个人理解：(面向接口的插件式开发)

类在容器中注册，成为Bean。容器(对象池)协调管理，并对外提供对象创建的服务。

Concept

POJO：不依赖任何环境的特定类或接口的类，常见于model/domain

Bean：由Spring容器实例化和装配的对象

**Inversion of Control / Dependency Injection**

·对象间的依赖关系，由IoC容器集中管理被依赖对象的生命周期。

·控制被反转之后，获得依赖对象的过程由自身管理变为IoC容器动态注入。

**Dependency inversion principle**

·上层模块不应该依赖底层模块，它们都应该依赖于抽象。

·抽象不应该依赖于细节，细节应该依赖于抽象。

注入方式:

·constructor(不允许循环依赖)

·setter

注入配置：

·XML

·注解

·Java Config

**常用注解**

|  |  |
| --- | --- |
| @Autowired | 自动装配 |
| @Service | 一个class定义为一个bean |
| @Repository |  |
| @Qualifier | autowired有多个选择时，指定一个实例 |

------------------------------------------

注解：

1. BeanFactory
2. SpringMVC

事务：实现数据和操作的完整性