# Spring Life Cycle

To explain the concept of Spring life cycle, lets change the beanLifeCycle-applicationContext.xml file.

<beans…>

```
 <bean id="myService"
      class="com.branch.springwork.MyEmployeeService" />

    <bean id="myTeam"
      class="com.branch.springwork.MarkettingTeam"
      init-method = "doMyStartupStuff"
      destroy-method = "doMyCleanupStuff">

       <!-- Set up the constructor injection -->
       <constructor-arg ref="myService" />
    </bean>
```

</beans>

Notice that in xml file we have added init-method and destroy-method.

```
init-method = "doMyStartupStuff"
destroy-method = “doMyCleanupStuff"
```

doMyStartupStuff and doMyCleanupStuff are implemented as a public void in Markettingteam.java file.

```
// add an init method
public void doMyStartupStuff() {
      System.out.println("MarkettingTeam: doMyStartupStuff--- inside
method doMyStartupStuff");
}

// add a destroy method
public void doMyCleanupStuff() {
      System.out.println("MarkettingTeam: doMyCleanupStuff---: inside
method doMyCleanupStuffYoYo");
}
```

**Console output:**

```
MarkettingTeam: doMyStartupStuff--- inside method doMyStartupStuff

Marketting team:--- Working hard to make customer happy
Marketting team:--- Market Sale is Current and Progressive!

MarkettingTeam: doMyCleanupStuff---: inside method doMyCleanupStuff
```

**Conclusion:**

**Container Started** => Bean Instantiated => Dependencies Injected => Internal Spring Processing => Your Custom Init method => Bean is Ready for Use/Container is Shutdown => Your Custom Destroy Method =>**Stop**

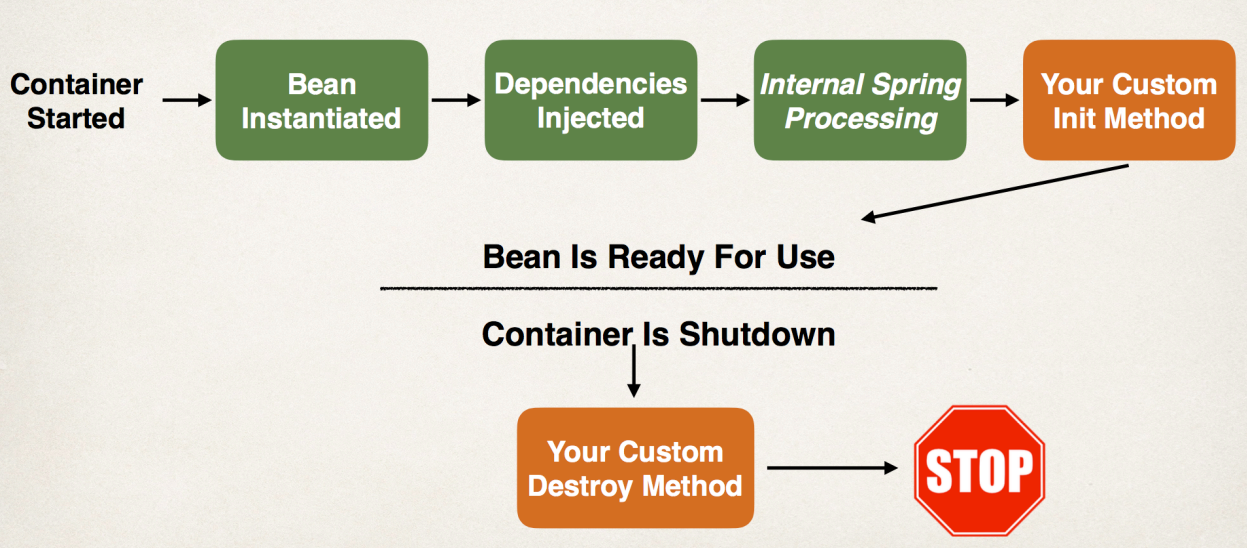


Image Reference: luv2code.com