

Manual
Recurso Code Implementation.
FullCopyConvert Data

Revisão: Novembro / 2016

Sumário

Bem-vindo ao FullCopyConvert Data! Estas instruções irão guiá-lo para utilizar o recurso Code Implementation(Implementação de Código).

Sobre o FullCopyConvert Data.....	3
Contato.....	3
1 – Code Implementation - Inicio.	4
2 – Code Implementation – Criando Funções próprias.	6

Sobre o FullCopyConvert Data.

FullCopyConvert é uma ferramenta especializada em conversão e migração de dados, oferece uma forma confortável e fácil de converter suas informações de um banco de dados para outro. Com o FullCopyConvert você poderá converter as informações das seguintes bases de dados disponíveis.



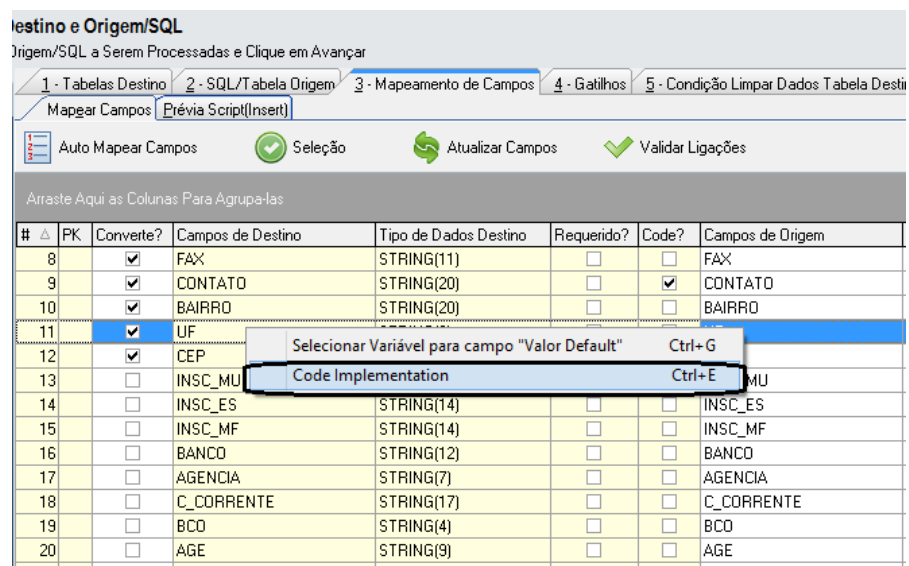
Contato.

O suporte do FullCopyConvert Data é feito através de e-mail. É necessário apenas enviar um e-mail para suporte@fullcopyconvert.com.br relatando o problema ocorrido ou mesmo uma dúvida.

1 – Code Implementation - Inicio.

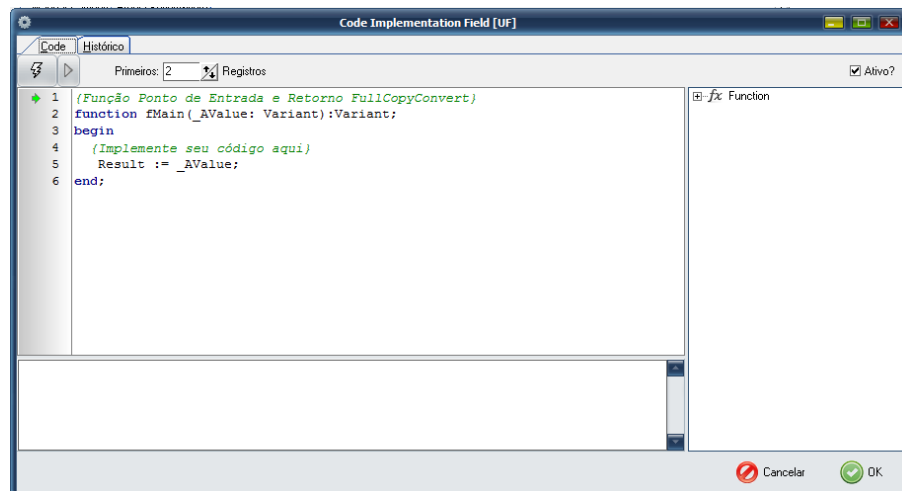
O **FullCopyConvert** conta com um recurso de **implementação de código pascal**. Onde você poderá utilizar funções nativas do FullCopyConvert ou mesmo poderá criar suas próprias funções. Este recurso é bastante útil principalmente nas importações de arquivos **csv, excel e access**, pois conta com pouquíssimas funções para realização de tratamentos dos dados. Mas **nada impede** que esse recurso possa ser **utilizado para outros tipos de bases de dados** que se deseja realizar a migração ou conversão dos dados.

