# JAVAMAIL API - FETCHING EMAILS

In the previous chapter we learnt how to check emails. Now let us see how to fetch each email and read its content. Let us write a Java class **FetchingEmail** which will read following types of emails:

- Simple email
- Email with attachment
- Email with inline image

Basic steps followed in the code are as below:

- Get the Session object.
- Create POP3 store object and connect to the store.
- Create Folder object and open the appropriate folder in your mailbox.
- Retrieve messages.
- Close the folder and store objects respectively.

## Create Java Class

Create a java Create a java class file **FetchingEmail**, contents of which are as below:

```
package com.tutorialspoint;

import java.io.BufferedOutputStream;
import java.io.BufferedReader;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.Date;
import java.util.Properties;

import javax.mail.Address;
import javax.mail.Folder;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.Multipart;
import javax.mail.NoSuchProviderException;
import javax.mail.Part;
import javax.mail.Session;
import javax.mail.Store;

public class FetchingEmail {

   public static void fetch(String pop3Host, String storeType, String user,
      String password) {
      try {
         // create properties field
         Properties properties = new Properties();
         properties.put("mail.store.protocol", "pop3");
         properties.put("mail.pop3.host", pop3Host);
         properties.put("mail.pop3.port", "995");
         properties.put("mail.pop3.starttls.enable", "true");
         Session emailSession = Session.getDefaultInstance(properties);
         // emailSession.setDebug(true);

         // create the POP3 store object and connect with the pop server
```

```
        Store store = emailSession.getStore("pop3s");

        store.connect(pop3Host, user, password);

        // create the folder object and open it
        Folder emailFolder = store.getFolder("INBOX");
        emailFolder.open(Folder.READ_ONLY);

        BufferedReader reader = new BufferedReader(new InputStreamReader(
      System.in));

        // retrieve the messages from the folder in an array and print it
        Message[] messages = emailFolder.getMessages();
        System.out.println("messages.length---" + messages.length);

        for (int i = 0; i < messages.length; i++) {
           Message message = messages[i];
           System.out.println("---------------------------------");
           writePart(message);
           String line = reader.readLine();
           if ("YES".equals(line)) {
              message.writeTo(System.out);
           } else if ("QUIT".equals(line)) {
              break;
           }
        }

        // close the store and folder objects
        emailFolder.close(false);
        store.close();

     } catch (NoSuchProviderException e) {
        e.printStackTrace();
     } catch (MessagingException e) {
        e.printStackTrace();
     } catch (IOException e) {
        e.printStackTrace();
     } catch (Exception e) {
        e.printStackTrace();
     }
  }
  public static void main(String[] args) {

     String host = "pop.gmail.com";// change accordingly
     String mailStoreType = "pop3";
     String username =
        "abc@gmail.com";// change accordingly
     String password = "*****";// change accordingly

     //Call method fetch
     fetch(host, mailStoreType, username, password);

  }

  /*
  * This method checks for content-type
  * based on which, it processes and
  * fetches the content of the message
  */
  public static void writePart(Part p) throws Exception {
     if (p instanceof Message)
        //Call methos writeEnvelope
        writeEnvelope((Message) p);

     System.out.println("----------------------------");
     System.out.println("CONTENT-TYPE: " + p.getContentType());

     //check if the content is plain text
     if (p.isMimeType("text/plain")) {
        System.out.println("This is plain text");
        System.out.println("---------------------------");
        System.out.println((String) p.getContent());
     }
     //check if the content has attachment
     else if (p.isMimeType("multipart/*")) {
        System.out.println("This is a Multipart");
        System.out.println("---------------------------");
        Multipart mp = (Multipart) p.getContent();
        int count = mp.getCount();
        for (int i = 0; i < count; i++)
           writePart(mp.getBodyPart(i));
     }
     //check if the content is a nested message
     else if (p.isMimeType("message/rfc822")) {
        System.out.println("This is a Nested Message");
        System.out.println("---------------------------");
        writePart((Part) p.getContent());
     }
     //check if the content is an inline image
     else if (p.isMimeType("image/jpeg")) {
        System.out.println("--------> image/jpeg");
        Object o = p.getContent();

        InputStream x = (InputStream) o;
        // Construct the required byte array
        System.out.println("x.length = " + x.available());
        while ((i = (int) ((InputStream) x).available()) > 0) {
           int result = (int) (((InputStream) x).read(bArray));
           if (result == -1)
        int i = 0;
        byte[] bArray = new byte[x.available()];

           break;
        }
        FileOutputStream f2 = new FileOutputStream("/tmp/image.jpg");
        f2.write(bArray);
     }
     else if (p.getContentType().contains("image/")) {
        System.out.println("content type" + p.getContentType());
        File f = new File("image" + new Date().getTime() + ".jpg");
        DataOutputStream output = new DataOutputStream(
           new BufferedOutputStream(new FileOutputStream(f)));
           com.sun.mail.util.BASE64DecoderStream test =
                (com.sun.mail.util.BASE64DecoderStream) p
                 .getContent();
        byte[] buffer = new byte[1024];
        int bytesRead;
        while ((bytesRead = test.read(buffer)) != -1) {
           output.write(buffer, 0, bytesRead);
        }
     }
     else {
        Object o = p.getContent();
        if (o instanceof String) {
           System.out.println("This is a string");
           System.out.println("---------------------------");
           System.out.println((String) o);
        }
        else if (o instanceof InputStream) {
           System.out.println("This is just an input stream");
           System.out.println("---------------------------");
           InputStream is = (InputStream) o;
           is = (InputStream) o;
           int c;
           while ((c = is.read()) != -1)
              System.out.write(c);
        }
        else {
```

```
            System.out.println("This is an unknown type");
            System.out.println("---------------------------");
            System.out.println(o.toString());
         }
      }

   }
   /*
   * This method would print FROM,TO and SUBJECT of the message
   */
   public static void writeEnvelope(Message m) throws Exception {
      System.out.println("This is the message envelope");
      System.out.println("---------------------------");
      Address[] a;

      // FROM
      if ((a = m.getFrom()) != null) {
         for (int j = 0; j < a.length; j++)
         System.out.println("FROM: " + a[j].toString());
      }

      // TO
      if ((a = m.getRecipients(Message.RecipientType.TO)) != null) {
         for (int j = 0; j < a.length; j++)
         System.out.println("TO: " + a[j].toString());
      }

      // SUBJECT
      if (m.getSubject() != null)
         System.out.println("SUBJECT: " + m.getSubject());

   }

}
```

