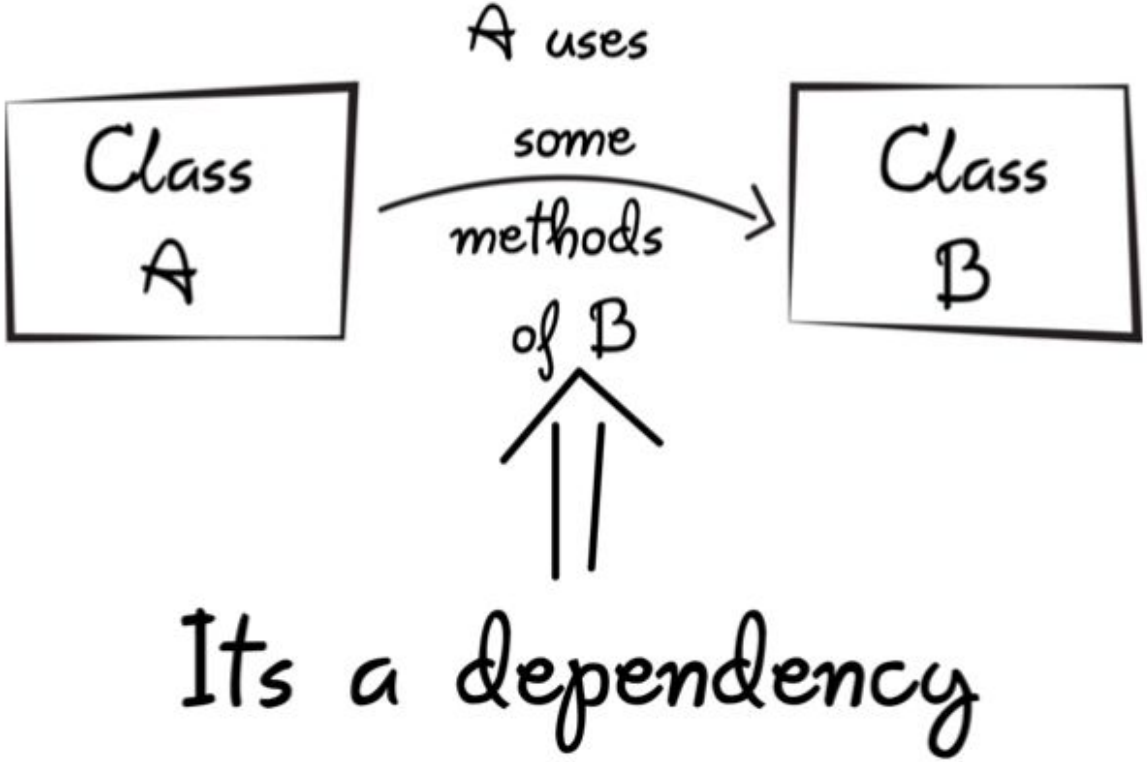
https://www.martinfowler.com/articles/injection.html

