This is the mimer-discussion.html file created by Athina Verroiopoulou to show step by step the construction of the mimer assignment:

The first part, was to accomplish the return of my own MIME header, from MyWebServer in order to trigger the client-side handler. To do that, I had to make a. xyz file in my directory and name it “mimer-data”. Then, I changed MyWebServer code in order to serve. xyz files. (add character.endsWith(".xyz") line and datatype = "application/xyz" line to identify the application/xyz datatype.) A file that ends with “xyz”, is a data type file that has a special format and contain atomic coordinates. Usually is divided in a line for read the total number of atoms, another line for potential comments and another that have the data (the atomic coordinates).

To continue, I had to download and install the “shim.bat” file in my directory. Then I connected my MIME type with shim.bat by following the provided instructions. First, I changed my Windows registry by opening the Registry Editor Window and look at the folder with the path of “HKEY\_CLASSES\_ROOT\MIME\Database\Content Type”. When i found it, I had to add the “application/xyz” key and the “Extension/.xyz” string for this new key. So i end up with the name “Extension” and Data value “.xyz”.

Now the MIME type is connected with the shim.bat. I should also associate the. xyz file with the handler (the shim.bat). In windows 10 I selected the .xyz file, then right click on it, open with, and then I searched for the shim.bat file. I checked the option for open always with that and tested if everything works. I used firefox by typing “<http://localhost:2540/>[mimer-data.xyz](http://localhost:2540/mimer-data.xyz) “ in the browser or simple by clicking the “mimer-call.html” link, while webserver was running. Then another cmd window will pop up in the browser that executes the shim file. The first pause is to observer the development and Handler directory and identify the name of the temporary file that data is stored. The second is to execute the handler and view the temporary data, but is not yet available. Meanwhile, in the WebServer’s output I can see the right MIME and the data that the. xyz file have inside.

The second step is to make a Handler that can read environmental variables and with the help of my shim.bat, call it from the server. To do that I added the line “java -Dfirstarg=%1 Handler” in the shim.bat in order to execute the handler.java. Then I made the handler.java as was instructed from the instructions. In the command line, I typed “shim mimer-data.xyz” and then I called the Webserver and clicked the “mimer-call.html” webpage. Now in the cmd window that pop up from browser, i could observe that Handler.class is called. Also, the data of mimer-data.xyz and the temporary file location that these data are stored are showed. After the handler runs and when we are in the second pause, the temporary file location is stored in an output file that is deleted after the pause. The server will still show the same information and the command window with “shim mimer-data.xyz” will execute the handler and show the “mimer-data.xyz” but is not able to load any file yet.

Until now I manage to request from the browser my. xyz file and make the Webserver able to identify the correct datatype and send the correct data and MIME header back to the client. The browser that will receive the header from the server will write the data into a temporary file and then will call a handler to pass the name of the created file as an argument. Then the handler will open the file and read and handle the data.

The third step is to create a java program that can serialize our data to XML. To do that, I created the BCClient.java file, as was instructed from the instructions This program will communicate with the server with the help of a back channel that maintains the server socket at port 2570 and send there the xml. With the use of Xstream library we can serialize and deserialize XML objects. To instantiate the class we put this line “ Xstream xstream = new Xstream()” in the code. In order to use it I had to download the xstream and xpp3 jar file and place them in the jdk/lib/ which is located at “C:\Program Files\Java\jdk1.8.0\_111\lib” on my computer. Then I downloaded the “jcxclient.bat” and changed the classpath to point into the path above. This will allow me to compile BCClient.java file and create its .class file. Then I downloaded the rxclient.bat and changed the classpath to point into the path above. I commented the server code, in order to see that the xml serialization is working. When I compile “BCClient.java” file I can enter a string that will be done in XML and stored into a temporary directory (disk). By changing the code I can see the xml output (the serialized data) and the deserialized data output in my terminal.

The fourth step is to manage a communication between the server and my handler throughout a backchannel. To accomplish that, I had to change MyWebServer program in order to handle an XML stream. I inserted the suggested code of BCWorker and BCLooper in my WebServer. Then I downloaded the “jcx.bat” and “rx.bat” file. I edit them in order to change the path, so that it points the right jdk and call MyWebServer.java. This will allow me to compile MyWebServer.java. file, create its .class file and allow server to find runtime libraries for Xstream. Uncomment the parts of BCClient that was in comments so that is connects with the server. To verify that the back-channel communication is working I had to compile at the same time MyWebServer.java and BCClient.java file. First, i run the “jcx.bat” file, then the “rx.bat” file for compiling the MyWebServer.java file. Then the “jcxclient.bat” file and rxclient.bat file for compiling the BCClient.java file. I pressed what I want to show in the XML and observe the output both in client and server. From the output, we can see that it reads the data, make them into an xml, send them to server and wait for server’s response of acknowledgement to terminate and print the receipt. On the server side, we can see the xml data that is on its back channel and then the same data deserialized into a symbolic form.

The last step is to combine the Handler (code snippet) and BCClient (a back-channel handler) into a new java file with the name BCHandler. This program is our client that take as input our. xyz file and return the right mime type and the data first in xml form and then in their deserialized form. After I created the program I changed the “jcxclient” file in order to create the BCHandler.class file. Then I had to change the shim.bat file to compile the file and see the output in the terminal. After that step was successful I started my WebServer and clicked on the html link from “mimer-call”. In the new browser window that is connected with the updated shim.bat file I can see the data as it is in the file, then in XML form and then their symbolic / deserialized form. In the server, I can see the MIME type and the data in the above sequence. The temporary location of the file is "C:\Users\Athina\AppData\Local\Temp\mimer-data.xyz" and we can see there the data.

At the end of this assignment we have a web server that reads the data of “mimer-data.xyz” file and send back the correct mime header and data. The firefox browser will write the data of “mimer-data.xyz” file to a temporary folder, the name of which will be passed as “argOne” with the help of call shim.bat. At the end this file will be removed. When shim.bat is running having the name of the file we can see in the first pause the existing status and then the execution of BCHandler. There the name of the file will be passed, the file will open, show the data in a symbolic form and then into an xml form. The xml data will be sent to web server with a back-channel communication and then we can see in server the xml data and their symbolic form.