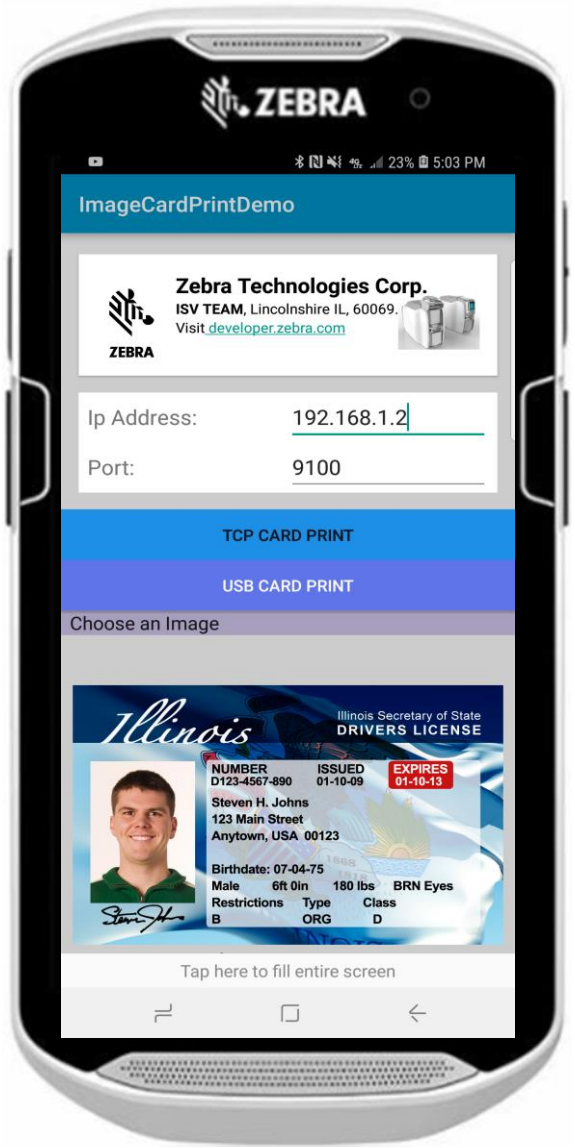
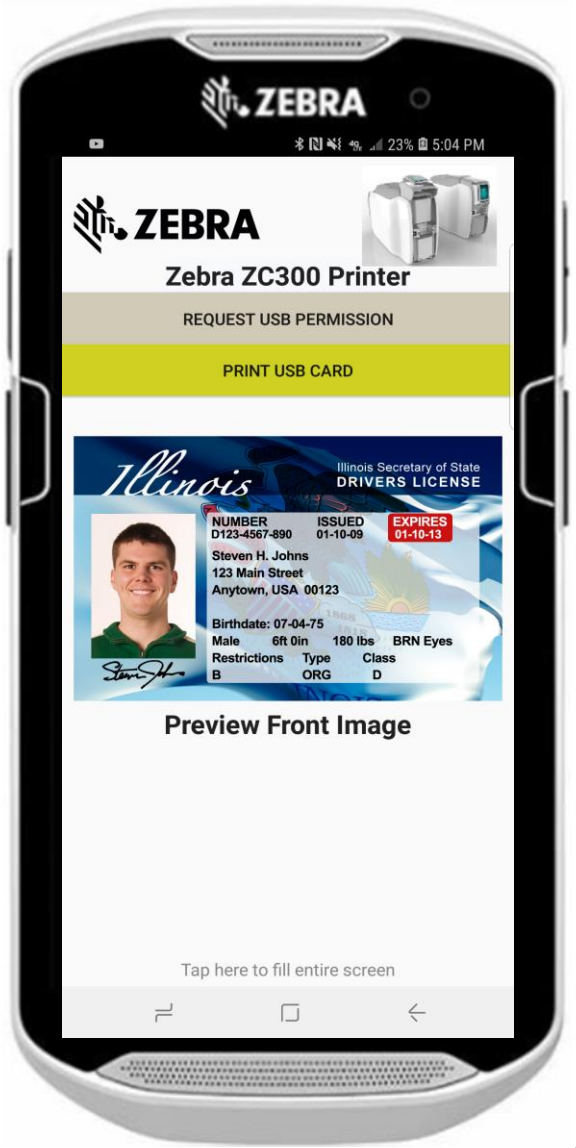
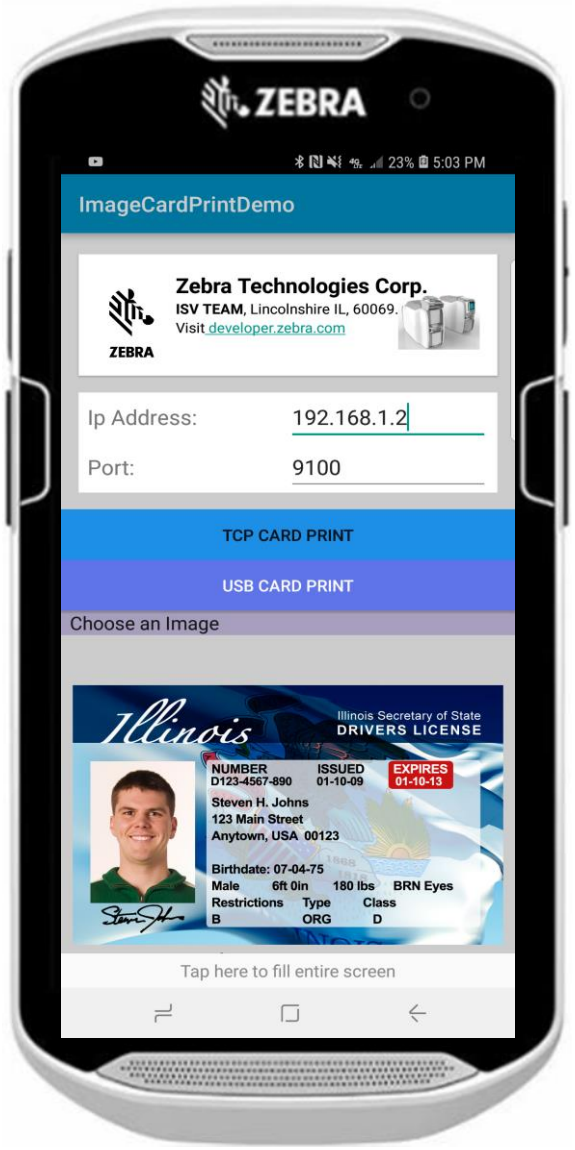
Developing my first Android App with ZC100/300





