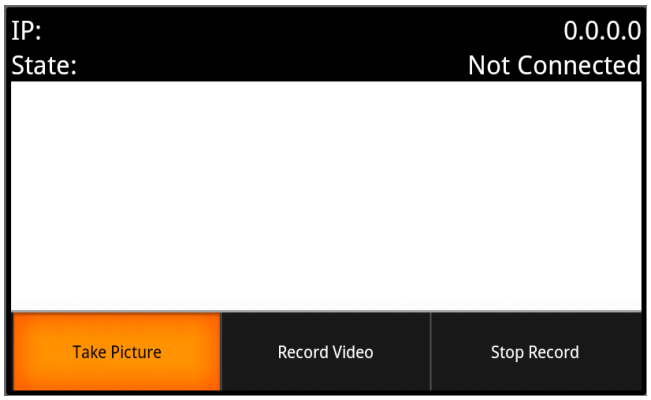
Lab 4	Remote camera – Learn to use camera module on the phone

Target

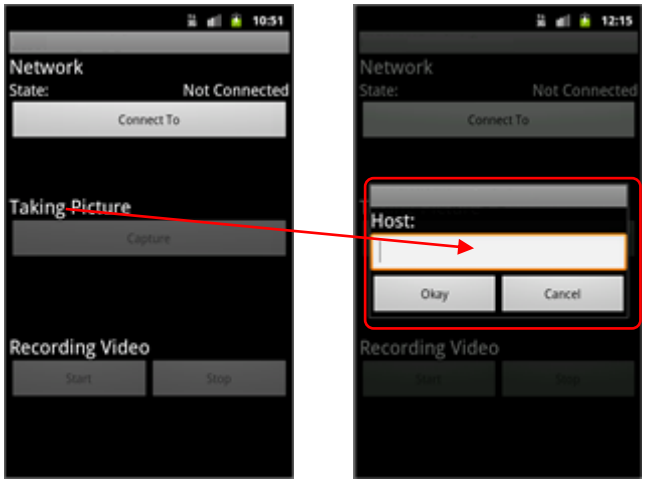
Camera

- Full screen
- Displaying IP
- Displaying Connection State
- Camera Preview
- TCP connection server
- Wake Lock
- A menu for camera action

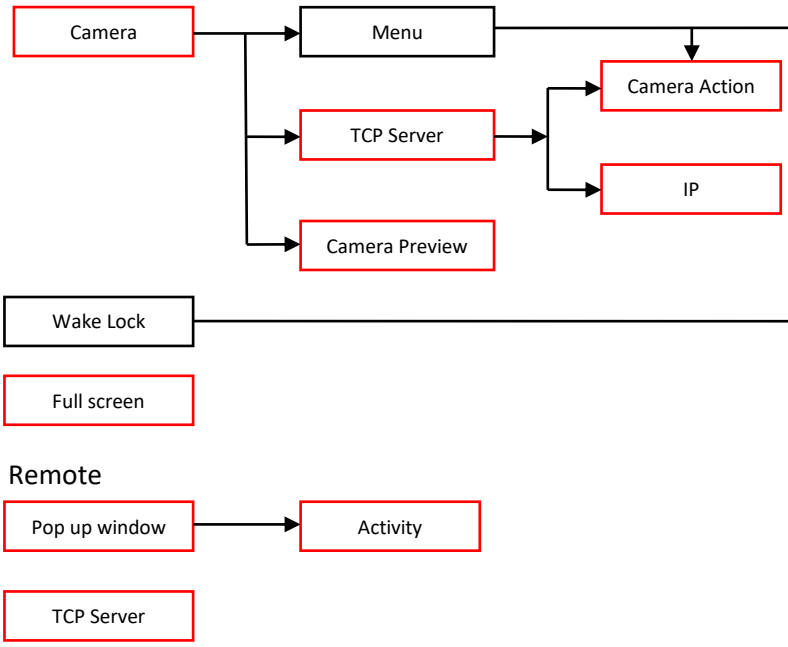


Remote

- Displaying connection state
- A button for connecting
- A dialog to input connection host
- Some buttons for controlling camera
- TCP connection client



