Grupos do Google

Tutorial para Geração de Boletos Bancoob / Sicoob

Giorgio Torres 22/01/2014 11:17

Postado no grupo: **JRimum Community**

Caros, bom dia.

Finalizei a homologação dos boletos gerados pela biblioteca Bopepo da JRimum (daqui pra frente chamarei apenas de api) no Sicoob.

Segue abaixo uns workarounds que tive que fazer para sair de acordo com a especificação.
Em anexo segue os documentos de layout para elaboração de boleto e arquivos remessa/retorno do banco Sicoob.

Vamos lá.

Problema 1) O primeiro problema que tive foi com relação à modalidade da carteira que aparecia na linha digitável.

Não sou especialista no assunto, mas pelo que li na internet, a modalidade da carteira deveria estar relacionada ao tipo de cobrança: Cobrança sem registro: Modalidade 2;
Cobrança com registro: Modalidade 1. <Quem souber mais sobre o assunto favor comentar sobre isso>
No entanto essa ligação não acontece na api. Mesmo "setando" a modalidade da conta corrente diretamente

```
ContaBancaria conta = new
ContaBancaria(BancosSuportados.BANCOOB.create());
conta.setModalidade(new Modalidade("2"));
```

não afeta a saída na linha digitável do boleto. Ou seja, o trecho acima não faz nada além de gastar memória.
Na verdade, acho que a equipe da JRimum poderia dar uma olhada nos dados da classe Carteira. Tenho um leve pressentimento de que há dados redundantes ou desnecessários.

Solução:

Procurando aqui na comunidade encontrei a solução no seguinte post: https://groups.google.com/d/msg/jrimum-community/l2hdl62nMMs/HZ5EVw-Xp2kJ
Depois de criado o título bancário para a conta bancária previamente instanciada, é necessário sobrescrever a informação da modalidade através dos parâmetros bancários: Segue o código abaixo:

```
titulo.setParametrosBancarios(new
ParametrosBancariosMap("ModalidadeDeCobranca", 2));
```

Isso foi suficiente para resolver o problema da modalidade errada que aparecia na linha digitável.

Problema 2) Uma das coisas que a api nos confunde é com relação à criação do objeto ContaBancaria, onde temos que especificar, dentre outras coisas, o número da conta do cedente.

Na api existe o método "ContaBancaria::setNumeroDaConta", onde, intuitivamente, coloquei o número da conta bancária.
Na verdade, para o banco Sicoob, temos que colocar nesse método o Código do Cedente. Esse código não é o número da conta bancária, mas sim um outro número que o banco disponibiliza para identificação do cliente (cedente).

Solução:

Ligar no banco (agência) e pedir o CÓDIGO do cliente (falar o nome do cliente ou o número da conta corrente).
Esse código é parecido com o número da conta: existe o número e um DV.
Por causa de um outro problema que tive, colocar o código inteiro INCLUSIVE com o DV.
Mais pra frente falarei o porquê.

Problema 3) Geração do Nosso Número:

O Sicoob deixa um espaço de 7 dígitos para o nosso número e 1 dígito para o DV do nosso número.
Na classe Titulo da api, tem método para setar o nosso número e outro para setar o DV. Intuitivamente
coloquei o nosso número de 7 dígitos no método setNossoNumero e o DV no método setDigitoDoNossoNumero.
Porém o método setNossoNumero da api espera um código de 8 dígitos e lança uma excessão caso é passado alguma
String de tamanho diferente (tanto maior que 8 quanto menor que 8).

Solução:

No método setNossoNumero colocar o nosso número gerado pelo seu sistema com 7 dígitos (para o Sicoob) e concatenar o dv junto ao número.
Acabei deixando o DV setado também no método setDigitoNossoNumero, porém não sei se ele é usado internamente na geração do boleto.
Código:

```
// nossoNumero.length() == 7
// nossoNumeroDV.length() == 1
titulo.setNossoNumero(nossoNumero+nossoNumeroDV);
titulo.setDigitoDoNossoNumero(nossoNumeroDV);
```

Problema 4) Problema na construção do DV do Nosso Número.

A construção do nosso número com o DV do sicoob utiliza o código da agência, o código do cliente, o dígito do código
do cliente e o nosso número gerado pelo sistema com 7 dígitos (colocar 0 zero à esquerda caso o número seja menor).

Solução: Lembrar que o código do cliente no objeto Cedente já está com o DV.

Problema 5) O último problema é corrigir no boleto gerado (impresso em pdf) o resultado dos workarounds feitos anteriormente.

Depois de alterar o código do cedente, o DV vai aparecer duas vezes nos campos do boleto gerado. O nosso número
também irá sair com o DV duplicado.

Solução: Utilizar a sobrescrita dos campos do boleto (http://www.jrimum.org/bopepo/wiki/Componente/Documentacao/Tutoriais/SobrecritaDeCampos)

```
boleto.addTextosExtras("txtRsNossoNumero", nossoNumero+"-
"+nossoNumeroDV);
boleto.addTextosExtras("txtFcNossoNumero", nossoNumero+"-
"+nossoNumeroDV);
boleto.addTextosExtras("txtRsAgenciaCodigoCedente", numAgencia +
"-" + dvAgencia + " / " + codigoCedente + "-" + dvCodigoCedente);
boleto.addTextosExtras("txtFcAgenciaCodigoCedente", numAgencia +
"-" + dvAgencia + " / " + codigoCedente + "-" + dvCodigoCedente);
```

Por fim, gostaria de parabenizar à toda equipe da JRimum. Apesar desses problemas, a api funciona muito bem. Acredito que
esses problemas acabam acontecendo por causa da lógica de negócio de cada banco, e como a api é bastante abrangente

acaba sendo nosso trabalho adequá-la ao uso.

Qualquer dúvida estarei visitando constantemente a comunidade.

Abraço a todos.