

**Property Editors**

**PropertyEditors are used to edit the property values. You can do the following using PropertyEditors**

* **Convert the value from one data type to another data type**
* **Convert the value from one format to another format.**

**o**

**Convert the Property Value from One Data Type to Another Data Type. Ex:**

**"B99-101" -> StudentId Object "36000,20000,16000" => Fee Object**

**Convert the Property Value from One Format to Another Format Ex:**

**"123" -> "JLC-123" "JLC-123" -> "123"**

**Steps to develop and Register custom property editors**

1. **Write your own editor class by extending PropertyEditorSupport class available in java.beans package.**
2. **Override the following method in Custom Editor class**

**public void setAsText(String txt)**

1. **Write the required conversion logic inside setAsText() method. Ex:**

**public class FeeEditor extends PropertyEditorSupport { public void setAsText(String txt) {**

**String str[]=txt.split(",");**

**double totalFee = Double.parseDouble(str[0]); double feePaid = Double.parseDouble(str[1]); double feeBal = Double.parseDouble(str[2]);**

**Fee myfee=new Fee();**

**myfee.setTotalFee(totalFee);**

**myfee.setFeePaid(feePaid);**

**myfee.setFeeBal(feeBal);**

**setValue(myfee);**

**}**

**}**

1. **Register the Custom property editors with spring container as follows.**

**@InitBinder**

**public void regsiterMyEditors(WebDataBinder binder) { binder.registerCustomEditor(Fee.class, new FeeEditor());**

**…..**

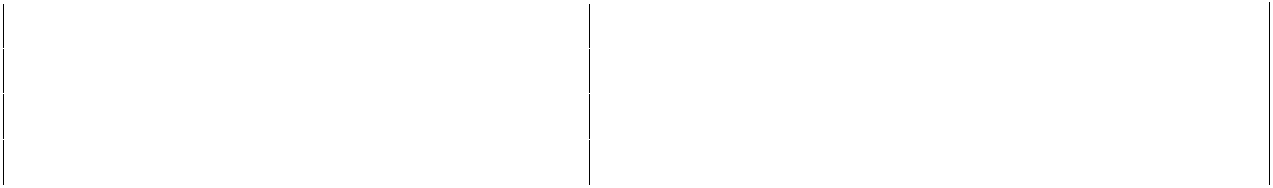
**}**



**www.coursecube.com** **120** **Spring-5 Study Guide**



**Lab45: Files required**



|  |  |  |  |
| --- | --- | --- | --- |
| **1.** | **Lab45.java** | **2.** | **Student.java** |
|  |  |  |  |
| **3.** | **StudentID.java** | **4.** | **StudentIDEditor.java** |
|  |  |  |  |
| **5.** | **Fee.java** | **6.** | **FeeEditor.java** |
|  |  |  |  |
| **7.** | **JLCAppConfig.java** |  |  |
|  |  |  |  |



**1. Lab43.java**

**package com.coursecube.spring;**

**import org.springframework.context.ApplicationContext;**

**import org.springframework.context.annotation.AnnotationConfigApplicationContext; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class Lab45 {**

**public static void main(String[] args) {**

**ApplicationContext ctx=new AnnotationConfigApplicationContext(JLCAppConfig.class); System.out.println("---------Now Spring Container is Ready-----");**

**Student stu=ctx.getBean("mystudent",Student.class);**

**System.out.println(stu);**

**}**

**}**



**2. Student.java**

**package com.coursecube.spring;**

**import javax.annotation.Resource;**

**/\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class Student {**

**@Resource(name="mystuId")**

**StudentID stuId; //F.I**

**String sname; //C.I**

**String email; //C.I**

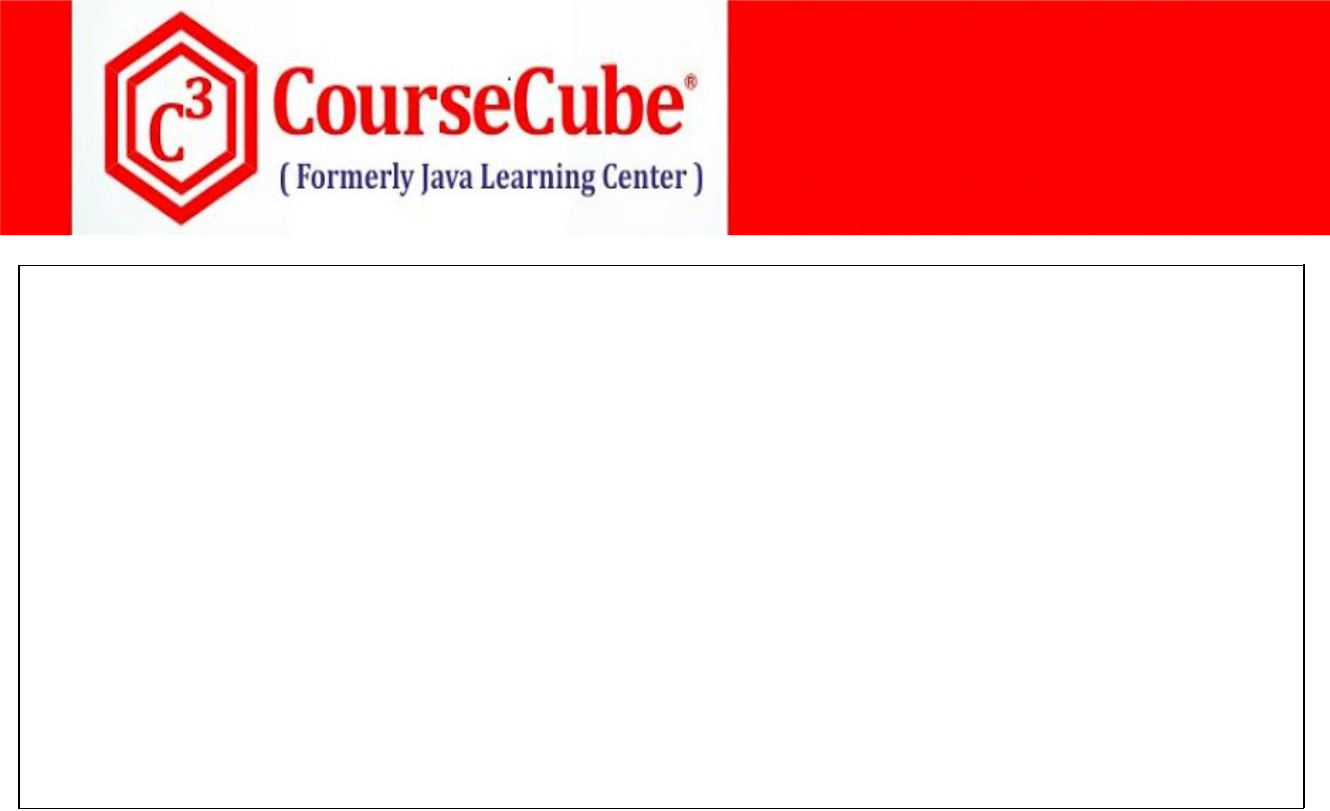
**String course; //C.I**

**@Resource(name="myfee")**

**Fee fee;** **//F.I**



**www.coursecube.com** **121** **Spring-5 Study Guide**



**public Student() {}**

**public Student(String sname, String email, String course) { super();**

**this.sname = sname;**

**this.email = email;**

**this.course = course;**

**}**

**//Setters and Getters**

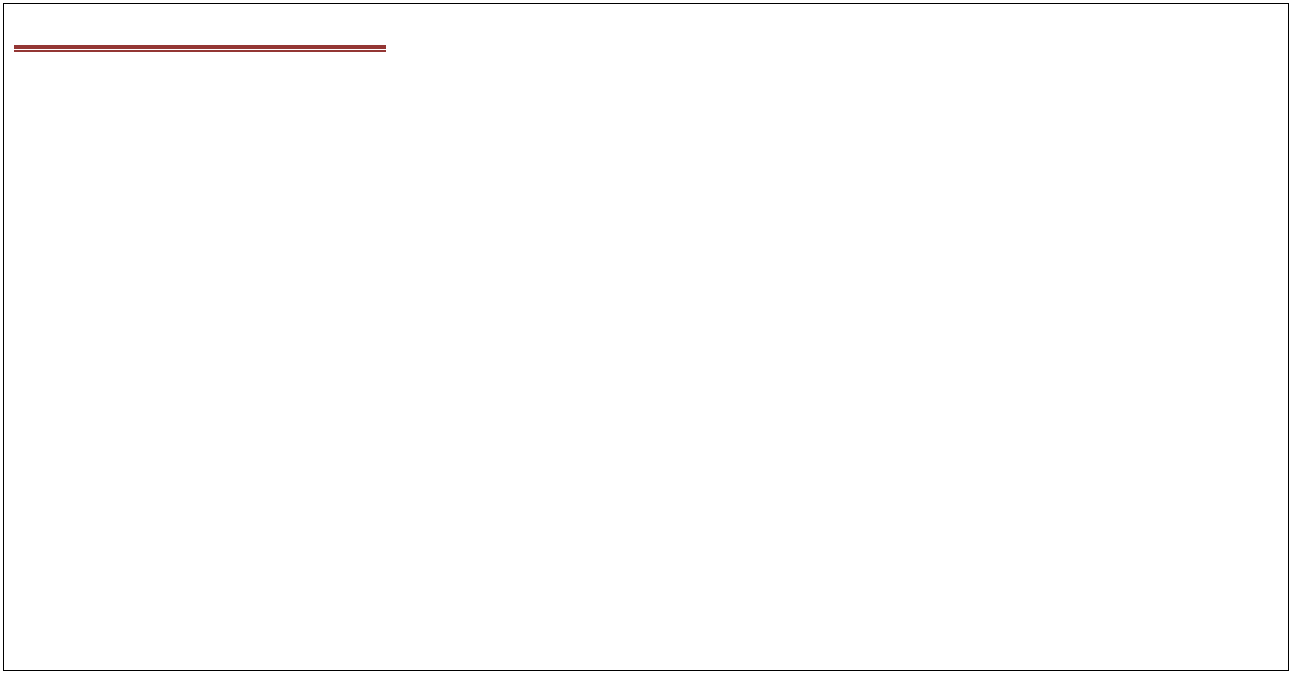
**@Override**

**public String toString() {**

**return stuId + "\t" + sname + "\t" + email + "\t" + course + "\t" + fee ;**

**}**

**}**



**3. StudentID.java**

**package com.coursecube.spring;**

**/\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class StudentID {**

**private int sid; //S.I**

**private String bid; //S.I**

**//Setters and Getters**

**@Override**

**public String toString() {**

**return bid + "\t" + sid ;**

**}**

**}**



1. **StudentIDEditor.java package com.coursecube.spring;**

**import java.beans.PropertyEditorSupport;**

**/\***

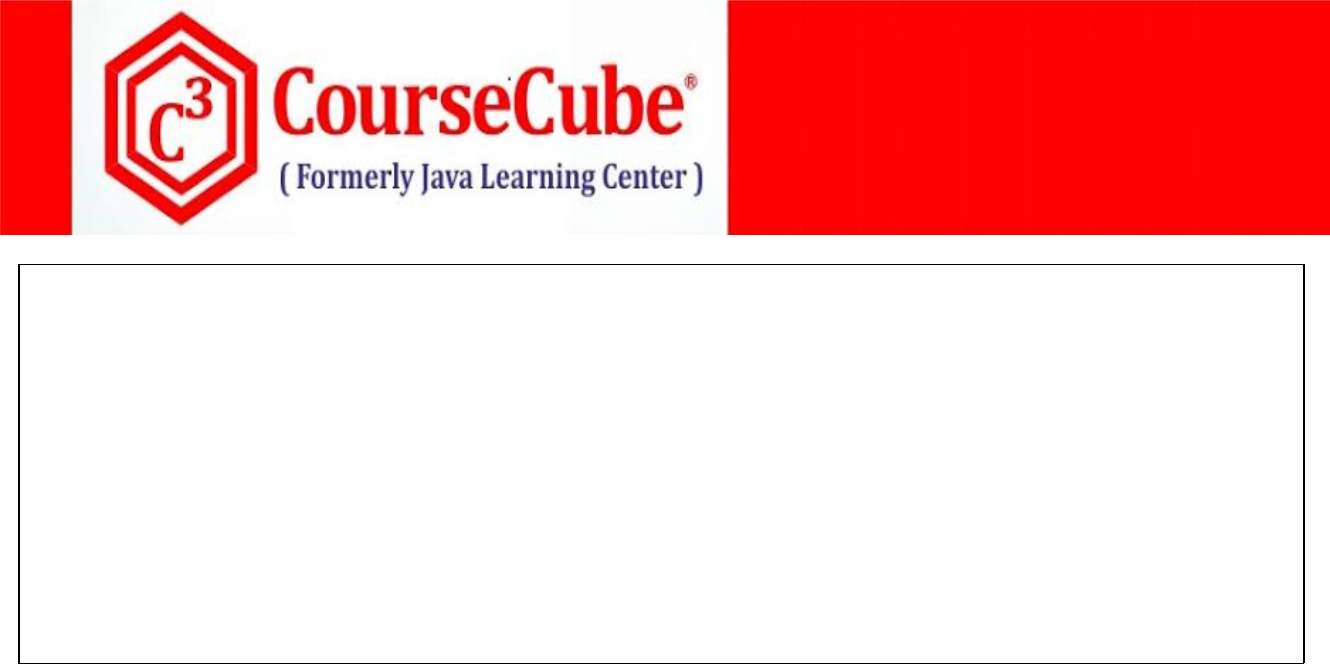
* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class StudentIDEditor extends PropertyEditorSupport { public void setAsText(String txt) {**