Program design (To understand what you need and what you have)

You need to have	You learned in last few labs
Camera  Remote 	TextView (Lab 1) EditText (Lab 1) ImageView (Lab 1) Button (Lab 1) SurfaceView (Lab 2) Accelerometer (Lab 2) Menu (Lab 2) Wake Lock (Lab 2)

Program procedure (**This lab focus only camera API up to android 4.4. Do not build this lab with android 5.0 or above)**

Task 1 – Set up the camera and preview

Task 2 – Create the function for take picture and record video

Task 3 – Create TCP server

Task 4 – Create the remote apps

Task 1	Set up the camera and preview

Knowledge learn in this task:

None

Procedure of the task:

Step 1
Create new project

Project name: Lab_4_camera(or your own one)

Build Target: Android 4.4

Package name: ece.course.lab_4.camera

Step 2
Create the SurfaceView with SurfaceHolder.Callback

01) Create a class call "CameraPreviewView"
extends "SurfaceView" and **implements** "SurfaceHolder.Callback"

02) Add variables
- Name: mSupportedPreviewSizes Type: List<Size>
- Name: mPreviewSize Type: Size
- Name: mCamera Type: Camera

03) Add Constructor
Code:
```
public CameraPreviewView(Context context, AttributeSet attr) {
	super(context, attr);
	getHolder().addCallback(this);
	//getHolder().setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
}
```

04) Add function "onMeasure"
Code:
```
protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
	final int width  = resolveSize(getSuggestedMinimumWidth(), widthMeasureSpec);
	final int height = resolveSize(getSuggestedMinimumHeight(), heightMeasureSpec);
	setMeasuredDimension(width, height);
	if (mSupportedPreviewSizes != null)
		mPreviewSize = getOptimalPreviewSize(mSupportedPreviewSizes, width, height);
}
```

05) Add function "surfaceChanged"
Code:
```
public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {
	try {
		if (mCamera != null) mCamera.setPreviewDisplay(holder);
	} catch (Exception exception) { }
}
```

06) Add function “surfaceCreated”

Code:

```
public void surfaceCreated(SurfaceHolder holder) {
	Camera.Parameters parameters = mCamera.getParameters();
	parameters.setPreviewSize(mPreviewSize.width, mPreviewSize.height);
	mCamera.setParameters(parameters);
	mCamera.startPreview();
}
```

07) Add function “surfaceDestoryed”

Code:

```
public void surfaceDestroyed(SurfaceHolder holder) {
	if (mCamera != null) mCamera.stopPreview();
}
```

08) Add function “setCamera”

Code:

```
public void setCamera(Camera camera) {
	mCamera = camera;
	if (mCamera != null) {
		mSupportedPreviewSizes = mCamera.getParameters().getSupportedPreviewSizes();
		requestLayout();
	}
}
```

09) Add function “getOptimalPreviewSize”

Code:

