Spring – Autowiring

Autowiring in the Spring framework can inject dependencies automatically. The Spring container detects those dependencies specified in the configuration file and the relationship between the beans. This is referred to as **Autowiring in Spring**. To enable Autowiring in the Spring application we should use **@Autowired** annotation. Autowiring in Spring internally uses constructor injection. An autowired application requires fewer lines of code comparatively but at the same time, it provides very little flexibility to the programmer

Modes of Autowiring

Modes	Description
No	This mode tells the framework that autowiring is not supposed to be done. It is the default mode used by Spring.
byName	It uses the name of the bean for injecting dependencies.
byType	It injects the dependency according to the type of bean.
Constructor	It injects the required dependencies by invoking the constructor.
Autodetect	The autodetect mode uses two other modes for autowiring – constructor and byType.

1. No

This mode tells the framework that autowiring is not supposed to be done. It is the default mode used by Spring.

```
<bean id="state" class="sample.State">
 <property name="name" value="UP" />
</bean>
<bean id="city" class="sample.City"></bean>
```

2. byname

It uses the name of the bean for injecting dependencies. However, it requires that the name of the property and bean must be the same. It invokes the setter method internally for autowiring.

```
<bean id="state" class="sample.State">
 <property name="name" value="UP" />
</bean>
<bean id="city" class="sample.City" autowire="byName"></bean>
```

3. byType

It injects the dependency according to the type of the bean. It looks up in the configuration file for the class type of the property. If it finds a bean that matches, it injects the property. If not, the program throws an error. The names of the property and bean can be different in this case. It invokes the setter method internally for autowiring.

```
<bean id="state" class="sample.State">
 <property name="name" value="UP" />
</bean>
<bean id="city" class="sample.City" autowire="byType"></bean>
```

4. constructor

It injects the required dependencies by invoking the constructor. It works similar to the “byType” mode but it looks for the class type of the constructor arguments. If none or more than one bean are detected, then it throws an error, otherwise, it autowires the “byType” on all constructor arguments.

```
<bean id="state" class="sample.State">
 <property name="name" value="UP" />
</bean>
<bean id="city" class="sample.City" autowire="constructor"></bean>
```

5. autodetect

The autodetect mode uses two other modes for autowiring – constructor and byType. It first tries to autowire via the constructor mode and if it fails, it uses the byType mode for autowiring. It works in Spring 2.0 and 2.5 but is deprecated from Spring 3.0 onwards.

```
<bean id="state" class="sample.State">
 <property name="name" value="UP" />
```

```
</bean>
<bean id="city" class="sample.City" autowire="autodetect"></bean>
```

The **@Autowired** annotation in the Spring Framework is used for automatic dependency injection. It can be applied to fields, setter methods, or constructors, and it tells Spring to automatically inject the necessary dependencies at runtime.

Constructor injection is often considered a good practice because it ensures that the object is fully initialized when it's created, and it can make your classes more testable and easier to reason about.