IOC（Inversion of Control，控制反转）是spring的核心，贯穿始终。

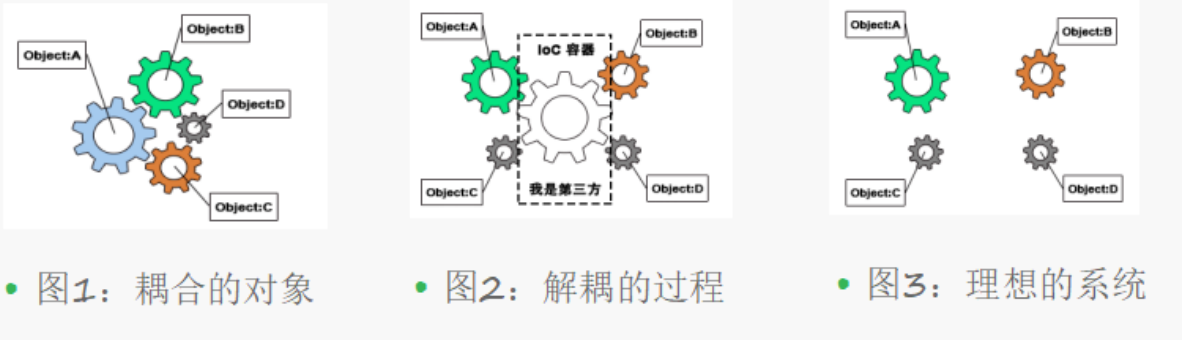
所谓IOC，对于spring框架来说，就是由spring来负责控制对象的生命周期和对象间的关系：

传统开发模式：对象之间互相依赖

IOC开发模式：IOC容器安排对象之间的依赖

举例：找女朋友　传统模式要相亲一个一个去找　IOC模式就是通过中介来根据需要找

IOC理论的背景

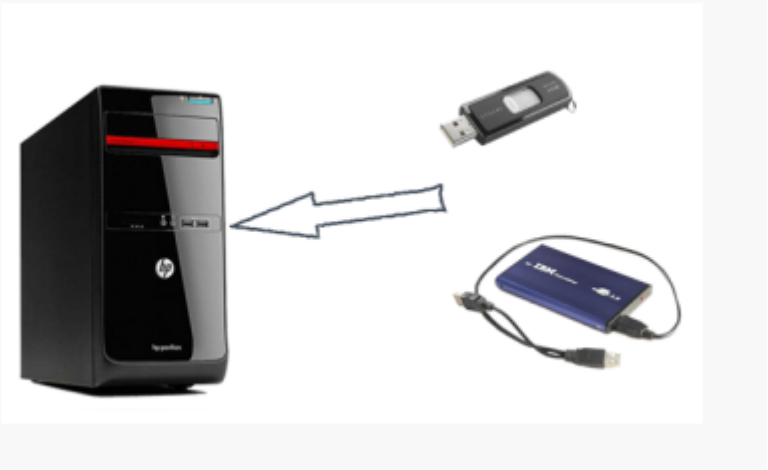


DI依赖注入（Dependency Injection）

IOC的另外的名字叫做依赖注入（Dependency Injection），

所谓的依赖注入，就是由IOC容器在运行期间，动态地将某种依赖关系注入到对象之中。

所以，依赖注入(DI)和控制反转(IOC)是从不同的角度的描述的同一件事情，就是指通过引入IOC容器，利用依赖关系注入的方式，实现对象之间的解耦



IOC的好处

IOC在编程过程中不会对业务对象构成很强的侵入性，

使用IOC之后，对象具有更好的可实行性，可重用性和可扩展性：

降低组件之间的耦合度： U盘和电脑独立 只要符合USB接口标准

提高开发效率和产品质量：U盘和电脑可以单独测试　分工明确
统一标准，提高模块的复用性：符合USB接口标准　USB可以反复利用
模块具有热插拔特性：USB热插拔特性　对象生成放在配置文件里实现

IOC通俗的理解如下：
IOC控制反转：
说的是创建对象实例的控制权从代码控制剥离到IOC容器控制，
实际就是你在xml文件控制，侧重于原理
DI依赖注入：
说的是创建对象实例时，为这个对象注入属性值或其它对象实例，侧重于实现