* **[Home](http://www.codejava.net/)**
* **[Java Core](http://www.codejava.net/java-core)**
* **[Java SE](http://www.codejava.net/java-se)**
* **[Java EE](http://www.codejava.net/java-ee)**
* **[Frameworks](http://www.codejava.net/frameworks)**
* **[IDEs](http://www.codejava.net/ides)**
* **[Servers](http://www.codejava.net/servers)**
* **[Coding](http://www.codejava.net/coding)**
* **[Books](http://www.codejava.net/books)**
* **[Videos](http://www.codejava.net/videos)**
* **[Java Skills Test](https://tests4geeks.com/test/java" \t "_blank)**

* [Home](http://www.codejava.net/)
* [Frameworks](http://www.codejava.net/frameworks)
* Spring

Top of Form

Search... 

Bottom of Form

[](http://www.codejava.net/component/banners/click/246)

Top of Form

**Free Java**

**Tutorial Videos**

* Name:



* Email:



* 
* [We respect your privacy](http://www.getresponse.com/permission-seal?lang=en" \t "_blank)

Bottom of Form

**[Sending e-mail with Spring MVC](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc)**

Last Updated on 20 January 2016   |   [Print](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc?tmpl=component&print=1&page=" \o "Print) [Email](http://www.codejava.net/component/mailto/?tmpl=component&template=protostar&link=8e42962654b875009047e275e967036688bddee7" \o "Email)

|  |  |
| --- | --- |
| [**Java Hosting from just $5/mo. Get it now!**](https://www.superjavahost.com/prices) | [**Download Aspose for all your file format manipulation needs**](http://www.aspose.com/java/total-component.aspx?utm_source=CodeJava&utm_medium=banner&utm_campaign=CodeJava-Textforums) |

**Table of contents:**

[1.Spring framework’s support for e-mail](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "SpringEmail)

[2.Required jar files](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "RequiredJars)

[3.Creating e-mail sending form](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "EmailForm)

[4.Configuring SMTP server settings and Spring MVC](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "Configuration)

[5.Creating Spring MVC controller class](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "SpringController)

[6.Creating result page and error page](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "ResultAndErrorPage)

[7.Run the application](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "RunApp)

[8.Download Eclipse project/WAR file](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "Download)

This tutorial provides a sample spring MVC application that allows user sending an e-mail message by filling a web form. The e-mail form looks like following screenshot:

In this tutorial, you are supposed to familiar with Java EE development as well as developing Spring MVC-based applications.

**1. Spring framework’s support for e-mail**

Based on [JavaMail](http://www.oracle.com/technetwork/java/javamail/index.html" \t "_blank), Spring framework provides high-level abstraction API which greatly simplifies e-mail sending process. Let’s take a brief look at this API in the following class diagram:

To send e-mail messages, we can use an implementation of interface MailSender – theJavaMailSenderImpl class which is built upon on JavaMail. It’s convenient to configure this implementation as a bean in Spring’s context:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | <bean id="mailSender" class="org.springframework.mail.javamail.JavaMailSenderImpl">      <!-- SMTP settings -->      <property name="host" value="SMTP\_HOST" />      <property name="port" value="SMTP\_PORT" />      <property name="username" value="USER\_NAME" />      <property name="password" value="PASSWORD" />        <property name="javaMailProperties">          <!-- additional properties specific to JavaMail -->          <props>              <prop key="mail.transport.protocol">smtp</prop>              <prop key="mail.smtp.auth">true</prop>              <prop key="mail.smtp.starttls.enable">true</prop>          </props>      </property>  </bean> |

This bean holds properties for SMTP and JavaMail and can be injected to a business/service class which needs to send an e-mail, for example:

|  |  |
| --- | --- |
| 1 | mailSender.send(email); |

In which email is an object of a type that implements MailMessage interface, such as SimpleMailMessageclass. We can construct the email object as follows:

|  |  |
| --- | --- |
| 1  2  3  4 | SimpleMailMessage email = new SimpleMailMessage();  email.setTo(toAddress);  email.setSubject(subject);  email.setText(body); |