1. Agora vamos ver um pouco na prática como iremos utilizar esse recurso no FullCopyConvert. Abra o FullCopyConvert e **vá até a 4° (quarta etapa)**. Selecione uma tabela que deseja trabalhar e depois de informar os dados da Aba 2 – SQL/Tabela Origem, **clique na aba 3 – Mapeamento de Campos**. Localize o campo que deseja realizar um tratamento e clique com o botão direito do mouse e selecione a opção **“Code Implementation”** ou mesmo pressione **“Ctrl + E”**. Como demonstra imagem abaixo:

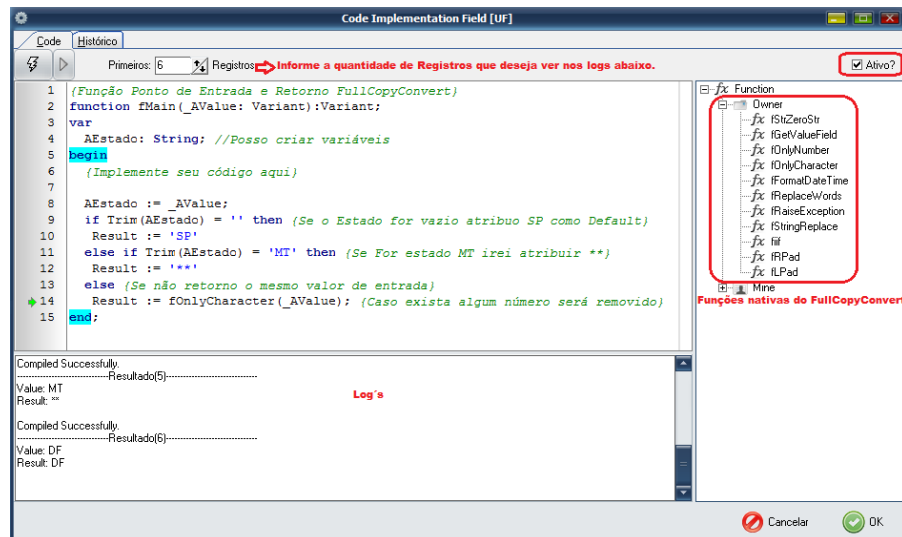


2. Após acessar a opção mencionada anteriormente note que irá abrir a seguinte tela. E já irá retornar uma função chamada **“fMain”**. Essa função é o ponto de entrada de cada coluna que se deseja realizar o tratamento. A função **“fMain” não pode ser alterada e não poderá acrescentar mais parâmetros de entrada** nesta função. A função **“fMain”** serve como um ponto

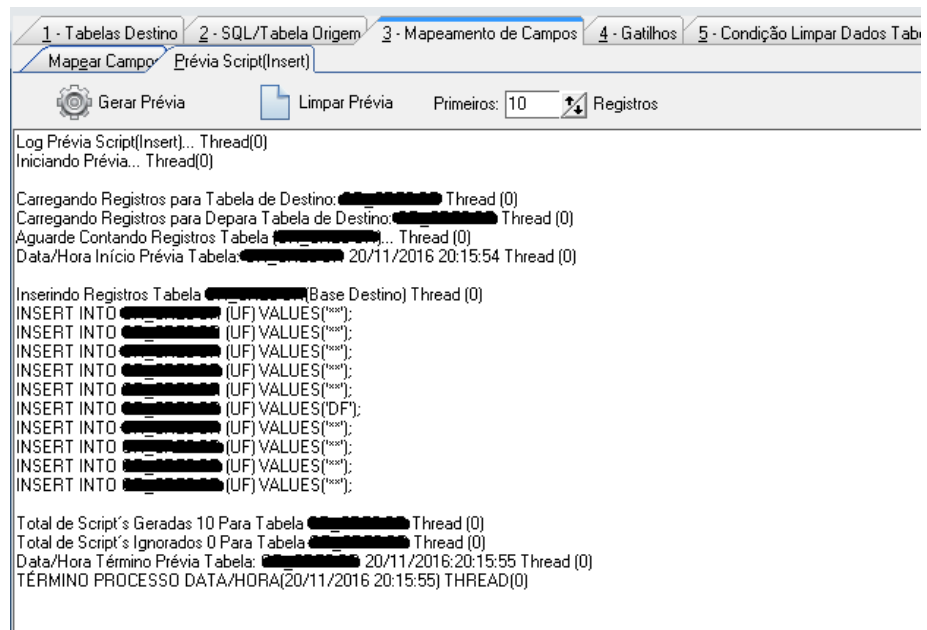
de entrada do FullCopyConvert. A cada passada de registro nesta determinada coluna o FullCopyConvert irá realizar a chamada da função fMain e irá retornar os dados que da própria função gerar. É muito importante que exista o ponto de retorno da função no caso Result. Veja abaixo a função original sem alteração na mesma.



