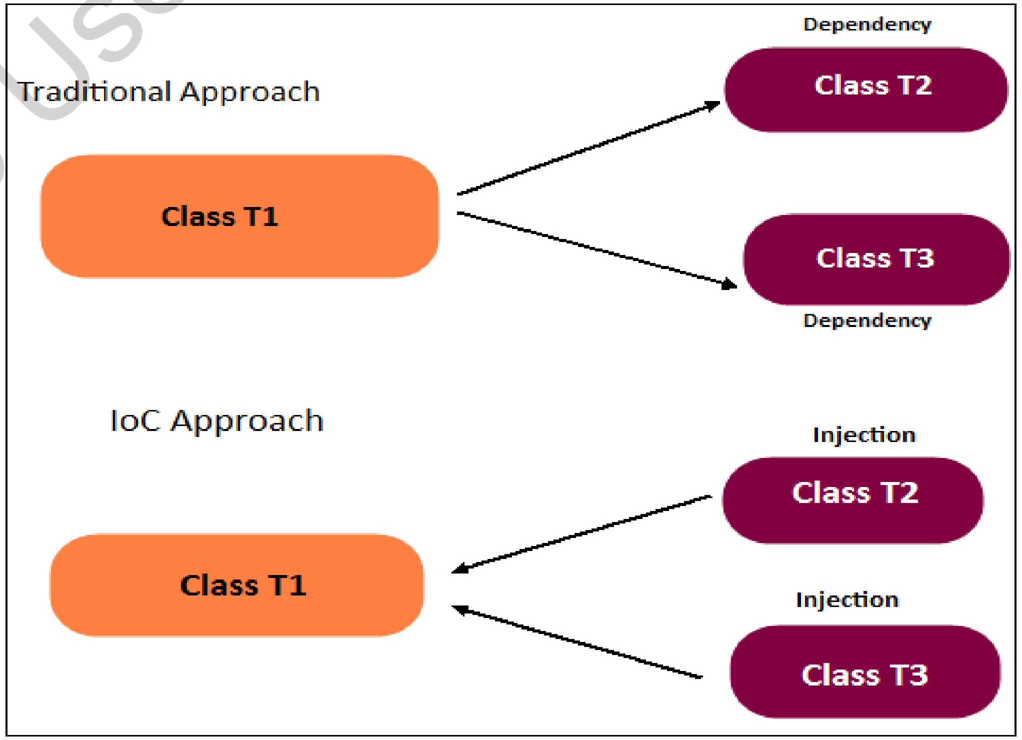
SPRING CORE | Session 2

OBJECTIVES

- Explain how to use Inversion of Control in your programs
- Explain the basics of aspect-oriented programming
- Describe new features of Spring 5.0

INVERSION OF CONTROL

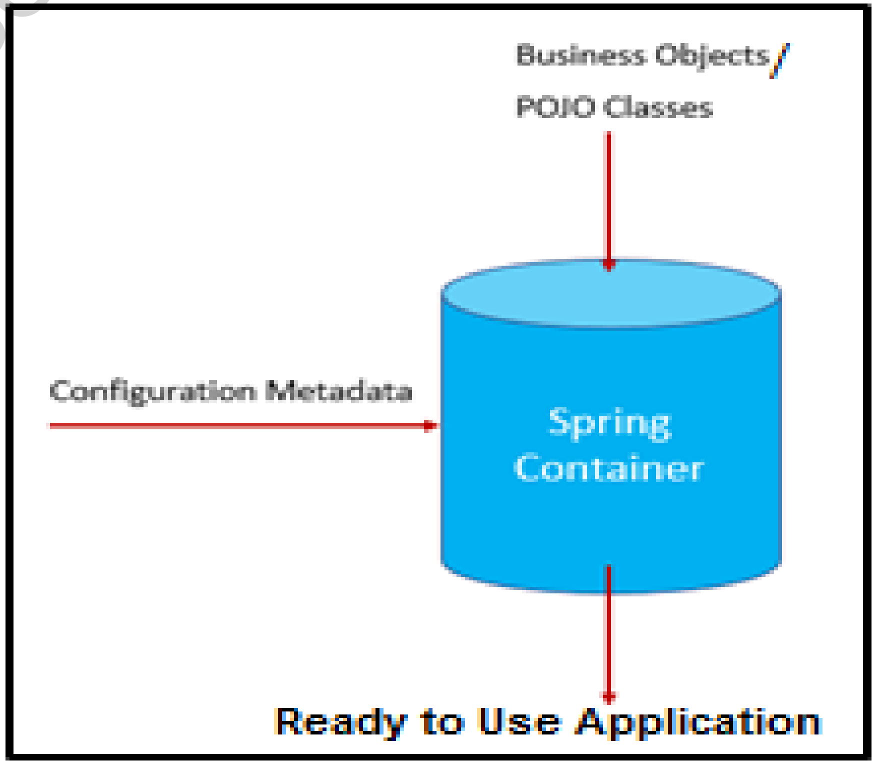
- Inversion of Control (IoC) is a design principle that:
  - Involves control of objects or custom-written portions of a program being transferred to a framework.
  - Enables creation of loosely-coupled objects.
  - Provides flexibility is coding programs.