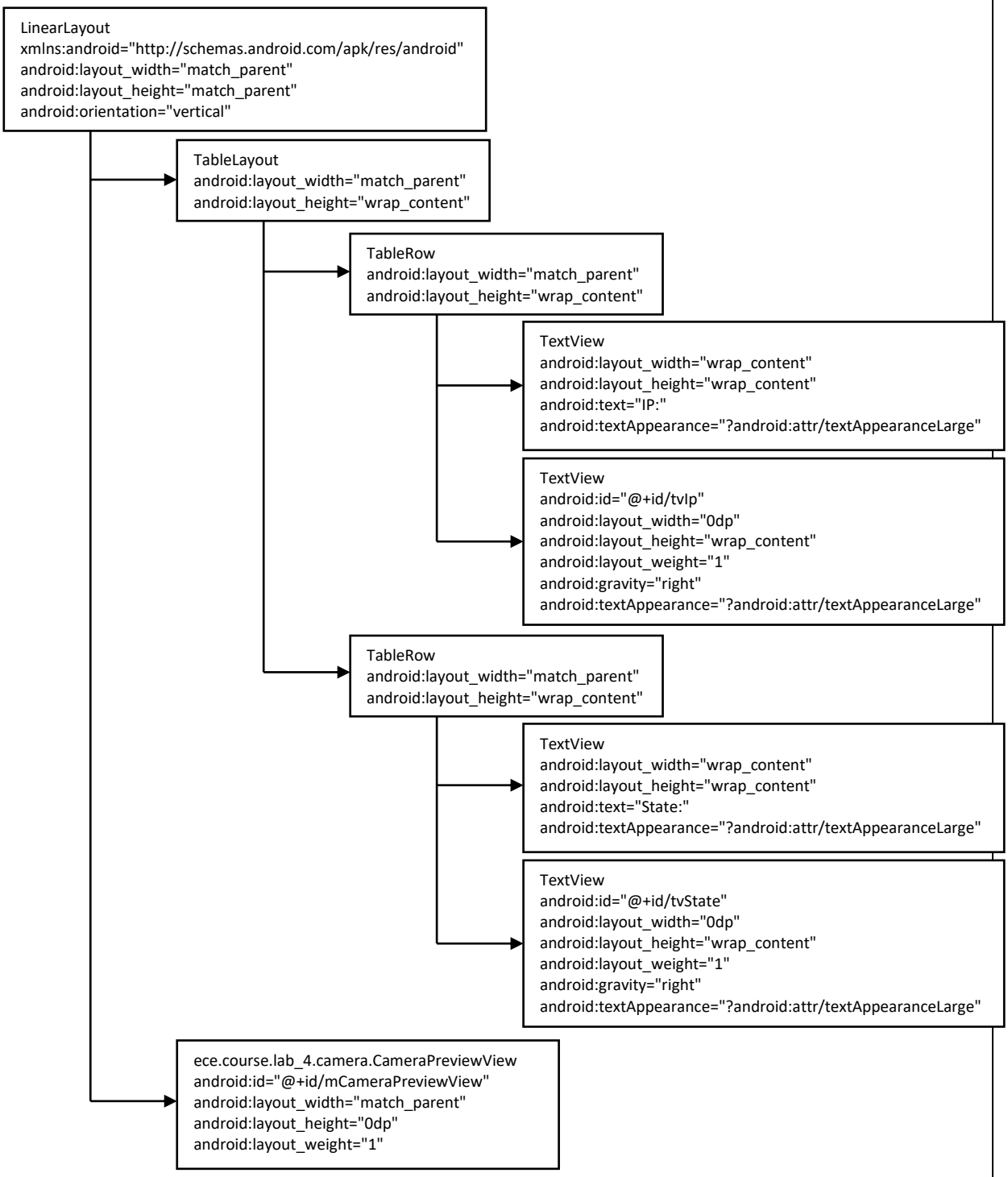
```
private Size getOptimalPreviewSize(List<Size> sizes, int w, int h) {
	if (sizes == null) return null;
	final double ASPECT_TOLERANCE = 0.1;
	double targetRatio = (double) w / h;
	double minDiff = Double.MAX_VALUE;
	int targetHeight = h;
	Size optimalSize = null;
	for (Size size : sizes) {
		double ratio = (double) size.width / size.height;
		if (Math.abs(ratio - targetRatio) > ASPECT_TOLERANCE)
			continue;

		if (Math.abs(size.height - targetHeight) < minDiff) {
			optimalSize = size;
			minDiff = Math.abs(size.height - targetHeight);
		}
	}
	if (optimalSize == null) {
		minDiff = Double.MAX_VALUE;
		for (Size size : sizes) {
			if (Math.abs(size.height - targetHeight) < minDiff) {
				optimalSize = size;
				minDiff = Math.abs(size.height - targetHeight);
			}
		}
	}
	return optimalSize;
}
```

