装载Bean

在 Spring IOC 容器读取 Bean 配置创建 Bean 实例之前, 必须对它进行实例化. 只有在容器实例化后, 才可以从 IOC 容器里获取 Bean 实例并使用.

配置Bean

```
<bean scope=“singleton|prototype|request|session” />
singleton: 单例 默认值[每次创建都是原来的那一份]
prototype: 原型  [每次创建都是一个全新的Bean]
request: 一次请求有效( java web开发中)
session: session级有效(java web开发中)
```

获取Bean

获取Bean有两种方式

1. BeanFactory: IOC 容器的基本实现.
    1. 工厂设计模式，创建分发各种bean。配置好它们之间的写作关系，参与bean的生命周期。
2. 应用上下文(ApplicationContext):建立在bean工厂基础之上，提供系统架构服务。
    - ApplicationCotext,spring更加高级的容器。功能强大：
        1. 提供文本信息解析工具，包括对国际化支持。
        2. 提供载入文件资源的通用方法，如图片。
        3. 可以向注册为监听器的bean发送事件。

    在很少的情况下，使用BeanFactory，如在移动设备。

补充：BeanFactory 是 Spring 框架的基础设施，面向 Spring 本身；ApplicationContext 面向使用 Spring 框架的开发者，几乎所有的应用场合都直接使用 ApplicationContext 而非底层的 BeanFactory。

BeanFactory

BeanFactory factory = new XmlBeanFactory(new ClassPathResource("applicationContext.xml"));
bean工厂只把bean的定义信息载进来，用到的时候才实例化。factory.getBean("mybean");就可得到一个bean。

BeanFactory是一个接口。

应用上下文(ApplicationContext)

ApplicationContext同样也是一个接口，它的父接口BeanFactory。而子接口的功能往往比父类更加强大，说以开发过程中，我们主要使用ApplicationContext。（接口与类之间的关系去看文档，或者看源码就可以得知。）

- ApplicationContext 的主要实现类：
    1. ClassPathXmlApplicationContext(ConfigurableApplicationContext 接口)：从类路径下加载配置文件
    2. FileSystemXmlApplicationContext(ConfigurableApplicationContext 接口): 从文件系统中加载配置文件。

总结：

ApplicationContext和BeanFactory加载Bean时，BeanFactory创建的时候，不会自动实例化Bean，而是在getBean的时候再去加载。而应上下文的方式会在创建的时候去自动加载。加载Bean是会消耗内存的，web项目追求响应速度至上，在加上内存（硬件）慢慢低廉，所以加载方式，我们会去选择ApplicationContext。