ZEBRA

Developing my first Android App with ZC100/300



Use Case – Select an image and send to print front-side

Basic Requirements:

Devices

- Zebra Card Printer – ZC300
- Zebra Mobile Computer – TC51

Communication

- TC51(WiFi) – ZC300 through Ethernet

Formatting Language

- Print Job (xml)

Platform

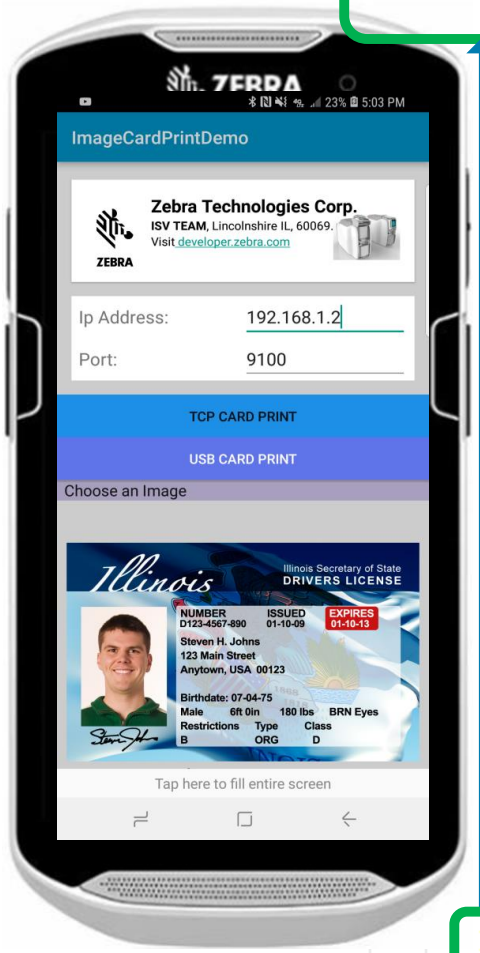
- Android

Data

- Local app

ZEBRA

Developing my first Android App with ZC100/300



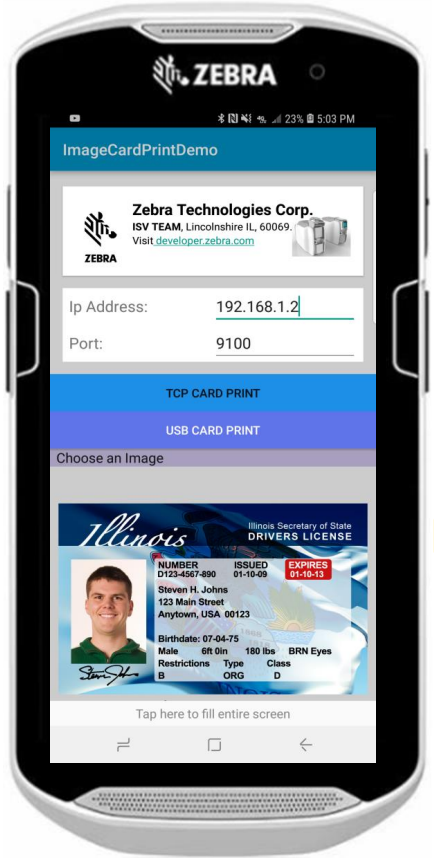

```
buttonCardNetwork.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        Thread threadmac = new Thread(new Runnable() {
            @Override
            public void run() {

                try {
                    Connection connection = null;
                    ZebraCardPrinter printer = null;
                    try {
                        ip = ipAddressEditText.getText().toString();
                        System.out.format("Status: %s%n", ip);
                        port = portNumberEditText.getText().toString();
                        System.out.format("Status: %s%n", port);
                        tcpCardHandler.getPrinterStatusOverTcp(ip, port);
                        connection = new TcpConnection(ip, Integer.parseInt(port));
                        connection.open();
                        printer = ZebraCardPrinterFactory.getInstance(connection);
                        if (printer != null) {
                            printthisCard.printCard(printer, uriTransferPath, context);
                        }
                    } catch (Exception e) {
                        System.out.format("Status: %s%n", e.getMessage());

                    finally {
                        try {
                            if (pri
                                printer
                                printer
                            }
                        } catch (ZebraCa
                            e.printStackT
                        }
                        if (connection !=
                            try {
                                connectio
                                connectio
                            } catch (Conn
                                e.printSt
                            }
                        }
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
        threadmac.start();
    }
});
```

```
protected void printCard(ZebraCardPrinter printer, String pathImage, final Context context) throws ZebraCardException, ConnectionException,
        IOException, SettingsException, ZebraIllegalArgumentException {
    ZebraGraphics graphics = null;
    try {
        List<GraphicsInfo> graphicsData = new ArrayList<~>();
        graphics = new ZebraCardGraphics(printer);

        generatePrintJobImage(graphics, graphicsData, pathImage);

        int jobId = printer.print( E 1, graphicsData);
        pollJobStatus(printer, jobId, context);

        final JobStatusInfo jStatus = printer.getJobStatus(jobId);
        this.runOnUiThread(new Runnable() {
            public void run() {
                Toast.makeText(context.getApplicationContext(), text: "Job completed: " + jStatus.printStatus, Toast.LENGTH_LONG).show();
                System.out.format("Job Completed: %s%n", jStatus.printStatus);
            }
        });

    } finally {
        if (graphics != null) {
            graphics.clear();
            graphics.close();
        }
    }
}
```

Using a Thread to open a quick connection

ZEBRA

Developing my first Android App with ZC100/300




```
protected void printCard(ZebraCardPrinter printer, String pathImage, final Context context) throws ZebraCardException, ConnectionException,
        IOException, SettingsException, ZebraIllegalArgumentException {
    ZebraGraphics graphics = null;
    try {
        List<GraphicsInfo> graphicsData = new ArrayList<GraphicsInfo>();
        graphics = new ZebraCardGraphics(printer);

        generatePrintJobImage(graphics, graphicsData, pathImage);

        int jobId = printer.print( i: 1, graphicsData);
        pollJobStatus(printer, jobId, context);

        final JobStatusInfo jStatus = printer.getJobStatus(jobId);
        this.runOnUiThread(new Runnable() {
            public void run() {
                Toast.makeText(context.getApplicationContext(), text: "Job completed: " + jStatus.pri    us, Toast.LENGTH_LONG).show();
                System.out.format("Job Completed: %s%n", jStatus.printStatus);
            }
        });


    } finally {
        if (graphics != null) {
            graphics.clear();
            graphics.close();
        }
    }
}
```