Step 3

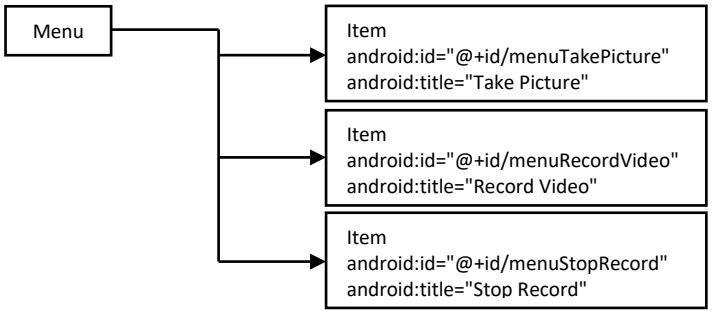
Set up the main layout

Set up the layout as following:



Step 4
Set up the menu

Set up the menu as following:



Step 5
Update the main activity

01) Add variables

- Name: mPowerManager — Type: PowerManager
- Name: mWakeLock — Type: WakeLock
- Name: mCameraPreviewView — Type: CameraPreviewView
- Name: mCamera — Type: Camera
- Name: mCameraId — Type: int

02) Update function "onCreate"

Code:

```
public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        if (android.os.Build.VERSION.SDK_INT > 9)
        {
            StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
            StrictMode.setThreadPolicy(policy);
        }
        getWindow().addFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN);
        getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
        setContentView(R.layout.main);
        if(!getPackageManager().hasSystemFeature(PackageManager.FEATURE_CAMERA)) {
                Toast.makeText(this, "No Camara On The Phone Leaving...",
                        Toast.LENGTH_SHORT).show();
                finish();
        }
        int numberOfCameras = Camera.getNumberOfCameras();
        boolean hvBackCamera = false;
        CameraInfo cameraInfo = new CameraInfo();
        for (int i = 0; i < numberOfCameras; i++) {
                Camera.getCameraInfo(i, cameraInfo);
                if (cameraInfo.facing == CameraInfo.CAMERA_FACING_BACK) {
                        mCameraId = i;
                        hvBackCamera = true;
                        break;
                }
        }
        mPowerManager = (PowerManager) getSystemService(POWER_SERVICE);
        mWakeLock = mPowerManager.newWakeLock(PowerManager.PARTIAL_WAKE_LOCK,
                getClass().getName());
        if (!hvBackCamera) {
                Toast.makeText(this, "The Apps only support the back Camara, Leaving...",
                        Toast.LENGTH_SHORT).show();
                finish();
```

```
        }
        mCameraPreviewView = (CameraPreviewView) findViewById(R.id.mCameraPreviewView);
}
```

03) Add function “onResume”

Code:

```
protected synchronized void onResume() {
        super.onResume();
        mWakeLock.acquire();
        mCamera = Camera.open(mCameraId);
        mCameraPreviewView.setCamera(mCamera);
}
```

04) Add function “onPause”

Code:

```
protected synchronized void onPause() {
        mWakeLock.release();
        if (mCamera != null) {
                mCameraPreviewView.setCamera(null);
                mCamera.stopPreview();
                mCamera.release();
                mCamera = null;
        }
        super.onPause();
}
```

05) Add function “onCreateOptionsMenu”

Code:

```
public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu, menu);
        return true;
}
```

06) Update function “onOptionsItemSelected”

Code:

```
public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
        case R.id.menuTakePicture :
                return true;
        case R.id.menuRecordVideo :
                return true;
        case R.id.menuStopRecord :
                return true;
        }
        return false;
}
```

Step 6

Ask for permissions (can refer to lab 2- Accelerometer, task 4, step 2)