Inversion of Control Principle

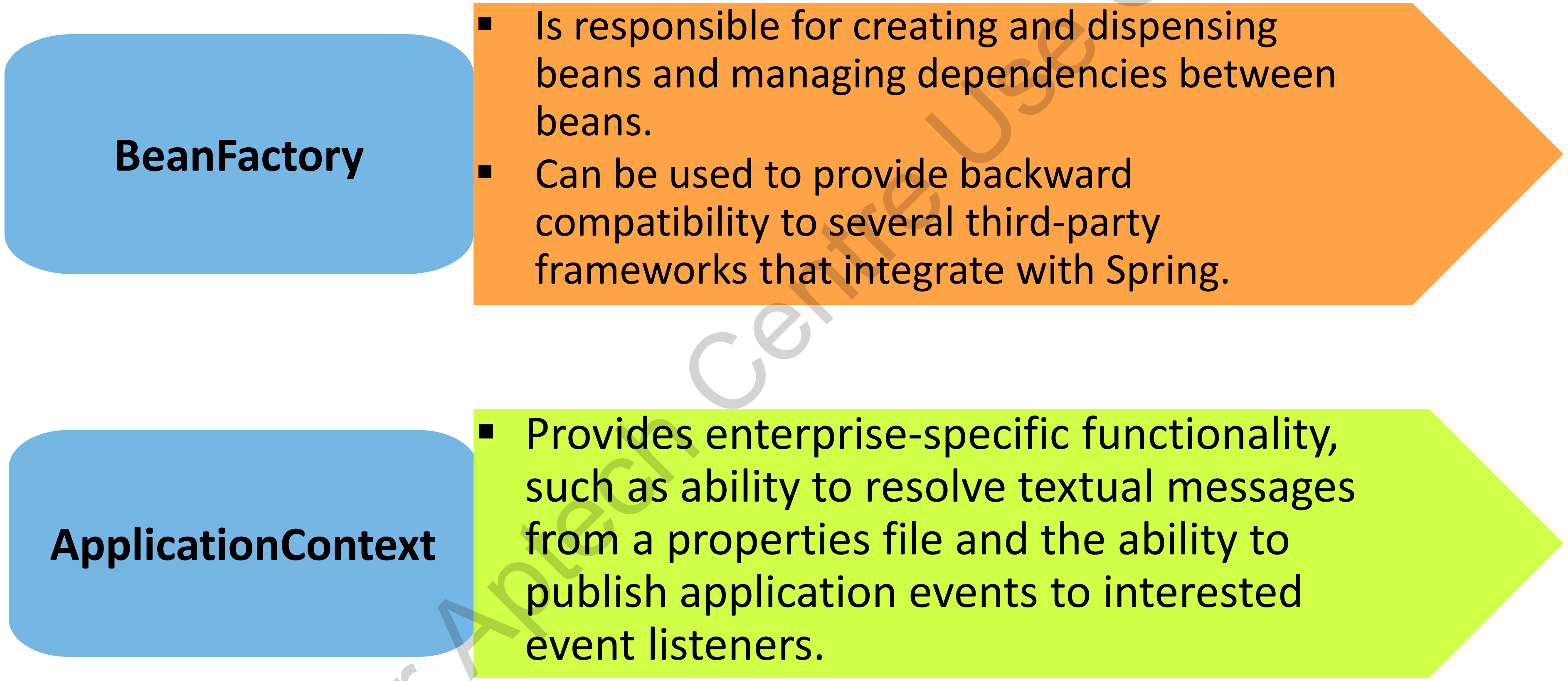
CONTAINERS AND SPRING CONTAINERS (1-2)

- Container is an application program or a subsystem in which a component (program building block) runs.
- Containers have following properties:
  - Access: To access the objects of the container.
  - Storage: To store the objects of the container.
  - Traversal: To traverse the objects of the container.
- Spring Framework implements IoC through the Spring container, which uses Dependency Injection to manage application components.



