Principais modificações ACBr trunk2

O objetivo desse documento é descrever as principais modificações efetuadas no amplo Refactoring que foi promovido no “trunk2”, além de explicar o que motivou essas mudanças e ainda como você pode ajustar os seus fontes, para compilar seus projetos no novo “trunk2”

Objetivos a serem atingidos com o Refactoring

- Permitir configuração de endereços URL através de arquivos .INI
- Compatibilizar ACBrNFe (e derivados) com Lazarus e Linux
- Melhor estrutura dos diretórios
- Reduzir IFDEFs
- Reduzir problemas com UTF8 e acentuação nas diversas IDEs suportadas
- Evitar duplicidade de código
- Código centralizado
- Padronização de rotinas

Tarefas a serem efetuadas no Trunk2

- ACBrInstall
- Testes de compilação em Delphi XE e Delphi 7
- Testes generalizados no ACBrNFe
- Testes nos Exemplos em Delphi
- Migração e testes com alguns modelos de DANFE
- Ver TODO.TXT na pasta Exemplos

Reestruturação dos diretórios

A pasta **Exemplos** foi reestruturada para que os Demos fiquem em subpastas de acordo com o Package de cada componente. Por exemplo, os demos de ACBrCEP, ACBrMail, ACBrConsultaCNPJ estão dentro da pasta: Exemplos\ACBrTCP\, assim como os demos de ACBrSPED e ACBrSintegra, estão dentro da pasta ACBrTXT.

A mesma restruturação, focada em subpastas por Packages, pode ser vista nos diretórios “Fontes” e “Pacotes”

Todos os fontes de componentes e projetos de Terceiros foram migrados para a pasta “Fontes\Terceiros” e em suas respectivas pastas. Isso visa dar maior visibilidade e

crédito aos diversos componentes e projetos, que o ACBr utiliza. A instalação desses componentes ocorrerá de forma transparente, ou seja, não é necessário instalar nenhum desses componentes manualmente, os Packages do ACBr possuem as dependências necessárias para utilização desses fontes.

Todos os componentes que tem como finalidade a geração de Arquivos TXT, foram agrupados na pasta ACBrTXT; são eles: ACBrConvenio115, ACBrLFD, ACBrPAF, ACBrSEF2, ACBrSintegra, ACBrSPED, SintegraSultan.

Os componentes que geram Documentos Fiscais Eletrônicos foram agrupados na pasta ACBrDFe (Dfe = Documento Fiscal Eletrônico); são eles: ACBrNFe, ACBrCTe, ACBrGNRE, ACBrMDFe, ACBrNFSe. Esses componentes sofreram um amplo refactoring, e centenas de linhas de código foram suprimidas e unificadas em um Classe em comum a todos eles... a ACBrDFe.

Hierarquia de dependência dos Packages

- ACBrComum → Synapse
- ACBrDiversos → ACBrComum
- PCNComum → ACBrDiversos
- ACBrOpenSSL → ACBrComum
- ACBrSerial → ACBrDiversos, ACBrOpenSSL
- ACBrTXTComum → ACBrDiversos,
- ACBrConvenio115 → ACBrTXTComum, ACBrOpenSSL
- ACBrLFD → ACBrTXTComum
- ACBrPAF → ACBrTXTComum, ACBrOpenSSL
- ACBrSEF2 → ACBrTXTComum, PCNComum
- ACBrSintegra → ACBrTXTComum
- ACBrSPED → ACBrTXTComum
- ACBrTCP → ACBrDiversos
- ACBrTEFD → ACBrComum
- ACBr_Boleto → ACBrTCP
- ACBr_BoletoFC_Fortes → ACBr_Boleto, fortes324laz
- ACBr_BoletoFC_LazReport → ACBr_Boleto, lazreportpdfexport
- ACBrDFeComum → ACBrOpenSSL, ACBrTCP, PCNComum
- ACBrNFe → ACBrDFeComum
- ACBrCTe → ACBrDFeComum
- ACBrGNRe → ACBrDFeComum

- ACBrMDFe → ACBrDFeComum
- ACBrNFSe → ACBrDFeComum
- ACBr_SAT → PCNComum
- ACBr_SAT_ECFVirtual → ACBr_SAT, ACBrSerial
- ACBr_SAT_Extrato_ESCPOS → ACBr_SAT, ACBrDFeComum, ACBrSerial
- ACBr_SAT_Extrato_Fortes → ACBr_SAT, ACBrDFeComum, fortes324laz

Mudanças em ACBrComum

- Componente **ACBrEAD** removido de ACBrComum. O mesmo foi movido para ACBrOpenSSL.
  Motivo: Esse componente é totalmente dependente das Units do OpenSSL

