FactoryBean 适用于 Bean 的创建过程比较复杂的场景，比如数据库连接池的创建。

|  |
| --- |
| public interface FactoryBean<T> {  T getObject() throws Exception;  Class<T> getObjectType();  boolean isSingleton();  } |

假设需要创建一个 Person 的 Bean，那么就需要一个 Car 的实例；如果 Car 的实例创建很麻烦，就可以用一个FactoryBean<Car>把创建 Car 的复杂过程包装起来。

配置上，尽管Person的car属性是Car类型的，但是我们在装配一个Person的Bean时，让其car属性引用一个FactoryBean<Car>类型的Bean也是可以的，spring会负责中间的处理过程。

|  |
| --- |
| public class Person {  private Car car ;  private void setCar(Car car){ this.car = car; }  } |
| **public** **class** MyCarFactoryBean **implements** FactoryBean<Car> {  **private** String make;  **private** **int** year;  **public** **void** setMake(String m) { **this**.make = m; }  **public** **void** setYear(**int** y) { **this**.year = y; }  @Override **public** Car getObject() {  // 这里我们假设 Car 的实例化过程非常复杂，反正就不是几行代码可以写完的那种  CarBuilder cb = new CarBuilder();  **if** (year != 0) cb.setYear(**this**.year);  **if** (StringUtils.*hasText*(**this**.make)) cb.setMake(**this**.make);  **return** cb.buildCar();  }  @Override **public** Class<Car> getObjectType() {  **return** Car.**class**;  }  @Override **public** **boolean** isSingleton() {  **return** **false**;  }  } |
| <bean class = *"com.javadoop.MyCarFactoryBean"* id = *"car"*>  <property name = *"make"* value = *"Honda"*/>  <property name = *"year"* value = *"1984"*/>  </bean>  <bean class = *"com.javadoop.Person"* id = *"josh"*>  <property name = *"car"* ref = *"car"*/>  </bean> |

现在还用 xml 配置 Bean 依赖的越来越少了，更多时候，我们可能会采用 java config 的方式来配置，这里有什么不一样呢？

配置Person的Bean时，把 MyCarFactoryBean 看成是一个简单的 Bean 就可以了，不必理会它是不是FactoryBean。

|  |
| --- |
| @Configuration  **public** **class** TestConfiguration {  @Bean  **public** MyCarFactoryBean carFactoryBean() {  MyCarFactoryBean cfb = **new** MyCarFactoryBean();  cfb.setMake("Honda");  cfb.setYear(1984);  **return** cfb;  }  @Bean  **public** Person aPerson() {  Person person = **new** Person();  // 注意这里的不同  person.setCar(carFactoryBean().getObject());  **return** person;  }  } |

当配置文件中<bean>的class属性配置的实现类是FactoryBean时，通过 getBean()方法返回的不是FactoryBean本身，而是FactoryBean#getObject()方法所返回的对象，相当于FactoryBean#getObject()代理了getBean()方法