Spring Container

CONTAINERS AND SPRING CONTAINERS (2-2)



BeanFactory

- Is responsible for creating and dispensing beans and managing dependencies between beans.
- Can be used to provide backward compatibility to several third-party frameworks that integrate with Spring.

ApplicationContext

- Provides enterprise-specific functionality, such as ability to resolve textual messages from a properties file and the ability to publish application events to interested event listeners.

Types of Spring Containers

DEPENDENCY INJECTION

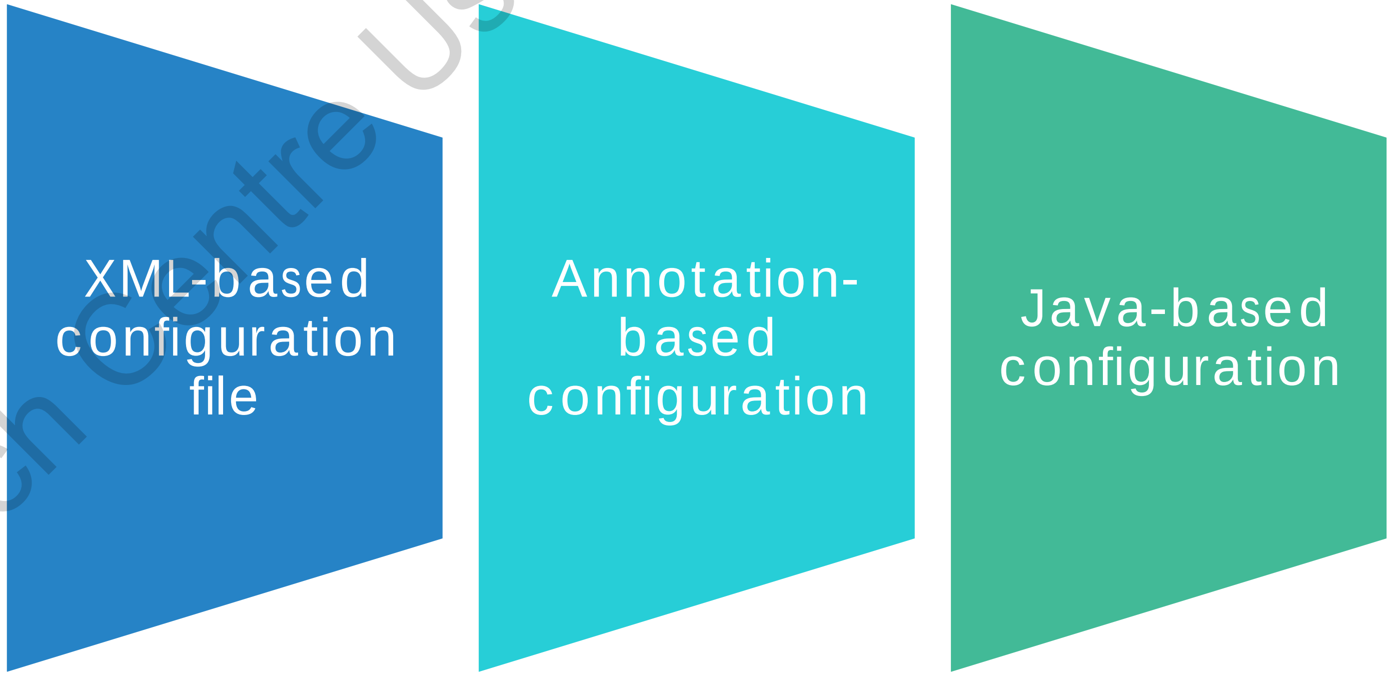
- DI is a design principle that is used to implement IoC in Spring Framework.
- DI links the classes used in an application while keeping them independent of each other.
- DI is of two types:
  - ❑ Constructor-based
  - ❑ Setter-based

Constructor-based DI	Setter-based DI
• The container invokes a class constructor with multiple arguments, each representing a dependency on the other class.	• The container calls setter methods on the beans after invoking a no-argument constructor or no-argument static factory method to instantiate the bean.

