Myserver.java

```
/* Author: Yang Meng (01679623)
 * DataCommunicationI-ProgrammingAssignment-1 Server
 * Last edit at 10/17/2016
 */
import java.io.*;
import java.net.*;

public class Myserver {
	public static void main(String args[]) {

		ServerSocket ss = null;
		try
		{
			// Create a socket named ss to listen on a specific port: args[0]
			try
			{
				ss = new ServerSocket(Integer.parseInt(args[0]));
			}
			//If server failed to start, display this information to user
			catch (IOException e)
			{
				System.err.println("Server could not listen on "+ args[0] +" port\n" );
				System.exit(1);
			}
			//If server starts successfully, print out the success message
			System.out.println("********Server has been successfully started********");

			// Server starts listening and waits until receive the message "exit"
			while(!args[0].equals("exit")) {
				Socket client = null;
				try
				{
					// Wait for a client to connect until get a socket
					client = ss.accept();
					System.out.println("********Connection with client has been set up********");
				}

				catch (IOException e)
				{
					// If server can not connected to the client, print out the message
					System.err.println("Connection with client failed\n");
					System.exit(1);
				}
```

```
// Communicate with the client through the socket
BufferedReader in = new BufferedReader(new InputStreamReader(client.getInputStream()));
PrintWriter out = new PrintWriter(new OutputStreamWriter(client.getOutputStream()));

//Receive the control command from client and send the server's response
String line;
String FileName= "";
String[] Request = null;
StringBuffer StringContent = new StringBuffer("");
try
{
	while((line = in.readLine()) != null)
	{
		System.out.println(line);
		if (Request == null)
		{
			Request= line.split(" ");
			continue;
		}
		if (line.length() == 0) break;
		StringContent.append(line + "\r\n");
	}
	FileName = Request[1];

	// Only accept GET and PUT, other requests will be denied
	if(!(Request[0].equals("GET")) && !(Request[0].equals("PUT")))
	{
		out.println("Please use only GET and PUT command\n");
	}
	out.flush();
}
//If server fail to read the client's Request, print out the message
catch(IOException ioe)
{
	System.out.println("Exception while reading the Request" + ioe);
}

//Execute GET method
if(Request[0].equals("GET"))
{
	System.out.println("********Start to GET********");

	// Read file from local server, and then send it to the client
	int ch;
	InputStream From_File = null;
	File file_get = new File("Container/" +FileName);
```

```
        try {

            From_File = new FileInputStream(file_get);

            while((ch = From_File.read()) != -1)
            {
                StringContent.append((char)ch);
            }

            From_File.close();
            System.out.println("File contents :"+StringContent);

            // Start sending server's reply, server is using HTTP 1.0 protocol
            // Send HTTP protocol Version & status code
            out.println("HTTP/1.0 200 OK");
            // /Send the type of data
            out.println("Content-Type: text/plain");
            out.println(StringContent + "\n");
            out.flush();

        }
        //If server could not find the file, throws the FileNotFoundException
        catch(FileNotFoundException e)
        {
            System.out.println("File " + file_get.getAbsolutePath() + " could not be found");
            out.println("404 Not Found");
            out.flush();
        }
        //Server throws IOException when server can not read the file
        catch(IOException ioe)
        {
            System.out.println("Exception occurs while reading the file" + ioe);
        }
        System.out.println("********File "+ FileName +" has been sent successfully********");
    }
    //GET command stops here


    //Execute PUT command
    if (Request[0].equals("PUT"))
    {
        System.out.println("********Start to PUT********");
        FileName = "Container/" + FileName;

        //Create a file called Creat_File
        OutputStream Creat_File = new FileOutputStream(FileName);
```

```
                    //Write to this file
                    PrintWriter Write_File = new PrintWriter(FileName);
                    Write_File.println(StringContent);

                    //Send the response to client
                    out.println("200 OK File successfully Created");
                    out.flush();

                    Creat_File.close();
                    Write_File.close();
                }
                //PUT command stops here

                // Close the streams and socket
                out.close();
                in.close();
                client.close();
            }
            // Loop stops here
            ss.close();
        }

        //Print out the error message if anything wrong happens
        catch (Exception e)
        {
            System.err.println(e);
            System.err.println("Usage: java HttpServer <port>: " + args[1]);
        }

    }
    //Main function ends here
}
```