POSITIVO

Quick_Start-Servico_de_Impressão para_POS-V1.2

Introdução

Este Quick Start tem como objetivo facilitar a integração e o uso do serviço de impressão em nossos Point of Sale (POS). O serviço de impressão da L400 oferece duas alternativas de integração: diretamente via AIDL ou através de uma biblioteca que encapsula a implementação do AIDL.

Opções de Integração

AIDL (Android Interface Definition Language)

A integração via AIDL permite uma comunicação direta com o serviço de impressão. Os arquivos necessários estão disponíveis no link compartilhado pelo **Portal do Desenvolvedor** em:

- **/Impressora/Printer_Api_v4.pdf** - Documentação mais recente da impressora
- **/Impressora/AIDL** - Os arquivos AIDL e um exemplo de implementação do service connection e uso das funções (PrinterApp.zip)

Documentacao google sobre AIDL https://developer.android.com/guide/components/aidl

Android Library (AAR e JAR)

Oferecemos também uma biblioteca (AAR e JAR) que encapsula a implementação do AIDL, simplificando o processo de integração. Essa biblioteca disponibiliza as funções de impressão para serem usadas diretamente, como implementado no MainActivity, mas sem a necessidade de codificar a parte de binding.

- **/Impressora/AndroidLibrary** - AAR e JAR encapsulando a implementação do AIDL (equivalente ao código do PrinterManager do PrinterApp), disponibilizando as funções de impressão.

Integração com Delphi

Caso você utilize Delphi, também oferecemos suporte para integração com essa plataforma.

- **/Impressora/Deplhi** - JAR e Bridge file (.pas), para ser usado diretamente em Delphi.

Documentação Disponível

Antes de começar a integração, é recomendado revisar a documentação relevante:

- **Printer API** - Nesta documentação, você encontrará detalhes sobre as funções disponíveis para impressão.
- **AIDL** - Aqui, você encontrará os arquivos AIDL e um exemplo de implementação do service connection.
- **Android Library (AAR e JAR)** - Nesta pasta, você poderá encontrar a biblioteca que encapsula o AIDL para uso mais simples.

Instruções de Instalação

AIDL

Para utilizar a integração via AIDL, siga estes passos:

1. Baixe o arquivo **PrinterApp.zip** e extraia o conteúdo.
2. Copie os arquivos AIDL para o diretório do seu projeto Android.
3. Implemente a conexão do serviço usando o exemplo fornecido no arquivo `PrinterMananger.java`.

```
import br.com.positivo.printermodule.Printer;
import br.com.positivo.printermodule.PrinterCallback;

public class MainActivity extends Activity implements View.OnClickListener {
    private Context mContext; // Contexto da atividade
    private PrinterManagerListener mListener; // Listener para gerenciamento da impressora
    private IPrinterCallback mCallback = null; // Callback para manipular eventos da impressora
    private IPrinterService mPrinterService; // Serviço da impressora

    // ServiceConnection para lidar com a conexão e desconexão do serviço da impressora
    private ServiceConnection mConnectionService = new ServiceConnection() {
        @Override
        public void onServiceDisconnected(ComponentName name) {
            Log.d(TAG, "Service Disconnected");
            mPrinterService = null;
        }

        @Override
        public void onServiceConnected(ComponentName name, IBinder service) {
            mPrinterService = IPrinterService.Stub.asInterface(service);
            mListener.onServiceConnected(); // Notifica o ouvinte que o serviço está conectado
            Log.d(TAG, "Service Connected");
        }
    };
```

```
    public void onPrinterStart() {
        mCallback = new IPrinterCallback.Stub() {
            @Override
            public void onException(int code, final String msg) throws RemoteException {
                Log.w(TAG, "onException(" + code + "," + msg + ")");
            }

            @Override
            public void onLength(long current, long total) throws RemoteException {
                currentLen = current;
                totalLen = total;
            }

            public void onComplete() {
                Log.i(TAG, "onComplete()");
            }

            @Override
            public void onRealLength(double realCurrent, double realTotal) throws RemoteException {
                realCurrentLen = realCurrent;
                realTotalLen = realTotal;
                Log.i(TAG, "realCurrent=" + realCurrent + ", realTotal=" + realTotal);
                android.util.Log.i(TAG,"End time: " + new Date().getTime());
            }
        };

        Intent intent = new Intent();
        intent.setPackage("com.xcheng.printerservice");
        intent.setAction("com.xcheng.printerservice.IPrinterService");
        mContext.startService(intent);
        mContext.bindService(intent, mConnectionService, Context.BIND_AUTO_CREATE);
        Log.d("PrinterSample", "onPrinterStarted");
    }
}
```