3. Vamos a um pequeno exemplo do que podemos fazer nessa função. **Veja os comentários.**



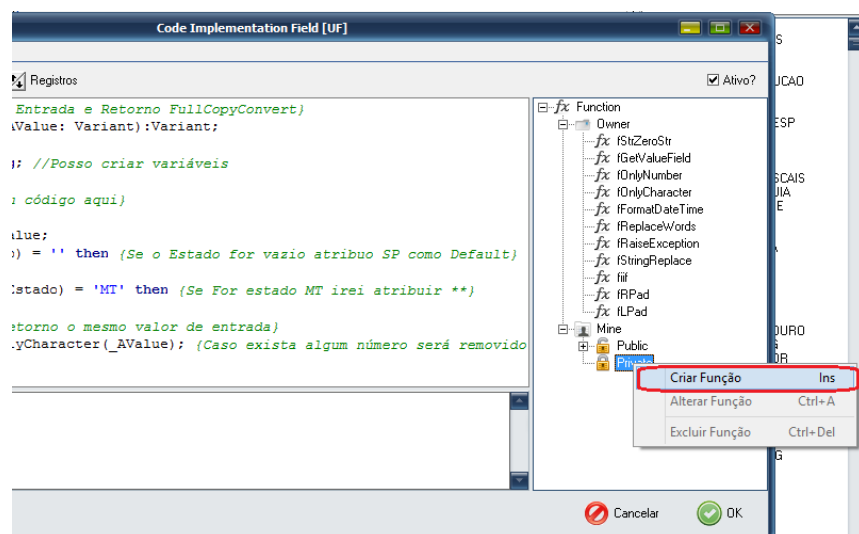
4. No exemplo da tela acima podemos ver que fizemos alguns tratamentos: se o valor for vazio, se for MT ou se não entrou em algumas das condições, iremos remover qualquer número que retornar no valor. Veja um exemplo de como será gerado o script.



2 – Code Implementation – Criando Funções próprias.

Além de utilizar as funções nativas no FullCopyConvert, você poderá criar suas próprias funções. E poderá utilizá-las na chamada da função “fMain”.

1. Agora vamos ver um pouco na prática como iremos utilizar esse recurso no FullCopyConvert. Acesse a opção de “Code Implementatin” como relatado no item 1.1 anteriormente. Expanda o **treeview Function** e irá notar o item “Mine”, que corresponde as funções próprias.