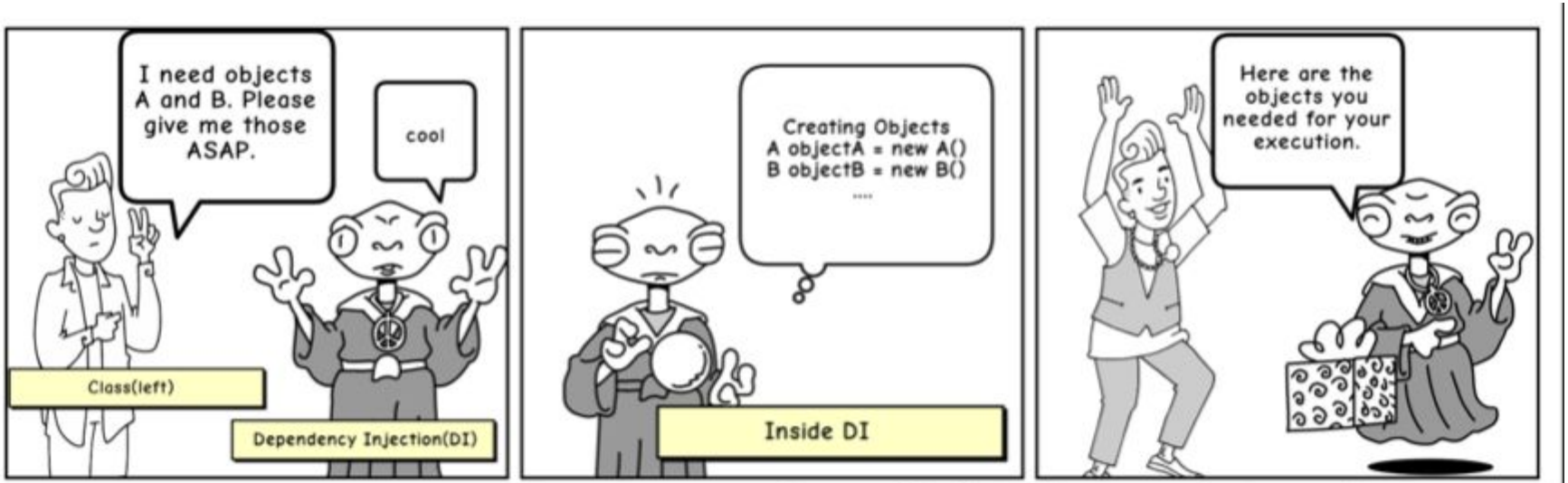
Inversion of Control (IoC)

Concept in application development

Don’t call me, I will call you

Dependency Injection

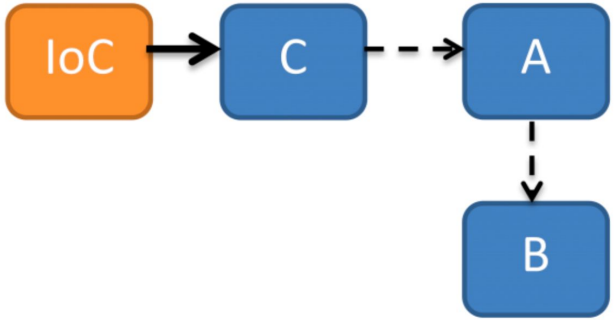



I need objects A and B. Please give me those ASAP.
cool
Class(left)
Dependency Injection(DI)
Creating Objects
A objectA = new A()
B objectB = new B()
....
Inside DI
Here are the objects you needed for your execution.
This comic was created at www.MakeBeliefsComix.com. Go there and make one now!