4. Lembre-se de sempre iniciar o serviço antes de chamá-lo e com tempo hábil para que ele possa ser iniciado.

```
public class MainActivity extends Activity implements View.OnClickListener,PrinterManager.PrinterManagerListener {
    public static final String TAG = "PrinterSample";
    private PrinterManager mPrinterManager;
    private Context mContext;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mContext = this.getApplicationContext();

        initLayout();
        updateLayout(false);

        mPrinterManager = new PrinterManager(this,this);
        mPrinterManager.onPrinterStart();
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        mPrinterManager.onPrinterStop();

    }

    @Override
    public void onServiceConnected() {
        mPrinterManager.printerInit();          //iniciando o servico
```

```
    }
}
```

Android Library (AAR e JAR)

Para utilizar a biblioteca Android, siga estes passos:

1. Baixe a biblioteca **printer-library.aar** e adicione-a ao seu projeto Android como uma dependência.
2. Agora você pode usar as funções de impressão diretamente no seu código, sem se preocupar com a comunicação com o serviço.

```
import br.com.positivo.printermodule.Printer;
import br.com.positivo.printermodule.PrinterCallback;

public class MainActivity extends Activity implements View.OnClickListener {
    public Printer mPrinter; // Instância da classe Printer utilizada para imprimir
    private Context mContext; // Contexto da atividade

    // Callback para lidar com eventos de impressão
    private PrinterCallback mCallback = new PrinterCallback() {
        @Override
        public void onError(int i, String s) {
            // Manipula erros durante o processo de impressão, se necessário
        }

        @Override
        public void onRealLength(double v, double v1) {
            // Manipula o comprimento real da impressão, se necessário
        }

        @Override
        public void onComplete() {
            // Manipula a conclusão da impressão, se necessário
        }
    };

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mContext = this.getApplicationContext(); // Obtém o contexto da atividade
        mPrinter = new Printer(mContext); // Inicializa a instância da classe Printer
    }

    // Método para manipular o clique no botão de impressão
    private void onPositivoReceiptClicked() {
        final Bitmap positivo_title = Bitmap.createScaledBitmap(BitmapFactory.decodeResource(
                mContext.getResources(),
                R.drawable.positivo), 265, 54, false);

        Executors.newSingleThreadExecutor().execute(new Runnable() {
            @Override
            public void run() {
                try {
                    if (!mPrinter.isReady()) {
                        waitForPrinter(); // Aguarda até que a impressora esteja pronta
                    }

                    mPrinter.printBitmap(positivo_title, mCallback); // Imprime a imagem com o callback
                } catch (Exception e) {
                    // Lidar com exceções, se ocorrerem durante a impressão
                    e.printStackTrace();
                }
            }
```

```
        });
    }

    // Método para aguardar até que a impressora esteja pronta
    private void waitForPrinter() {
        synchronized (this) {
            try {
                Log.d(TAG, "Print: waitForPrinter");
                wait(1000); // Aguarda 1 segundo (pode ser ajustado)
            } catch (InterruptedException e) {
                Log.d(TAG, "Print: Error waiting for initialization", e);
                e.printStackTrace();
            }
        }
    }
}
```

Integração com Delphi

Para integrar com Delphi, siga estes passos:

1. Baixe os arquivos **printer.jar** e **PrinterBridge.pas**.
2. Adicione o arquivo .jar à biblioteca do seu projeto Delphi.
3. Use o arquivo .pas como bridge para acessar as funções de impressão do serviço.

```
printer := TJprintermodule_Printer.JavaClass.init(TAndroidHelper.Context.getApplicationContext);
```

Uso do Bound Service

O serviço de impressão da L400 é um Bound Service:

https://developer.android.com/guide/components/bound-services

Significa que é necessário vincular seu aplicativo a ele através do AIDL. A inicialização do serviço é assíncrona, portanto, é importante criar a instância do Printer em um momento adequado e realizar a impressão em outro.