Add following uses permissions

- android.permission.ACCESS_WIFI_STATE
- android.permission.CAMERA
- android.permission.INTERNET
- android.permission.RECORD_AUDIO
- android.permission.WAKE_LOCK
- android.permission.WRITE_EXTERNAL_STORAGE

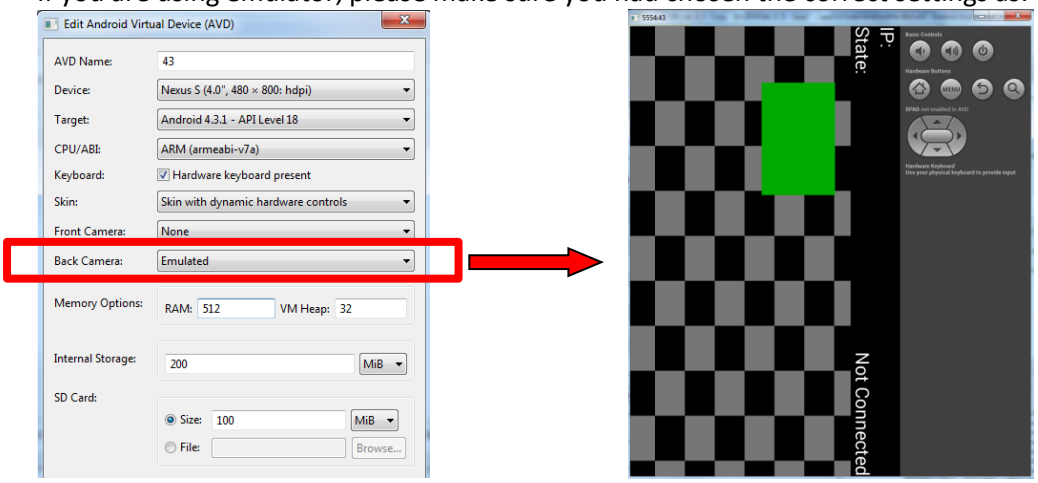
Open AndroidManifest.xml and add android:screenOrientation="landscape". Just like this

```
<activity
        android:name=".MainActivity"
        android:label="@string/app_name" android:screenOrientation="landscape">
```

Step 7

Test the apps.

If you are using emulator, please make sure you had chosen the correct settings as:



Task 2	Create the function for take picture and record video

Knowledge learn in this task:

None

Procedure of the task:

Step 1

Update the MainActivity

01) Add variables

- Name: mMediaRecorder Type: MediaRecorder
- Name: isRecording Type: Boolean Value: false
- Name: TAG_DEBUG Type: String Value: “MainActivity”

02) Update function “onOptionsItemSelected”

Code:

```
public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
        case R.id.menuTakePicture :
                takePicture();
                return true;
        case R.id.menuRecordVideo :
                recordVideo();
                return true;
        case R.id.menuStopRecord :
                stopRecord();
                return true;
        }
        return false;
```

```
}
```

03) Add variable mPictureCallBack (Type: “PictureCallBack”)

Code:

```
private PictureCallback mPictureCallBack = new PictureCallback() {
	public void onPictureTaken(byte[] data, Camera camera) {
		try {
			File pictureFile = getOutputFile(true);
			FileOutputStream fileOutputStream = new FileOutputStream(pictureFile);
			fileOutputStream.write(data);
			fileOutputStream.close();
		} catch (Exception exception) { }
		mCamera.startPreview();
	}
};
```

04) Add function “takePicture”

Code:

```
private void takePicture() {
	mCamera.takePicture(null, null, mPictureCallBack);
}
```

05) Add function “recordVideo”

Code:

```
private void recordVideo() {
	if (isRecording)
		return;
	if (prepareVideoRecorder()) {
		mMediaRecorder.start();
		isRecording = true;
	} else {
		releaseMediaRecorder();
	}
}
```

06) Add function “stopRecord”