> *You can set the debug on by uncommenting the statement emailSession.setDebug*true
> *;*

## Compile and Run

Now that our class is ready, let us compile the above class. I've saved the class FetchingEmail.java to directory : **/home/manisha/JavaMailAPIExercise**. We would need the jars *javax.mail.jar* and *activation.jar* in the classpath. Execute the command below to compile the class *boththejarsareplacedin/home/manisha/directory* from command prompt:

```
javac -cp /home/manisha/activation.jar:/home/manisha/javax.mail.jar: FetchingEmail.java
```

Now that the class is compiled, execute the below command to run:

```
java -cp /home/manisha/activation.jar:/home/manisha/javax.mail.jar: FetchingEmail
```

## Verify Output

You should see the following message on the command console:

```
messages.length---3
------------------------------------
This is the message envelope
---------------------------
FROM: XYZ <xyz@gmail.com>
TO: ABC <abc@gmail.com>
SUBJECT: Simple Message
----------------------------
CONTENT-TYPE: multipart/alternative; boundary=047d7b343d6ad3e4ea04e8ec6579
```

```
This is a Multipart
---------------------------

----------------------------
CONTENT-TYPE: text/plain; charset=ISO-8859-1
This is plain text
---------------------------
Hi am a simple message string....

--
Regards
xyz

This is the message envelope
---------------------------
FROM: XYZ <xyz@gmail.com>
TO: ABC <abc@gmail.com>
SUBJECT: Attachement
----------------------------
CONTENT-TYPE: multipart/mixed; boundary=047d7b343d6a99180904e8ec6751
This is a Multipart
---------------------------

----------------------------
CONTENT-TYPE: text/plain; charset=ISO-8859-1
This is plain text
---------------------------
Hi I've an attachment.Please check

--
Regards
XYZ

----------------------------
CONTENT-TYPE: application/octet-stream; name=sample_attachement
This is just an input stream
---------------------------
Submit your Tutorials, White Papers and Articles into our Tutorials Directory. This is a
tutorials database where we are keeping all the tutorials shared by the internet
community for the benefit of others.


This is the message envelope
---------------------------
FROM: XYZ <xyz@gmail.com>
TO: ABC <abc@gmail.com>
SUBJECT: Inline Image
----------------------------
CONTENT-TYPE: multipart/related; boundary=f46d04182582be803504e8ece94b
This is a Multipart
---------------------------

----------------------------
CONTENT-TYPE: text/plain; charset=ISO-8859-1
This is plain text
---------------------------
Hi I've an inline image


[image: Inline image 3]

--
Regards
XYZ

----------------------------
CONTENT-TYPE: image/png; name="javamail-mini-logo.png"
content typeimage/png; name="javamail-mini-logo.png"
```

Here you can see there are three emails in our mailbox. First a simple mail with message "Hi am a simple message string....". The second mail has an attachment. The contents of the attachment are also printed as seen above. The third mail has an inline image.

Processing math: 100%