**System.out.println("StudentIDEditor - setAsText()"); //txt => B99-101**



**www.coursecube.com** **122** **Spring-5 Study Guide**



**String str[]=txt.split("-");**

**String bid=str[0];**

**int sid=Integer.parseInt(str[1]);**

**StudentID stuId=new StudentID();**

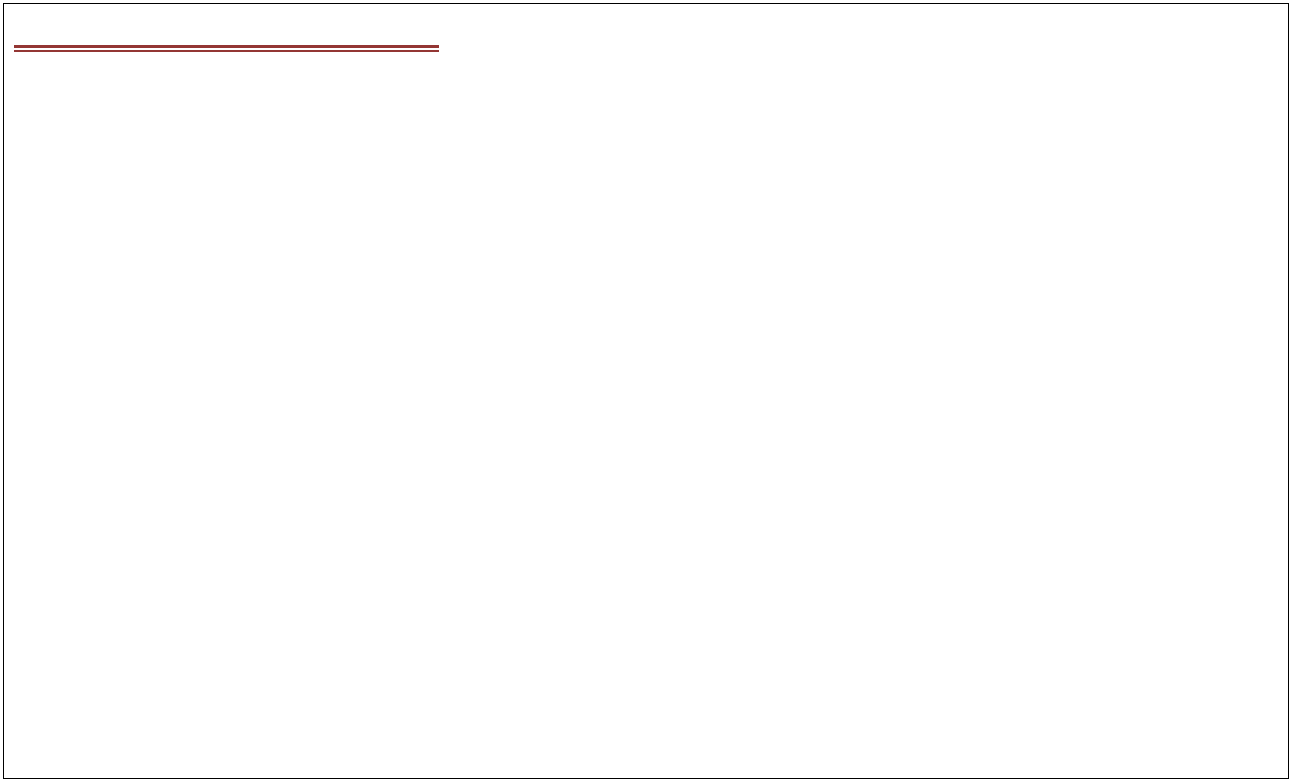
**stuId.setBid(bid);**

**stuId.setSid(sid);**

**setValue(stuId);**

**}**

**}**



**5. Fee.java**

**package com.coursecube.spring;**

**/\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class Fee {**

**private double totalFee; //S.I**

**private double feePaid; //S.I**

**private double feeBal; //S.I**

**public Fee() {}**

**//Setters and Getters**

**@Override**

**public String toString() {**

**return totalFee + "\t " + feePaid + "\t " + feeBal ;**

**}**

**}**



**6. FeeEditor.java**

**package com.coursecube.spring;**

**import java.beans.PropertyEditorSupport;**

**/\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

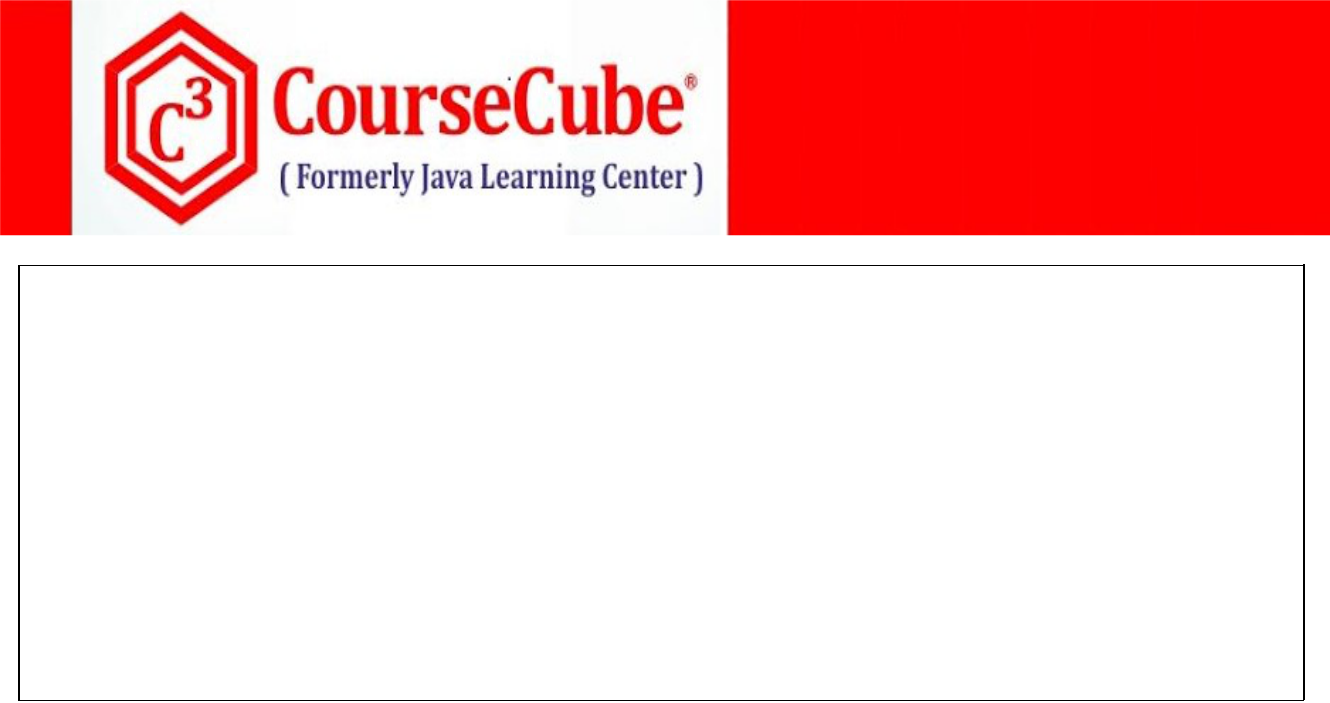
**public class FeeEditor extends PropertyEditorSupport { public void setAsText(String txt) {**

**System.out.println("FeeEditor - setAsText()");**

**//txt => "36000,20000,16000"**



**www.coursecube.com** **123** **Spring-5 Study Guide**



**String str[]=txt.split(",");**

**double totalFee = Double.parseDouble(str[0]);**

**double feePaid = Double.parseDouble(str[1]);**

**double feeBal = Double.parseDouble(str[2]);**

**Fee myfee=new Fee();**

**myfee.setTotalFee(totalFee);**

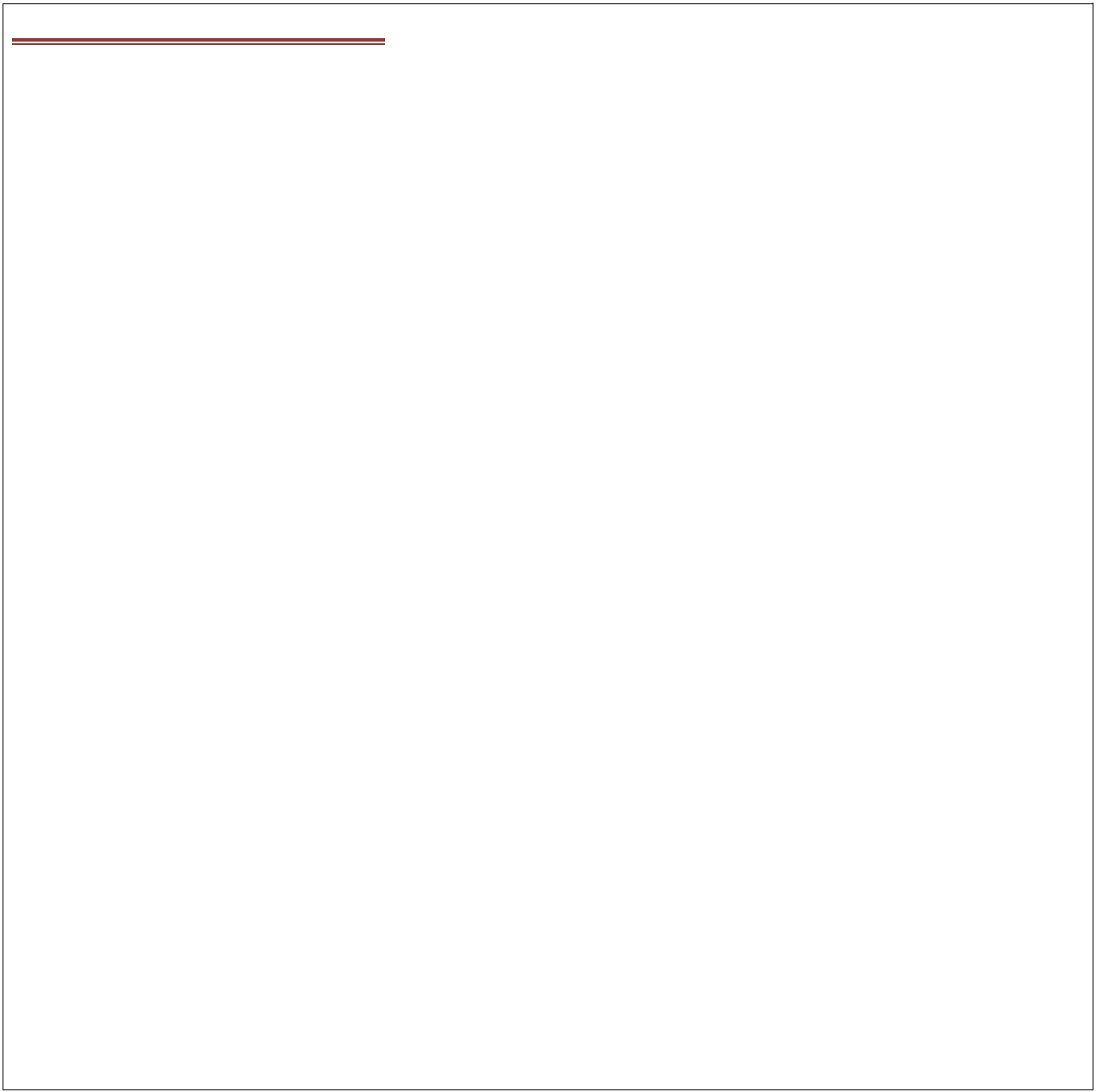
**myfee.setFeePaid(feePaid);**

**myfee.setFeeBal(feeBal);**

**setValue(myfee);**

**}**

**}**



1. **JLCAppConfig.java package com.coursecube.spring;**

**import org.springframework.context.annotation.Bean;**

**import org.springframework.context.annotation.Configuration; import org.springframework.web.bind.WebDataBinder; import org.springframework.web.bind.annotation.InitBinder; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**@Configuration**

