Spring core

# Spring context

The Spring Framework implements the Inversion of Control (IoC) principle (IoC is also known as dependency injection (DI)). What is inversion of control (IoC) - <https://stackoverflow.com/questions/3058/what-is-inversion-of-control>.

It is a process whereby objects define their dependencies only through:

* constructor arguments;
* arguments to a factory method;
* 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.

The basis for Spring Framework’s IoC container are packages:

* org.springframework.beans;
* org.springframework.context.

The [BeanFactory](https://docs.spring.io/spring-framework/docs/5.3.10/javadoc-api/org/springframework/beans/factory/BeanFactory.html) interface provides an advanced configuration mechanism capable of managing any type of object. [ApplicationContext](https://docs.spring.io/spring-framework/docs/5.3.10/javadoc-api/org/springframework/context/ApplicationContext.html) is a sub-interface of BeanFactory. It adds:

* Easier integration with Spring’s AOP features;
* Message resource handling (for use in internationalization);
* Event publication;
* Application-layer specific contexts such as the WebApplicationContext for use in web applications.

In short, the **BeanFactory** provides the configuration framework and basic functionality, and the **ApplicationContext** adds more enterprise-specific functionality.

The container gets its instructions on what objects to instantiate, configure, and assemble by reading configuration metadata. The configuration metadata is represented in:

* XML;
* Java annotations;
* Java code.

# References

* IDE for developing enterprise applications using Spring Framework and Spring Boot, the new generation of Spring Tools provides world-class development support for your Spring applications - <https://spring.io/tools>.