The [Core Container](https://docs.spring.io/spring/docs/5.0.0.BUILD-SNAPSHOT/spring-framework-reference/html/beans.html#beans-introduction) consists of the spring-core, spring-beans, spring-context, spring-context-support, and spring-expression (Spring Expression Language) modules.

The spring-jdbc module provides a [JDBC](https://docs.spring.io/spring/docs/5.0.0.BUILD-SNAPSHOT/spring-framework-reference/html/jdbc.html#jdbc-introduction)-abstraction layer that removes the need to do tedious JDBC coding and parsing of database-vendor specific error codes.

The spring-tx module supports [programmatic and declarative transaction](https://docs.spring.io/spring/docs/5.0.0.BUILD-SNAPSHOT/spring-framework-reference/html/transaction.html) management for classes that implement special interfaces and for all your POJOs (Plain Old Java Objects).

**Spring Framework Artifacts**

| **GroupId** | **ArtifactId** | **Description** |
| --- | --- | --- |
| org.springframework | spring-aop | Proxy-based AOP support |
| org.springframework | spring-aspects | AspectJ based aspects |
| org.springframework | spring-beans | Beans support, including Groovy |
| org.springframework | spring-context | Application context runtime, including scheduling and remoting abstractions |
| org.springframework | spring-context-support | Support classes for integrating common third-party libraries into a Spring application context |
| org.springframework | spring-core | Core utilities, used by many other Spring modules |
| org.springframework | spring-expression | Spring Expression Language (SpEL) |
| org.springframework | spring-instrument | Instrumentation agent for JVM bootstrapping |
| org.springframework | spring-instrument-tomcat | Instrumentation agent for Tomcat |
| org.springframework | spring-jdbc | JDBC support package, including DataSource setup and JDBC access support |
| org.springframework | spring-jms | JMS support package, including helper classes to send and receive JMS messages |
| org.springframework | spring-messaging | Support for messaging architectures and protocols |
| org.springframework | spring-orm | Object/Relational Mapping, including JPA and Hibernate support |
| org.springframework | spring-oxm | Object/XML Mapping |
| org.springframework | spring-test | Support for unit testing and integration testing Spring components |
| org.springframework | spring-tx | Transaction infrastructure, including DAO support and JCA integration |
| org.springframework | spring-web | Web support packages, including client and web remoting |
| org.springframework | spring-webmvc | REST Web Services and model-view-controller implementation for web applications |
| org.springframework | spring-websocket | WebSocket and SockJS implementations, including STOMP support |

Logging is a very important dependency for Spring because a) it is the only mandatory external dependency, b) everyone likes to see some output from the tools they are using, and c) Spring integrates with lots of other tools all of which have also made a choice of logging dependency.

IoC is also known as dependency injection (DI). It is a process whereby objects define their dependencies, that is, the other objects they work with, only through constructor arguments, arguments to a factory method, or properties that are set on the object instance after it is constructed or returned from a factory method. The container then injects those dependencies when it creates the bean. This process is fundamentally the inverse, hence the name Inversion of Control (IoC), of the bean itself controlling the instantiation or location of its dependencies by using direct construction of classes, or a mechanism such as the Service Locator pattern.

 A bean is an object that is instantiated, assembled, and otherwise managed by a Spring IoC container.

The Spring container can autowire relationships between collaborating beans.

@Primary is an effective way to use autowiring by type with several instances when one primary candidate can be determined. When more control over the selection process is required, Spring’s @Qualifier annotation can be used.

The Spring Expression Language (SpEL for short) is a powerful expression language that supports querying and manipulating an object graph at runtime.