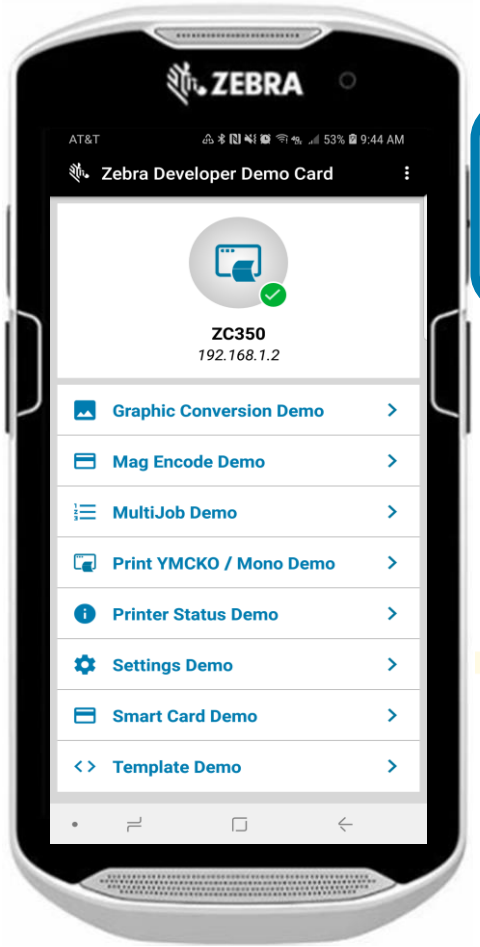
Building the layers

Sending the print job to the printer

Monitoring the job status

ZEBRA

Developing my first Android App with ZC100/300

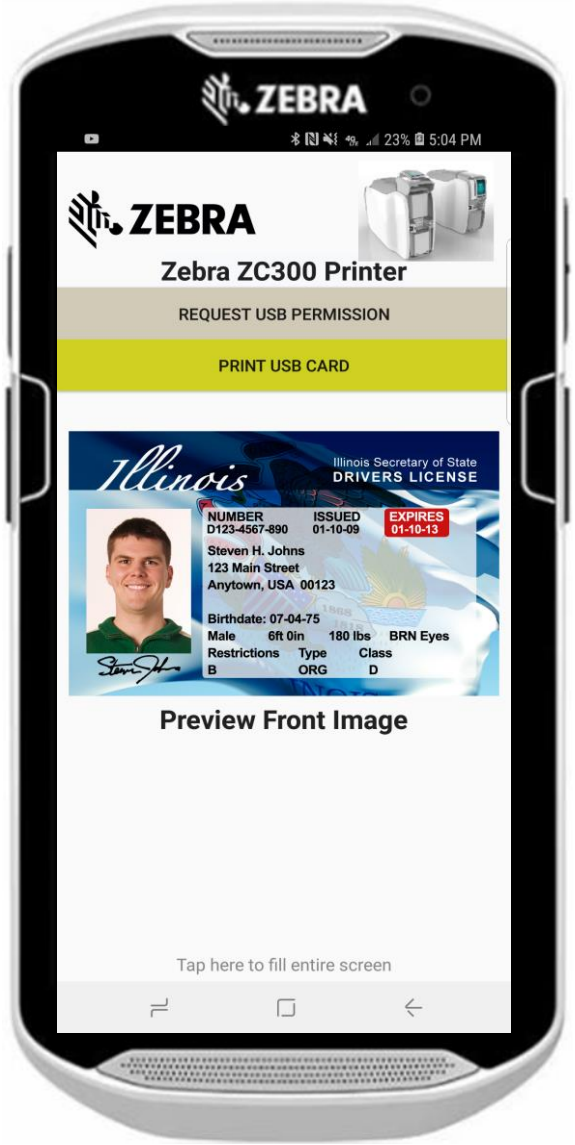



```
@Override
protected void onPreExecute() {
    super.onPreExecute();

    if (onManualConnectionListener != null) {
        onManualConnectionListener.onManualConnectionStarted();
    }
}

@Override
protected Void doInBackground(Void... voids) {
    TcpConnection connection = null;

    try {
        connection = getTcpConnection(ipAddress);
        connection.open();

        Map<String, String> discoveryDataMap = DiscoveryUtilCard.getDiscoveryDataMap(
                                                nection);

        String model = discoveryDataMap.get("MODEL");
        if (model != null) {
            if(!model.toLowerCase().contains("zxp1") && !model.toLowerCase()
                                                    tains("zxp3")) {
                printer = new DiscoveredCardPrinterNetwork(discoveryDataMa
            } else {
                throw new ConnectionException(weakContext.get().getStri
                                                (R.string.printer_model_not_supported));
            }
        } else {
            throw new SettingsException(weakContext.get().getStri
                                        (R.string.no_printer_model_found));
        }
    } catch (Exception e) {
        exception = e;
    } finally {
        ConnectionHelper.cleanUpQuietly( zebraCardPrinter
                                        1, connection);
    }

    return null;
}

@Override
protected void onPostExecute(Void aVoid) {
    super.onPostExecute(aVoid);

    if (onManualConnectionListener != null) {
        onManualConnectionListener.onManualConnectionFinished(exception, printer);
    }
}
```

