**Spring framework** is built on [dependency injection](http://www.journaldev.com/2410/spring-dependency-injection-example-with-annotations-and-xml-configuration) and we inject the class dependencies through spring bean configuration file.

Usually we provide bean configuration details in the spring bean configuration file and we also specify the beans that will be injected in other beans using ref attribute. But Spring framework provides autowiring features too where we don’t need to provide bean injection details explicitly.

There are different ways through which we can autowire a spring bean.

1. autowire byName – For this type of autowiring, setter method is used for dependency injection. Also the variable name should be same in the class where we will inject the dependency and in the spring bean configuration file.
2. autowire byType – For this type of autowiring, class type is used. So there should be only one bean configured for this type in the spring bean configuration file.
3. autowire by constructor – This is almost similar to autowire byType, the only difference is that constructor is used to inject the dependency.
4. autowire by autodetect – If you are on Spring 3.0 or older versions, this is one of the autowire options available. This option was used for autowire by constructor or byType, as determined by Spring container. Since we already have so many options, this option is deprecated.

@Autowired annotation – We can use this annotation for spring bean autowiring. This annotation can be applied on variables and methods for autowiring byType. We can also use this annotation on constructor for constructor based autowiring.

For this annotation to work, we also need to enable annotation based configuraiton in spring bean configuration file. This can be done by **context:annotation-config** element or by defining a bean of type org.springframework.beans.factory.annotation.AutowiredAnnotationBeanPostProcessor.