Spring:

What is Spring?

* Spring mean a family of project made by Spring Framework

Spring Framework aimed at removing complexity and make certain things easier to access; reducing boiler plate code.

* Lead to several Spring projects built on top of Spring Framework tailored to different things.
* Spring Boot: Provided a new and faster way of Spring-based applications. Simplified a deployment process
* Spring Cloud: Simplified development making use of distributed architecture.

Why Spring?

* Spring makes it easy to create Java enterprise applications.
* Flexible, modular, backward compatible.
* Large and active community.
* Continually innovates and evolves.

Why Spring Boot:

* Make getting started with Spring quick and easy
* Can build different type of Spring Project

Notable Features:

* Auto-configuration
* Standalone
* Opinionated

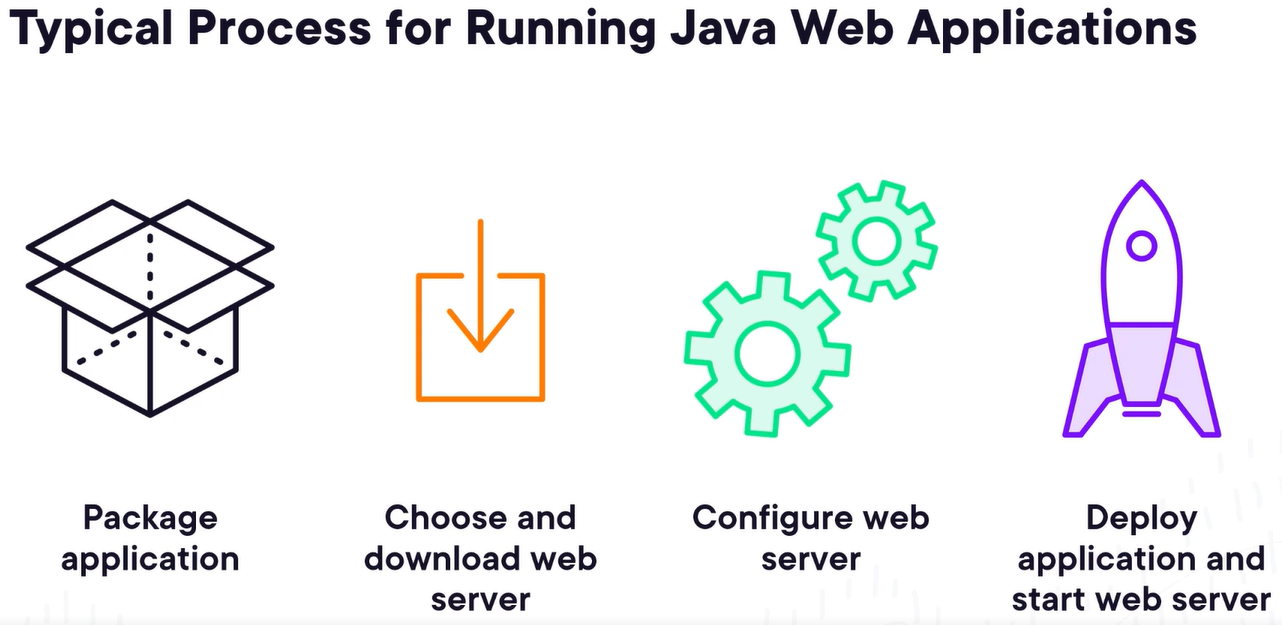
Auto-configuration:

Automatically configure Spring application based on the dependencies added to it.

Setting up Auto-configuration is effortless (adding @EnableAutoConfiguration)

Standalone:

Typical Java Web Application:



Springboot:

Package -> run.

Opinionated:

Favor convention over configuration. Is designed to get up and run as quickly as possible possible.

Soft opinionated, easy to override.

Spring Foundation:

Software Framework: Universal, reuseable software environment. Provide particular functionality as part of a larger software platform to facilitate development of software application, products and solutions.

A screenshot of a computer

AI-generated content may be incorrect.

Enterprise Java development at the time was complicated.

Spring Framework became foundation for Spring Projects.

6 key area of Spring Project:

* Core
* Web
* AOP (aspect oriented programming)
* Data Access
* Integration
* Testing

Spring Core:

A close-up of a computer screen

AI-generated content may be incorrect.

Foundational module of Spring Framework

Provide support for validation, data binding, type conversion, …

Center is Dependency Injection.

Objects might dependent on other objects. (example, computer object depend on HD and Memory)

Dependency Injection is dealing with the way objects fulfill their dependency objects.

1. Objects fulfill its own dependency. (easy, but tightly coupled)
2. Declare what it depends on and sth else fulfill the dependency. (flexible, loosely coupled)

Spring Core is a dependency injection container: creates and maintain objects and their dependencies.

Spring Web:

* Framework for handling web request, via Spring Web MVC or Spring Web WebFlux.

Spring Web MVC:

(Java) Servlet:

An object receive a request and generate a response based on that request.

A two rectangular cards with text

AI-generated content may be incorrect.

Challenge:

* Low level API (easy to mix up concern)
* Hard to use
* Less productive

Spring Web MVC:

A group of rectangular objects

AI-generated content may be incorrect.