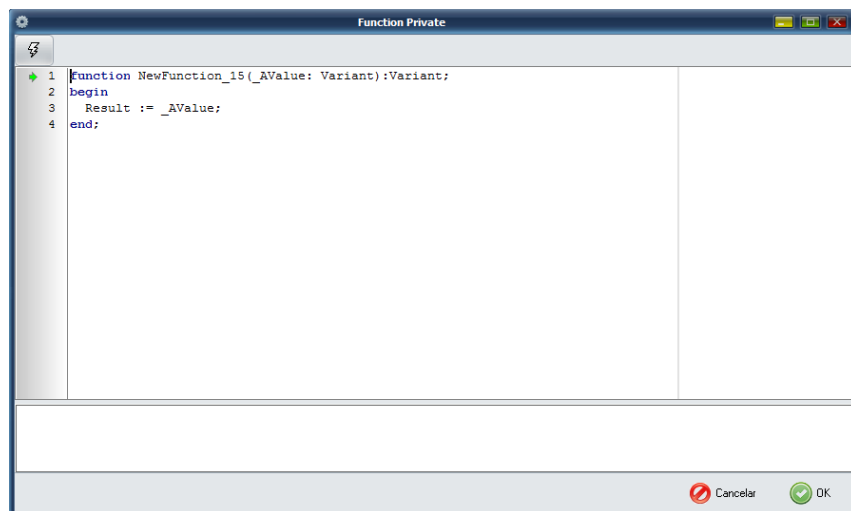


2. Note na imagem acima que você pode escolher em criar uma função **public ou private**.

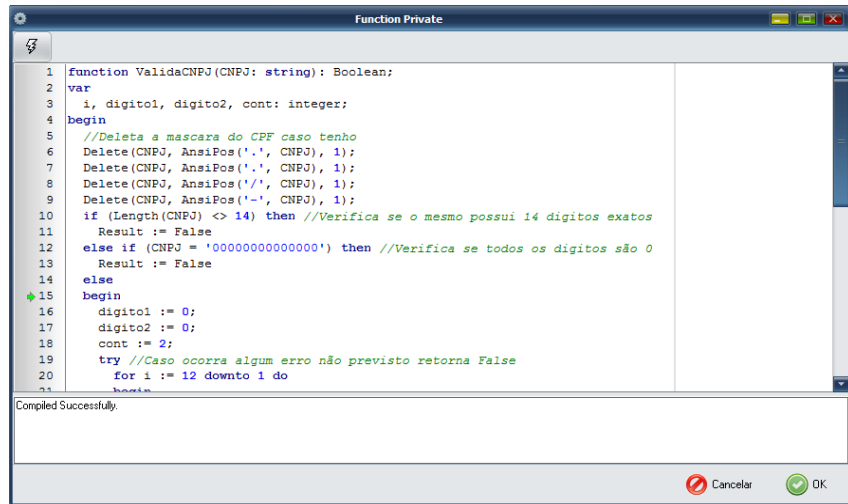
Public – Funções públicas podem ser reaproveitadas em outros projetos de conversão e migração.

Private – Funções privadas podem ser somente utilizadas no projeto de conversão atual.

3. Selecione em qual irá se enquadrar sua função e clique com o botão direito do mouse e selecione a opção **“Criar Função”** Como demonstra a imagem do item 1.