Class Dependencies

Service Location / Active Calling

IoC / DI / Auto-Wiring / Passive Calling



Types of Dependency Injection

Constructor injection

Property/Setter injection

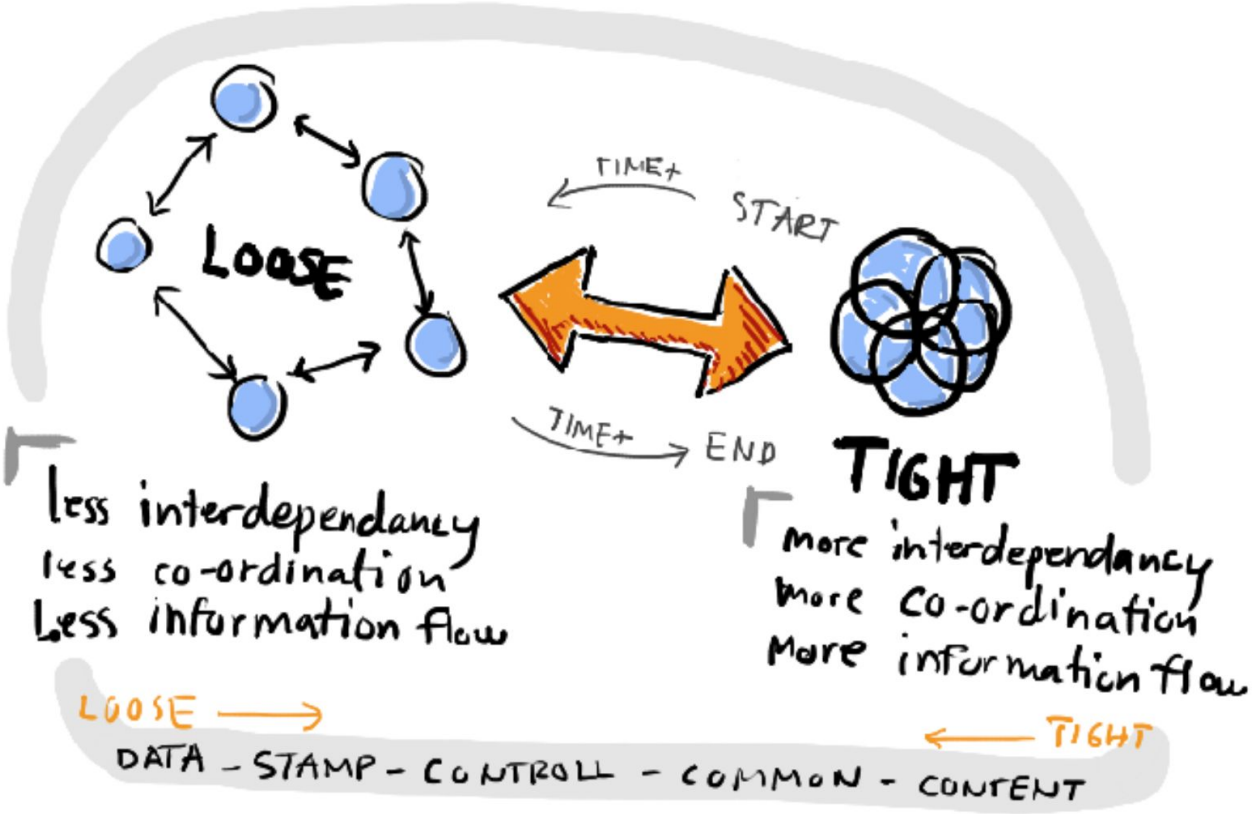
Method injection

Interface injection

Benefit of Dependency Injection

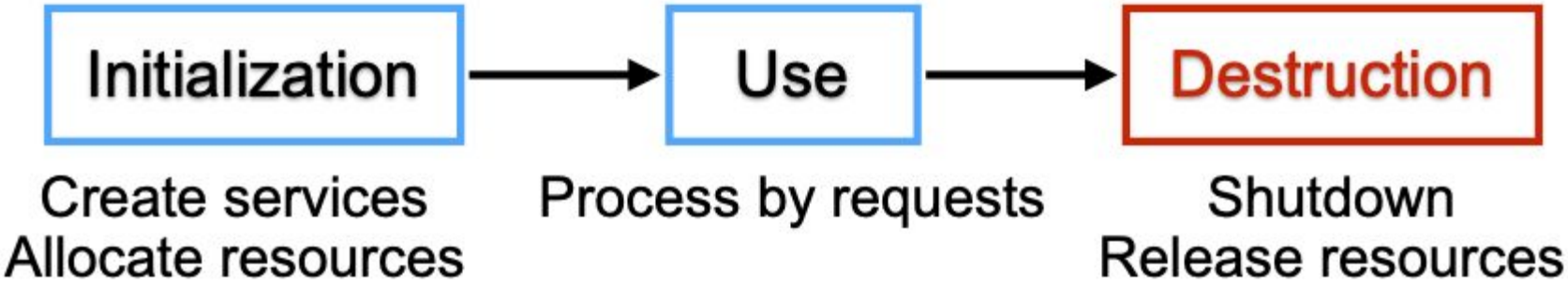
- Reduces noise in your code Reduces object coupling
- Reduces defects that arise from incorrect construction
- Focus on the API contract

