

مستند شیوه برقراری ارتباط با

برنامه پرداخت دستگاه P1000

تهیه کننده : حامد سلیمی

تاریخ تنظیم سند : چهارم مرداد ماه 1399

نسخه اول

هدف از این مستند ایجاد دفترچه راهنمای چگونگی برقراری ارتباط با اپلیکیشن اندرویدی دستگاه P1000 می باشد. به منظور درک بهتر یک پروژه تحت عنوان CPSDK تهیه شده است که در اختیار شرکت های ثانویه قرار می گیرد.

این پروژه از سه بخش اصلی تشکیل شده است.

بخش اول : تشکیل کلاس Boot به منظور اخذ permission های مورد نیاز برنامه می باشد.

بخش دوم : تشکیل یک کلاس BroadCast به منظور دریافت پاسخ از برنامه پارسایان می باشد.

بخش سوم : توسط این بخش درخواست های مختلف به اپلیکیشن ارسال می شود.

در این برنامه پرداخت امکاناتی فراهم آمده است تا شرکت های ثانویه بتوانند تراکنش های خرید، خرید دولتی، موجودی، خرید شارژ(تکی و مستقیم) و پرداخت قبض را انجام دهند.

الف – پیاده سازی اخذ Permission

از آنجا که سیستم عامل دستگاه های P1000 اندروید 9 می باشد بنابراین باید permission های مورد نیاز به صورت runtime اخذ شود به همین منظور برنامه های ثانویه می بایست حتما کلاس BootActivity که در کد نمونه قرار داده شده است را پیاده کنند تا permission های مورد نیاز را از طریق این کلاس به برنامه بدهند.

ب- پیاده سازی انجام تراکنش

برنامه های ثانویه برای انجام تراکنش نیاز دارند که ابتدا یک intent به همراه یک action با عنوان "com.pec.smartpos.cpsdk" تعریف کرده و دو مقدار نام شرکت و نوع تراکنش را به این intent با عنوان های CompanyName و transactionType مقداردهی کنند، البته مقداردهی های دیگری نیز وجود دارد که بسته نوع تراکنش متفاوت می باشد که در ادامه توضیح آن آورده شده است.

پس از مقداردهی کردن می بایست intent را با تابع startActivity اجرا کرد. پس از این، اپلیکیشن پارسیان با اکتیویتی مربوطه اجرا می گردد که آماده انجام تراکنش مورد نظر است. در این مرحله، برنامه پارسیان تراکنش را انجام می دهد و پاسخ را در غالب کلاس Response که این کلاس نیز در کد نمونه وجود دارد، به برنامه بازمی گرداند که این باز گرداندن به صورت sendBroadcast می باشد.

بنابراین برنامه ثانویه می بایست یک Receiver به همراه IntentFilter با عنوان "com.pec.ThirdCompany" رجیستر کند تا قادر باشد توسط این کلاس پاسخ را دریافت کند. در این حالت می توان در onReceive کلاس Receiver توسط intentی که به آن داده شده است پاسخ را توسط کلید های کلاس Response دریافت کرد. در زیر کدهای مربوط به تعریف و رجیستر کردن Receiver با اکشن مورد نظر آورده شده است.

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
    IntentFilter intentFilter = new IntentFilter();  
    intentFilter.addAction("com.pec.ThirdCompany");  
    myBroadCast = new MyBroadCast();  
    registerReceiver(myBroadCast, intentFilter);  
    activity = this;  
}
```

```
onActivityResult = true;
```

در شکل زیر شیوه دریافت پاسخ از برنامه پارسایان توسط intent که به Receiver داده شده است آورده شده است. همانطور که دیده می شود تمامی مقادیر توسط تابع getStringExtras با کلید هایی که در کلاس Response قرار دارند دریافت شده است.