Connection doing on Background

ZEBRA

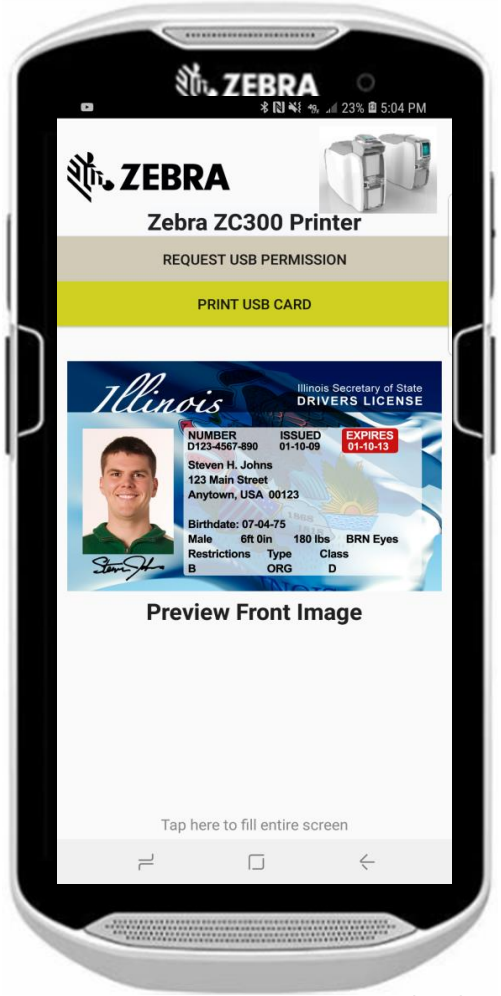
Developing my first Android App with ZC100/300




```
IntentFilter filter = new IntentFilter(ACTION_USB_PERMISSION);


// Catches intent indicating if the user grants permission to use the USB device
private final BroadcastReceiver mUsbReceiver = (context, intent) → {
        String action = intent.getAction();
        if (ACTION_USB_PERMISSION.equals(action)) {
            synchronized (this) {
                UsbDevice device = (UsbDevice) intent.getParcelableExtra(UsbManager.EXTRA_DEVICE);
                if (intent.getBooleanExtra(UsbManager.EXTRA_PERMISSION_GRANTED, defaultValue: false)) {
                    if (device != null) {
                        hasPermissionToCommunicate = true;
                    }
                }
            }
        }
};
```

Usb permission to have access to USB port First, then, you can open the port for communication