Tight coupling
Loose coupling

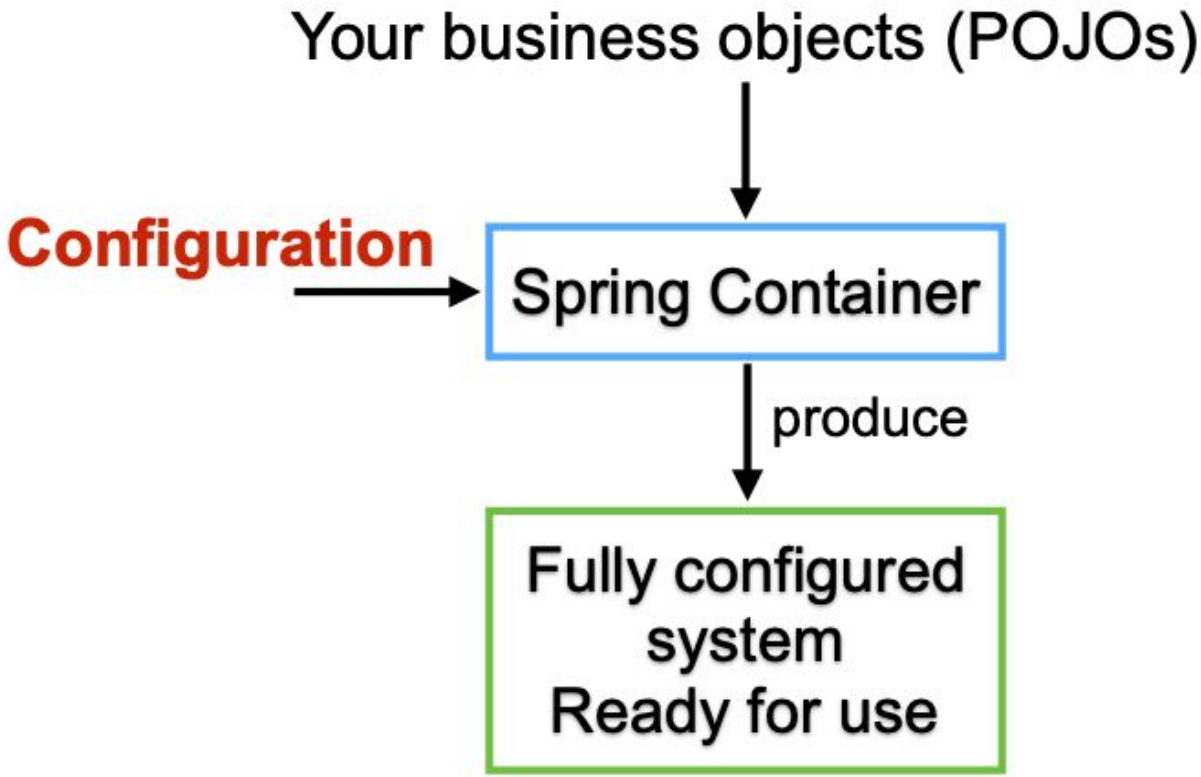


Back to Spring Framework

Application Lifecycle



Spring IoC container



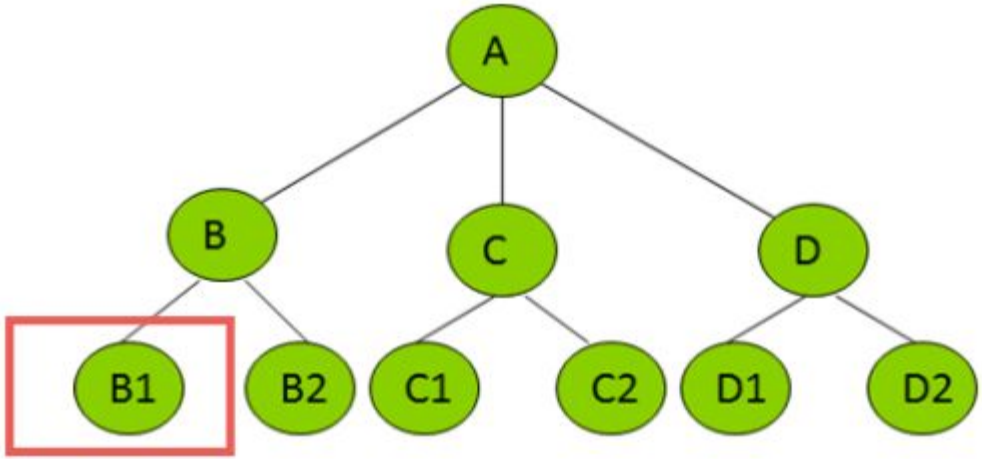
Configuration

From XML file

Annotation-based configuration (2.5)