```
import com.google.android.material.snackbar.Snackbar;

public class MyBroadCast extends BroadcastReceiver {

    private Response response = new Response();
    @SuppressWarnings("WrongConstant")
    @Override
    public void onReceive(Context context, Intent intent) {

        String str = "Hi I got your message : ";
        response.PS = intent.getStringExtra(Response.PrintStatus);
        if(response.PS != null){
            if(response.PS.equals("FINISHED")) {
                Log.e( tag: "MyBroadCastReceiver", msg: "***" + "Print FINISHED");
                str += "Print FINISHED";
            }
        }
        else {
            response.RS = intent.getStringExtra(Response.ResponseCode);
            if(response.RS != null){
                if(response.RS.equals("05")) {
                    Log.e( tag: "MyBroadCastReceiver", msg: "***" + "Cancelled");
                    str += "Transaction Cancelled";
                }
                else {
                    response.trxType = intent.getStringExtra(Response.TransactionType);
                    response.CN = intent.getStringExtra(Response.CardNumber);
                    response._RRN = intent.getStringExtra(Response.RRN);
                    response.SN = intent.getStringExtra(Response.SerialNumber);
                    response.TN = intent.getStringExtra(Response.TerminalNumber);
                    response.DT = intent.getStringExtra(Response.DateTime);
                    response.DI = intent.getStringExtra(Response.DateTime);
                    response.IH = intent.getStringExtra(Response.IssueNumber);
                    response.ZH = intent.getStringExtra(Response.ZoneNumber);
                    response.KRN = intent.getStringExtra(Response.KRN);
                    response.CH = intent.getStringExtra(Response.CashNumber);
                    response.LXJLBC = intent.getStringExtra(Response.LXJLBC);
                }
            }
        }
    }
}
```

```

        response._RRN = intent.getStringExtra(Response.RRN);
        response.SN = intent.getStringExtra(Response.SerialNumber);
        response.TN = intent.getStringExtra(Response.TerminalNumber);
        response.DT = intent.getStringExtra(Response.DateTime);
        switch (response.trxType) {
            case "Sale":
            case "BillPayment":
                response.AM = intent.getStringExtra(Response.Amount);
                break;
            case "Balance":
                response.BL = intent.getStringExtra(Response.Balance);
                break;
            case "SingleCharge":
                response._Pin = intent.getStringExtra(Response.Pin);
                response._Serial = intent.getStringExtra(Response.Serial);
                break;
            case "QuickCharge":
                response._Phone = intent.getStringExtra(Response.Phone);
                break;
        }
        Log.e( tag: "MyBroadCastReceiver", msg: "*" + response.toString());
        str += "\n" + response.toString();
    }
}

Snackbar.make(MainActivity.activity.findViewById(R.id.main),
    str,
    Snackbar.ANIMATION_MODE_SLIDE).
    show();
}
,
MyBroadCast > onReceive()

```

```

MyBroadCast > onReceive()
{
    str = "";
    response = new Response();
    response._RRN = intent.getStringExtra(Response.RRN);
    response.SN = intent.getStringExtra(Response.SerialNumber);
    response.TN = intent.getStringExtra(Response.TerminalNumber);
    response.DT = intent.getStringExtra(Response.DateTime);
    response.trxType = intent.getStringExtra(Response.TrxType);
    response.AM = intent.getStringExtra(Response.Amount);
    response.BL = intent.getStringExtra(Response.Balance);
    response.Pin = intent.getStringExtra(Response.Pin);
    response.Serial = intent.getStringExtra(Response.Serial);
    response.Phone = intent.getStringExtra(Response.Phone);
    response.toString();
}

```

تراکنش خرید :

```
public void onClickSale(View view) {  
  
    Intent i = new Intent(TAGS.Action);  
    i.putExtra(TAGS.CompanyName, value: "pecco");  
    i.putExtra(TransactionType.transactionType, TransactionType.Sale);  
  
    i.putExtra(TAGS.AM, value: "1000");  
    startActivity(i);  
}
```

TAG.Action = “com.pec.smartpos.cpsdk”

باید دقت شود که از کلاس TransactionType نوع متناسب تراکنش انتخاب شود و در قسمت نام شرکت باید بجای مقدار pecco نام شرکته مربوطه و بجای عدد 1000 مقدار مورد نظر برای خرید قرار داده شود.

تراکنش خرید دولتی:

```
public void onClickGovSale(View view) {  
  
    Intent i = new Intent(TAGS.Action);  
    i.putExtra(TAGS.CompanyName, value: "pecco");  
    i.putExtra(TransactionType.transactionType, TransactionType.SaleGov);  
  
    i.putExtra(TAGS.AM, value: "1000");  
    i.putExtra(TAGS.GovID, value: "383026282140125006203003336820");  
    startActivity(i);  
}
```

TAG.Action = “com.pec.smartpos.cpsdk”

باید دقت شود که از کلاس TransactionType نوع متناسب تراکنش انتخاب شود و در قسمت نام شرکت باید بجای مقدار pecco نام شرکته مربوطه، بجای عدد 1000 مقدار مورد نظر برای خرید و همچنین بجای عدد فوق شناسه دولتی مورد نظر قرار داده شود.

تراکنش موجودی:

```
public void onClickBalance(View view) {  
  
    Intent i = new Intent(TAGS.Action);  
    i.putExtra(TAGS.CompanyName, value: "pecco");  
    i.putExtra(TransactionType.transactionType, TransactionType.Balance);  
    startActivity(i);  
}
```

TAG.Action = “com.pec.smartpos.cpsdk”

باید دقت شود که از کلاس TransactionType نوع متناسب تراکنش انتخاب شود و در قسمت نام شرکت باید بجای مقدار pecco نام شرکته مربوطه قرار داده شود.

تراکنش قبض:

```
public void onClickBill(View view) {  
  
    Intent i = new Intent(TAGS.Action);  
    i.putExtra(TAGS.CompanyName, value: "pecco");  
    i.putExtra(TransactionType.transactionType, TransactionType.BillPayment);  
  
    i.putExtra(TAGS.BillName, value: "اب"); // Bill name  
    i.putExtra(TAGS.BI, value: "1257617"); // Bill ID  
    i.putExtra(TAGS.PI, value: "121225"); // Payment ID  
    startActivity(i);  
}
```

TAG.Action = “com.pec.smartpos.cpsdk”

باید دقت شود که از کلاس TransactionType نوع متناسب تراکنش انتخاب شود و در قسمت نام شرکت باید بجای مقدار pecco نام شرکته مربوطه، همچنین نام قبض، شناسه قبض و شناسه پرداخت مطابق فوق در intent قرار داده شود.

تراکنش شارژ تکی:

```
public void onClickSingleCharge(View view) {  
  
    Intent i = new Intent(TAGS.Action);  
    i.putExtra(TAGS.CompanyName, value: "pecco");  
    i.putExtra(TransactionType.transactionType, TransactionType.SingleCharge);  
  
    i.putExtra(TAGS.ChargeAM, value: "20000");  
    i.putExtra(TAGS.Operator, TAGS.IranCell);  
    startActivity(i);  
}
```

TAG.Action = “com.pec.smartpos.cpsdk”

باید دقت شود که از کلاس TransactionType نوع متناسب تراکنش انتخاب شود و در قسمت نام شرکت باید بجای مقدار pecco نام شرکتی مربوطه، همچنین باید نام مقدار شارژ و نوع اپراتور، مطابق فوق در intent قرار داده شود. دقت شود که مبلغ باید معتبر باشد و هر مبلغی نمی توان وارد کرد.

مبالغ معتبر برای خرید شارژ ایرانسل : دوهزارتومن، پنج هزار تومن، ده هزار تومن

مبالغ معتبر برای خرید شارژ همراه اول : هزار تومن، دوهزارتومن، پنج هزار تومن، ده هزار تومن، بیست هزارتومن

مبالغ معتبر برای خرید شارژ تالیا : دوهزارتومن، دوهزارتومن، پنج هزار تومن، ده هزار تومن

مبالغ معتبر برای خرید شارژ رایتل : دوهزارتومن، پنج هزار تومن، ده هزار تومن، بیست هزارتومن

تراکنش شارژ مستقیم:

```
public void onClickQuickCharge(View view) {  
  
    Intent i = new Intent(TAGS.Action);  
    i.putExtra(TAGS.CompanyName, value: "pecco");  
    i.putExtra(TransactionType.transactionType, TransactionType.QuickCharge);  
  
    i.putExtra(TAGS.ChargeAM, value: "12345");  
    i.putExtra(TAGS.phone, value: "09010575701");  
    startActivity(i);  
}
```

TAG.Action = “com.pec.smartpos.cpsdk”

باید دقت شود که از کلاس TransactionType نوع متناسب تراکنش انتخاب شود و در قسمت نام شرکت باید بجای مقدار pecco نام شرکت مربوطه، همچنین باید مقدار شارژ و شماره تلفن، مطابق فوق در intent قرار داده شود.

ج- پیاده سازی عملیات پرینت

عملیات پرینت به دو دسته : پرینت رسید تراکنش و رسید شرکت های ثانویه تقسیم می گردد.

الف- رسید تراکنش

به منظور دریافت رسید تراکنش صرفا کافیهست که برنامه ثانویه یک intent با Action با عنوان "com.pec.smartpos.cpsdk" تولید کند و نام شرکت خود را به عنوان CompanyName و نوع پرینت را از کلاس printType انتخاب کند و سپس RRN تراکنش مورد نظر به همراه respCode آن را در intent قرار دهد و آن را startActivity نماید. بدیهی است به منظور عملیات پرینت برنامه های ثانویه می بایست بعد از دریافت پاسخ مقادیر دریافتی را ذخیره کنند تا در هنگام پرینت به مقادیر فوق دسترسی داشته باشند که این عمل می تواند توسط دیتابیس SQLite انجام شود.

```
public void onClickPrint(View view) {  
  
    Intent i = new Intent(TAGS.Action);  
    i.putExtra(TAGS.CompanyName, value: "pecco");  
    // select print type from PrintType class  
    i.putExtra(PrintType.printType, PrintType.receiptSale);  
  
    i.putExtra(TAGS.RRN, value: "720579393417");  
    i.putExtra(TAGS.respCode, value: "00");  
    startActivity(i);  
}
```

ب- رسید شرکت

برای عملیات چاپ رسیدهای مخصوص شرکتهای می بایست این رسید به بخش های مختلف تقسیم گردد و به صورت یک لیستی از bitmap درآیند و به تابع saveFile داده شود. پس از آن باید یک intent ساخته شود و نام شرکت، نوع پرینت و تعداد bitmap ها در intent ساخته شده قرار داده شود و نهایتا با تابع startActivity اجرا شود. در شکل زیر نحوه استفاده آورده شده است. (تابع saveFile داخل برنامه وجود دارد)

```
public void onClickPrintBitMap(View view) {
    try {
        Bitmap bitmap = BitmapFactory.decodeStream(getAssets().open( fileName: "header2.jpg"));
        List<Bitmap> bitmapList = new ArrayList<>();
        bitmapList.add(bitmap);
        bitmapList.add(bitmap);
        saveFile(bitmapList, companyName: "pecco");

        Intent i = new Intent(TAGS.Action);
        i.putExtra(TAGS.CompanyName, value: "pecco");
        i.putExtra(PrintType.printType, PrintType.receiptBitMap);
        i.putExtra(TAGS.NumOfBitMap, bitmapList.size());
        startActivity(i);
    }
    catch (IOException e) {
        e.printStackTrace();
    }
}
```

دقت شود که در متد Receiver در کلاس MyBroadcast ابتدا وضعیت پرینت چک می شود که این وضعیت به صورت زیر چک می شود.