That’s for a simple mail message (plain text). In case if we want to send HTML e-mail or attach files to the e-mail, we can use MimeMailMessage class with the help of MimeMessagePreparator class andMimeMessageHelper class. For example, sending an e-mail in HTML format with an attachment:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | mailSender.send(new MimeMessagePreparator() {    public void prepare(MimeMessage mimeMessage) throws MessagingException {      MimeMessageHelper message = new MimeMessageHelper(mimeMessage, true, "UTF-8");      message.setFrom(fromEmail);      message.setTo(toEmail);      message.setSubject("A file for you");      message.setText("<b>See the attached</b>", true);      message.addAttachment("CoolStuff.doc", new File("CoolStuff.doc"));    }  }); |

The following table summarizes the interfaces and classes provided in org.springframework.mail package:

|  |  |
| --- | --- |
| **org.springframework.mail** | |
| [*MailSender*](http://static.springsource.org/spring/docs/3.0.0.RC1/javadoc-api/org/springframework/mail/MailSender.html) | [*MailMessage*](http://static.springsource.org/spring/docs/3.0.0.RC1/javadoc-api/org/springframework/mail/MailMessage.html) |
| [*JavaMailSender*](http://static.springsource.org/spring/docs/3.0.0.RC1/javadoc-api/org/springframework/mail/javamail/JavaMailSender.html) | [SimpleMailMessage](http://static.springsource.org/spring/docs/3.0.0.RC1/javadoc-api/org/springframework/mail/SimpleMailMessage.html) |
| [JavaMailSenderImpl](http://static.springsource.org/spring/docs/3.0.0.RC1/javadoc-api/org/springframework/mail/javamail/JavaMailSenderImpl.html) | [MimeMailMessage](http://static.springsource.org/spring/docs/3.0.0.RC1/javadoc-api/org/springframework/mail/javamail/MimeMailMessage.html) |
| [MimeMessagePreparator](http://static.springsource.org/spring/docs/3.0.0.RC1/javadoc-api/org/springframework/mail/javamail/MimeMessagePreparator.html) | [MimeMessageHelper](http://static.springsource.org/spring/docs/3.0.0.RC1/javadoc-api/org/springframework/mail/javamail/MimeMessageHelper.html) |

Click on a link in the table to see API documentation for the corresponding interface/class.

**2. Required jar files**

The application requires the following jar files copied to its WEB-INF\lib directory:

|  |  |
| --- | --- |
| **Required jar files** | |
| [JavaMail](http://www.oracle.com/technetwork/java/javamail/index-138643.html) | mail.jar |
| [Spring framework](http://www.springsource.org/download/community) | spring-beans-3.2.0.RELEASE.jar  spring-context-3.2.0.RELEASE.jar  spring-context-support-3.2.0.RELEASE.jar  spring-core-3.2.0.RELEASE.jar  spring-expression-3.2.0.RELEASE.jar  spring-web-3.2.0.RELEASE.jar  spring-webmvc-3.2.0.RELEASE.jar |
| [Apache Commons Logging](http://commons.apache.org/logging/download_logging.cgi) | commons-logging-1.1.1.jar |

**NOTE:**Click on a hyperlink in the table above to download the corresponding software.

The sample application we are going to build contains the following key files:

* + EmailForm.jsp: displays an e-mail form.
  + Result.jsp: shows successful message after the e-mail has been sent.
  + Error.jsp: shows error message in case of an exception is thrown.
  + SendEmailController.java: the Spring controller class that takes input from e-mail form, calls Spring’smailSender to send the e-mail, and redirects user to either successful page or error page.
  + spring-mvc.xml: Spring’s context configuration file. Here we will configure SMTP server settings and various properties for JavaMail.
  + web.xml: web deployment descriptor file.

**3. Creating e-mail sending form**

Create a JSP file called EmailForm.jsp with the following HTML code:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36 | <%@ page language="java" contentType="text/html; charset=UTF-8"      pageEncoding="UTF-8"%>  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"       "http://www.w3.org/TR/html4/loose.dtd">  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <title>Email with Spring MVC</title>  </head>  <body>      <center>          <h1>Sending e-mail with Spring MVC</h1>          <form method="post" action="sendEmail.do">              <table border="0" width="80%">                  <tr>                      <td>To:</td>                      <td><input type="text" name="recipient" size="65" /></td>                  </tr>                  <tr>                      <td>Subject:</td>                      <td><input type="text" name="subject" size="65" /></td>                  </tr>                  <tr>                      <td>Message:</td>                      <td><textarea cols="50" rows="10" name="message"></textarea></td>                  </tr>                  <tr>                      <td colspan="2" align="center">                          <input type="submit" value="Send E-mail" />                      </td>                  </tr>              </table>          </form>      </center>  </body>  </html> |

This is a simple form with three fields: *To*, *Subject* and *Message* – which are necessary attributes for a simple outgoing e-mail message. On submitting this form, the action named “*sendEmail.do*” will be called, as specified by the form’s action attribute. We will implement a Spring controller class for handling this action in the next section.

