Bộ môn: Kỹ Thuật Phần Mềm

Lập trình WWW *Java*

.

Bean Scope - Bean Life Cycle

Bean Scopes

Scope refers to the lifecycle of a bean

How long the bean will live

How many instances are created

How is the bean shared in the spring environment

Default Scope: Singleton

- **Spring Container** creates **only one instance** of the bean by default
- It's **cached in memory**
- **All requests for that bean** will return a **shared reference** to the **same bean**

Default Scope: Singleton

```
Coach theCoach = context.getBean("trackCoach", Coach.class);

...

Coach alphaCoach = context.getBean("trackCoach", Coach.class);
```


```
<bean id="trackCoach"
      class="com.se.springdemo.libs.TrackCoach"
      scope="singleton" >
</bean>
```

Explicitly specify Bean Scope

File: *ApplicationContext.xml*

```
<bean id="trackCoach"
    class="com.se.springdemo.libs.TrackCoach"
    scope="singleton" >
</bean>
```

Additional Spring Bean Scopes

Scopes	Description
Singleton	Create single shared instance of the bean – default scope.
Prototype	Create a new bean instance for each container request.
Request	Scoped for HTTP web request. Only used for web apps.
Session	Scoped for HTTP web session. Only used for web apps.
Global session	Scoped for global HTTP web session. Only used for web apps.

Prototype Scope

File: *beanScope-applicationContext.xml*

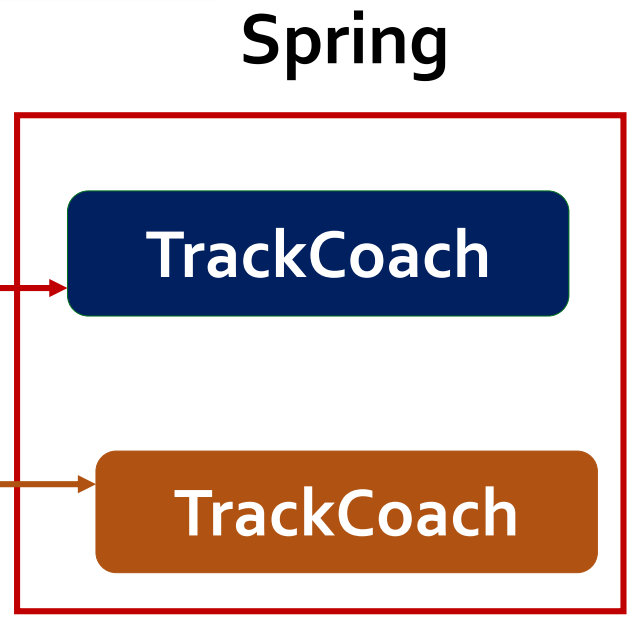
```
<bean id="trackCoach"
    class="com.se.springdemo.libs.TrackCoach"
    scope="prototype" >
</bean>
```

File: *BeanScopeDemoApp.java*

```
Coach theCoach = context.getBean("trackCoach", Coach.class);

...

Coach alphaCoach = context.getBean("trackCoach", Coach.class);
```