```

response.PS = intent.getStringExtra(Response.PrintStatus);
if(response.PS != null) {
    if(response.PS.equals("FINISHED")) {
        Log.e( tag: "MyBroadCastReceiver", msg: "***" + "Print FINISHED");
        str += "Print FINISHED";
    }
}
}

```

معرفی کلاس های TransactionType و PrintType

این کلاس ها صرفا حاوی یکسری مقادیر ثابت برای مقداردهی به **intent** هایی است که برای تراکنش و یا پرینت ساخته می شوند. در شکل زیر این دو کلاس آورده شده است.

```

public class TransactionType {

    public static final String transactionType = "transactionType";
    // types of transaction
    public static final String Sale = "Sale";
    public static final String SaleGov = "Sale";
    public static final String Balance = "Balance";
    public static final String BillPayment = "BillPayment";
    public static final String SingleCharge = "SingleCharge";
    public static final String QuickCharge = "QuickCharge";
}

```

مقدار **transactionType** به عنوان کلید (**key**) و باقی مقادیر بسته به نوع تراکنش موردنظر به عنوان مقدار (**value**) به **intent** داده می شوند.

```
public class PrintType {

    public static final String printType = "printType";
    // types of print
    public static final String receiptBitMap = "receiptBitMap";
    public static final String receiptSale = "receiptSale";
    public static final String receiptBalance = "receiptBalance";
    public static final String receiptSingleCharge = "receiptSingleCharge";
    public static final String receiptQuickCharge = "receiptQuickCharge";
    public static final String receiptGovPayment = "receiptGovPayment";
    public static final String receiptBillPayment = "receiptBillPayment";
}
```

```
)
    receiptGovPayment = "receiptGovPayment";
    receiptSingleCharge = "receiptSingleCharge";
    receiptQuickCharge = "receiptQuickCharge";
    receiptBalance = "receiptBalance";
    receiptSale = "receiptSale";
    receiptBitMap = "receiptBitMap";
    printType = "printType";
}
```

مقدار **printType** به عنوان کلید (**key**) و باقی مقادیر بسته به نوع پرینت موردنظر به عنوان مقدار (**value**) به **intent** داده می شوند.

معرفی کلاس های TAGS و Response

این دو کلاس نیز حاوی هیچ گونه متدی نمی باشد و صرفا حاوی یکسری کلید و مقدار می باشند.

```
public class TAGS {  
      
    public static final String Action = "com.pec.smartpos.cpsdk";  
    public static final String CompanyName = "CompanyName";  
  
    public static final String AM = "AM";  
    public static final String GovID = "GovID";  
    public static final String BillName = "BillName";  
    public static final String BI = "BI";  
    public static final String PI = "PI";  
  
    public static final String MCI = "همراه اول";  
    public static final String IranCell = "ایرانسل";  
    public static final String Talia = "تالیا";  
    public static final String Ritel = "رایتل";  
    public static final String ChargeAM = "ChargeAM";  
    public static final String Operator = "Operator";  
    public static final String phone = "phone";  
  
    public static final String RRN = "RRN";  
    public static final String respCode = "respCode";  
  
    public static final String BitMap = "BitMap";  
    public static final String NumOfBitMap = "NumOfBitMap";  
}
```

```
public class Response {

    public static final String TransactionType = "TransactionType";
    public static final String ResponseCode = "ResponseCode";
    public static final String CardNumber = "CardNumber";
    public static final String RRN = "RRN";
    public static final String SerialNumber = "SerialNumber";
    public static final String TerminalNumber = "TerminalNumber";
    public static final String DateTime = "DateTime";
    public static final String Amount = "Amount";
    public static final String Balance = "Balance";
    public static final String Pin = "Pin";
    public static final String Serial = "Serial";
    public static final String Phone = "Phone";

    public String trxType = "";
    public String RS = "";
    public String CN = "";
    public String _RRN = "";
    public String SN = "";
    public String TN = "";
    public String DT = "";
    public String AM = "";
    public String BL = "";
    public String _Pin = "";
    public String _Serial = "";
    public String _Phone = "";

    @Override
    public String toString() {...}

}
```