**public class JLCAppConfig{**

**/\***

**@Bean(name="mystuId")**

**public StudentID getStuId() {**

**StudentID stuId= new StudentID();**

**stuId.setBid("B-99");**

**stuId.setSid(101);**

**return stuId;**

**}**

**@Bean(name="myfee")**

**public Fee getFee() {**

**Fee fee= new Fee();**

**fee.setTotalFee(36000);**

**fee.setFeePaid(20000);**

**fee.setFeeBal(16000);**

**return fee;**

**}**

**\*/**

**@Bean(name="mystuId")**

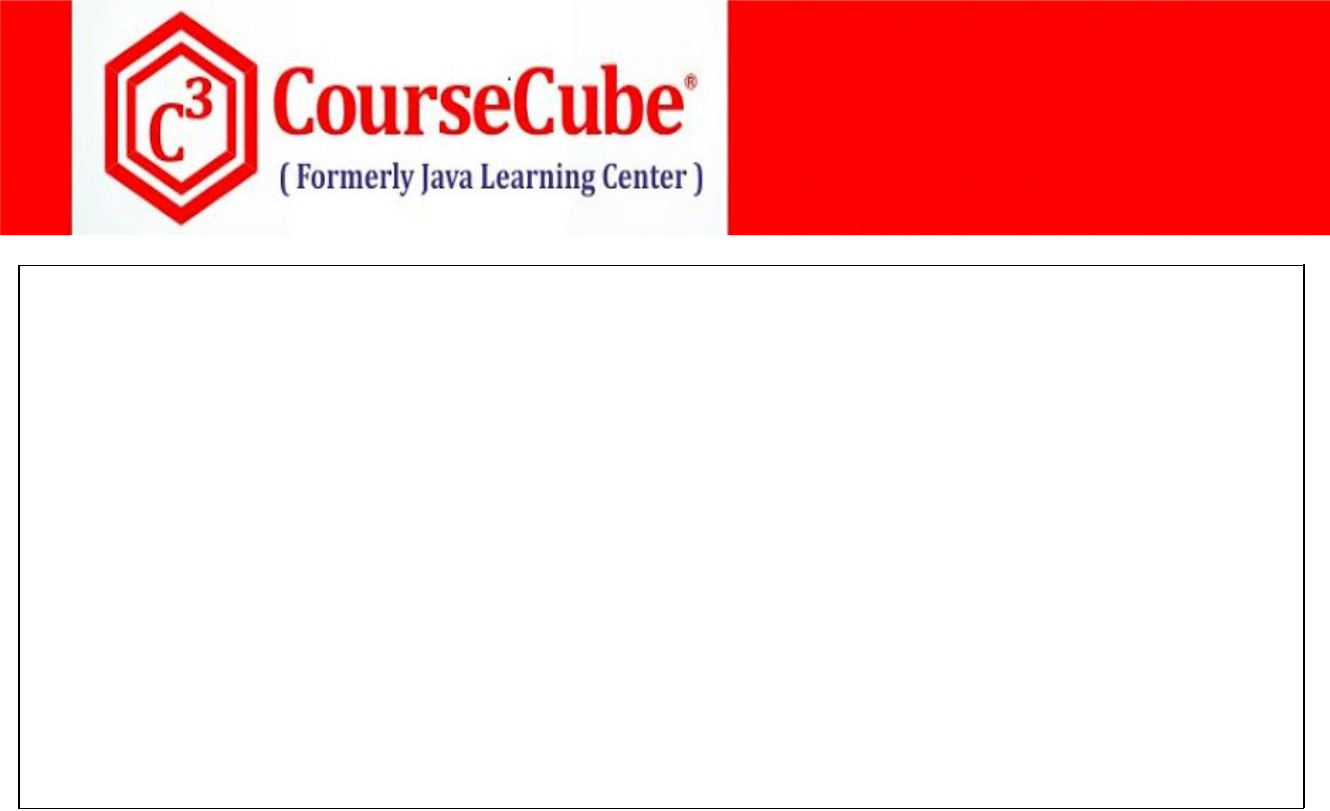
**public String getStuId() {**

**return "B99-101";**

**}**



**www.coursecube.com** **124** **Spring-5 Study Guide**



**@Bean(name="myfee")**

**public String getFee() {**

**return "36000,20000,16000";**

**}**

**@Bean(name="mystudent")**

**public Student getStudent() {**

**Student stu= new Student("Srinivas","sri@cc.com","Full Stack Course"); return stu;**

**}**

**@InitBinder**

**public void registerMyEditors(WebDataBinder mydataBinder) { mydataBinder.registerCustomEditor(StudentID.class, new StudentIDEditor()); mydataBinder.registerCustomEditor(Fee.class, new FeeEditor());**

**}**

**}**



**Events and Listeners**

**Event Handling mechanism is provided with Spring Container called ApplicationContext . ApplicationContext Container handles two types of Events:**

* 1. **Standard or System Events**
  2. **Custom or Application Events.**

**1) Standard Events or System Events**

**Spring provides the following standard events: ContextStartedEvent**

**ContextRefreshedEvent ContextStoppedEvent ContextClosedEvent**

**When start() method is called on ConfigurableApplicationContext then ContextStartedEvent will be published by the Container.**

**When refresh() method is called on ConfigurableApplicationContext then ContextRefreshedEvent will be published by the Container.**

**You should not call refresh() method . if you call then Exception will be thrown: java.lang.IllegalStateException: GenericApplicationContext does not support multiple refresh attempts: just call 'refresh' once**

**When stop() method is called on ConfigurableApplicationContext then ContextStoppedEvent will be published by the Container.**

**When close() method is called on ConfigurableApplicationContext then ContextClosedEvent will be published by the Container.**



**www.coursecube.com** **125** **Spring-5 Study Guide**



**Writing Listeners**

**There are two ways to write Listeners.**

* 1. **Writing Listener with ApplicationListener interface**
  2. **Writing Listener with @EventListener annotation.**

1. **Writing Listener with ApplicationListener interface**
2. **Write your own listener class by implementing ApplicationListener interface public class JLCListener implements ApplicationListener<ApplicationEvent>{**

**}**

1. **Override the following method**

**@Override**

**public void onApplicationEvent(ApplicationEvent event) {**

**}**

**3) Write the required logic inside onApplicationEvent() method.**

1. **Writing Listener with @EventListener annotation.**
2. **Write your own listener class public class MyListener {**

**}**

1. **Write your own method in Custom listener class with the parameter of Event type.**

**public void handleMyEvent(ApplicationEvent event) {**

**}**

**3) Mark that method with @EventListener annotation.**

**@EventListener**

**public void handleMyEvent(ApplicationEvent event) {**

**}**

**4) Write the required logic inside that method method.**



**www.coursecube.com** **126** **Spring-5 Study Guide**



**Register Listeners with Container**

**Just the Configure the Bean in Java Configuration class.**

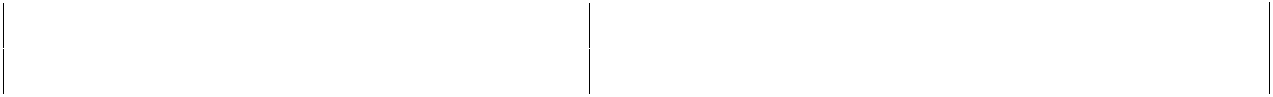
**@Bean**

**public JLCListener getJLCListener() {**

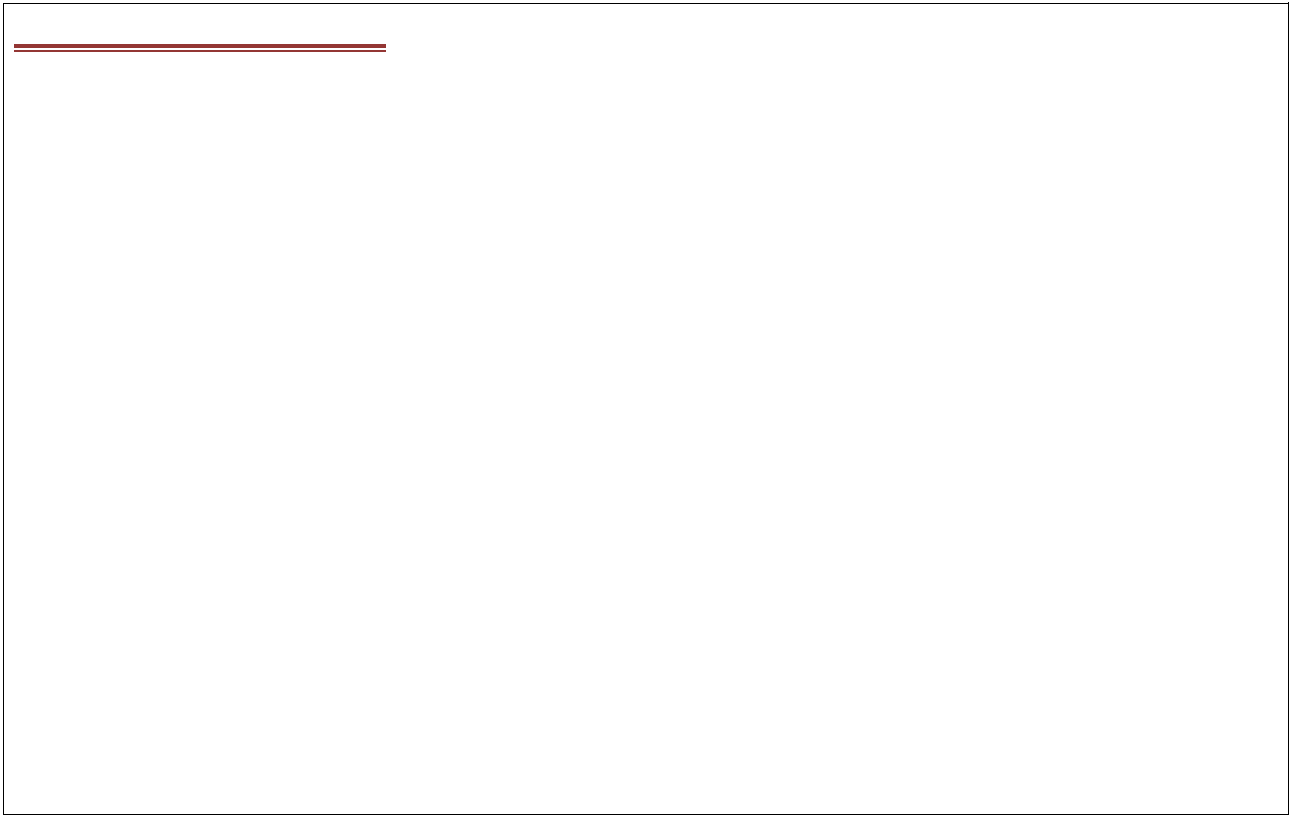
**return new JLCListener();**

**}**

**Lab46: Files required**



|  |  |  |  |
| --- | --- | --- | --- |
| **1.** | **Lab46.java** | **2.** | **JLCListener.java** |
|  |  |  |  |
| **3.** | **MyListener.java** | **4.** | **JLCAppConfig.java** |
|  |  |  |  |