Java-based configuration (3.0)

Spring IoC container



Bean B1

Break 15 minutes

Beans

Bean Definitions

Package-qualified class name
Bean behavioral (scope, lifecycle, callback)
Reference to other beans
Other configuration setting to create new object

Spring IoC container manages one of more beans

Beans are created with configuration

Bean Definitions

Property	Section
Class	Instantiating beans
Name	Naming beans
Scope	Beans scopes
Constructor arguments	Dependency Injection
Properties	Dependency Injection
Autowiring mode	Autowiring collaborators
Lazy initialization mode	Lazy-initialized beans

Bean Scopes

Scope	Description
singleton	Single instance for each container
prototype	Single bean definition to any number of object instances.
request	HTTP request
session	HTTP session
application	ServletContext

Change scope of bean

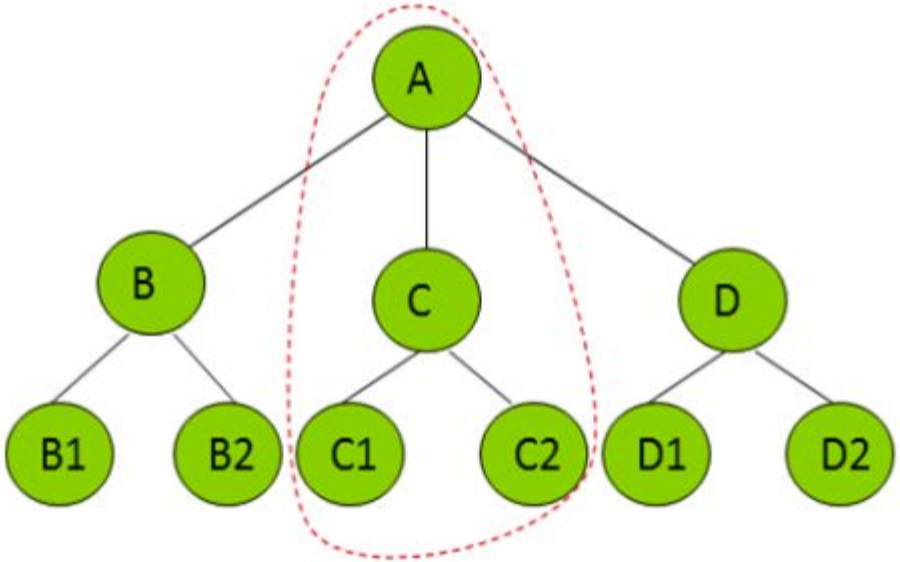
```
@Component
@Scope(ConfigurableBeanFactory.SCOPE_PROTOTYPE)
public class RealRandom implements MyRandom {

    public String number;

    @Override
    public int nextInt(int bound) {
        return new Random().nextInt(bound);
    }

}
```

Lazy-load dependencies



BeanFactory ?