This book: **[Spring in Action](http://www.amazon.com/gp/product/1935182358/ref=as_li_tf_tl?ie=UTF8&camp=1789&creative=9325&creativeASIN=1935182358&linkCode=as2&tag=code0ac-20" \t "_blank)** helps you learn the latest features, tools, and practices including Spring MVC, REST, Security, Web Flow, and more.

**4. Configuring SMTP server settings and Spring MVC**

Create a Spring context configuration file called spring-mvc.xml with the following XML code:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40 | <?xml version="1.0" encoding="UTF-8"?>  <beans xmlns="http://www.springframework.org/schema/beans"      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"      xmlns:context="http://www.springframework.org/schema/context"      xsi:schemaLocation="http://www.springframework.org/schema/beans      http://www.springframework.org/schema/beans/spring-beans-3.0.xsd      http://www.springframework.org/schema/context      http://www.springframework.org/schema/context/spring-context-3.0.xsd">        <context:component-scan base-package="net.codejava.spring" />        <bean id="mailSender" class="org.springframework.mail.javamail.JavaMailSenderImpl">          <property name="host" value="smtp.gmail.com" />          <property name="port" value="587" />          <property name="username" value="youremail" />          <property name="password" value="yourpassword" />          <property name="javaMailProperties">              <props>                  <prop key="mail.transport.protocol">smtp</prop>                  <prop key="mail.smtp.auth">true</prop>                  <prop key="mail.smtp.starttls.enable">true</prop>              </props>          </property>      </bean>        <bean id="viewResolver"          class="org.springframework.web.servlet.view.InternalResourceViewResolver">          <property name="prefix" value="/" />          <property name="suffix" value=".jsp" />      </bean>        <bean          class="org.springframework.web.servlet.handler.SimpleMappingExceptionResolver">          <property name="exceptionMappings">              <props>                  <prop key="java.lang.Exception">Error</prop>              </props>          </property>      </bean>  </beans> |

This configuration is pretty straightforward:

* + <context:component-scan ... />: tells Spring to scan the package net.codejava.spring for initializing components which are annotated by Spring annotations (It’s a Spring controller class, in case of this application).
  + Bean mailSender: this is the important part because it declares a Spring bean for e-mail implementation – the JavaMailSenderImpl class and configures SMTP server settings, which is for a Gmail account in this case. This bean will be injected to a Spring controller class which will be covered in the next section.
  + Bean viewResolver: maps logical view names to real JSP file names.
  + Bean SimpleMappingExceptionResolver: maps all exceptions of type java.lang.Exception to be handled by Error.jsp page.

The web deployment descriptor file (web.xml) is configured as follows:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | <?xml version="1.0" encoding="UTF-8"?>  <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"      xmlns="http://java.sun.com/xml/ns/javaee"      xmlns:web="http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"      xsi:schemaLocation="http://java.sun.com/xml/ns/javaee          http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"      id="WebApp\_ID" version="3.0">    <display-name>EmailSpringMVC</display-name>        <servlet>          <servlet-name>SpringController</servlet-name>          <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>          <init-param>              <param-name>contextConfigLocation</param-name>              <param-value>/WEB-INF/spring-mvc.xml</param-value>          </init-param>          <load-on-startup>1</load-on-startup>      </servlet>        <servlet-mapping>          <servlet-name>SpringController</servlet-name>          <url-pattern>\*.do</url-pattern>      </servlet-mapping>      <welcome-file-list>      <welcome-file>EmailForm.jsp</welcome-file>    </welcome-file-list>  </web-app> |

It declares Spring controller servlet with its context configuration file (/WEB-INF/spring-mvc.xml). The controller is configured to handle all requests whose URL end with pattern: *\*.do.*And the default page when accessing the application is the email form (EmailForm.jsp).

This video tutorial: **[The Java Spring Tutorial](http://click.linksynergy.com/link?id=d1Fh24*EiQI&offerid=323058.65830&type=2&murl=https%3A%2F%2Fwww.udemy.com%2Fjavaspring%2F&u1=CodeJavaArticle" \t "new)http://ad.linksynergy.com/fs-bin/show?id=d1Fh24*EiQI&bids=323058.65830&type=2&subid=0** help you discover how to master the Spring framework instantly.