Adv:

* Higher level API (Clear separation of concern)
* Easier to use
* More productive

MVC: Model, View, Controller.

Spring WebFlux:

Reactive Programming:

* A way of programming that focus on streams of data and how they change.
* React rather than wait.
* Spring WebFlux
* A different way of handling web request
* Asynchronous execution
* Doesn’t block (wait)

Traditional Flow:

A screen shot of a message

AI-generated content may be incorrect.

When request is waiting on something else, it is considered blocked.

Reactive Flow:

A screen shot of a message

AI-generated content may be incorrect.

Doesn’t wait. When the operation is done, the code will be notified, if not it will continue executing other operation. When done, code will execute subsequences steps.

Spring AOP:

AOP:

Aspect-oriented Programming. Aim to increase modularity by allowing the separation of cross-cutting concern.

Without AOP, solving concern that are cross-cutting result in scattered and duplicated code.

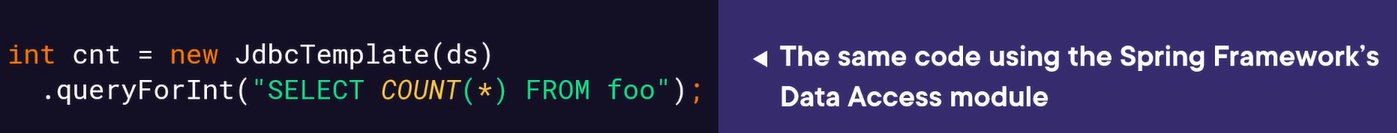
Spring Data Access:

* Make it easier to develop app that interact with data

Without Spring Data Access:  
A screenshot of a computer program

AI-generated content may be incorrect.

With Spring Data Access:



Mapped vendor-specific error into set of exceptions.

Testing data is also easier.

Spring Integration:

Integration is a multisided problem:

1. How to expose operations to other system?
2. How to invoke or run operations on other systems?

Different ways to expose Operations:

* RMI
* Messaging systems
* Web services

Spring Integration make it easy to expose and invoke web services

A screenshot of a computer screen

AI-generated content may be incorrect.

Spring Testing:

A close-up of a computer screen

AI-generated content may be incorrect.

Spring Data:

Provide a familiar and consistent Spring-based programming model for data access while still retaining the special traits of the underlying data store.

Extend the data access capabilities provided by Spring Framework.

Spring Data add new ways of interacting with relational databases, and support many different type of database.

Spring Cloud:

Provide tools for dev to quickly build distributed system.

Microservice is a small application well-scoped to a single purpose or domain.

Spring Security:

Framework for securing Spring-based application.

Adv of Spring:

* Rock-solid, well engineered
* Stood the test of time
* Huge community
* Built-in IDE support
* Scalable

Disadv:

* Too much magic (things under the hood)
* Steep learning curve
* Increase size of final deliverable
* Hard to debug
* Add memory overhead.
* Complexity grown over time

What is Spring?

A purple gear with a arrow pointing to it

AI-generated content may be incorrect.

Dependency Injection: Remove hard-coded wiring in app, using a framework to inject dependency resources where they are needed.

Spring: Framework built around reducing complexity around JEE.

POJO based (plain old java object).

Unobtrusive

AOP/Proxies

Best Practices

The problem Spring trying to solve:

* Testability
* Maintainability
* Scalability
* Complexity
* Business Focus

The Solution:

* Configuration
* Focus on Business need
* Testing
* Annotation or XML Based
* Interface Based

How it works:

POJO, HashMap, Registry.

Spring configuration using java:

* No applicationContext.xml
* Didn’t exist at first, but this is java and people don’t like XML

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

Scope and Autowiring:

Scope is not pattern.

5 scope inside Spring:

Valid in any config:

* Singleton
* Prototype

Valid only in web-aware Spring projects:

* Request
* Session
* Global

Singleton:

Restrict instantiation of a class to 1 object

Default bean scope

Single instance per Spring container/Application context

A screenshot of a computer program

AI-generated content may be incorrect.

Prototype:

* Guranteed unique instance per request
* Opposite of Singleton

Web Scopes:

Request Scopes: return a bean per http request, for the lifecycle of a bean request.

Session: return a bean per http session, live if the user session is alive.

GlobalSession: return a bean per application.

Autowired:

* Add @ComponentScan to configuration file
* Mark bean wanted as autowired
* Can choose by name and type

Stereotypes Annotation:

* @Component (same thing as @Bean)
* @Repository (denote a class being used as repository object)
* @Service (where to put business logic)

Spring Configuration Using XLM:

Why use XML:

* First approach
* Simpler
* Separation of concern

A blue and white sign with text

AI-generated content may be incorrect.

Bean Definition:

Namespaces:

A computer screen shot of a computer code

AI-generated content may be incorrect.

Spring created this namespace to simply act like a dictionary for the properties we can use to create a bean inside of our application.

XML Declaration:

A computer screen shot of a service

AI-generated content may be incorrect.

Beans:

* Essentially classes
* Defining beans = replace keyword new
* Define class, use interface

A blue and white text on a white background

AI-generated content may be incorrect.

Autowire:

Spring automatically wires bean

4 types of autowires:

* byType: autowired if exactly 1 bean of that property type exist within container.
* byName: