Fetching data from server using Java ME

The basic purpose of this article is to demonstrate how to download a file from a remote server using an `HttpConnection`, and view its contents in a `Form`.

Important Points

- One benefit of using an `HttpConnection` (as compared to a `StreamConnection`) is the availability of a method to check for the length of the remote data. The data can then be read in one fell-swoop, in contrast to reading one character/byte at a time.
- Before leaving the `viewFile()` method, check the status of the input stream and the `HttpConnection`, and close if necessary.

Sample code for client

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import javax.microedition.io.*;
import java.io.*;

public class FileViewer extends MIDlet implements CommandListener {
 private Display display; // Reference to Display object
 private TextBox tbViewer; // View file contents in a textbox
 private Command cmView; // Command to view file
 private Command cmExit; // Command to exit
 private String url = "https://www.kernel.org/pub/software/scm/git/docs/v1.7.9.1/git-check-attr.txt";

 public FileViewer() {
  display = Display.getDisplay(this);

  // Define commands
  cmView = new Command("View", Command.SCREEN, 2);
  cmExit = new Command("Exit", Command.EXIT, 1);

  tbViewer = new TextBox("Viewer", "", 250, TextField.ANY);
  tbViewer.addCommand(cmView);
  tbViewer.addCommand(cmExit);
  tbViewer.setCommandListener(this);
 }

 public void startApp() {
  display.setCurrent(tbViewer);
 }

 private void viewFile() throws IOException
 {
  int iResponseCode;
  HttpConnection http = null;
  InputStream iStrm = null;
  try {
   http = (HttpConnection) Connector.open(url);
   // ----------------
   // Client Request
   // ----------------
   // 1) Set request method
   http.setRequestMethod(HttpConnection.GET);
   // 2) Set header information (this header is optional)
   http.setRequestProperty("User-Agent",
     "Profile/MIDP-1.0 Configuration/CLDC-1.0");
   // 3) Set body/data - No data for this request
   // ----------------
   // Server Response
   // ----------------
   // 1) Get response status message and code
   iResponseCode = http.getResponseCode();
   System.out.println("Msg: " + http.getResponseMessage());
   System.out.println("Code: " + iResponseCode);
   // 2) Check response status
   if (iResponseCode == HttpConnection.HTTP_OK)
   {
    // 3) Get data and show the file contents
    iStrm = http.openInputStream();
    int length = (int) http.getLength();
    if (length > 0)
    {
     byte serverData[] = new byte[length];
     iStrm.read(serverData);
     tbViewer.setString(new String(serverData));
    }
   }
   else
    System.out.println("No data can be read if response was not HTTP_OK");
  }catch (Exception e){
   e.printStackTrace();
  }
  finally
  {
   // Close the stream
   if (iStrm != null)
    iStrm.close();
   // Close the connection
   if (http != null)
    http.close();
  }
 }

 public void pauseApp()
 {

 }

 public void destroyApp(boolean unconditional)
```


(C) Copyright Nokia 2013. All rights reserved.