Code:

```
private void stopRecord() {
	if (!isRecording)
		return;
	mMediaRecorder.stop();
	releaseMediaRecorder();
	mCamera.lock();
	mCamera.stopPreview();
	mCamera.startPreview();
	isRecording = false;
}
```

07) Add function “releaseMediaRecorder”

Code:

```
private void releaseMediaRecorder() {
	if (mMediaRecorder != null) {
		mMediaRecorder.reset();
		mMediaRecorder.release();
		mMediaRecorder = null;
		mCamera.lock();
	}
```

```
}
```

08) Add function “prepareVideoRecorder”

Code:

```
private boolean prepareVideoRecorder() {
        mMediaRecorder = new MediaRecorder();
        mCamera.unlock();
        mMediaRecorder.setCamera(mCamera);
        mMediaRecorder.setAudioSource(MediaRecorder.AudioSource.CAMCORDER);
        mMediaRecorder.setVideoSource(MediaRecorder.VideoSource.CAMERA);
        mMediaRecorder.setOutputFormat(MediaRecorder.OutputFormat.MPEG_4);
        mMediaRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.DEFAULT);
        mMediaRecorder.setVideoEncoder(MediaRecorder.VideoEncoder.DEFAULT);
        File mediaFile = getOutputFile(false);
        mMediaRecorder.setOutputFile(mediaFile.toString());
        mMediaRecorder.setPreviewDisplay(mCameraPreviewView.getHolder().getSurface());
        try {
                mMediaRecorder.prepare();
      return true;
        } catch (Exception exception) { }
        releaseMediaRecorder();
        return false;
}
```

For recording sound

Set up the output format

09) Add function “getOutputFile”

Code:

```
private File getOutputFile(boolean isPicture) {
        File storageDir = new File(
                Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURES),
                "USTECE_Lab_4_2");

        if (!storageDir.exists()) {
                if (!storageDir.mkdirs()) {
                        Log.d(TAG_DEBUG, "failed to create directory");
                        return null;
                }
        }
        String timeStamp = new SimpleDateFormat("yyyyMMdd_HHmmss").format(new Date());
        if (isPicture) {
                return new File(storageDir.getPath() + File.separator
                        + "IMG_"+ timeStamp + ".jpg");
        } else {
                return new File(storageDir.getPath() + File.separator
                        + "VID_"+ timeStamp + ".mp4");
        }
}
```

Step 2

Test the apps

Task 3	Create TCP server

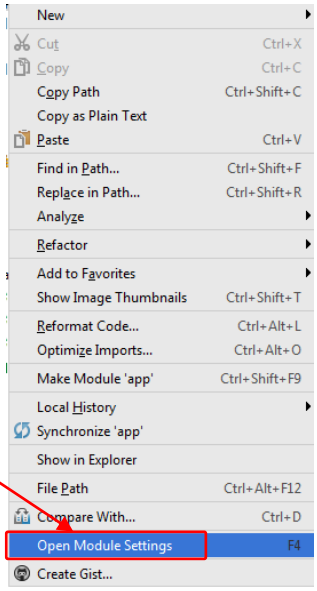
Knowledge learn in this task:

None
Procedure of the task:
Step 1 Add library

01) Download file: Lab_4_Lib.jar

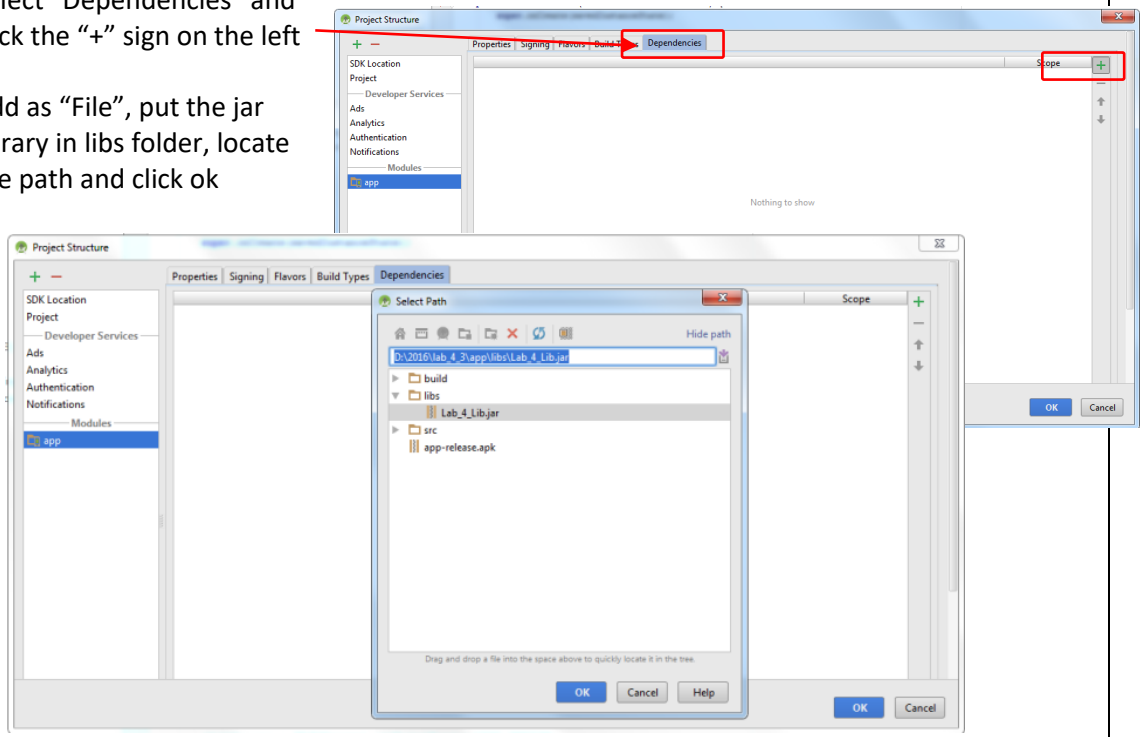
02) Add library to project

1) Right click the App module and select “Open Module Settings”



2) Select “Dependencies” and click the “+” sign on the left

3) Add as “File”, put the jar library in libs folder, locate the path and click ok



Step 2

Update the MainActivity

01) Add constants

- Name: APPS_PORT Type: int Value: 1234
- Name: MSG_PICTURE_CAPTURE Type: byte Value: (byte)**'P'**
- Name: MSG_VIDEO_RECORDING_START Type: byte Value: (byte)**'V'**
- Name: MSG_VIDEO_RECORDING_STOP Type: byte Value: (byte)**'S'**

02) Add variables

- Name: mWifiManager Type: WifiManager
- Name: mServer Type: NetworkConnectionServer
- Name: tvState Type: TextView
- Name: tvIp Type: TextView

03) Update function "onCreate"

Code:

```
public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        getWindow().addFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN);
    getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
        setContentView(R.layout.main);
        if(!getPackageManager().hasSystemFeature(PackageManager.FEATURE_CAMERA)) {
                Toast.makeText(this, "No Camara On The Phone Leaving...",
                        Toast.LENGTH_SHORT).show();
                finish();
        }
        int numberOfCameras = Camera.getNumberOfCameras();
        boolean hvBackCamera = false;
        CameraInfo cameraInfo = new CameraInfo();
        for (int i = 0; i < numberOfCameras; i++) {
                Camera.getCameraInfo(i, cameraInfo);
                if (cameraInfo.facing == CameraInfo.CAMERA_FACING_BACK) {
                        mCameraId = i;
                        hvBackCamera = true;
                        break;
                }
        }
        if (!hvBackCamera) {
                Toast.makeText(this, "The Apps only support the back Camara, Leaving...",
                        Toast.LENGTH_SHORT).show();
                finish();
        }
        mPowerManager = (PowerManager) getSystemService(POWER_SERVICE);
        mWakeLock = mPowerManager.newWakeLock(PowerManager.PARTIAL_WAKE_LOCK,
                getClass().getName());
        mWifiManager = (WifiManager) getApplicationContext().getSystemService(WIFI_SERVICE);
        if (mWifiManager == null) {
                Toast.makeText(this, "There is no wifi Module in the phone, leave...",
                        Toast.LENGTH_LONG).show();
                finish();
        }
        int ipInteger = mWifiManager.getConnectionInfo().getIpAddress();
        mCameraPreviewView = (CameraPreviewView) findViewById(R.id.mCameraPreviewView);
        tvState = (TextView) findViewById(R.id.tvState);
        tvState.setText("Not Connected");
        tvIp = (TextView) findViewById(R.id.tvIp);
        tvIp.setText(getIpString(ipInteger));
}
```

