06 Spring_配置详解_属性注入

Spring

一:实体类

```
public class Users {
      private  Integer  id;
      private  String name;
 //对应的set get  toString 方法
}
```

二：set方法注入

1、引用类型

1、配置文件

```
 <!--Set方法注入  -->
 <bean   name="users01" class="com.spring.bean.Users">
  <property name="name" value="陈老师" ></property>
  <property name="id"  >
      <value>12</value>
  </property>
 </bean>
```

2、测试

```
    @org.junit.Test
    public  void getVoid01(){
        //1.创建容器对象
    ClassPathXmlApplicationContext classPath= new
ClassPathXmlApplicationContext("applicationContext.xml");
        //2.找容器要对象
    Users bean = (Users)classPath.getBean("users01");
        System.out.println("bean:"+bean);
```

```
    }
```

2、值类型

1、实体类

```
public class Car {
    private String name;
    private  String clor;
    //对应的get set toString 方法
    }
```

```
public class Users {
      private  Integer  id;
      private  String name;
      private  Car  car;
    //对应的get set toString 方法
    }
```

2、配置文件

```
 <bean   name="users01" class="com.spring.bean.Users">
      <property name="name" value="陈老师" ></property>
      <property name="id" value="12" ></property>
      <!--name:属性名
           ref:指定配置的car对象
        -->
      <property name="car" ref="car"></property>
 </bean>
 <!--将对象配置在容器中  -->
 <bean name="car" class="com.spring.bean.Car">
     <property name="name" value="宝马"></property>
     <property name="clor" value="红色"></property>
 </bean>
```

3、测试

> 同上

```
<terminated> Test.getVoid01 [JUnit] D:\JDK\jdk-8u101-windows-x64\jdk\bin\javaw.exe (2018年8月14日 上午5:18:37)
log4j:WARN No appenders could be found for logger (org.springfr
log4j:WARN Please initialize the log4j system properly.
创建了一个对象
bean:Users [id=12, name=陈老师, car=Car [name=宝马, clor=红色]]
```

三：构造函数注入

> 1、实体类同上

1、配置文件

```
<!-- 构造函数的注入
  注意:constructor-arg的个必须和构造函数参数一致,顺序可调整
  name:构造函数的参数名
  value:值
  ref:指向对应的对象bean
  index:指向构造函数的参数顺序
  type:指向构造函数参数的类型(在类型不同参数名相同)
-->
<bean name="users01" class="com.spring.bean.Users">
      <constructor-arg name="id" value="12" index="1"></constructor-arg>
      <constructor-arg name="name" value="123" index="0" type="java.lang.Integer" ></constructor-arg>
      <constructor-arg name="car" ref="car" index="2"></constructor-arg>
 </bean>
<bean name="car" class="com.spring.bean.Car">
     <property name="name" value="奔驰" ></property>
     <property name="clor" value="红色"></property>
 </bean>
```

2、测试

> 1、同上

```
log4j:WARN No appenders could be found for logger (org.springf
log4j:WARN Please initialize the log4j system properly.
构造函数的注入
bean:Users [id=12, name=tom, car=Car [name=奔驰1, clor=红色]]
```

四：P名称空间注入(了解)

> 1、实体类|测试同上

1、配置文件

> 1、注意必须导入冥名空间xmlns:p="http://www.springframework.org/schema/p"
> 2、P名称空间注入本质就是set方法

```
<bean name="car" class="com.spring.bean.Car">
     <property name="name" value="奔驰" ></property>
     <property name="clor" value="红色"></property>
 </bean>
 <!-- P:名称空间注入
      导入P名称的冥名空间
      |-值类型:p:属性="值"
      |-对象类型 ： P：属性名-ref="值"
 -->
 <bean name="users03" class="com.spring.bean.Users"
    p:name="小陈" p:id="123" p:car-ref="car">
</bean>
```

五：spel(spring Expression Language)注入(了解)

> 1、Sring的表达式语言
> 2、实体类|测试同上

1、配置文件

```
<!--spel注入
  |- 取其他类型的值来当成自己的值
   #{bean的名称.属性名}
  |-自定义
   #{值}
   ref不能使用#{}的方式
-->
 <bean name="users04" class="com.spring.bean.Users">
      <property name="name" value="#{users03.name}" ></property>
      <property name="id" value="#{1230}"  ></property>
      <property name="car" ref="car" ></property>
</bean>
<bean name="car" class="com.spring.bean.Car">
     <property name="name" value="奔驰" ></property>
     <property name="clor" value="红色"></property>
 </bean>
```