Dicas de Implementação

Para utilizar o serviço de impressão:

1. Crie a instância do Printer na inicialização do componente (por exemplo, no método onCreate).

```
// MainActivity.java -> onCreate()
@Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mContext = this.getApplicationContext();

        mPrinterManager = new PrinterManager(this,this);
        mPrinterManager.onPrinterStart();
    }
```

1. Realize a impressão (por exemplo, utilizando a função `printText`) no momento esperado (por exemplo, em um botão ou evento).

```
private void onQRcodeClicked(){
        ThreadPoolManager.getInstance().executeTask(new Runnable() {
            @Override
            public void run() {
                try {
                    //PrinterManager lida com as callbacks e com as chamadas
                    mPrinterManager.printQRCode("www.positivotecnologia.com.br");
                    mPrinterManager.printText("\n");
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

@Override
public void onClick(View v) {
        int id = v.getId();
        switch(id){
            case R.id.id_printer_qrcode:
                  onQRcodeClicked();
                  break;
        }
  }
```

Exemplos de Uso

Aqui estão alguns exemplos básicos de uso do serviço de impressão:

Kotlin

```
private val mConnectionService: ServiceConnection = object : ServiceConnection {
        override fun onServiceDisconnected(name: ComponentName) {
            mPrinterService = null
        }

        override fun onServiceConnected(name: ComponentName, service: IBinder) {
            mPrinterService = IPrinterService.Stub.asInterface(service)
            printerInit()
        }
    }

override fun print(
        success: () -> Unit,
        warning: (message: String) -> Unit,
        error: (message: String) -> Unit
    ) {
        this.success = success
        this.warning = warning
        this.error = error
        context = Application.context
        onPrinterStart()
    }

    private fun onPrinterStart() {
        val intent = Intent()
        intent.setPackage("com.xcheng.printerservice")
        intent.action = "com.xcheng.printerservice.IPrinterService"
        context?.startService(intent)
        context?.bindService(intent, mConnectionService, Context.BIND_AUTO_CREATE)
    }

    fun printerInit() {
        try {
            mPrinterService?.printerInit(printerInit(object : IPrinterCallback.Stub() {
```

```
                override fun onComplete() {
                    val bitmap1 = super.bitmapToPrint(600)
                    bitmap1.copy(bitmap1.config, false).scaled(380)?.let {
                        printBitmap(it)
                    }
                }

                override fun onException(p0: Int, p1: String?) {
                    error?.invoke(p1 ?: "Tente novamente")
                }

                override fun onLength(p0: Long, p1: Long) {}

                override fun onRealLength(p0: Double, p1: Double) {}
            }))
        } catch (e: Exception) {
            e.printStackTrace()
            FirebaseCrashlytics.getInstance().recordException(e)
        }
    }

    private fun onPrinterStop() {
        try {
            context?.unbindService(mConnectionService)
        } catch (e: Exception) {
            e.printStackTrace()
            FirebaseCrashlytics.getInstance().recordException(e)
        }
    }

    private fun printBitmap(bitmap: Bitmap?) {
        try {
            mPrinterService?.printBitmap(bitmap, this)
        } catch (e: Exception) {
            e.printStackTrace()
            FirebaseCrashlytics.getInstance().recordException(e)
        }
    }

    override fun onLength(p0: Long, p1: Long) { }

    override fun onRealLength(p0: Double, p1: Double) { }

    override fun asBinder(): IBinder {
        return Binder()
    }

    override fun onException(p0: Int, p1: String?) {
        error?.invoke(p1 ?: "Tente novamente")
    }

    override fun onComplete() {
        onPrinterStop()
        success?.invoke()
    }
```

FAQ (Perguntas Frequentes)

O código de exemplo fornecido utiliza diretamente a interface AIDL. A interface AIDL ainda precisa ser encapsulada como uma interface Delphi?

💡 Não, você pode fornecer diretamente, pois o PrinterModule_1.1.0 ja faz o encapsulamento.

A interface PrintWrapPaper é capaz de imprimir novas linhas? O mesmo que ('\n')?

💡 Nao, o **PrintWrapPaper** avança o papel de acordo com o valor passado por parâmetro e mostra a linha de corte.

OnComplete significa que a tarefa atual é impressa?

💡 Quando toda a impressão e completada, se você usa mais de uma chamada ele imprime após finalizar todas.

“Espaço entre linhas” `KEY_LINESPACE` refere-se à distância entre linhas ou à altura da linha do valor impresso?

💡 Altura da linha

Preciso chamar o init (iniciar a impressora) toda vez que for imprimir alguma coisa?

💡 Não, init só precisa ser chamado uma vez e após o bound com o service.