**1. Lab46.java**

**package com.coursecube.spring;**

**import org.springframework.context.ConfigurableApplicationContext;**

**import org.springframework.context.annotation.AnnotationConfigApplicationContext; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class Lab46 {**

**public static void main(String[] args) {**

**ConfigurableApplicationContext ctx=new AnnotationConfigApplicationContext(JLCAppConfig.class); System.out.println("---------Now Spring Container is Ready-----");**

**//ctx.refresh();**

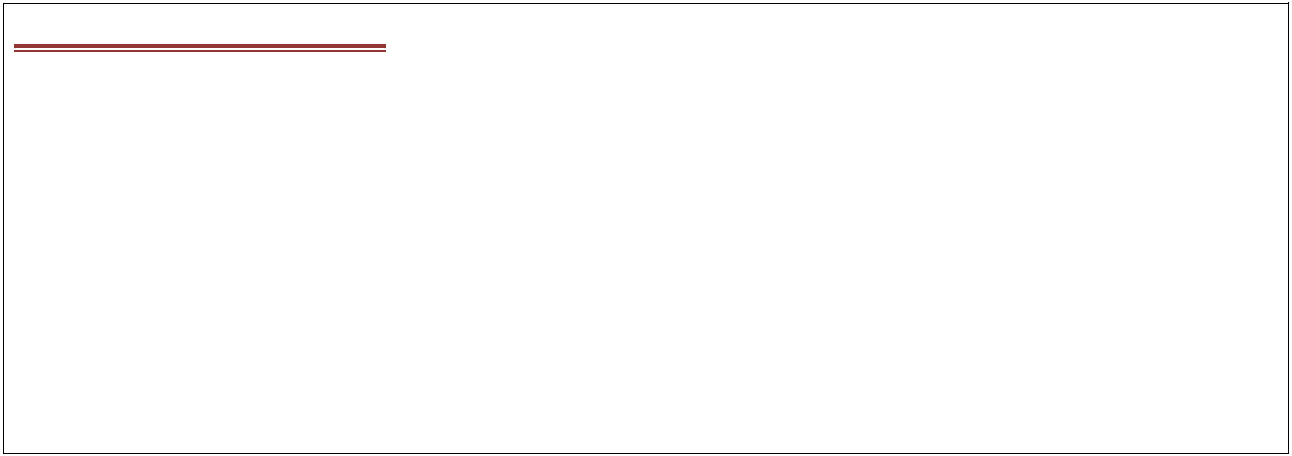
**ctx.start();**

**ctx.stop();**

**ctx.close();**

**}**

**}**



1. **JLCListener.java package com.coursecube.spring;**

**import org.springframework.context.ApplicationEvent; import org.springframework.context.ApplicationListener; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class JLCListener implements ApplicationListener<ApplicationEvent>{**



**www.coursecube.com** **127** **Spring-5 Study Guide**



**@Override**

**public void onApplicationEvent(ApplicationEvent event) { System.out.println("JLCListener - onApplicationEvent()"); System.out.println("ApplicationEvent is Raised"); System.out.println(event.getClass().getSimpleName());**

**System.out.println("-----------------------------");**

**}**

**}**



1. **MyListener.java package com.coursecube.spring;**

**import org.springframework.context.ApplicationEvent; import org.springframework.context.event.EventListener; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class MyListener {**

**@EventListener**

**public void handleMyEvent(ApplicationEvent event) { System.out.println("MyListener - handleMyEvent()"); System.out.println("ApplicationEvent is Raised"); System.out.println(event.getClass().getSimpleName());**

**System.out.println("-----------------------------");**

**}**

**}**



1. **JLCAppConfig.java package com.coursecube.spring;**

**import org.springframework.context.annotation.Bean;**

**import org.springframework.context.annotation.Configuration;**

**@Configuration**

**public class JLCAppConfig{**

**@Bean**

**public JLCListener getJLCListener() {**

**return new JLCListener();**

**}**

**@Bean**

**public MyListener getMyListener() {**

**return new MyListener();**

**}**

**}**



**www.coursecube.com** **128** **Spring-5 Study Guide**



**Writing Application Events**

**There are Two ways to write Application or Custom Events.**

* 1. **Writing Events with ApplicationEvent class**
  2. **Writing Events without ApplicationEvent class**

1. **Writing Events with ApplicationEvent class**
2. **Write your Event class by extending ApplicationEvent public class BatchEvent extends ApplicationEvent{ //1**

**}**

1. **Declare the variables to hold the information about the Event.**

**public class BatchEvent extends ApplicationEvent{ BatchTO batchTO; //2**

**}**

1. **Write two argument constructor in the event class**
   1. **First Argument is Source of Event**
   2. **Second Argument is Information of the Event**

**Public class BatchEvent extends ApplicationEvent{ BatchTO batchTO;**

**public BatchEvent(Object source,BatchTO batchTO) { //3 super(source);**

**this.batchTO = batchTO;**

**}**

**}**

**4) Wrtie Getter method**

**public class BatchEvent extends ApplicationEvent{ BatchTO batchTO;**

**public BatchEvent(Object source,BatchTO batchTO) { super(source);**

**this.batchTO = batchTO;**

**}**

**public BatchTO getBatchTO() { //4**

**return batchTO;**

**}**

**}**



**www.coursecube.com** **129** **Spring-5 Study Guide**



1. **Writing Events with ApplicationEvent class**

**1) Write your Event class**

**public class BatchEvent { //1**

**}**

1. **Declare the variables to hold the information about the Event.**

**public class BatchEvent extends ApplicationEvent{ BatchTO batchTO; //2**

**}**

1. **Write One argument constructor in the event class a. One argument is Information of the Event**

**Public class BatchEvent extends ApplicationEvent{ BatchTO batchTO;**

**public BatchEvent(BatchTO batchTO) { //3**

**this.batchTO = batchTO;**

**}**

**}**

**4) Wrtie Getter method**

**public class BatchEvent extends ApplicationEvent{ BatchTO batchTO;**

**public BatchEvent(BatchTO batchTO) {**

**this.batchTO = batchTO;**

**}**

**public BatchTO getBatchTO() { //4**

**return batchTO;**

**}**

**}**

**Publishing Application Events**

**You can publish the events either with ApplicationContext or ApplicationEventPublisher**

**BatchEvent batchEvent = new BatchEvent(....);**

**ctx.publishEvent(batchEvent);**

**or**

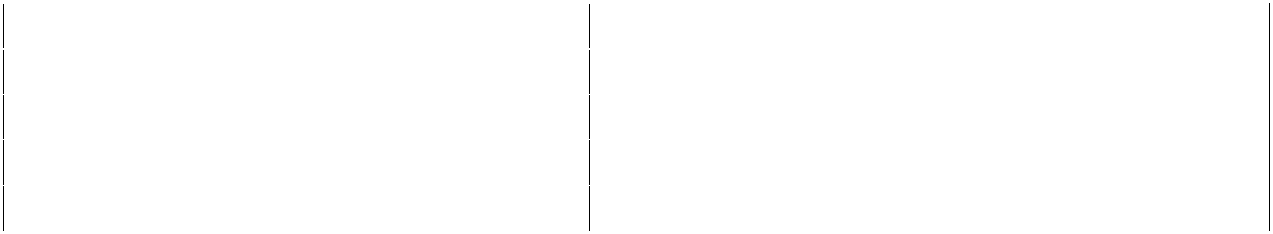
**publisher.publishEvent(batchEvent);**



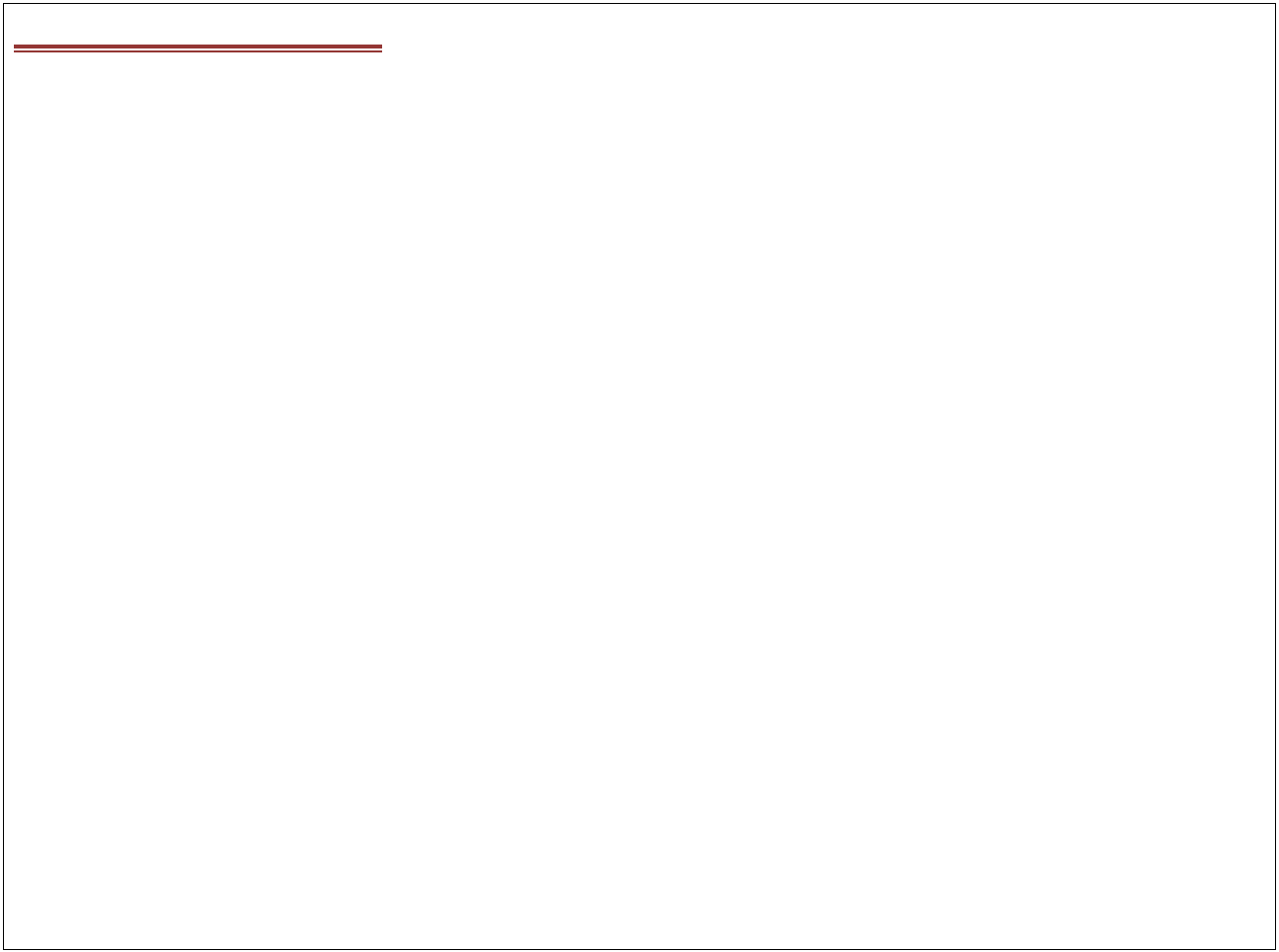
**www.coursecube.com** **130** **Spring-5 Study Guide**



**Lab47: Files required**



|  |  |  |  |
| --- | --- | --- | --- |
| **1.** | **Lab47.java** | **2.** | **JLCManagerService.java** |
|  |  |  |  |
| **3.** | **BatchTO.java** | **4.** | **BatchEvent.java** |
|  |  |  |  |
| **5.** | **WorkShopTO.java** | **6.** | **WorkShopEvent.java** |
|  |  |  |  |
| **7.** | **BatchListener.java** | **8.** | **WorkShopListener.java** |
|  |  |  |  |
| **9.** | **JLCAppConfig.java** |  |  |
|  |  |  |  |