Interface defines basic functionality for Spring container

Factory design pattern

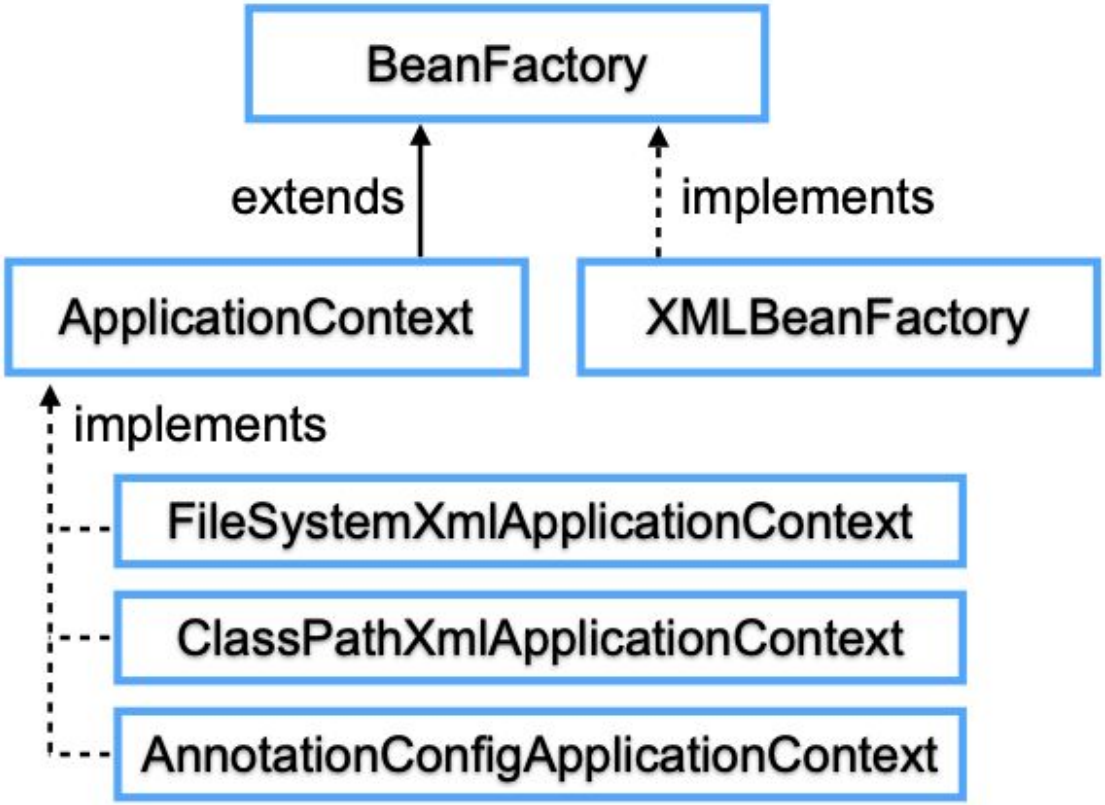
Load beans from configuration source

Instantiated the bean when requested

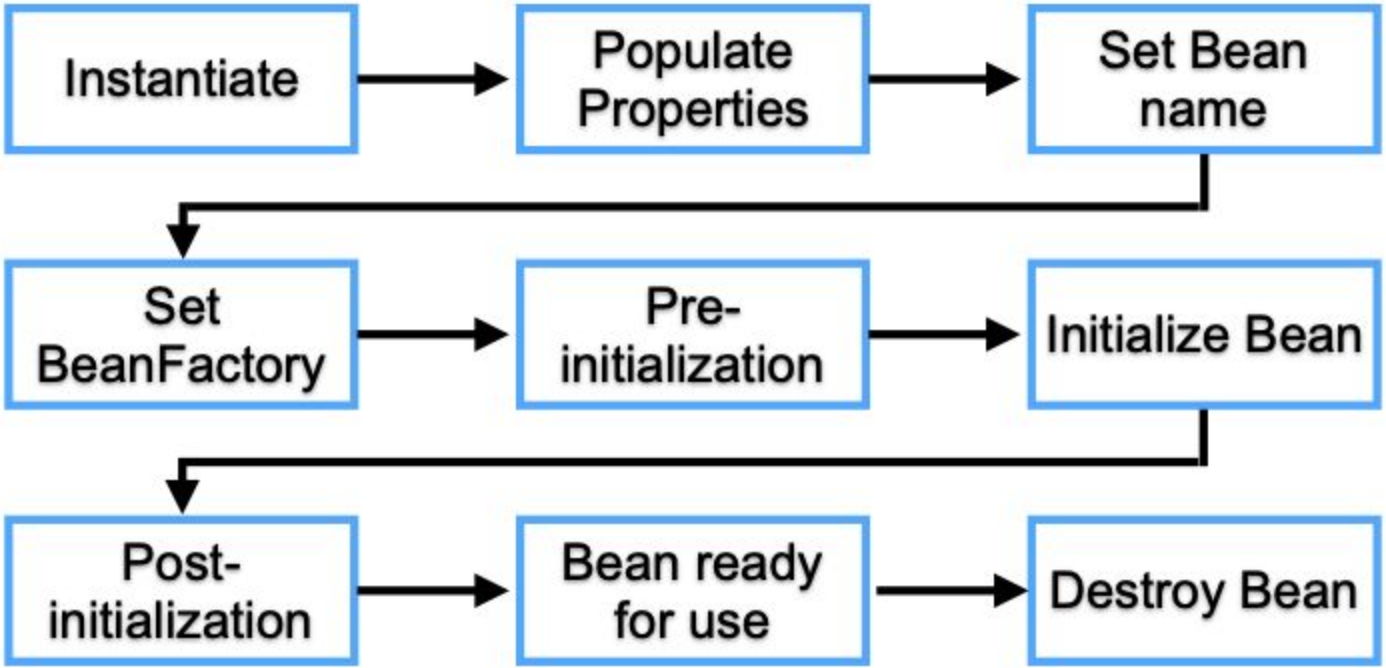
Wire dependencies and properties for beans