**5. Creating Spring MVC controller class**

In order to handle submission from the e-mail form, we need to create a Spring controller class as follows:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41 | package net.codejava.spring;  import javax.servlet.http.HttpServletRequest;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.mail.SimpleMailMessage;  import org.springframework.mail.javamail.JavaMailSender;  import org.springframework.stereotype.Controller;  import org.springframework.web.bind.annotation.RequestMapping;  import org.springframework.web.bind.annotation.RequestMethod;    @Controller  @RequestMapping("/sendEmail.do")  public class SendEmailController {        @Autowired      private JavaMailSender mailSender;        @RequestMapping(method = RequestMethod.POST)      public String doSendEmail(HttpServletRequest request) {          // takes input from e-mail form          String recipientAddress = request.getParameter("recipient");          String subject = request.getParameter("subject");          String message = request.getParameter("message");            // prints debug info          System.out.println("To: " + recipientAddress);          System.out.println("Subject: " + subject);          System.out.println("Message: " + message);            // creates a simple e-mail object          SimpleMailMessage email = new SimpleMailMessage();          email.setTo(recipientAddress);          email.setSubject(subject);          email.setText(message);            // sends the e-mail          mailSender.send(email);            // forwards to the view named "Result"          return "Result";      }  } |

This controller class is quite simple. It is declared as a Spring MVC controller by the annotation @Controller, and is mapped to the e-mail form’s action by the @RequestMapping annotation. We inject the mailSenderbean declared in spring-mvc.xml file into this controller through the private field also named mailSender. The injection is done automatically by Spring as we use the @Autowired annotation.

The method doSendEmail()is responsible for capturing input from e-mail form, creating aSimpleMailMessage object and sending the e-mail by invoking the send() method on the mailSender bean. The e-mail is in plain text format. Finally, it returns a view named “*Result*” which causes Spring to use theviewResolver to find and load appropriate JSP file (Result.jsp).

**6. Creating result page and error page**

Code the Result.jsp file as follows:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | <%@ page language="java" contentType="text/html; charset=UTF-8"      pageEncoding="UTF-8"%>  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"      "http://www.w3.org/TR/html4/loose.dtd">  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <title>Send e-mail result</title>  </head>  <body>      <center>          <h2>Thank you, your email has been sent.</h2>      </center>  </body>  </html> |

And code the Error.jsp as follows:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | <%@ page language="java" contentType="text/html; charset=UTF-8"      pageEncoding="UTF-8"%>  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"      "http://www.w3.org/TR/html4/loose.dtd">  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <title>Error</title>  </head>  <body>      <center>          <h2>Sorry, the email was not sent because of the following error:</h2>          <h3>${exception.message}</h3>      </center>  </body>  </html> |

As we can see, the result page simply tells the user that the e-mail has been sent, while the error page displays an error message if any exception thrown during the process of sending e-mail.

**7. Run the application**

So far we have created all the key pieces of the application. Let’s deploy it on a servlet container like Tomcat, and access the application by typing the following URL into browser’s address bar (your host name and port number maybe different, depending on server configuration):

*http://localhost:8080/EmailSpringMVC*

The e-mail form is displayed, type in required information:

Hit **Send E-mail** button, it may take a while for the e-mail to be sent. A successful message comes from the result page in case everything is going well:

In case of error (such as network failure or the SMTP server could not be reached), the error page displays:

You can download the sample application as an Eclipse project or deployable WAR file in the attachment section, and remember to update SMTP settings to match your e-mail account.

You may be also interested in:

* [Spring MVC: Send e-mail with attachments](http://www.codejava.net/frameworks/spring/spring-mvc-send-e-mail-with-attachments" \t "_blank)
* [Spring MVC beginner tutorial with Spring Tool Suite IDE](http://www.codejava.net/frameworks/spring/spring-mvc-beginner-tutorial-with-spring-tool-suite-ide" \t "_blank)
* [Sending e-mail with JSP, Servlet and JavaMail](http://www.codejava.net/java-ee/jsp/sending-e-mail-with-jsp-servlet-and-javamail" \t "_blank)
* [Send e-mail with attachments in Struts2](http://www.codejava.net/frameworks/struts/send-e-mail-with-attachments-in-struts2" \t "_blank)
* [How to start e-mail programming in Java](http://www.codejava.net/java-ee/javamail/how-to-start-e-mail-programming-in-java" \t "_blank)