**1. Lab47.java**

**package com.coursecube.spring;**

**import org.springframework.context.ApplicationContext;**

**import org.springframework.context.annotation.AnnotationConfigApplicationContext; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class Lab47 {**

**public static void main(String[] args) {**

**ApplicationContext ctx=new AnnotationConfigApplicationContext(JLCAppConfig.class); System.out.println("---------Now Spring Container is Ready-----");**

**JLCManagerService jlc = (JLCManagerService) ctx.getBean("myjlc");**

**BatchTO bto = new BatchTO("B-24", "28-Apr-2010", "6.30-8.30", 150); jlc.addBatch(bto);**

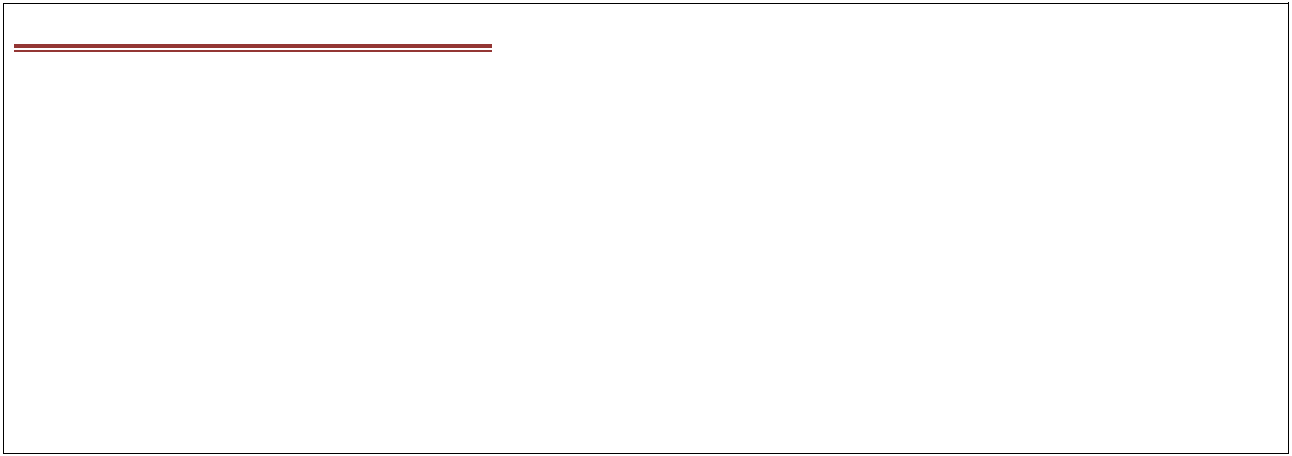
**System.out.println("---------------------");**

**WorkshopTO wsto = new WorkshopTO("Spring Boot ", 5000, 99);**

**jlc.addWorkshop(wsto);**

**}**

**}**



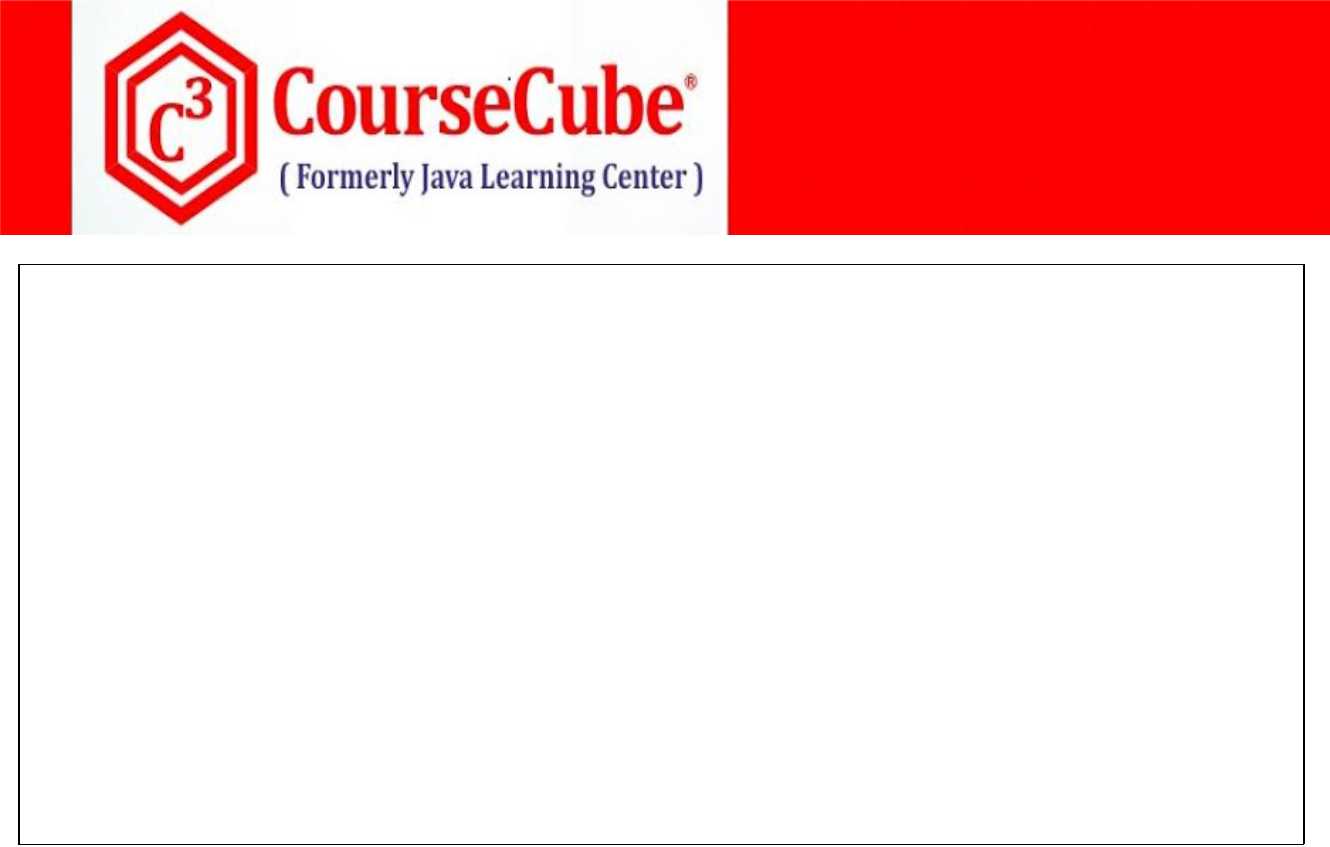
1. **JLCManagerService.java package com.coursecube.spring;**

**import org.springframework.beans.factory.annotation.Autowired; import org.springframework.context.ApplicationContext; import org.springframework.context.ApplicationEventPublisher; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**