Manage the bean lifecycle

BeanFactory ?



Lifecycle of BeanFactory

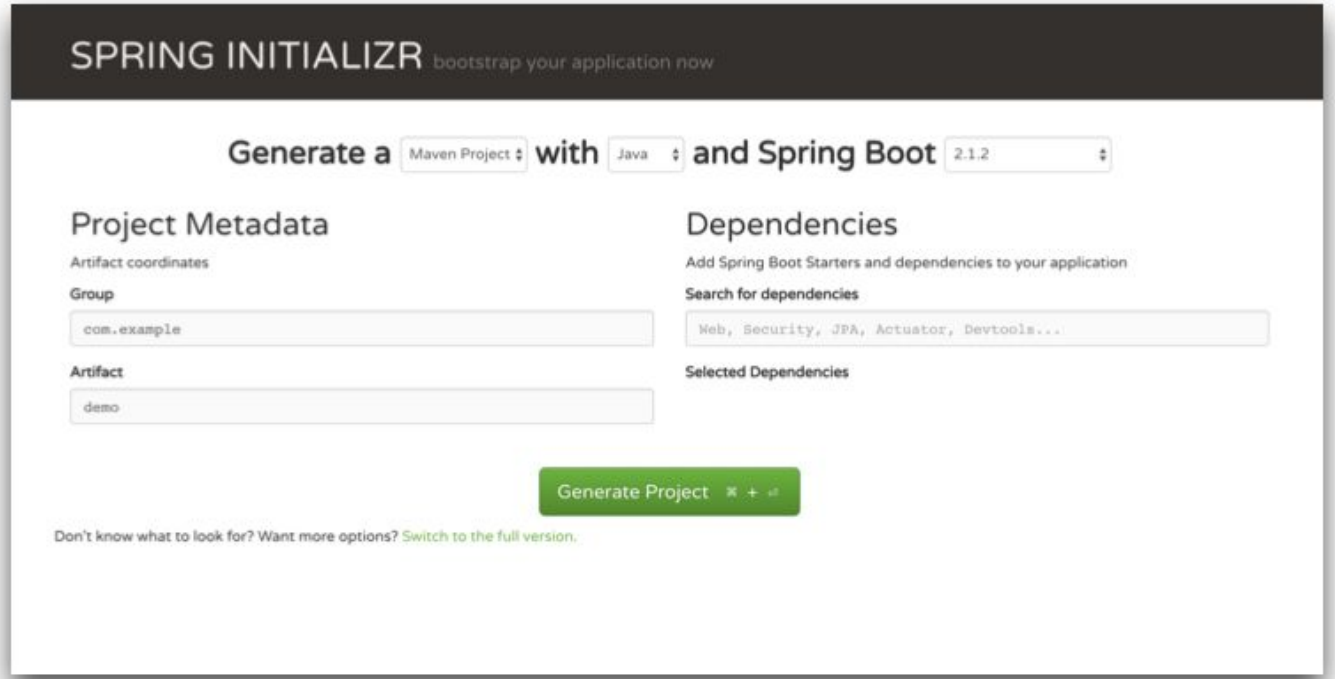


Let’s start

exercise

Create new Project

Use **spring initializr**



https://start.spring.io/

Layer of application

Using Spring to manage dependencies

@Component

@Autowired

Constructor and Setter injection

@Primary

@Qualifier

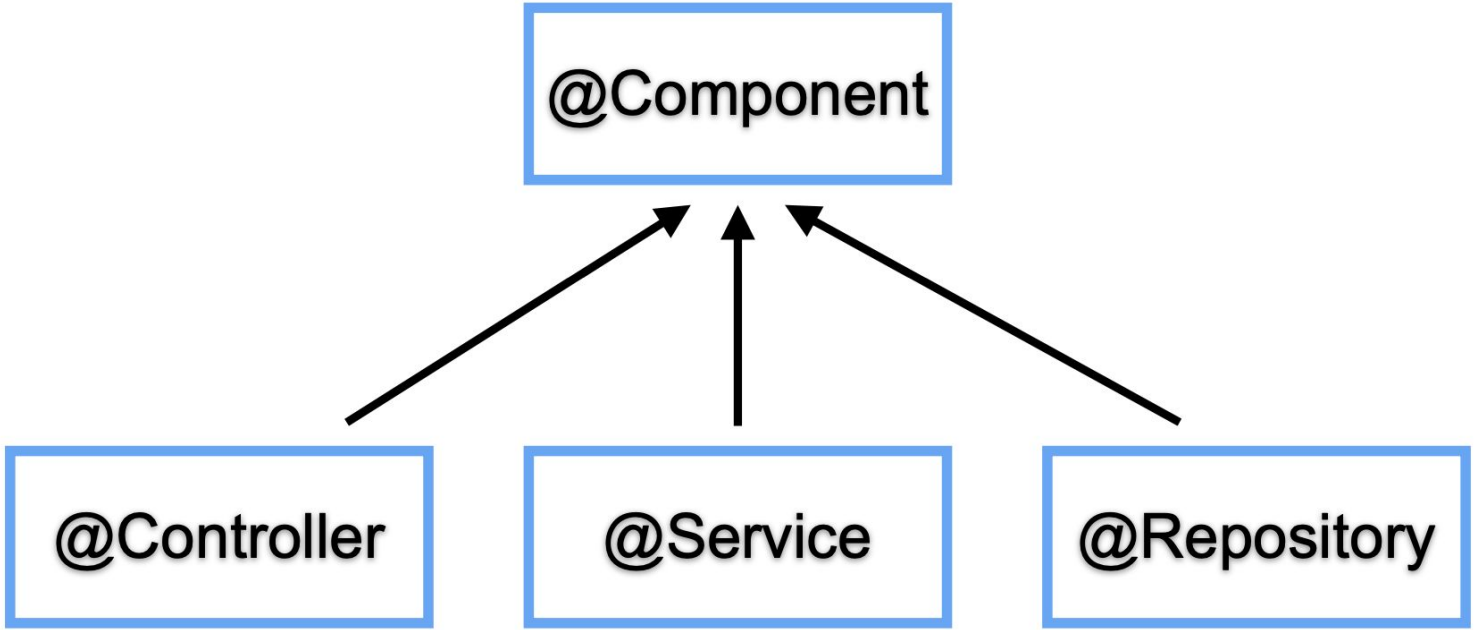
Scope of beans

Singleton
Prototype

What are difference of …

@Component
@Controller
@Service
@Repository

What are difference of …



What are difference of …

@Component

Generic stereotype for any component or bean

@Controller

Stereotype for the presentation layer (Spring MVC)

@Service

Stereotype for the service layer

@Repository

Stereotype for the persistence layer

Take a group Photo