JOÃO GABRIEL – DESENVOLVIMENTO DE ANDROID

Stone Pagamentos | Rua da Quitanda, 199 10º - Rio de Janeiro - RJ

Manual de integração

com Android



2014

Índice

[1 CONCEITOS BÁSICOS 3](#_Toc402979453)

[1.1 Informações de básicas referentes à integração 3](#_Toc402979454)

[1.2 Como Integrar com a Stone 3](#_Toc402979455)

[2 Como enviar uma transação? 3](#_Toc402979456)

[2.1 Simples e Rápido 3](#_Toc402979457)

[2.2 Entendendo o Método 5](#_Toc402979458)

[2.3 Criando conexão com o PINpad 6](#_Toc402979459)

[2.4 Retorno da Transação 8](#_Toc402979460)

[2.5 ReturnOfTransactionXML 11](#_Toc402979461)

[3 CANCELAMENTO 11](#_Toc402979462)

# CONCEITOS BÁSICOS

## Informações de básicas referentes à integração

* A integração requer android 2.3 ou superior;
* Um tablet ou celular com bluetooth;
* Utilizar as funções disponíveis no .jar de integração;
* Essa documentação foi criada dando como exemplos a aplicação ‘**StoneSDK**’, desenvolvido pela Stone.
* Essa SDK usa de suporte o jar XStream na versão 1.4.7 (**xstream-1.4.7.jar**).

## Como Integrar com a Stone

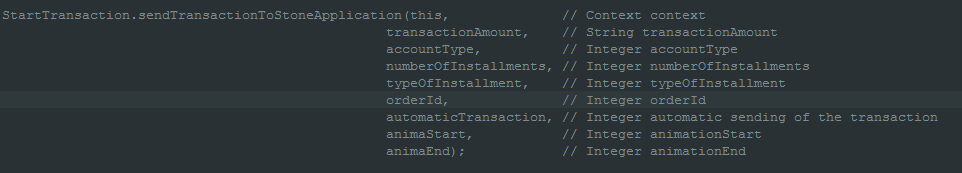
* Certifique-se que o .jar está na pasta ‘libs’ do seu projeto, como na imagem ao lado:
* Faça o import do jar da XStream;
* Após incluir o .jar de integração na pasta ‘libs’, você poderá iniciar a integração.

# Como enviar uma transação?

## Simples e Rápido (obsoleta)

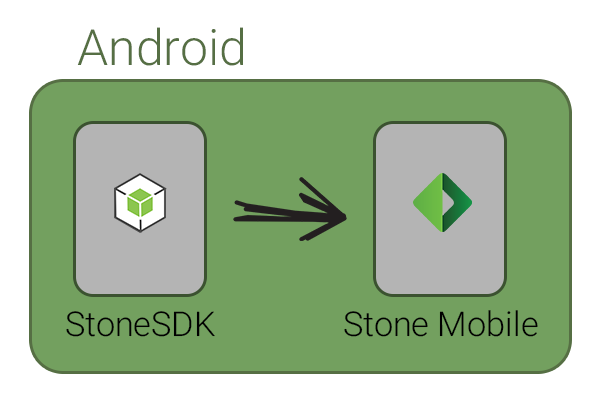
* Vamos entender como funciona a integração da Stone:

A primeira coisa é adicionar essa linha de código a aplicação



**Obs: Ao lado de cada método ou parâmetro estão comentados os tipos dos valores que estão sendo passados.**

* Entendendo o que acontece:



A seta representa o envio de valores passada na linha de código da imagem a cima.

O usuário terá uma transação de interface, sendo levado da sua aplicação, para a aplicação da Stone.

Veja o exemplo:



## Entendendo o Método (obsoleto)

O método ‘sendTransactionToStoneApplication()’ espera receber alguns valores como parâmetro,

sendo eles:

* Contexto da aplicação
* Valor da transação
* Tipo da compra
* Tipo de parcela
* Número de parcelas (se houve parcerlamento)
* Um identificador da transação
* Auto Envio
* Animações

Contexto da aplicação:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| Activity | getAplicationContext() | Contexto da aplicação |
| Fragment | getActivity. getAplicationContext() | Contexto da aplicação |

Valor da transação:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| String | **(seuEditText)**.getText().toString() | O valor deve ser passado em centavos |

Tipo da compra:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| Integer | 1 | Débito |
| Integer | 2 | Crédito |

Tipo da parcela *(Opcional)*:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| Integer | 0 | À VISTA |
| Integer | 1 | LOJISTA |
| Integer | 2 | EMISSOR |