DI Types

BEAN DEFINITION (1-2)

- Beans are objects of the Spring container.
- Spring container is provided with information about the beans and their dependencies, which is called Configuration Metadata.



Methods to provide Configuration Metadata

BEAN DEFINITION (2-2)

Following code is used to define a bean:

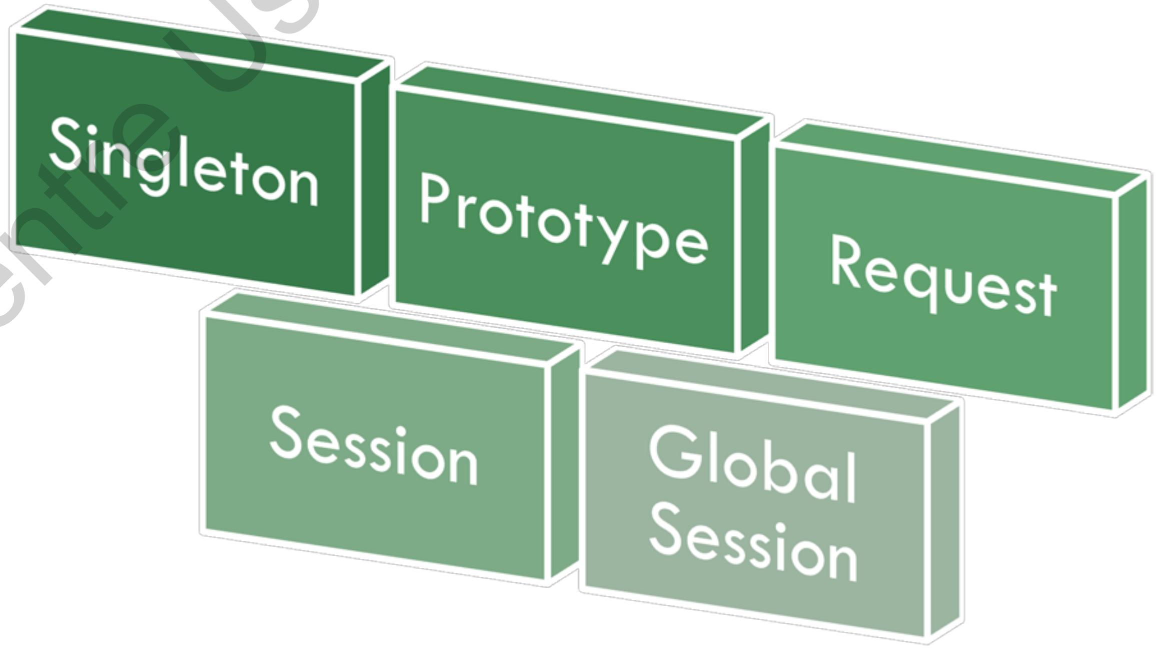
```
<!-- A simple bean definition -->
   <bean id = "..." class = "...">
      <!-- collaborators and configuration for the bean-->
   </bean>
```

OR

```
<!-- A bean definition with initialization method -->
   <bean id = "..." class = "..." init-method = "...">
   <!-- collaborators and configuration for this bean go here -->
   </bean>
```