Mudanças em ACBrUtil.pas

- Revisão de parâmetros e retornos de todos os métodos, usando “String” em vez de “AnsiString”, sempre que possível. Isso evita a conversão excessiva de String para AnsiString nas IDEs modernas como Delphi XE e Lazarus.
- Remoção de Warnings e revisão de métodos que utilizam **TFormatSettings**
- método: **DecodeToSys** renomeado para **DecodeToString**. Esse método foi revisado para retornar uma String compatível com a IDE em uso.
  - Se a String recebida for UTF8:
    - Delphi 7, converte a String para Ansi
    - Delphi XE, converte para UnicodeString (UTF16)
    - Lazarus converte de AnsiString para String
  - Se a String recebida for Ansi
    - No Delphi 7, converte de AnsiString para String
    - No Delphi XE, converte de Ansi para UnicodeString
    - No Lazarus, converte de Ansi para UTF8
- Método: **ParseText** foi refatorado e revisto para suportar diversas IDEs, usando uma chamada a DecodeToString
- **padL** foi renomeada para **PadRight**; **padR** foi renomeada para **PadLeft; padC** foi renomeada para **PadCenter; padS** foi renomeada para **PadSpace.**
  Motivo: Os métodos padL e padR, tiveram a sua nomenclatura errada desde a sua primeira versão (eles fazem o oposto do que o nome deles diz). E a mudança de nome é uma maneira de forçar a revisão das Units que utilizam os mesmos.
  Para corrigir seus fontes, use o recurso da IDE, “Find in Files” que permite localizar e substituir de forma automática todas as ocorrências desses métodos. Use por exemplo: Localizar: “padL(“ Substituir por: “PadRight(“.
- O método **Poem_Zeros** ganhou uma sobrecarga que permite receber um Inteiro

como parâmetro.

- **FloatToString** foi aprimorado: Converte um Double para String, semelhante a FloatToStr(), porém garante que não haverá separador de Milhar e o Separador Decimal será igual a "SeparadorDecimal" informado (o default é .(ponto))
- Novo método: **FormatFloatBr:** Faz o mesmo que FormatFloat, porém garante que o resultado final terá o separador de decimal = ',' e o separador de milhar como Ponto
- Novos métodos: **FormatDateBr** e **FormatDateTimeBr**: Que garantem que o separador de data utilizado no resultado final, será a "/" e o da hora ":"
- Novo método: **FloatMask** que retorna uma máscara baseada no número de decimais informado.
- Os seguintes métodos foram migrados de **ACBrDFeUtil.pas:** EstaVazio, NaoEstaVazio, EstaZerado, NaoEstaZerado, TamanhoIgual, TamanhoMenor
- Adicionado o método **ApplicationPath:** que retorna o Path da aplicação.
- Adicionado os métodos: UnZip(S: TStream) e UnZip(S: AnsiString)
- Método **StoD** refatorado para melhor performance
- Adicionado o método: **TranslateUnprintable** que permite traduzir em arquivos de Log caracteres de controle.
- Método: **IsNumber** renomeado para **VarIsNumber,** para tornar mais clara a finalidade do mesmo.

Mudanças em ACBr.inc

- Removidas as diretivas: ACBrNFeOpenSSL, ACBrCTeOpenSSL, ACBrNFSeOpenSSL, ACBrMDFeOpenSSL, ACBrGNREOpenSSL.
  Motivo: No Trunk2 não mais são necessárias diretivas de compilação para mudar de Capicom para OpenSSL.. Isso poderá ser feito em RunTime, apenas modificando uma propriedade de qualquer um dos componentes ACBrDFe.
- Introduzida a diretiva HAS_FORMATSETTINGS, que será ligada quando a IDE suportar o uso de TformatSettings.
- Introduzida a diretiva USE_libeay32, que será ligada quando a IDE suportar o uso dessa Unit (apenas Delphi)

Mudanças em ACBrBase.pas

- Revisão de métodos, usando "String" em vez de "AnsiString", sempre que possível.

Mudanças em ACBrReg.pas

- Criação de Tipos genéricos que podem ser utilizados em vários dos componentes ACBr. Isso evita ter que redeclarar os mesmos tipos nas Units de registro, de diversos componentes.
    - **TACBrFileProperty**: Utilizado quando o componente precisa de um OpenFileDialog
    - **TACBrFileINIProperty**: Utilizado quando o componente precisa de um OpenFileDialog com filtro para arquivos .INI
    - **TACBrDirProperty**: Utilizado em propriedades de componentes que precisam de um OpenDirDialog