Número de parcelas *(Opcional)*:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| Integer | De 2 à 12 | Apenas valores entre **2** e **12** |

Identificador da transação:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| Integer | n números | Identificador para o arquivo de conciliação |

## 

Auto envio:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| Integer | 1 | Para o usuário confirmar a transação |
| Integer | 2 | Enviar transação automáticamente |

Animações:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| Integer | R.anim.suaAnimação (Start) | Para o usuário confirmar a transação |
| Integer | R.anim.suaAnimação (End) | Enviar transação automáticamente |

O método ‘startNewTransaction()’ espera receber alguns valores como parâmetro,

sendo eles:

* Activity
* Transaction
* Animação incial
* Animação final

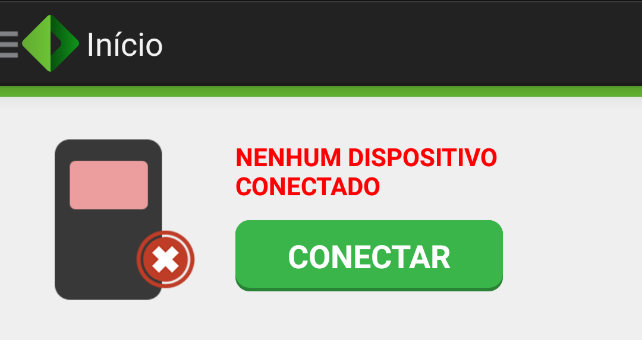
Contexto da aplicação:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| Activity | getAplicationContext() | Contexto da aplicação |
| Fragment | getActivity. getAplicationContext() | Contexto da aplicação |

Transaction (objeto):

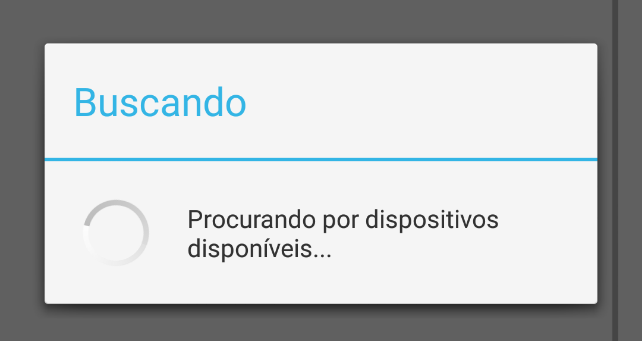
|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| String | setAmount() | Valor da transação |
| Integer | setTypeOfPurchase() | Tipo da compra (1- débito ou 2 - crédito ) |
| Integer | setNumberOfInstalments() | Número de parcelas da transação |
| Integer | setTypeOfInstalment() | Tipo de parcela (0 - à vista ou 1 - parcelamento) |
| Integer | setDemandId() | Uma numeração para a transação |
| boolean | setNeededConfirm() | Confirmação das informações pelo usuário |

## Criando conexão com o PINpad

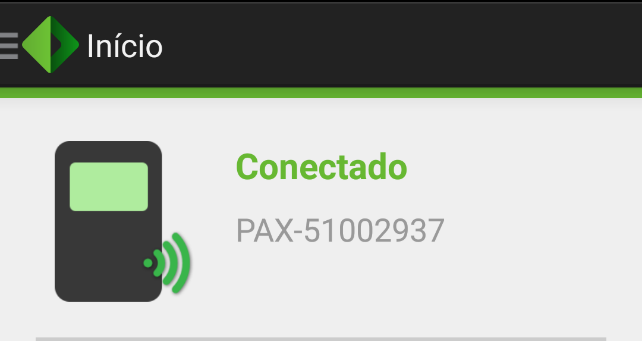


O aplicativo está encarregado de avisar quando há ou não uma conexão com algum PINpad.

Se o usuário não estiver conectado com nenhum dispositivo, o aplicativo exibirá um botão para buscar e conectar com algum dispositivo. Como na imagem ao lado.



Ao tocar em ‘**CONECTAR**’, será realizada uma busca e o usuário deverá selecionar em um dos dispositivos da lista para conectar.



Logo após a conexão for estabelecida, a tela principal será atualizada da seguinte forma:



A conexão com o PINpad pode ser criada automaticamente após a primeira conexão, para essa função ser habilitada, é necessário ir em **CONFIGURAÇÕES**, localizadana aba lateral e selecionar a opção de ‘Conexão automática’.

Quando esta opção está marcada, um serviço é ativado. Esse serviço é responsável por verificar a conexão com o PINpad a cada 5 minutos, se nenhum PINpad estiver conectado com o dispositivo, o mesmo tentará criar conexão com o último PINpad conectado com o dispositivo.

