|  |
| --- |
| Spring |
| Replaced-Method  public class ReplacedM implements MethodReplacer {  @Override  public Object reimplement(Object arg0, Method arg1, Object[] arg2) throws Throwable {    System.*out*.println(arg2);  return "hallo";  }  }  <bean id=*"T1000"* class=*"ru.javabegin.training.spring.impls.robot.ModelT1000"*  autowire=*"byType"* scope=*"prototype"*>  <replaced-method name=*"dance"* replacer=*"replacementTarget"*/>  </bean>  <bean id=*"replacementTarget"* class=*"ru.javabegin.training.spring.main.ReplacedM"*>  </bean> |
| Lookup-Method  public interface IRobotConveyer { //kann auch class oder abstract class sein    public Robot getRobot();  }  <bean id=*"RobotConveyer"* class=*"ru.javabegin.training.spring.main.IRobotConveyer"*>  <lookup-method name=*"getRobot"* bean=*"T1000"* />  </bean> |
| Autowired  <bean id=*"T1000"* class=*"ru.javabegin.training.spring.impls.robot.ModelT1000"*  autowire=*"byType"* scope=*"prototype"*>  //autowire versucht für alle in class *ModelT1000* befindlichen Properties entsprechnde bean im context zu finden.  <bean class=*"ru.javabegin.training.spring.impls.robot.ModelT1000"*  autowire-candidate=*"true"*>  <constructor-arg value=*"golded"* type=*"String"*/>  <constructor-arg value=*"2007"* type=*"int"*/>  <constructor-arg value=*"false"* type=*"boolean"*/>  </bean>  <bean class=*"ru.javabegin.training.spring.impls.robot.ModelT1000"*  autowire-candidate=*"true"*>  <constructor-arg value=*"gold"* />  <constructor-arg value=*"3219"* type=*"int"*/>  <constructor-arg value=*"true"* type=*"boolean"*/>  </bean>    <bean id=*"t1000Pool"* class=*"ru.javabegin.training.spring.impls.robot.ModelT1000"* autowire=*"byType"*/>  public void setRobotCollection(Collection<Robot> robotCollection) {  this.robotCollection = robotCollection;  }  So funktioniert es auch. Spring sieht Propertie Collection und sucht für diese Collection alle passende Beans. |