Usage:

First run server with

python Server.py server_port

Then in a separate terminal run the client with

python ClientLauncher.py server_host server_port RTP_port video_file

Her are the commands I used when running it

python Server.py 4000

python ClientLauncher.py 127.0.0.1 4000 7000 movie.Mjpeg

What does this application do?

When clientlauncher is ran it creates a client object and passes the arguments to it. The client uses the RTSP protocol to request video data from the server and display it on the client side. When the client receives an RTSP reply from the server it creates a new thread that listens for a reply and terminates upon receival. RTSP is a protocol with states. There are four states the player can be in the first is SETUP and is the initial state of the player. Upon an OK from the server the setup establishes an Real Time transfer Protocol (RTP) port to the server and transitions to the READY state. Once the client is READY the PLAY request to go to the PLAYING state. When the server OKs the server sends a stream of video frames to the listening port on the client. The client does not have to request each frame the server will send the frames until the video is done or the client sends a PAUSE request entering the READY state. At any time while the client is already setup the client can send a TEARDOWN request which tells the server to terminate the connection between the server and the client.

What did I do in my project?

The first thing I did was writing the request message for each request code so that the message is in the same format as shown in the assignment sheet. Every request sent also had to increment the sequence number and save the request message so the send request function can actually send it.

The next thing I did was handle the state transitions. This just involved assigning the correct next state after successful requests according the FSA on the assignment sheet. The third thing I did was on the RtpPacket class correctly positioning the bits to create the header for the RTP packets. I used the provided example bitwise operations to fill in the header which is a byte array so that it matched the diagram. I explain the operations in the comments.

What did I change from the finished project?

After the project was finished I wanted to work on optional work number 2. I moved the call to the setup function to when the player is initialized so that the button could be removed. I also removed the teardown button as when the player is exited a teardown request is sent by the exit handler so the button is not needed. I wanted to add a stop button but met challenges and could not get the implementation working. The plan was to have the app send SETUP if the PLAY button was hit and the client wasn't already set up. I got it working so the play button could do both but when the STOP button is hit it is supposed to send a TEARDOWN to save resources but once the TEARDOWN was sent I was never able to get it to correctly SETUP again.