**OBS**: Lembrando que se o aplicativo da Stone for fechado, a conexão será interrompida por alguns segundos, e se a opção de ‘Conexão automática’ estiver marcada, dentro de alguns segundos e conexão irá o serviço será ativado e irá conectar com o PINpad.

## Retorno da Transação

Quando o autorizador da Stone responder a transação, isso quer dizer se ela foi aprovada ou negada, o aplicativo da Stone irá retornar para o aplicativo que chamou o mesmo e enviará algumas informações que foram recebidas do autorizador.

O StoneSDK possui um método que fica responsável por verificar se há ou não um retorno de transação, o *getExternalInformations()*, é aconselhável que ele fique no método “*onResume()*” da *main* do seu projeto.

O aplicativo da Stone retornará um XML contendo as informações da transação.

A captura da resposta, dentro desse método, é feita da seguinte forma:



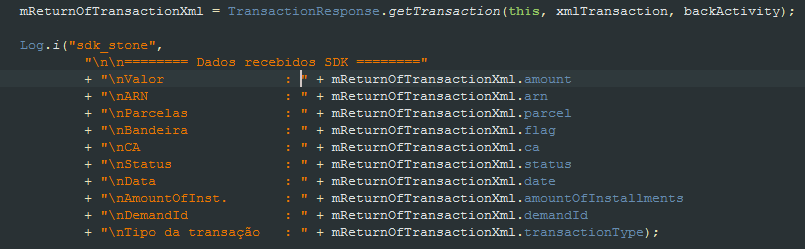
A variável ‘*backActivity*’ recebe as informações de uma *Intent* e verifica se é nula.



Se a variável não for nula, é instanciada uma variável do tipo *ReturnOfTransactionXml,* que é uma classe responsável por serializar e desserializar uma transação e tornar a mesma um objeto.

Se ela for diferente de ‘**null**’, o objeto será desserealizado, sendo assim, tornando-se manipulável através da variável ‘*backActivity*’.



Um exemplo de manipulação dessa variável pode ser visualizada na seguinte imagem:

**OBS**: repare na seguinte linha:

“ mReturnOfTransactionXml = TransactionResponse.*getTransaction*(**this**,xmlTransaction,backActivity); ”

A classe **TransactionResponse** está no sdkstoneapplication.jar, e o método ‘**getTransaction**’ deve receber como parâmetro:

* Contexto da aplicação
* XML da transação
* Bundle

Da seguinte forma:

Contexto da aplicação:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| Activity | getAplicationContext() | Contexto da aplicação |
| Fragment | getActivity. getAplicationContext() | Contexto da aplicação |

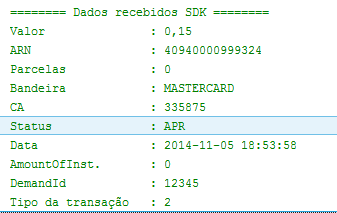
XML:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| String | String de retorno | A String ‘xml’ recebe uma String contida na variável ‘backActivity’. A mesma deve ser passada, apenas. |

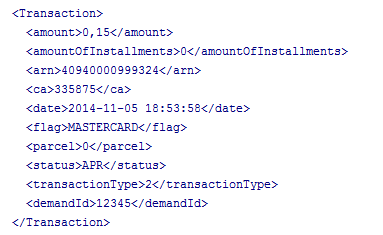
Bundle:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| Bundle | backActivity | O retorno da Intent |

O método ‘**getTransaction**’ retorna um objeto do tipo *ReturnOfTransaction*.



Após ele ser executado, no LogCat poderá ser visto a transação da seguinte forma:



Ao lado, um exemplo de XML, no caso, o que foi passado para o método ‘**getTransaction**’.

## ReturnOfTransactionXML

|  |  |  |
| --- | --- | --- |
| Tipo | Variável | Valor |
| String | Flag | MASTERCARD ou VISA |
| String | Arn | Numeração da transação |
| String | Amount | Valor da transação |
| String | Date | Formato: YYYY-MM-dd HH:mm:ss |
| String | CA | Código de autorização |
| String | Parcel | Número de parcelas |
| String | Status | APR ou DEC |
| String | TransactionType | 1 – débito | 2 – crédito |
| Long | DemandID | Número do ID do pedido |

Esta classe possui as variáveis dos seguintestipos:

**OBS**: Todos os valores que são passados, são retornados para futuras comparações, caso isso seja importante para o desenvolvedor.