BEAN DEFINITION INHERITANCE AND SCOPE

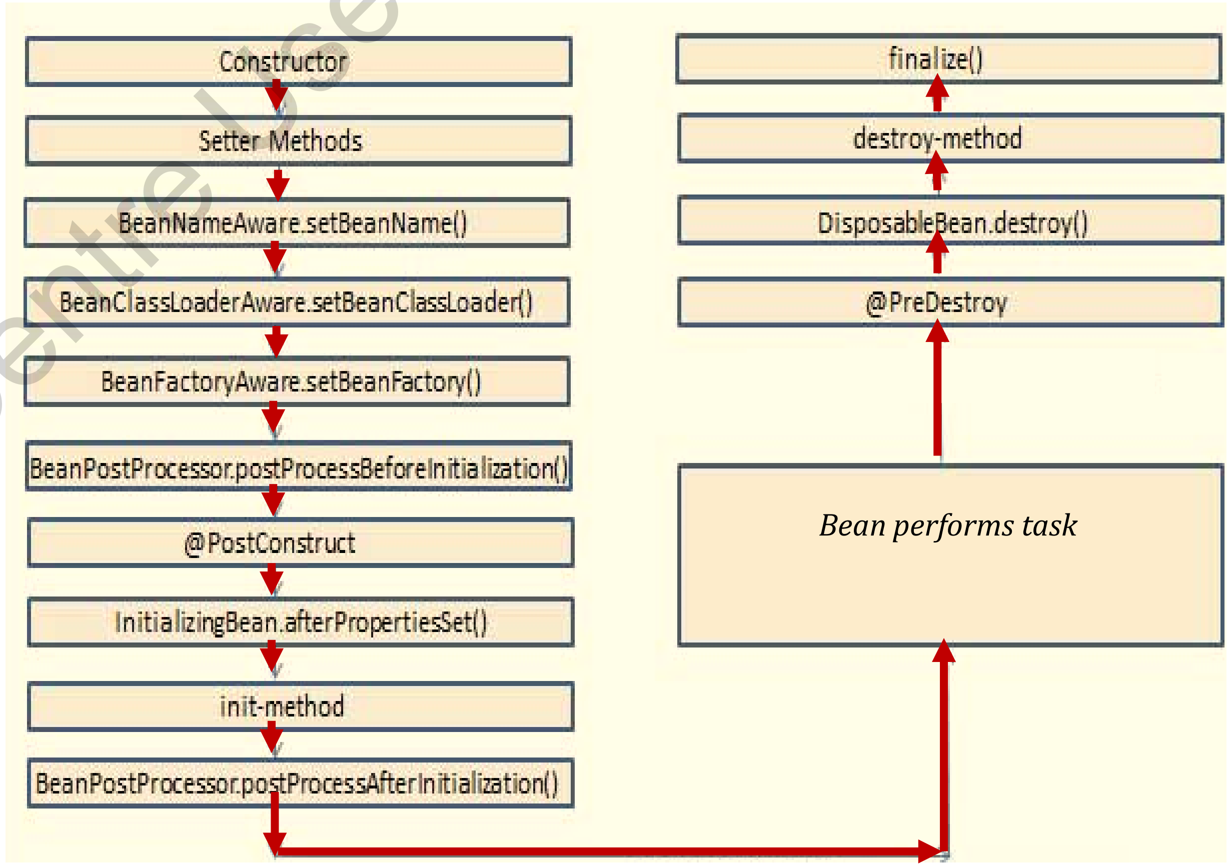
- Spring Framework supports inheritance of beans.
- You can define a parent bean and all child beans can inherit the configuration data.
- The bean scope helps to define and decide the type of bean instance returned from Spring container back to the caller.



Types of Bean Scopes

SPRING BEAN LIFECYCLE

- Spring beans have a defined lifecycle, from their initiation till their destructions.
- Spring beans factory manages the lifecycle of the Spring container beans.



Constructor
Setter Methods
BeanNameAware.setBeanName()
BeanClassLoaderAware.setBeanClassLoader()
BeanFactoryAware.setBeanFactory()
BeanPostProcessor.postProcessBeforeInitialization()
@PostConstruct
InitializingBean.afterPropertiesSet()
init-method
BeanPostProcessor.postProcessAfterInitialization()
Bean performs task
@PreDestroy
DisposableBean.destroy()
destroy-method
finalize()

Bean Lifecycle

AUTO-WIRING IN SPRING

- Spring container can auto-wire relationships between collaborating beans.
- The autowire attribute of the <bean/> element is used to specify autowire mode for a bean definition.
- Autowiring reduces the amount of XML configuration code required.

Autowire Mode	Description
no	This is the default autowiring mode and refers to no autowiring by default.
byName	Autowiring is done based on the name of the property; therefore, Spring searches for a bean with the same name as the property that needs to be set.
byType	Autowiring is done based on the type of the property; therefore, Spring searches for a bean with the same name as the property that needs to be set. If more than one match is found, the framework throws an exception. Use this mode to wire arrays and other typed-collections.
constructor	Autowiring is done based on constructor arguments, therefore, Spring searches for beans with the same type as the constructor arguments. Use this mode to wire arrays and other typed-collections.
autodetect	Autowiring is first done using autowire by constructor and then by byType, if required.

