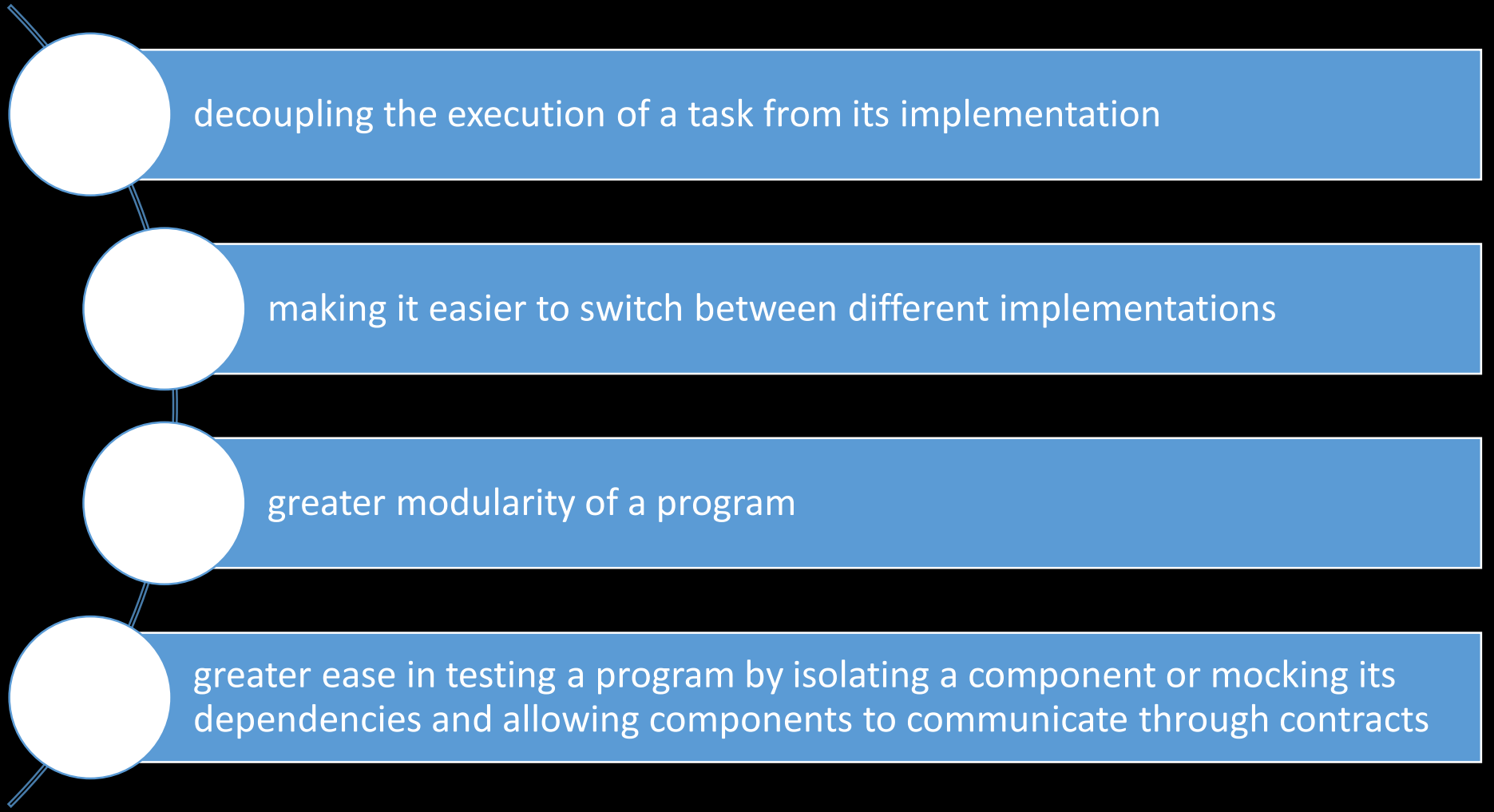
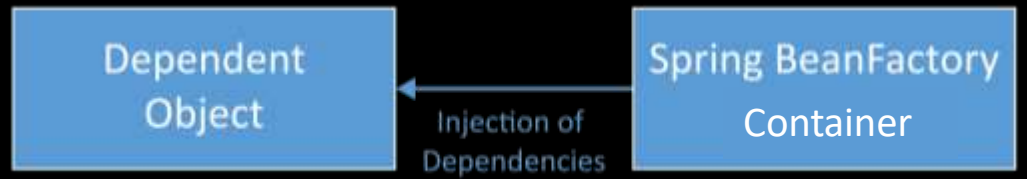
Spring IoC Container

The advantages of IoC

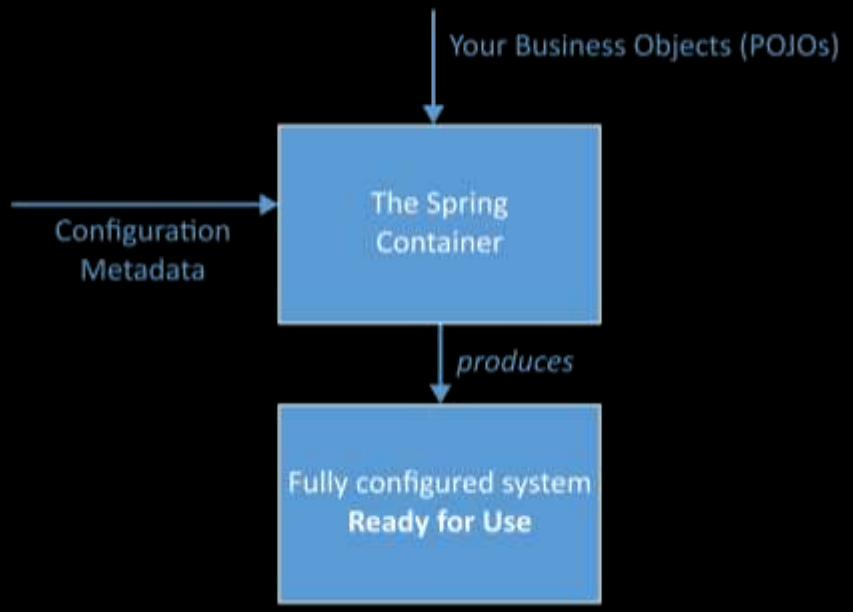


Inversion of Control in Spring



The container will create the objects, wire them together, configure them, and manage their complete life cycle from creation till destruction. The Spring container uses dependency injection (DI) to manage the components that make up an application.

Configuration metadata



Configuration metadata represents how you as an application developer tell the Spring container to instantiate, configure, and assemble the objects in your application.

Configuration metadata can be supplied in:

- XML format
- Annotation-based configuration
- Java-based configuration

Types of IoC containers in Spring

BeanFactory

```
Resource resource=new ClassPathResource("applicationContext.xml");
BeanFactory factory=new XmlBeanFactory(resource);
```

This is the simplest container providing the basic support for DI and is defined by the *org.springframework.beans.factory.BeanFactory* interface. The BeanFactory and related interfaces, such as BeanFactoryAware, InitializingBean, DisposableBean, are still present in Spring for the purpose of backward compatibility with a large number of third-party frameworks that integrate with Spring.

ApplicationContext

```
ApplicationContext context =
    new ClassPathXmlApplicationContext("applicationContext.xml");
```

This container adds more enterprise-specific functionality such as the ability to resolve textual messages from a properties file and the ability to publish application events to interested event listeners. This container is defined by the *org.springframework.context.ApplicationContext* interface.