# CANCELAMENTO

## Como fazer um Cancelamento

A partir dos valores que são passados de retorno após o envio de uma transação, é possível fazer o cancelamento de uma transação, se a mesma ainda permanecer no banco de dados do aplicativo da Stone.

Para fazer o cancelamento de uma transação, serão necessários:

* Código de autorização da transação
* Numeração da transação
* Contexto da aplicação

Código de autorização da transação :

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| String | ARN | Numeração da transação |
| String | CA | Código da transação |
| Context | getApplicationContext() | Contexto da aplicação |

O envio do cancelamento é muito parecido com o do envio de uma transação. Como pode ser visto na seguinte imagem:

## Resposta do Cancelamento

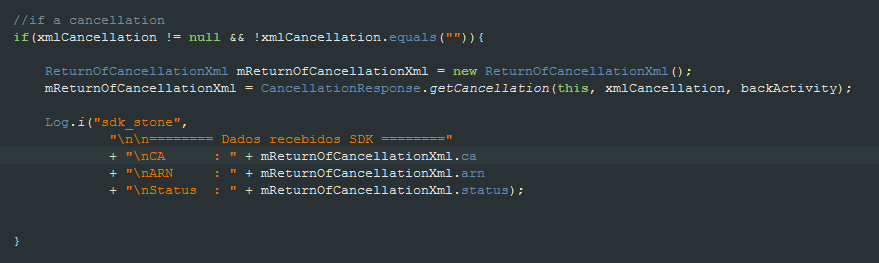
A resposta do cancelamento também é bastante similar à resposta da transação:

A String ‘xmlCancellation’ é a responsável por pegar a resposta da Intent do cancelamento.

Uma verificação se ela é numa ou vazia, se não for, exibir no LogCat as informações.



Exibindo as informações no LogCat:



A resposta da requisição de cancelamento possui como resposta:

* Código de autorização da transação
* Numeração da transação
* Status

Respostas do cancelamento:

|  |  |  |
| --- | --- | --- |
| Tipo | Passagem | Descrição |
| String | ARN | Numeração da transação |
| String | CA | Código da transação |
| String | Status | Approved – Cancelamento realizado com sucesso;  Declined – Cancelamento negado;  PartialApproved – Parcialmente aprovada;  TchenicalError – Erro técnico no proxy de cancelamento; |

**OBS**: Por padrão, a senha de cancelamento é “1234”, podendo ser alterada nas **CONFIGURAÇÕES** da aplicação.

# IMPRESSÃO

## Como imprimir

As impressoras dos Pinpads também podem ser utilizadas pela API de integração, para isso existe a classe StartPrint.

Na classe StartPrint, estão disponíveis três métodos:

• validateListSize() – responsável por validar o tamanho de cada linha e tamanho dos caractéres, da seguinte forma:

|  |  |  |
| --- | --- | --- |
| Tipo | Tamanho | Quantidade |
| Int | Small | Máximo de 48 caractéres |
| Int | Medium | Máximo de 48 caractéres |
| Int | Big | Máximo de 24 caractéres |

• putSpace() – responsável por adicionar quantidades de espaços no final da lista

• sendPrint() - responsável por enviar a lista de impressão para o app da Stone

Para imprimir, basta criar uma lista de PrintObject, esse objeto representa cada linha que será impressa, ele tem como propriedades:

|  |  |  |
| --- | --- | --- |
| Tipo | Nome | Conteúdo |
| String | Message | Conteúdo a ser impresso |
| Int | Size | SMALL, MEDIUM, BIG |
| Int | Align | LEFT, CENTER, RIGHT |
| String | TAG | Máximo de 512 caractéres |

**OBS: Ao ser passado “TAG” como tamanho ou alinhamento, será impresso um QR Code com o conteúdo que foi passado.**

## Resposta da impressão

Os commandos de impressão possuem uma resposta que devem ser esperadas junto às demais respostas na Main da aplicação.

Ex:

String xmlPrint = (seu bundle).getString(“xmlPrint”);

Essa String deve ser passada para o método PrintResponse.getPrint();

Ex:

ReturnOfPrintXml returnOfPrintXml = **new** ReturnOfPrintXml();

returnOfPrintXml = PrintResponse.*getPrint*(**this**, xmlPrint, backActivity);

Desta forma você terá um objeto do tipo ReturnOfPrintXml e poderá saber se a impressão foi concluída com sucesso pela propriedade printCode:

|  |  |  |
| --- | --- | --- |
| Tipo | Resposta | Significado |
| int | 1 | Impresso com sucesso |
| Int | 2 | Ocorreu um erro durante a impressão |
| Int | 3 | Pinpad conectado não possui impressoda (Ex: D200) |