Developing my first Android App with ZC100/300

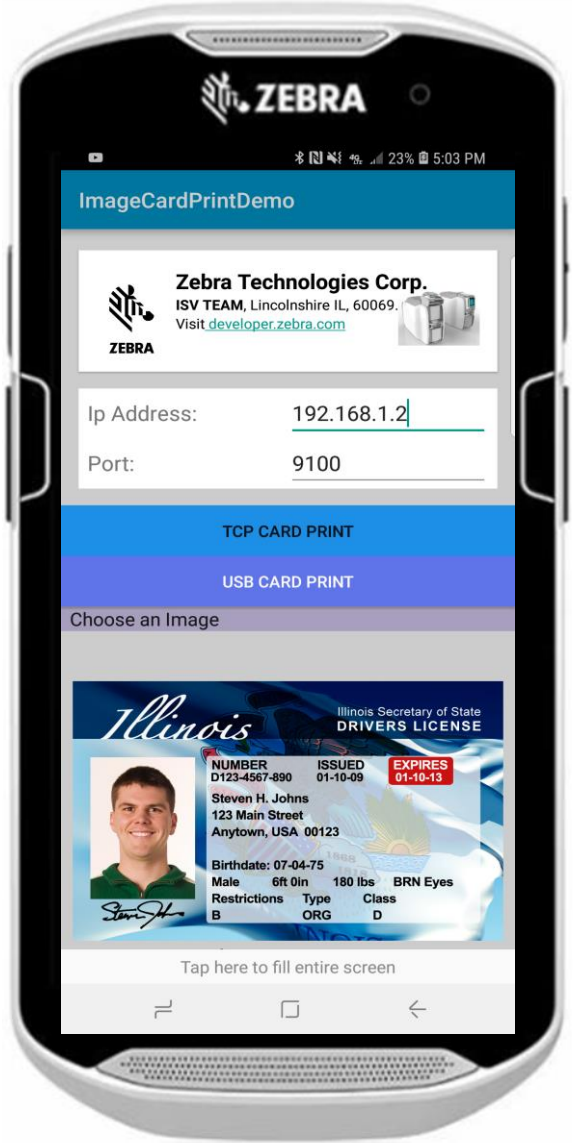


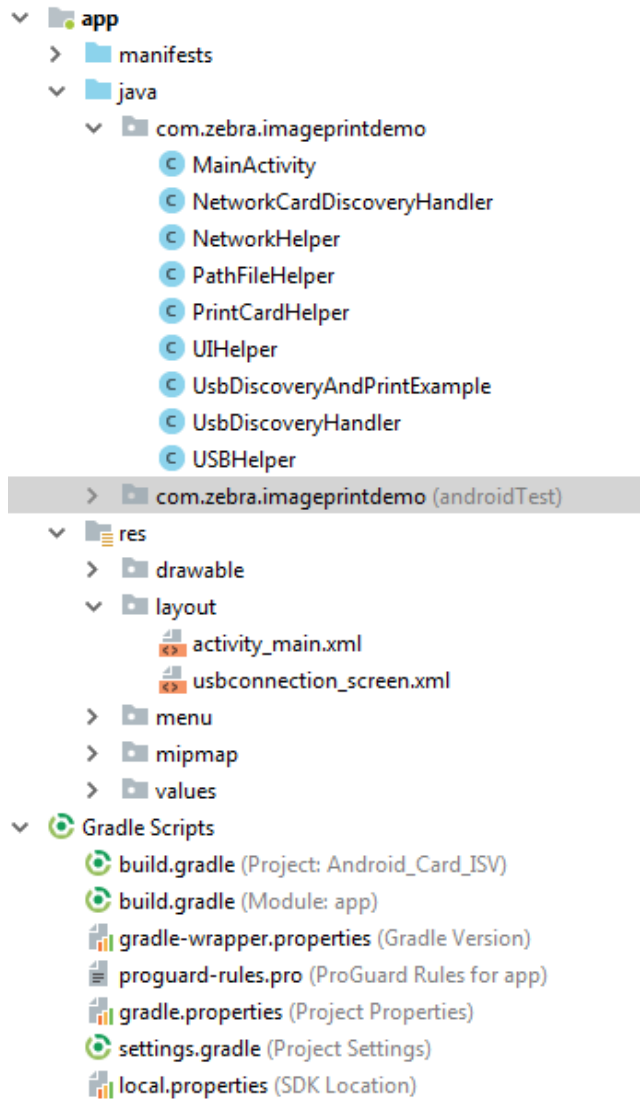

```
// Print button click
buttonPrint.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        Thread threadmac2 = new Thread(new Runnable() {
            @Override
            public void run() {
                try {
                    Connection connection = null;
                    ZebraCardPrinter printer = null;
                    try {
                        if (hasPermissionToCommunicate) {
                            try {
                                connection = discoveredPrinterUsb.getConnection();
                                usbHelper.getPrinterInfoOverUsb(connection);
                                connection.open();
                                printer = ZebraCardPrinterFactory.getInstance(connection);
                                if (printer != null) {
                                    printthisCard.printCard(printer, transferredImage, context);
                                }
                            } catch (final Exception e1) {
                                UsbDiscoveryAndPrintExample.this.runOnUiThread(new Runnable() {
                                    public void run() {
                                        Toast.makeText(getApplicationContext(), text: e1.getMessage() + e1.getLocalizedMessage(), Toast.LENGTH_LONG).show();
                                        System.out.format("Exception: %s%n", e1.toString());
                                    }
                                });
                            }
                        } else {
                            UsbDiscoveryAndPrintExample.this.runOnUiThread(new Runnable() {
                                public void run() {
                                    Toast.makeText(getApplicationContext(), "No permission to communicate", Toast.LENGTH_LONG).show();
                                    System.out.f