Prototype Scope

File: File: *BeanScopeDemoApp.java*

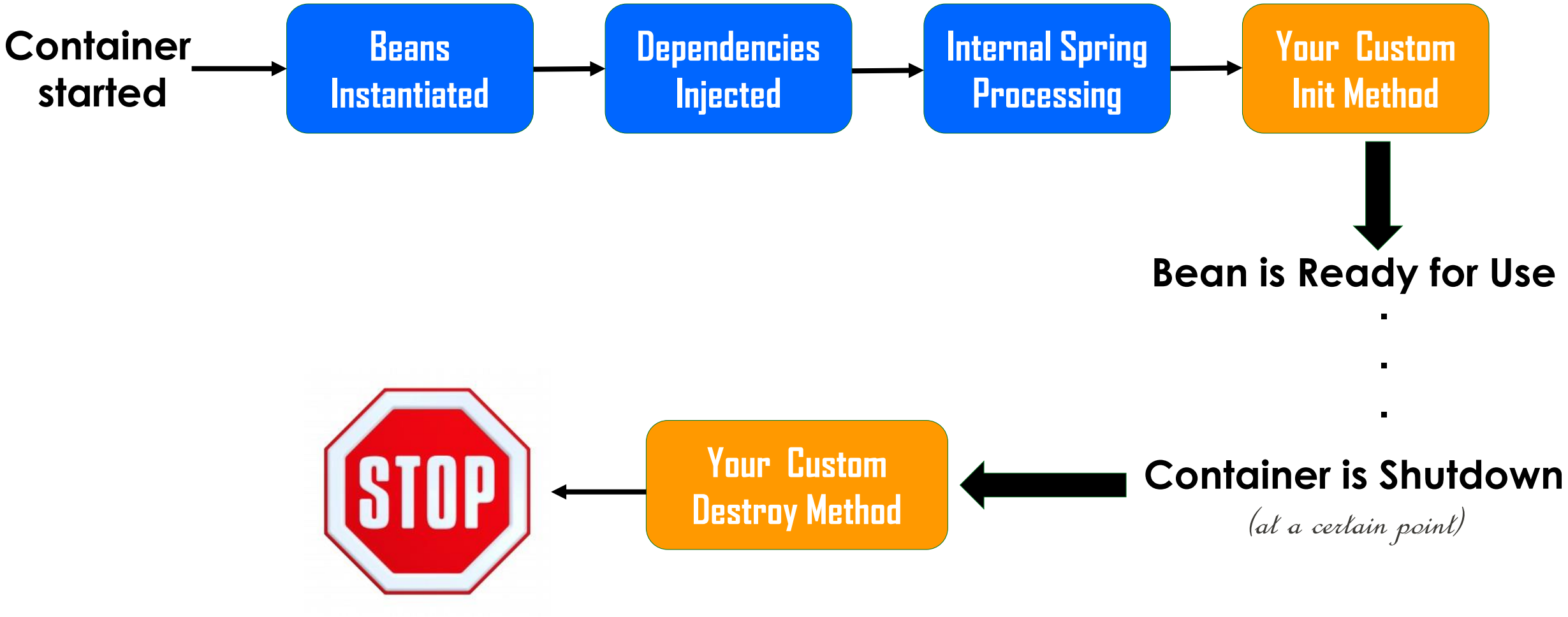
```
boolean result= (theCoach==alphaCoach);
//print out the result
System.out.println("\nPointing to the same object " + result);
System.out.println("\nMemory location for theCoach " + theCoach);
System.out.println("\nMemory location for alphaCoach " + alphaCoach);
```

```
Pointing to the same object false

Memory location for theCoach com.se.springdemo.libs.TrackCoach@21b2e768

Memory location for alphaCoach com.se.springdemo.libs.TrackCoach@57250572
```

Bean Lifecycle



Bean Lifecycle Methods

You can add **custom code** during **bean initialization:**

- Calling **custom business logic** methods
- Setting up handles to **resources** (databases, sockets files etc)

You can add **custom code** during **bean destruction:**

- Calling **custom business logic** methods
- Cleanup handles to **resources** (databases, sockets files etc)

Example of Bean Lifecycle Methods

File:*TrackCoach.java*

```
//init method
public void doMyStartupStuff() {
        System.out.println("TrackCoach: inside method doMyStartupStuff");
}
//destroy method
public void doMyCleanupStuff() {
        System.out.println("TrackCoach: inside method doMyStartupStuff");
}
```

File:*beanLifeCycle-applicationContext.xml*

```
<bean id="myCoach"
    class="com.se.springdemo.libs.TrackCoach"
    init-method="doMyStartupStuff"
    destroy-method="doMyCleanupStuff">
    <!-- set up constructor injection -->
    <constructor-arg ref="myFortune" />
</bean>
```

QUESTIONS