**www.coursecube.com** **131** **Spring-5 Study Guide**



**public class JLCManagerService {**

**@Autowired**

**private ApplicationEventPublisher publisher;**

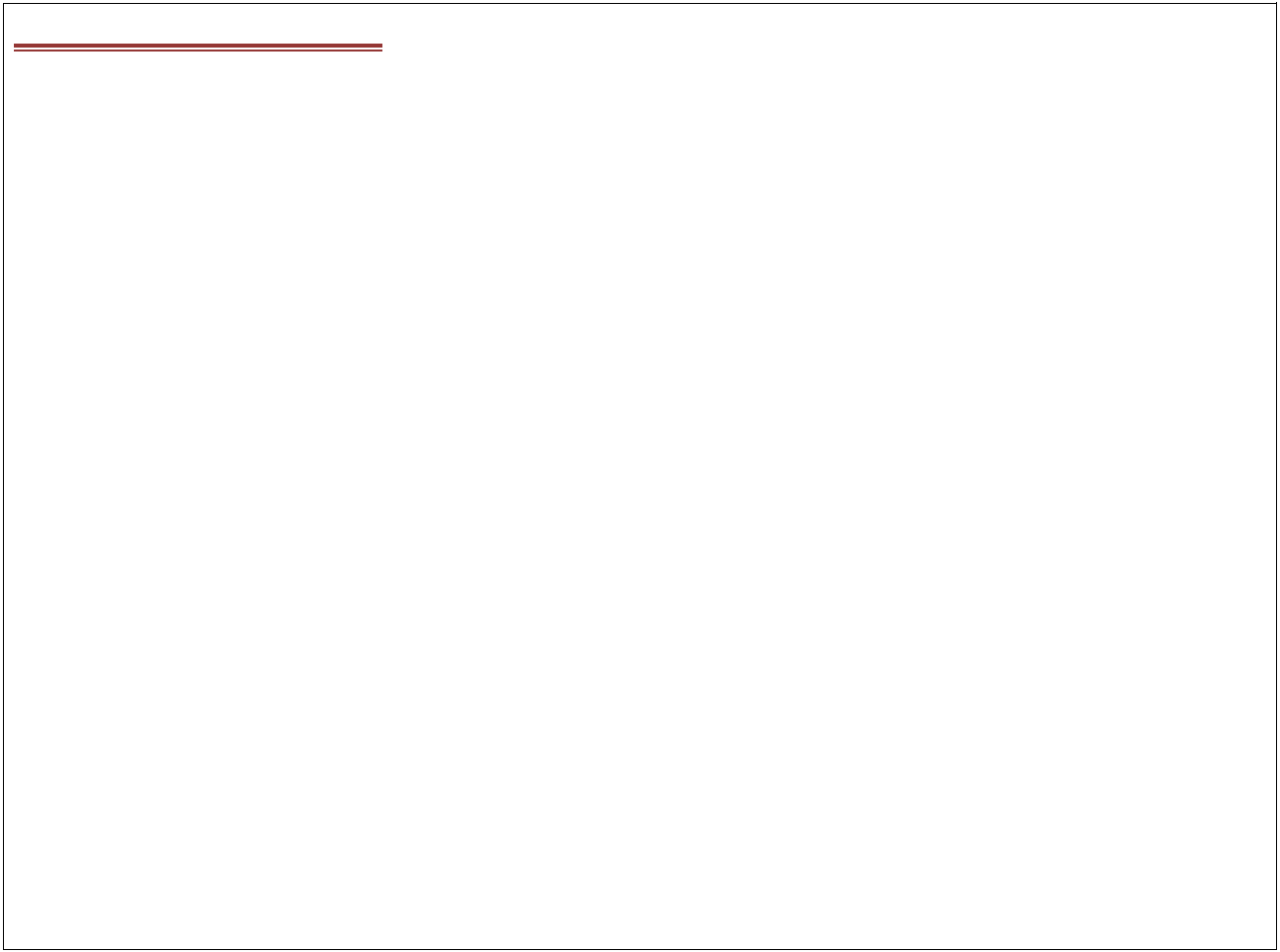
**@Autowired**

**private ApplicationContext ctx;**

**public void addBatch(BatchTO batchTO) { publisher.publishEvent(new BatchEvent(this,batchTO)); }**

**public void addWorkshop(WorkshopTO workshopTO) { ctx.publishEvent(new WorkshopEvent(this,workshopTO)); }**

**}**



**3. BatchTO.java**

**package com.coursecube.spring;**

**/\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class BatchTO {**

**String bid;**

**String sd;**

**String timings;**

**int nos;**

**public BatchTO(String bid, String sd, String timings, int nos) { this.bid = bid;**

**this.sd = sd;**

**this.timings = timings;**

**this.nos = nos;**

**}**

**public String toString() {**

**String msg = "JLC Announces... New Batch";**

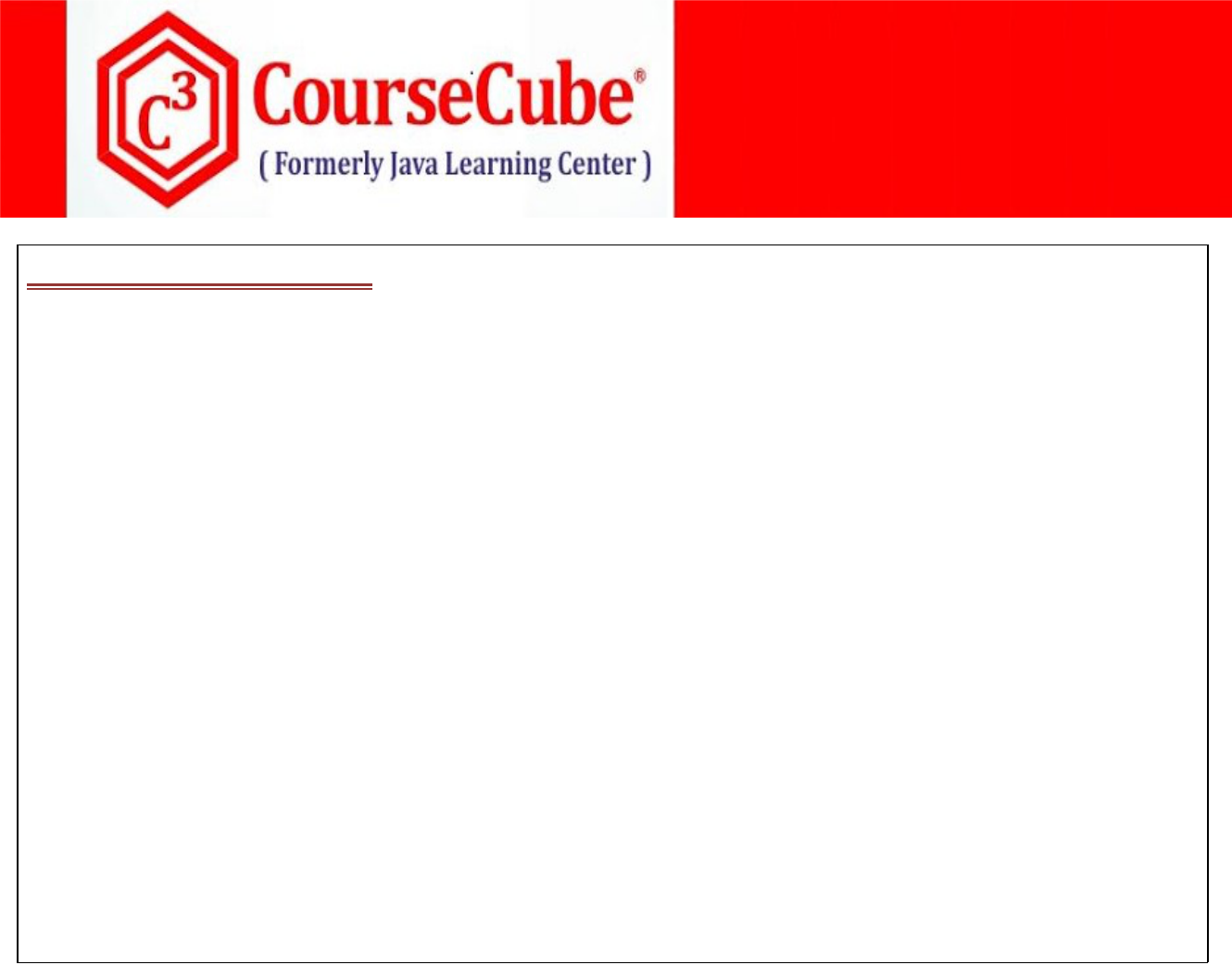
**msg = msg + "\n" + bid + "\t" + sd + "\t" + timings + "\t" + nos; return msg;**

**}**

**}**



**www.coursecube.com** **132** **Spring-5 Study Guide**



1. **BatchEvent.java package com.coursecube.spring;**

**import org.springframework.context.ApplicationEvent; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class BatchEvent extends ApplicationEvent{ BatchTO batchTO;**

**public BatchEvent(Object source,BatchTO batchTO) { super(source);**

**this.batchTO = batchTO;**

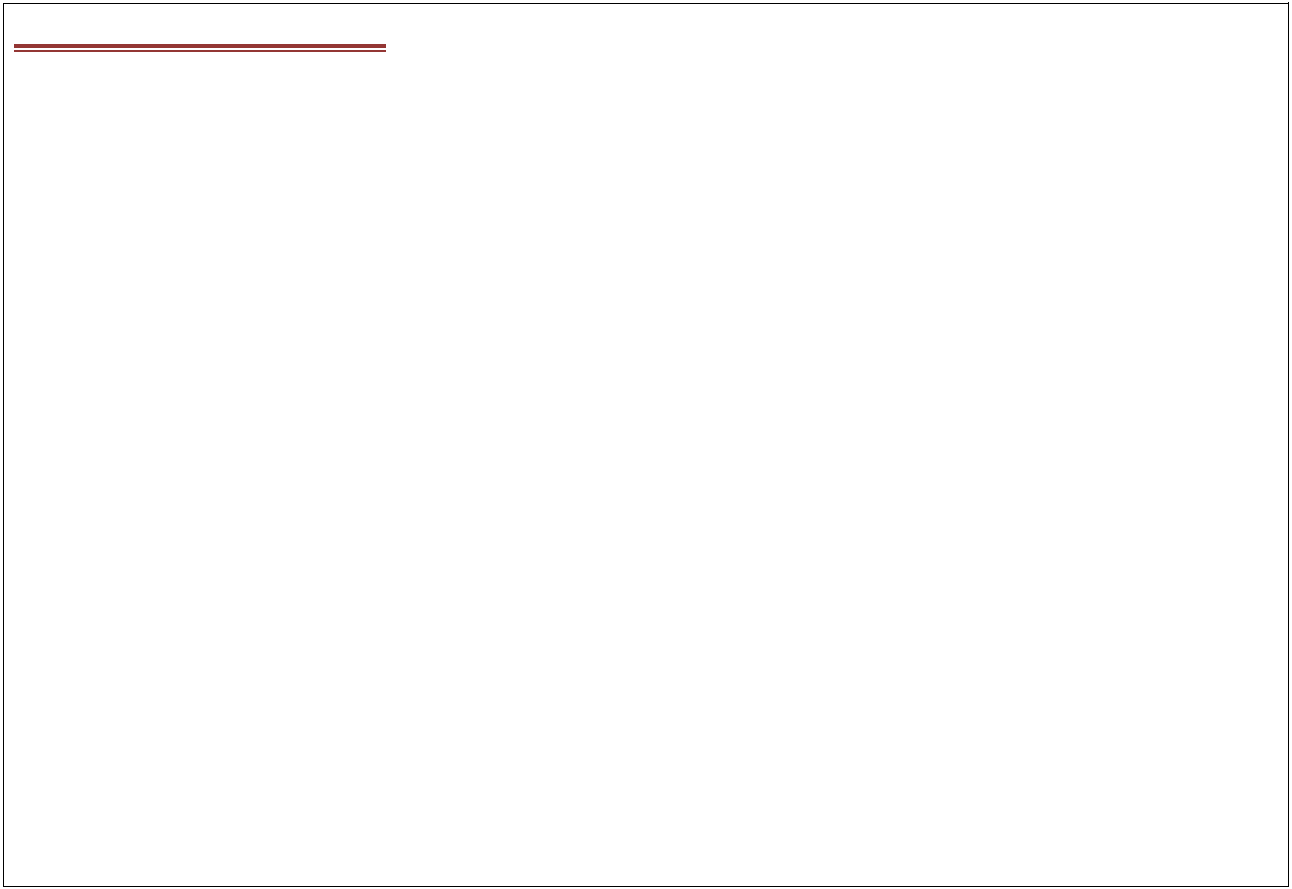
**}**

**public BatchTO getBatchTO() {**

**return batchTO;**

**}**

**}**



**5. WorkShopTO.java**

**package com.coursecube.spring;**

**/\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class WorkShopTO {**

**String topic;**

**double fee;**

**int nos;**

**public WorkShopTO(String topic, double fee, int nos) { this.topic = topic;**

**this.fee = fee;**

**this.nos = nos;**

**}**

**public String toString() {**

**String msg = "JLC Announces... New WorkShop";**

**msg = msg + "\n" + topic + "\t" + fee + "\t" + nos;**

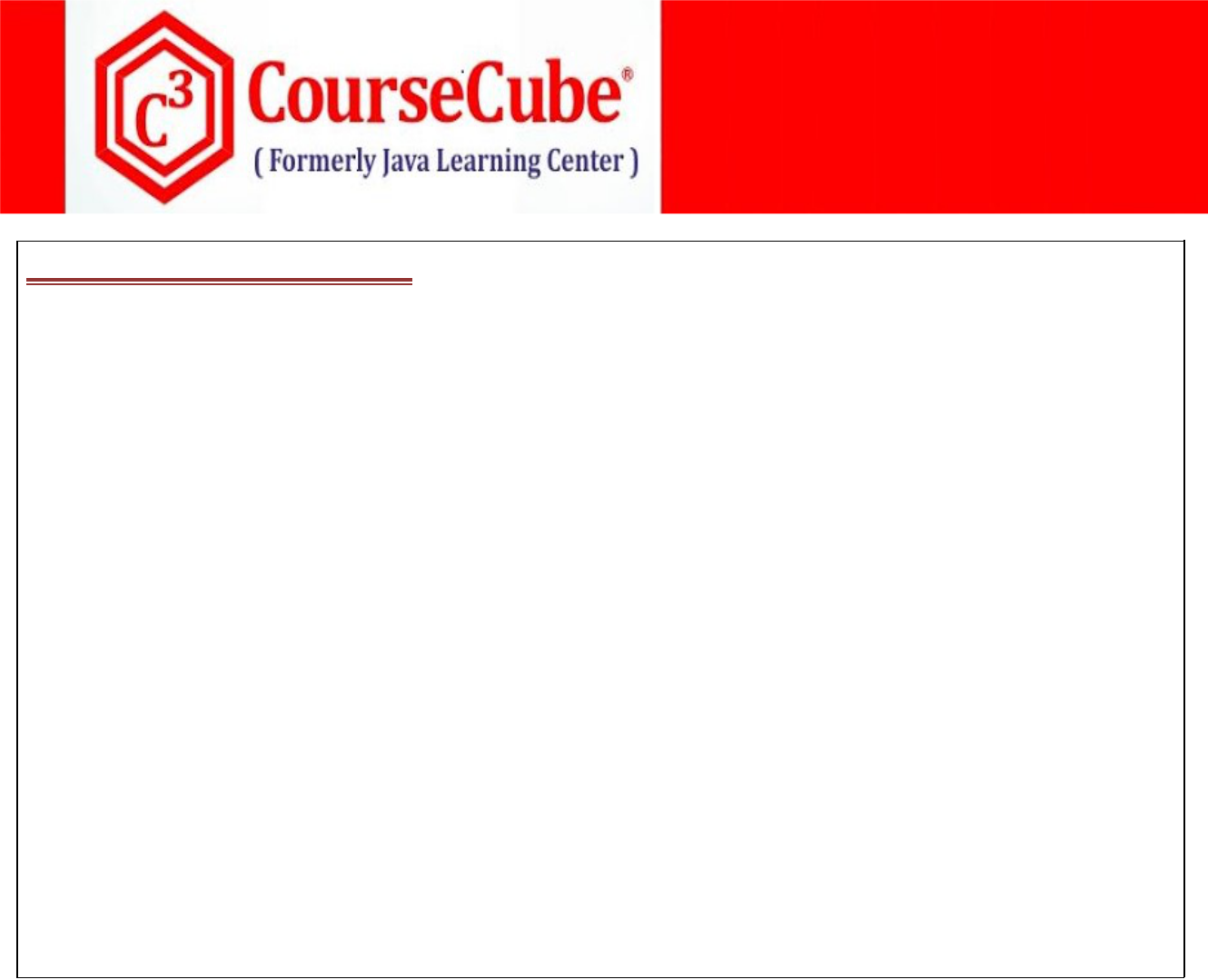
**return msg;**

**}**

**}**



**www.coursecube.com** **133** **Spring-5 Study Guide**



1. **WorkShopEvent.java**

**package com.coursecube.spring;**

**import org.springframework.context.ApplicationEvent; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class WorkShopEvent extends ApplicationEvent{**