**Share this article:**

|  |  |  |
| --- | --- | --- |
| **Attachments:** | | |
| [**EmailSpringMVC.war**](http://www.codejava.net/download-attachment?fid=71) | **[Deployable WAR file]** | **3907 kB** |
| [**EmailSpringMVC.zip**](http://www.codejava.net/download-attachment?fid=70) | **[Eclipse project]** | **3908 kB** |

**Add comment**

Top of Form

 



500 symbols left

 Notify me of follow-up comments

|  |  |  |
| --- | --- | --- |
|  | Get a new challengeGet an audio challengeHelp | http://www.google.com/recaptcha/api/img/clean/logo.png |
|  | [Privacy & Terms](http://www.google.com/intl/en/policies/) | |

Send

Bottom of Form

Comments

**1**2345

**0**[#48](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "comment-3687) **pradeep** 2016-04-14 06:45

hi,  
your code works fine.....

[Quote](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc)

**+1**[#47](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "comment-3578) **avadhoot** 2016-03-16 00:33

No mapping found for HTTP request with URI [/EmailSpringMVC/sendEmail.do] in DispatcherServlet with name 'SpringController'

[Quote](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc)

**0**[#46](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "comment-3558) **saidaReddy** 2016-03-08 23:25

this code is not working

[Quote](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc)

**0**[#45](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "comment-3557) **saidaReddy** 2016-03-08 21:24

i am trying to run the same code,but it is not run it showing error in HttpServletRequest.

[Quote](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc)

**0**[#44](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "comment-2801) **bhanupriya** 2015-10-20 02:28

sir i am getting this when i type my emailid in place of to   
Authentication failed; nested exception is javax.mail.AuthenticationFailedException: 535-5.7.8 Username and Password not accepted. Learn more at 535 5.7.8 [support.google.com/mail/answer/14257](https://support.google.com/mail/answer/14257" \t "_blank)nj1sm1937610pbc.2 - gsmtp   
can u please help me

[Quote](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc)

**+1**[#43](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "comment-2736) **sk** 2015-10-07 07:00

hiiii

[Quote](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc)

**+1**[#42](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "comment-2712) **harish** 2015-10-02 08:16

hi ,  
this is harish code is fine but i am getting error like org.springframework.beans.factory.BeanDefinitionStoreException: Failed to read candidate component class: file [D:\spring workspace\.metadata\.plugins\org.eclipse.wst.serve r.core\tmp0\wtpwebapps\EmailSpringMVC\WEB-INF\clas ses\net\codejava\spring\SendEmailController.class] ; nested exception is java.lang.IncompatibleClassChangeError: org/springframework/core/type/classreading/AnnotationMetadataReadingVisitor  
can any one explain what is this error   
Thanks in advance

[Quote](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc)

**0**[#41](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "comment-2630) **narasimharaju** 2015-09-14 00:19

hi raju,  
  
congregates you got job in MNC company

[Quote](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc)

**0**[#40](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "comment-2548) **sathesh** 2015-08-25 00:32

good effert

[Quote](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc)

**0**[#39](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \l "comment-2300) **Saras Kumar** 2015-07-01 03:48

good effort.

[Quote](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc)

**1**2345

[Refresh comments list](http://www.codejava.net/frameworks/spring/sending-e-mail-with-spring-mvc" \o "Refresh comments list)  
[RSS feed for comments to this post](http://www.codejava.net/component/jcomments/feed/com_content/207" \o "RSS feed for comments to this post" \t "_blank)

[JComments](http://www.joomlatune.com/" \o "JComments" \t "_blank)



**[JavaTipsEveryDay](http://javatipseveryday.com/" \t "_blank)**  [About](http://www.codejava.net/site/about)  [Advertise](http://www.codejava.net/advertise)  [Contribute](http://www.codejava.net/site/contribute)  [Contact](http://www.codejava.net/contactus)  [Terms of Use](http://www.codejava.net/site/terms)  [Privacy Policy](http://www.codejava.net/site/privacy-policy)  [Site Map](http://www.codejava.net/site/site-map)  [Newsletter](http://newsletter.codejava.net/)              
Copyright © 2012 - 2016 by www.codejava.net