                                }
                            });
                        }
                    } finally {
                        try {
                            if (printer != null)
                                printer.destroy(
                                printer = null;
                            }
                        } catch (ZebraCardExcept
                            e.printStackTrace();
                        }
                        if (connection != null)
                            try {
                                connection.close
                                connection = nul
                            } catch (ConnectionE
                                e.printStackTrac
                            }
                        }
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
        threadmac2.start();
    }
});
```

```
protected void printCard(ZebraCardPrinter printer, String pathImage, final Context context) throws ZebraCardException, ConnectionException,
        IOException, SettingsException, ZebraIllegalArgumentException {
    ZebraGraphics graphics = null;
    try {
        List<GraphicsInfo> graphicsData = new ArrayList<~>();
        graphics = new ZebraCardGraphics(printer);

        generatePrintJobImage(graphics, graphicsData, pathImage);

        int jobId = printer.print( c: 1, graphicsData);
        pollJobStatus(printer, jobId, context);

        final JobStatusInfo jStatus = printer.getJobStatus(jobId);
        this.runOnUiThread(new Runnable() {
            public void run() {
                Toast.makeText(context.getApplicationContext(), text: "Job completed: " + jStatus.printStatus, Toast.LENGTH_LONG).show();
                System.out.format("Job Completed: %s%n", jStatus.printStatus);
            }
        });

    } finally {
        if (graphics != null) {
            graphics.clear();
            graphics.close();
        }
    }
}
```