**WorkShopTO workshopTO;**

**public WorkShopEvent(Object source,WorkShopTO workshopTO) { super(source);**

**this.workshopTO = workshopTO;**

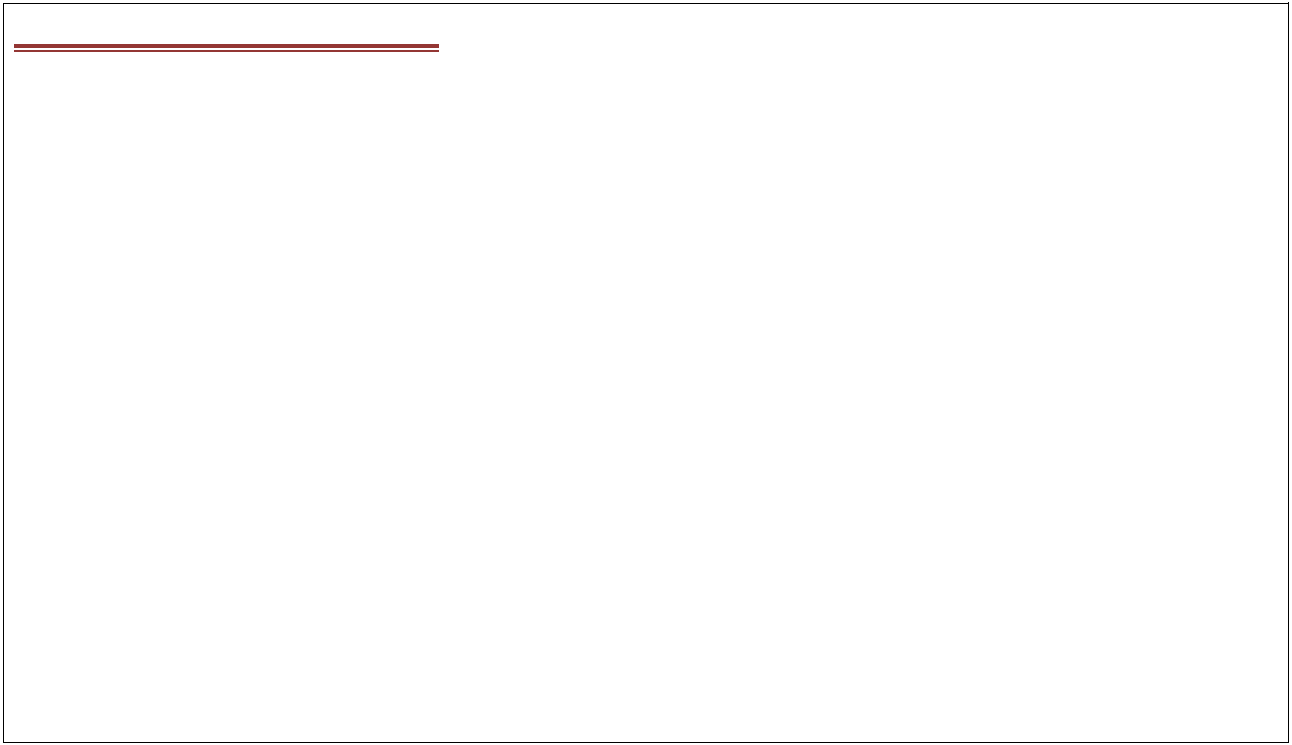
**}**

**public WorkShopTO getWorkshopTO() {**

**return workshopTO;**

**}**

**}**



**7.BatchListener.java**

**package com.coursecube.spring;**

**import org.springframework.context.ApplicationListener; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class BatchListener implements ApplicationListener<BatchEvent> {**

**@Override**

**public void onApplicationEvent(BatchEvent event) { System.out.println("BatchListener -> onApplicationEvent"); System.out.println("Batch event is raised"); System.out.println(event.getBatchTO());**

**}**

**}**



**www.coursecube.com** **134** **Spring-5 Study Guide**



1. **WorkShopListener.java package com.coursecube.spring;**

**import org.springframework.context.ApplicationListener; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

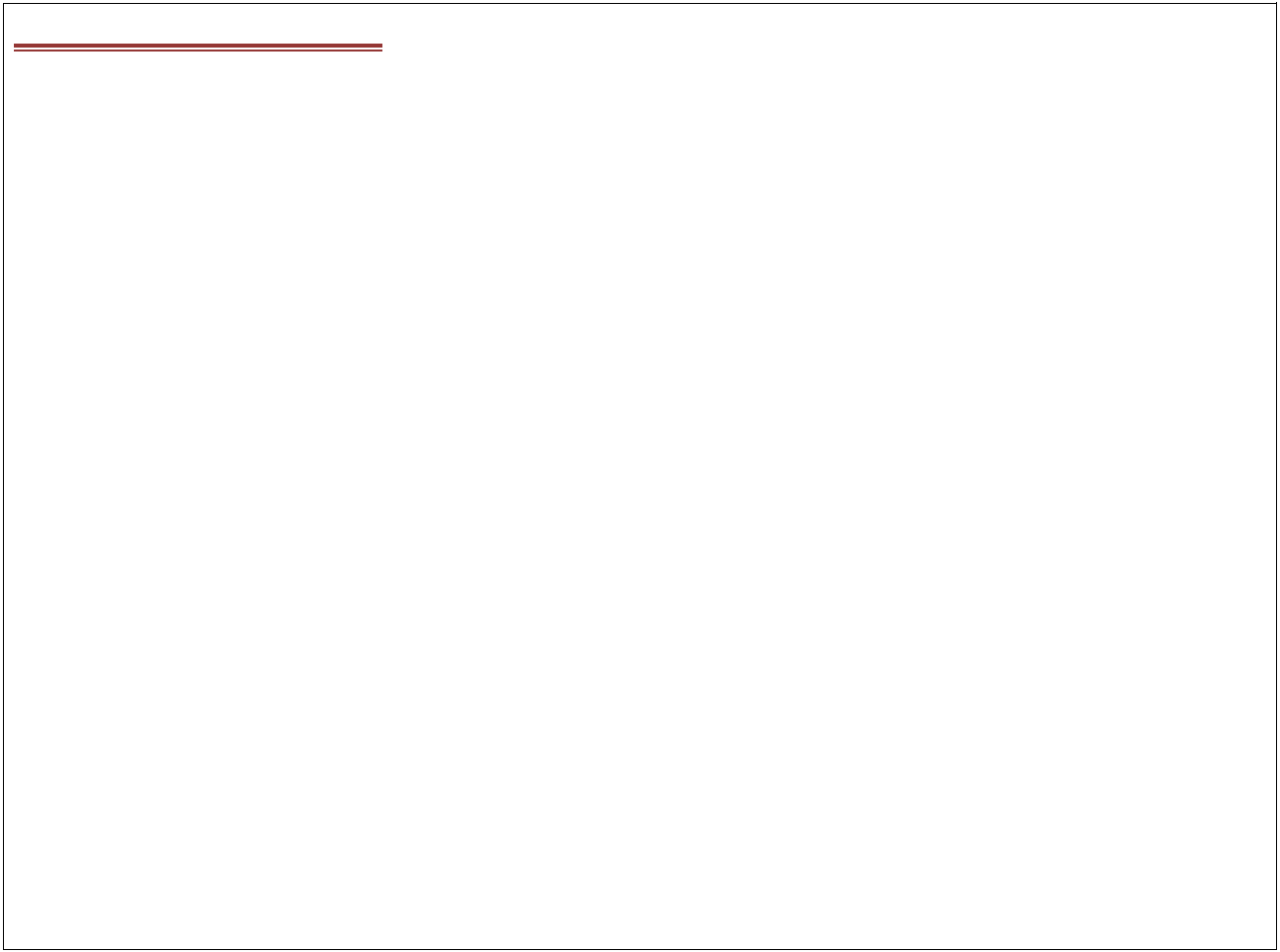
**public class WorkShopListener implements ApplicationListener<WorkShopEvent> {**

**@Override**

**public void onApplicationEvent(WorkShopEvent event) { System.out.println("WorkShopListener -> onApplicationEvent()"); System.out.println("WorkshopEvent is raised"); System.out.println(event.getWorkshopTO());**

**}**

**}**



1. **JLCAppConfig.java package com.coursecube.spring;**

**import org.springframework.context.annotation.Bean;**

**import org.springframework.context.annotation.Configuration; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**@Configuration**

**public class JLCAppConfig {**

**@Bean(name = "myjlc")**

**public JLCManagerService getManager() {**

**return new JLCManagerService();**

**}**

**@Bean**

**public BatchListener getBatchListener() {**

**return new BatchListener();**

**}**

**@Bean**

**public WorkShopListener getWorkShopListener() { return new WorkShopListener();**

**}**

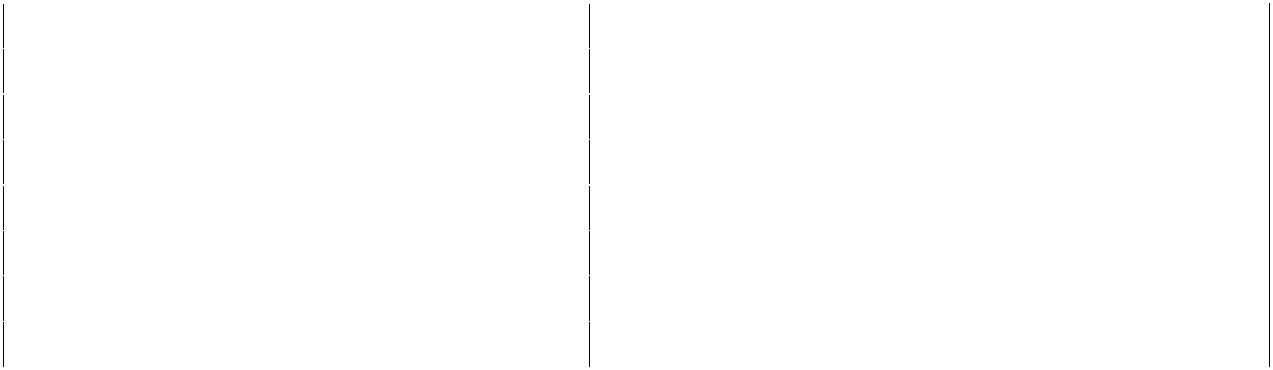
**}**



**www.coursecube.com** **135** **Spring-5 Study Guide**



**Lab48: Files required**



|  |  |  |
| --- | --- | --- |
| **1.** | **Lab48.java** | **Same as Lab47** |
|  |  |  |
| **2.** | **JLCManagerService.java** | **Updated in Lab48** |
|  |  |  |
| **3.** | **BatchTO.java** | **Same as Lab47** |
|  |  |  |
| **4.** | **BatchEvent.java** | **Updated in Lab48** |
|  |  |  |
| **5.** | **WorkShopTO.java** | **Same as Lab47** |
|  |  |  |
| **6.** | **WorkShopEvent.java** | **Updated in Lab48** |
|  |  |  |
| **7.** | **JLCListener.java** | **Newly Added in Lab48** |
|  |  |  |
| **8.** | **JLCAppConfig.java** | **Updated in Lab48** |
|  |  |  |



1. **JLCManagerService.java package com.coursecube.spring;**

**import org.springframework.beans.factory.annotation.Autowired; import org.springframework.context.ApplicationContext; import org.springframework.context.ApplicationEventPublisher; import org.springframework.stereotype.Component; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**@Component(value = "myjlc")**

