# Spring Framework

Spring Framework es una solución ligera para construir aplicaciones de grado empresarial. Está diseñado para no ser intrusivo, lo que significa que la gran mayoría de tu código no necesita declarar dependencias explicitas al framework, y de existir, son fáciles de remplazar.

Spring permite construir aplicaciones a partir de “plain old java objects” (POJOs) y aplicar servicios empresariales no intrusivos a dichos POJOs.

# Inyección de dependencias e inversión de control

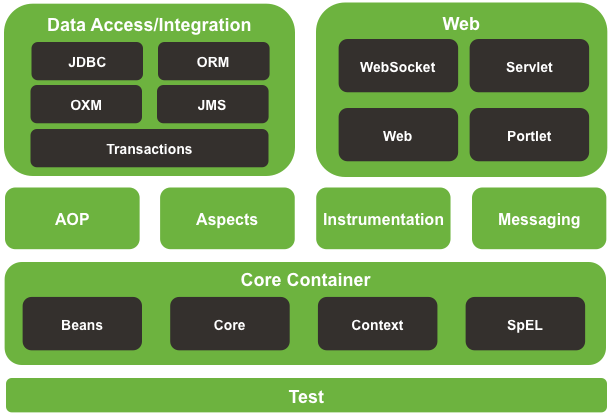
Una aplicación Java típicamente consiste de objetos que colaboran entre si para formar dicha aplicación. Es decir, los objetos en una aplicación tienen dependencias entre si.

La plataforma Java carece de lineamientos de como se deben organizar de forma coherente los bloques básicos que construirán una aplicación, dejando esta tarea a arquitectos y desarrolladores. A pesar de que pueden ser utilizados patrones como Factory, Abstract Factory, Builder, Decorator y Service Locator para organizar las diferentes clases y objetos que conforman una aplicación, estos patrones son simplemente eso: buenas practicas a las que se les dio un nombre, con una descripción de lo que el patrón hace, donde aplicarlo, los problemas que resuelve, etc. Los patrones son solamente buenas practicas, las cuales corresponde a los desarrolladores implementar.

La Inversión de Control de Spring Framework atacan este problema proveyendo una forma concisa de integrar componentes dispersos para formar una aplicación funcional y lista para usar.

# Módulos de Spring Framework

Spring Framework consiste de características organizadas en alrededor de 20 módulos. Estos módulos están agrupados en: Core Container, Data Access/Integration, Web, AOP, Instrumentation, Messaging y Test, como se muestra en el siguiente diagrama.



# Core Container

Core Container consiste de los módulos spring-core, spring-beans, spring-context, spring-context-support y spring-expression.

Los módulos spring-core y spring-beans proveen los fundamentos del framework, incluyendo Inversion de Control e Inyección de Dependencias. La clase BeanFactory es una sofisticada versión del patrón Factory. Remueve la necesidad de instanciar singletons y permite desacoplar la configuración y especificación de dependencias.

El módulo spring-context provee el acceso a objetos en una manera propia del framework muy similar al registro de JNDI. La interfaz ApplicationContext es el punto principal del módulo. Spring-context-support provee soporte para la integración de librerías de terceros en una aplicación de Spring.

# AOP e Instrumentation

El módulo spring-aop provee programación orientada a aspectos, permitiendo definir, por ejemplo, interceptores de métodos para desacoplar transparentemente código que implementa funcionalidad y que debe ser serparado.

El módulo spring-instrument provee soporte para instrumentación de clases e implementación de classloader para ser usados en ciertos servidores de aplicaciones.

# Data Access/Integration

La capa de Acceso a Datos/Integración consiste de los módulos de JDBC, ORM, OXM, JMS y Transaction.

El módulo spring-jdbc provee una capa de abstracción JDBC que remueve la necesidad del tedioso código JDBC, e interpretar códigos de error de base de datos específicas.

El módulo spring-tx soporta transacciones de forma programática y declarativa.

El módulo spring-orm provee capas de integración para los orm mas populares, incluyendo JPA, JDO y Hibernate.

# WEB

La capa web consiste de los módulos spring-web, spring-webmvc, spring-websocket y spring-webmvc-portlet.

El módulo spring-web provee integración con características web básicas, como carga de archivos y la inicialización del contenedor de inversión de control usando Servlet listeners y un contexto de aplicación web.

El módulo spring-webmvc contiene la implementación del MVC y REST Web Services para las aplicaciones web.