```
 {

 }

 public void commandAction(Command c, Displayable s)
 {
  if (c == cmView)
  {
   try {
    viewFile();
   } catch (Exception e)
   {
    e.printStackTrace();
   }
  } else if (c == cmExit)
  {
   destroyApp(false);
   notifyDestroyed();
  }
 }
}
```

Tip:

- Note that the connection will be actually opened and the request sent the first time any method that requires data to be received from the server is called, in this case `getResponseCode()`.
- If you need to use any of the values you got from the `HttpConnection` object more than once, it is advisable to store it in a local variable instead of calling the method several times, as shown above with the status code.

Most of this code can be used to download and display other types of files. Most likely you would only need to change the object in which you are going to display the file contents, and the method(s) used to put the data in it. To show this point the above code has been modified to add the option to download an image file. Note that the text file is now displayed in a `TextField`, and the image is displayed in an `ImageItem` and both of these objects are now added to a `Form`.

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import javax.microedition.io.*;
import java.io.*;

public class FileViewer1 extends MIDlet implements CommandListener
{
 private Display display; // Reference to Display object
 private Form form;
 private TextField tfViewer; // View text file contents
 private ImageItem iiViewer; // View image file contents
 private Command cmViewText; // Command to view text file
 private Command cmViewImage; // Command to view image file
 private Command cmExit; // Command to exit
 private String urlText = "https://www.kernel.org/pub/software/scm/git/docs/v1.7.9.1/git-check-attr.txt";
 private String urlImage = "http://a5.twimg.com/profile_images/3348709803/236c5e8be219e6bf690b67cec6202e9b_reasonably_small.png";

 private static final int MODE_TEXT = 0;
 private static final int MODE_IMAGE = 1;

 public FileViewer1()
 {
  display = Display.getDisplay(this);
  cmViewText = new Command("Download Text", Command.SCREEN, 2);
  cmViewImage = new Command("Download Image", Command.SCREEN, 2);
  cmExit = new Command("Exit", Command.EXIT, 1);
  form = new Form("File viewer");
  tfViewer = new TextField("Text Viewer", "", 5000, TextField.ANY);
  iiViewer = new ImageItem("Image Viewer", null, ImageItem.LAYOUT_CENTER,"");
  form.append(tfViewer);
  form.append(iiViewer);
  form.addCommand(cmViewText);
  form.addCommand(cmViewImage);
  form.addCommand(cmExit);
  form.setCommandListener(this);
 }

 public void startApp()
 {
  display.setCurrent(form);
 }

 private void viewFile(int mode) throws IOException
 {
  int iResponseCode;
  HttpConnection http = null;
  InputStream iStrm = null;
  // Set the appropriate URL
  String url = (mode==MODE_TEXT?urlText:urlImage);
  try {
   http = (HttpConnection) Connector.open(url);
   // Client Request
   // ----------------
   // 1) Set request method
   http.setRequestMethod(HttpConnection.GET);
   // 2) Set header information (this header is optional)
   http.setRequestProperty("User-Agent","Profile/MIDP-1.0 Configuration/CLDC-1.0");
   // Server Response
   // ----------------
   // 1) Get response status message and code
   iResponseCode = http.getResponseCode();
   System.out.println("Msg: " + http.getResponseMessage());
   System.out.println("Code: " + iResponseCode);
   // 2) Check response status
   if (iResponseCode == HttpConnection.HTTP_OK)
   {
    // 3) Get data and show the file contents
    iStrm = http.openInputStream();
    int length = (int) http.getLength();
    if (length > 0)
    {
     byte serverData[] = new byte[length];
     int bytesRead = iStrm.read(serverData);
     if (bytesRead == length)
     {
      // Set the data on a TextField or ImageItem as appropriate
```


(C) Copyright Nokia 2013. All rights reserved.

```
      if (mode==MODE_TEXT)
       tfViewer.setString(new String(serverData));
      else if (mode==MODE_IMAGE)
      {
       Image image = null;
       try{
        image = Image.createImage(serverData,0,length);

       } catch (Exception e){
        e.printStackTrace();
       };
       if (image != null){
        iiViewer.setImage(image);
       }
      }
     }
    }
   }
   else
    System.out.println("No data can be read if response was not HTTP_OK");
  }catch (Exception e){
   e.printStackTrace();
  }
  finally
  {
   // Close the stream
   if (iStrm != null)
    iStrm.close();
   // Close the connection
   if (http != null)
    http.close();
  }
 }

 public void pauseApp(){}

 public void destroyApp(boolean unconditional){}

 public void commandAction(Command c, Displayable s)
 {
  if (c == cmViewText)
  {
   try {
    viewFile(MODE_TEXT);
   } catch (Exception e)
   {
    e.printStackTrace();
   }
  }
  else if (c == cmViewImage)
  {
   try {
    viewFile(MODE_IMAGE);
   } catch (Exception e)
   {
    e.printStackTrace();
   }
  }
  else if (c == cmExit)
  {
   destroyApp(false);
   notifyDestroyed();
  }
 }
}
```

Note that the downloaded image could also be displayed on a `Canvas` using the `drawImage()` method.

Here is an example:

```
protected void paint(Graphics g){
g.drawImage(image,0, 0, Graphics.LEFT|Graphics.TOP);
}
```

Source Code

The project with the code included in the article can be downloaded here: Media:HttpTest.zip

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.