**public class JLCManagerService {**

**@Autowired**

**private ApplicationEventPublisher publisher;**

**@Autowired**

**private ApplicationContext ctx;**

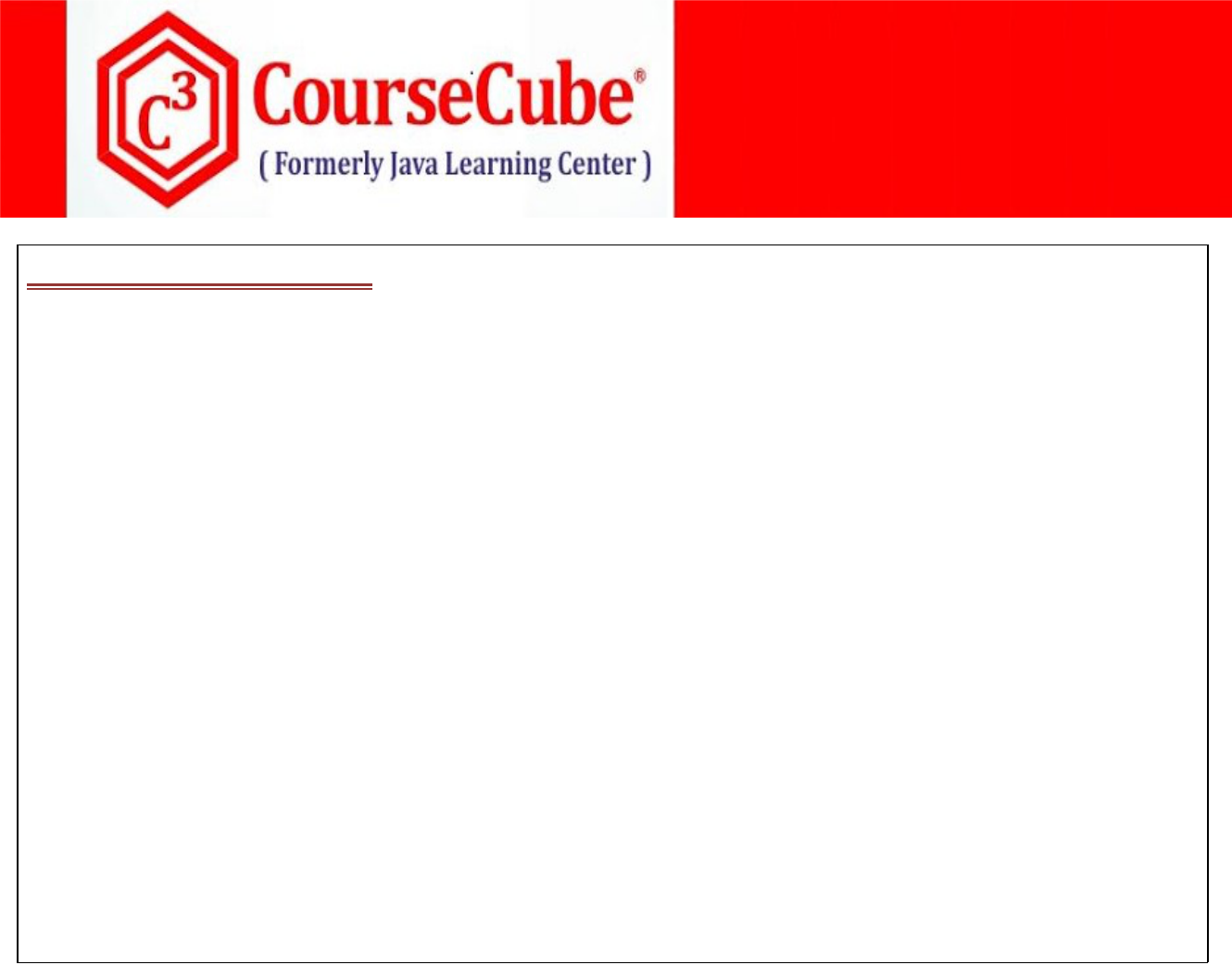
**public void addBatch(BatchTO batchTO) { publisher.publishEvent(new BatchEvent(batchTO)); }**

**public void addWorkshop(WorkshopTO workshopTO) { ctx.publishEvent(new WorkshopEvent(workshopTO)); }**

**}**



**www.coursecube.com** **136** **Spring-5 Study Guide**



1. **BatchEvent.java package com.coursecube.spring;**

**/\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class BatchEvent {**

**BatchTO batchTO;**

**public BatchEvent(BatchTO batchTO) {**

**this.batchTO = batchTO;**

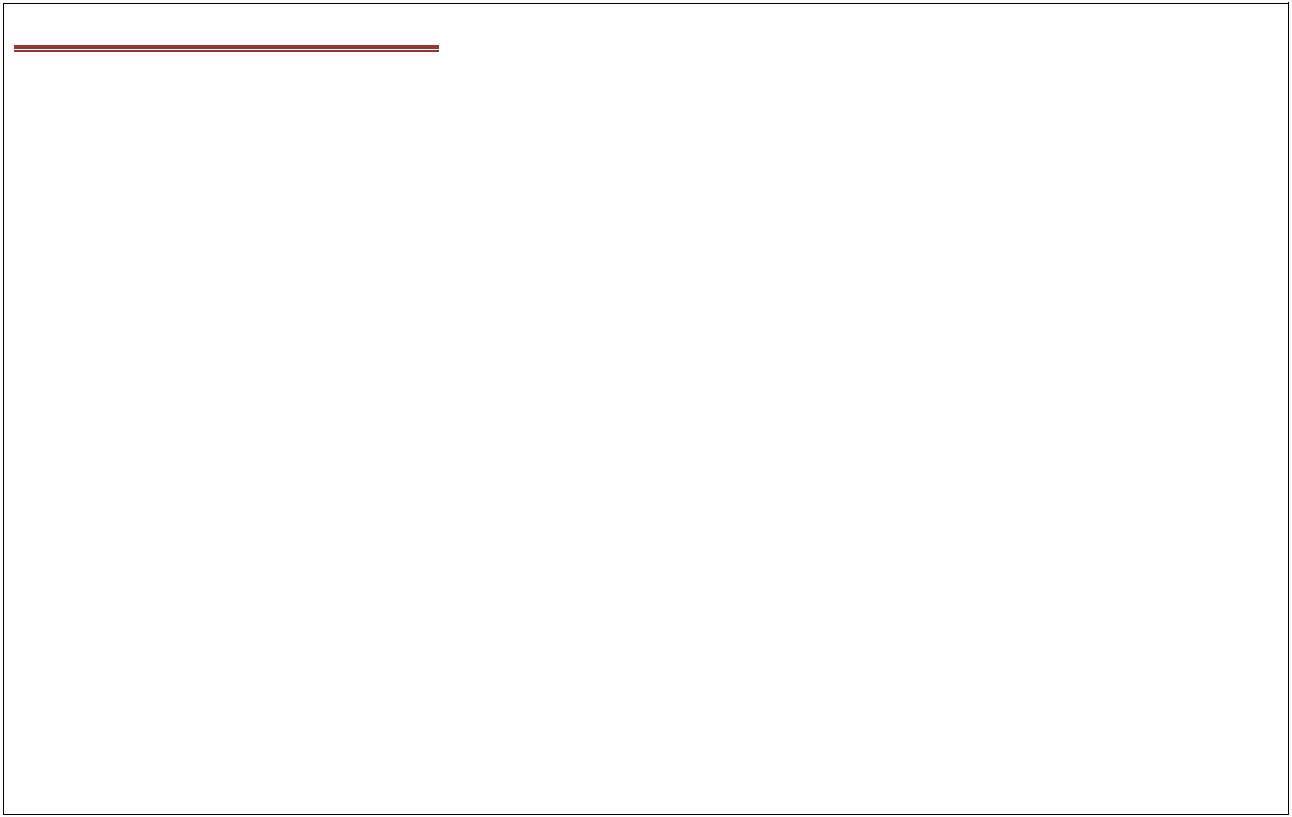
**}**

**public BatchTO getBatchTO() {**

**return batchTO;**

**}**

**}**



1. **WorkShopEvent.java**

**package com.coursecube.spring;**

**/\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**public class WorkshopEvent {**

**WorkshopTO workshopTO;**

**public WorkshopEvent(WorkshopTO workshopTO) { this.workshopTO = workshopTO;**

**}**

**public WorkshopTO getWorkshopTO() {**

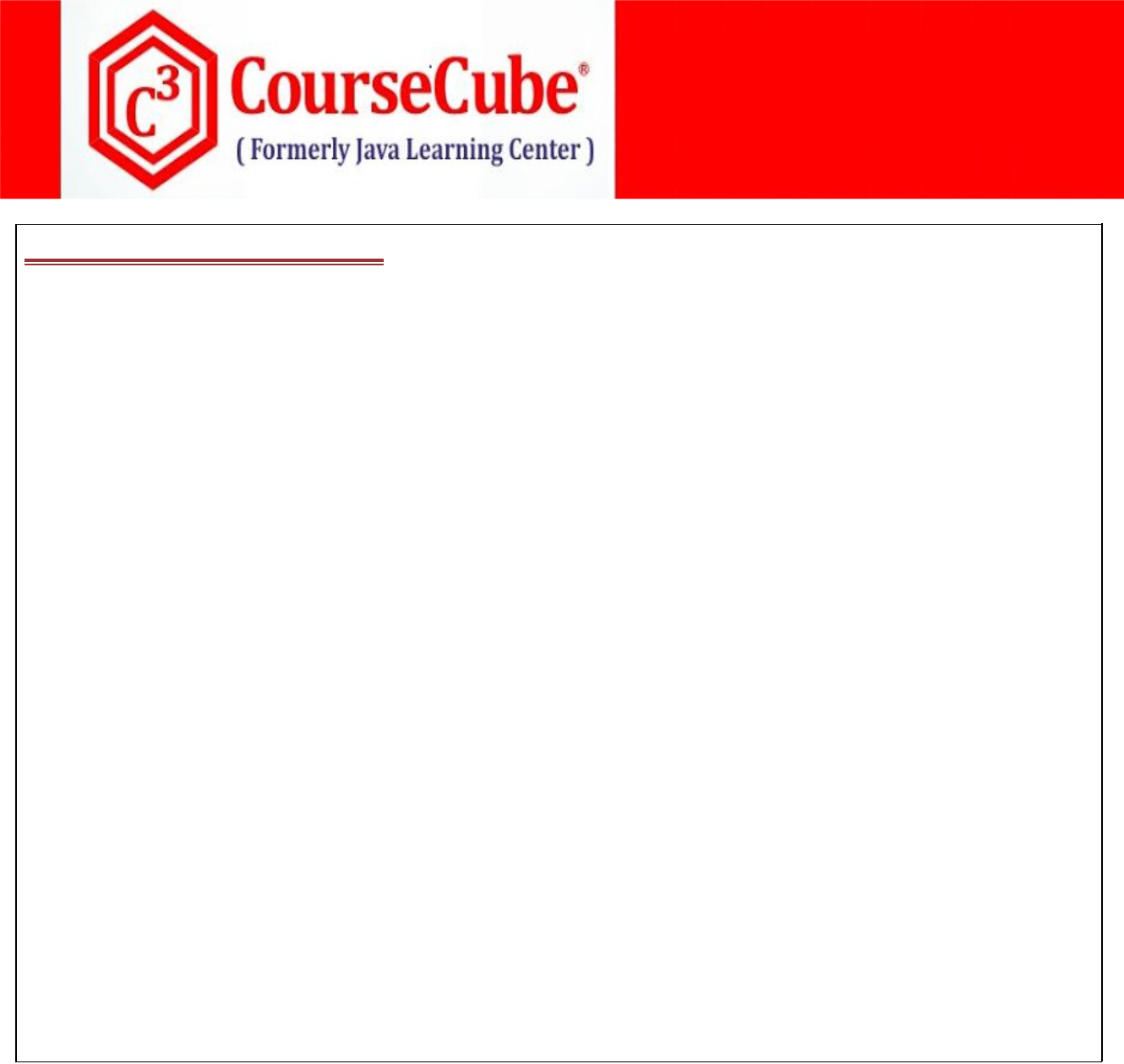
**return workshopTO;**

**}**

**}**



**www.coursecube.com** **137** **Spring-5 Study Guide**



**7.JLCListener.java**

**package com.coursecube.spring;**

**import org.springframework.context.event.EventListener; import org.springframework.stereotype.Component; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**@Component**

**public class JLCListener {**

**@EventListener**

**public void handleBatchEvent(BatchEvent event) { System.out.println("JLCListener -> handleBatchEvent"); System.out.println("Batch event is raised"); System.out.println(event.getBatchTO());**

**}**

**@EventListener**

**public void handleWorkshopEvent(WorkshopEvent event) { System.out.println("JLCListener -> handleWorkshopEvent()"); System.out.println("WorkshopEvent is raised"); System.out.println(event.getWorkshopTO());**

**}**

**}**



1. **JLCAppConfig.java package com.coursecube.spring;**

**import org.springframework.context.annotation.ComponentScan; import org.springframework.context.annotation.Configuration; /\***

* **@Author : Srinivas Dande**
* **@Company : CourseCube**
* **@Website : www.coursecube.com**
* **\*/**

**@Configuration**

**@ComponentScan(basePackages = { "com.coursecube.spring" })**

**public class JLCAppConfig {**

**}**



**www.coursecube.com** **138** **Spring-5 Study Guide**