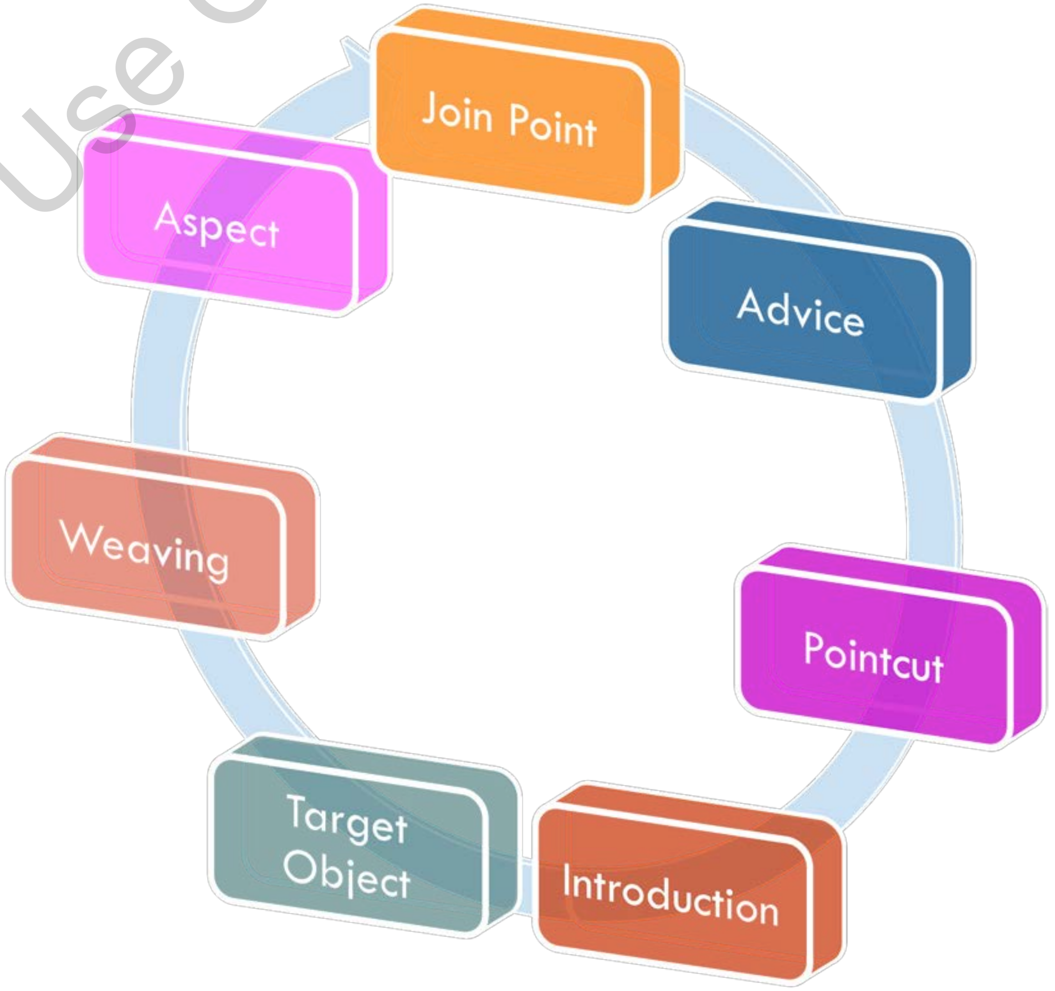
FACTORY METHOD

- Spring uses the **factory-method** attribute of a bean tag to delegate the instantiation of a class to the Factory class.
- The Factory class has a predefined static method that creates the instance of the required bean.
- Following code snippet shows how to instantiate a class using the factory method:

```
<bean id="userService" class="com.concretepage.UserService"
factory-method="createInstance">
```