ZEBRA

Developing my first Android App with ZC100/300





Four Helper file classes

- Network
- PathFile
- USB
- Print

ZEBRA

Developing my first Android App with ZC100/300

```java
public class NetworkHelper {

    public NetworkHelper() {}

    protected void getPrinterStatusOverTcp(String theIpAddress, String port) throws ConnectionException, SettingsException, ZebraCardException {
        Connection connection = new TcpConnection(theIpAddress, Integer.parseInt(port));
        ZebraCardPrinter zebraCardPrinter = null;

        try {
            connection.open();
            zebraCardPrinter = ZebraCardPrinterFactory.getInstance(connection);

            PrinterStatusInfo printerStatusInfo = zebraCardPrinter.getPrinterStatus();
            System.out.format("Status: %s%n", printerStatusInfo.status);
            System.out.format("Alarm: %s (%s)%n", printerStatusInfo.alarmInfo.value, printerStatusInfo.alarmInfo.description);
            System.out.format("Error: %s (%s)%n", printerStatusInfo.errorInfo.value, printerStatusInfo.errorInfo.description);
            System.out.format("Total jobs: %s%n", printerStatusInfo.jobsTotal);
            System.out.format("Pending jobs: %s%n", printerStatusInfo.jobsPending);
            System.out.format("Active jobs: %s%n", printerStatusInfo.jobsActive);
            System.out.format("Completed jobs: %s%n%n", printerStatusInfo.jobsComplete);

            PrinterInfo printerInfo = zebraCardPrinter.getPrinterInformation();
            System.out.format("Vendor: %s%n", printerInfo.vendor);
            System.out.format("Model: %s%n", printerInfo.model);
            System.out.format("SerialNumber: %s%n", printerInfo.serialNumber);
            System.out.format("OEM Code: %s%n", printerInfo.oemCode);
            System.out.format("Firmware Version: %s%n%n", printerInfo.firmwareVersion);

        } catch (ConnectionException e) {
            // Handle communications error here.
            e.printStackTrace();
        } finally {
            // Release resources and close the connection
            zebraCardPrinter.destroy();
            connection.close();
        }
    }
}
```

```
I/System.out: Status: 192.168.1.2
              Status: 9100
I/System.out: Status: idle
              Alarm: 0 (System: No error)
              Error: 0 (System: No error)
              Total jobs: 1
              Pending jobs: 0
              Active jobs: 0
I/System.out: Completed jobs: 1
              Vendor: Zebra
              Model: ZC350
              SerialNumber: C3J180400109
              OEM Code: ZUSA01
              Firmware Version: V201.01.01
```