04) Update function “onResume”

Code:

```
protected synchronized void onResume() {
	super.onResume();
	mWakeLock.acquire();
	mCamera = Camera.open(mCameraId);
	mCameraPreviewView.setCamera(mCamera);
	mServer = new NetworkConnectionServer(APPS_PORT, new Handler() {
		public void handleMessage(Message msg) {
			switch(msg.what) {
			case NetworkConnectionServer.MSG_HV_DATA :
				byte[] data = (byte[]) msg.obj;
				int length  = msg.arg1;
				for (int i = 0; i < length; i++) {
					byte[] msgData = new byte[1];
					switch(data[i]) {
					case MSG_PICTURE_CAPTURE :
						takePicture();
						Toast.makeText(MainActivity.this,
							"Picture Captured!!",
							Toast.LENGTH_SHORT).show();
						msgData[0] = MSG_PICTURE_CAPTURE;
						mServer.sendData(msgData);
						break;
					case MSG_VIDEO_RECORDING_START :
						recordVideo();
						Toast.makeText(MainActivity.this,
							"Video Recording Started!!",
							Toast.LENGTH_SHORT).show();
						msgData[0] = MSG_VIDEO_RECORDING_START;
						mServer.sendData(msgData);
						break;
					case MSG_VIDEO_RECORDING_STOP :
						stopRecord();
						Toast.makeText(MainActivity.this,
							"Video Recording Stopped!!",
							Toast.LENGTH_SHORT).show();
						msgData[0] = MSG_VIDEO_RECORDING_STOP;
						mServer.sendData(msgData);
						break;
					}
				}
				break;
			case NetworkConnectionServer.MSG_CONNECTED :
				tvState.setText("Connected");
				break;
			case NetworkConnectionServer.MSG_CONNECTION_LOST :
				tvState.setText("Connection Lost");
				break;
			}
		}
	});
	mServer.start();
}
```

05) Update function “onPause”

Code:

```
protected synchronized void onPause() {
	mWakeLock.release();
	if (mCamera != null) {
		mCameraPreviewView.setCamera(null);
		mCamera.stopPreview();
		mCamera.release();
		mCamera = null;
```

```
            }
            if(mServer != null){
                mServer.interrupt();
            }
            tvState.setText("Not Connected");
            super.onPause();
}
```

06) Add function “getIpString”

IP = 0xZZYYXXWW (Integer form)
= WW.XX.YY.ZZ (normal form)

Code:

```
private String getIpString(int ipInteger) {
            return (ipInteger & 0xFF) + "." + ((ipInteger >> 8) & 0xFF) + "."
                        + ((ipInteger >> 16) & 0xFF) + "." + ((ipInteger >> 24) & 0xFF);
}
```

Step 2

Test the apps

Task 4	Create the remote apps

Knowledge learn in this task:

None

Procedure of the task:

Step 1

Import the remote apps project

Project name: USTECE_Lab4_4

Step 2

Add the library

Add the library used in Task 3

Step 3

Build the apps and test it in emulator, then run it in android phone as remote control

Step 4

Demo to TA/IA