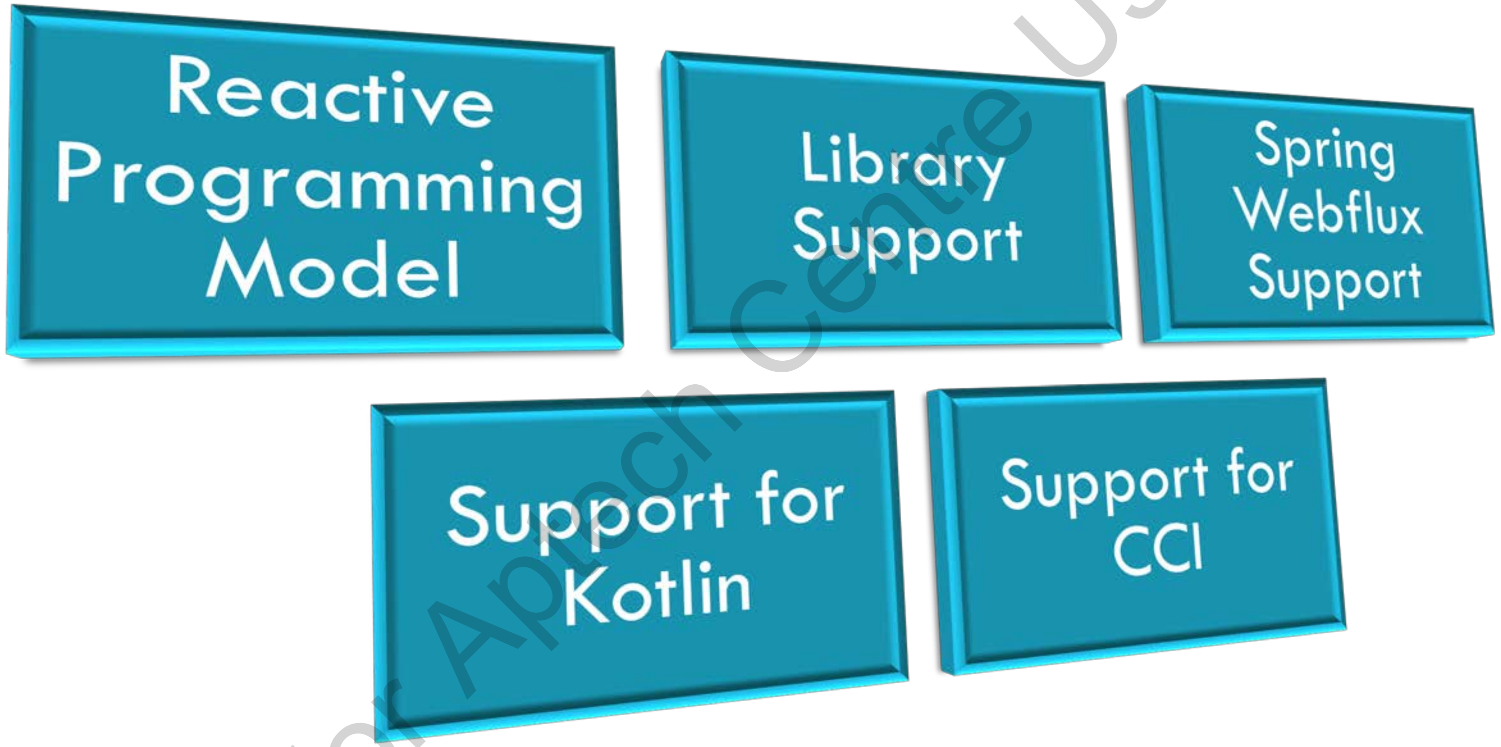
ASPECT-ORIENTED PROGRAMMING (AOP)

- Modular programming methodology that uses aspect as a module unit
- Isolates supporting functions from the main program's business logic, by using concerns



AOP Terms

NEW FEATURES I SPRING 5.0



Features of Spring 5.0

SUMMARY

- IoC is a design principle that provides modularity to application programs by transferring the control of objects to a framework.
- Spring Framework implements IoC through the following two spring containers:
  - BeanFactory
  - Applicationcontext
- DI is a design principle used to implement IoC in Spring Framework.
- Beans are objects of the Spring container.
- AOP methodology isolates supporting functions from the main program's business logic, by using concerns.
- Latest version of Spring framework is 5.0, which includes features, such as JDK baseline update, support for Reactive programming model, to name a few.