NetworkHelper & UsbHelper

- Creates a quick TCP/USB Connection to capture basic information of the printer to review printer status before to send the print jobs and also to verify printer information as vendor, model, serial number, OEM code and firmware version.

PathFile & UI

- To manage other subroutines needed to show messages and render the address of the files in Android Mobil Phone.

ZEBRA

Developing my first Android App with ZC100/300

```java
public class NetworkHelper {

    public NetworkHelper() {}

    protected void getPrinterStatusOverTcp(String theIpAddress, String port) throws ConnectionException, SettingsException, ZebraCardException {
        Connection connection = new TcpConnection(theIpAddress, Integer.parseInt(port));
        ZebraCardPrinter zebraCardPrinter = null;

        try {
            connection.open();
            zebraCardPrinter = ZebraCardPrinterFactory.getInstance(connection);

            PrinterStatusInfo printerStatusInfo = zebraCardPrinter.getPrinterStatus();
            System.out.format("Status: %s%n", printerStatusInfo.status);
            System.out.format("Alarm: %s (%s)%n", printerStatusInfo.alarmInfo.value, printerStatusInfo.alarmInfo.description);
            System.out.format("Error: %s (%s)%n", printerStatusInfo.errorInfo.value, printerStatusInfo.errorInfo.description);
            System.out.format("Total jobs: %s%n", printerStatusInfo.jobsTotal);
            System.out.format("Pending jobs: %s%n", printerStatusInfo.jobsPending);
            System.out.format("Active jobs: %s%n", printerStatusInfo.jobsActive);
            System.out.format("Completed jobs: %s%n%n", printerStatusInfo.jobsComplete);

            PrinterInfo printerInfo = zebraCardPrinter.getPrinterInformation();
            System.out.format("Vendor: %s%n", printerInfo.vendor);
            System.out.format("Model: %s%n", printerInfo.model);
            System.out.format("SerialNumber: %s%n", printerInfo.serialNumber);
            System.out.format("OEM Code: %s%n", printerInfo.oemCode);
            System.out.format("Firmware Version: %s%n%n", printerInfo.firmwareVersion);

        } catch (ConnectionException e) {
            // Handle communications error here.
            e.printStackTrace();
        } finally {
            // Release resources and close the connection
            zebraCardPrinter.destroy();
            connection.close();
        }
    }
}
```

NetworkHelper & UsbHelper

- Creates a quick TCP/USB Connection to capture basic information of the printer to review printer status before to send the print jobs and also to verify printer information as vendor, model, serial number, OEM code and firmware version.

PathFile & UI

- To manage other subroutines needed to show messages and render the address of the files in Android Mobil Phone.

ZEBRA


THANK YOU
Dankon
Maake
Ua Tsaug Rau Koj
Mochchakkeram
Terma Kasih
Multumesc
Merci
Raibh Maith Agat
Chokrane
Juspaxar
Mamana
Kiitos
Kia Ora
Grazie
Obrigado
Asante
Vinaka
Matondo
Dank Je
Merci
Nirringrazzjak
Obrigado
Obrigado
Multumesc
Spasibo
Cam on ban
Kiitos
Raibh Maith Agat
Merci
Mochchakkeram
Chokrane
Grazie
Welalin
Multumesc
Mochchakkeram
Obrigado
Matondo
Salamat
Nirringrazzjak
Matur Nuwun
Chokrane
Raibh Maith Agat
Kiitos
Salamat
Arigato
Obrigado
Mochchakkeram
Ua Tsaug Rau Koj
Maake
Spasibo