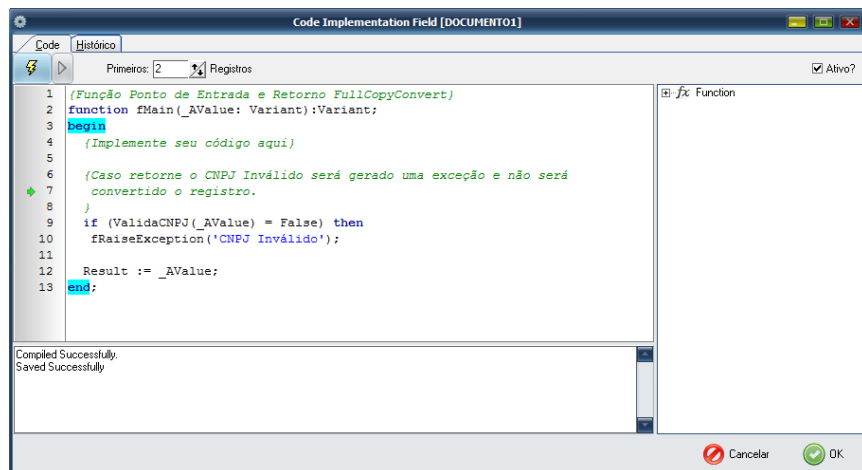


4. Como podemos observar na imagem acima o FullCopyConvert já criou uma estrutura de código, mas nada impede que possa ser alterada. Porém temos que atentar a um detalhe bastante importante: **O nome da função não pode ser iniciado com a letra “f”, pois todas as funções nativas do FullCopyConvert iniciam com a letra “f”.**
5. Segue um exemplo de Função criada mudamos para o nome **ValidaCNPJ**. Note que mudamos o nome da função e mudamos também os parâmetros de entrada e o retorno da função.






```
1 function ValidaCNPJ(CNPJ: string): Boolean;
2 var
3   i, digito1, digito2, cont: integer;
4 begin
5   //Deleta a mascara do CPF caso tenha
6   Delete(CNPJ, AnsiPos('.', CNPJ), 1);
7   Delete(CNPJ, AnsiPos('-', CNPJ), 1);
8   Delete(CNPJ, AnsiPos('/', CNPJ), 1);
9   Delete(CNPJ, AnsiPos('-', CNPJ), 1);
10  if (Length(CNPJ) <> 14) then //Verifica se o mesmo possui 14 digitos exatos
11    Result := False
12  else if (CNPJ = '00000000000000') then //Verifica se todos os digitos são 0
13    Result := False
14  else
15    begin
16      digito1 := 0;
17      digito2 := 0;
18      cont := 2;
19      try //Caso ocorra algum erro não previsto retorna False
20        for i := 12 downto 1 do
21          begin
```

6. Criada nossa função agora podemos utiliza-la na chamada da função principal “fMain”. Segue um print do exemplo da chamada da função.



```
1 (Função Ponto de Entrada e Retorno FullCopyConvert)
2 function fMain(_AValue: Variant):Variant;
3 begin
4   (Implemente seu código aqui)
5
6   (Caso retorne o CNPJ Inválido será gerado uma exceção e não será
7   convertido o registro.
8   )
9   if (ValidaCNPJ(_AValue) = False) then
10    fRaiseException('CNPJ Inválido');
11
12   Result := _AValue;
13 end;
```

7. Agora vamos analisar como será gerado o script. **Observação:** Note que nosso parâmetro de entrada é do tipo **Variant** e nosso parâmetro de entrada da Função ValidaCNPJ é do tipo **String**. Caso retorne Null você deverá converter para String o variant. Fazendo um VarToStr. Exemplo: if (ValidaCNPJ(VarToStr(_AValue))) = False) then. Caso não realize o tratamento poderá gerar a seguinte **exceção:** Could not convert variant of type (Null) into type (OleStr). **Os tratamentos para os tipos correspondente é muito importante para não gerar exceções desnecessárias.**

 Gerar Prévia
  Limpar Prévia
 Primeiros:  Registros

```

Log Prévia Script(Insert)... Thread(0)
Iniciando Prévia... Thread(0)

Carregando Registros para Tabela de Destino: ██████████ Thread (0)
Carregando Registros para De para Tabela de Destino: ██████████ Thread (0)
Aguarde Contando Registros Tabela ██████████... Thread (0)
Data/Hora Início Prévia Tabela: ██████████ 20/11/2016 21:44:32 Thread (0)

Inserindo Registros Tabela ██████████ (Base Destino) Thread (0)
Ocorreu o seguinte erro no Campo Tipo String: CNPJ Inválido
Campo Origem: DOCUMENTO1 Campo Destino: DOCUMENTO1
INSERT INTO ██████████ (UF, DOCUMENTO1) VALUES('xxx');
Ocorreu o seguinte erro no Campo Tipo String: CNPJ Inválido
Campo Origem: DOCUMENTO1 Campo Destino: DOCUMENTO1
INSERT INTO OR_CREDOR (UF, DOCUMENTO1) VALUES('xxx');

Total de Script's Geradas 2 Para Tabela ██████████ Thread (0)
Total de Script's Ignorados 0 Para Tabela ██████████ Thread (0)
Data/Hora Término Prévia Tabela: ██████████ 20/11/2016:21:44:32 Thread (0)
TÉRMINO PROCESSO DATA/HORA(20/11/2016 21:44:32) THREAD(0)
  
```

Note que a mensagem de erro é apresentada. Pois fizemos uma chamada no método `fRaiseException`.

Dúvidas entre em contato.

Questões gerais

info@fullcopyconvert.com.br

Vendas questões relacionadas

registro@fullcopyconvert.com.br

Apoiar

Problemas com o uso de nossos programas ou questões simplesmente técnicas?

suporte@fullcopyconvert.com.br

Requisito	Limite		Limites Trial	Limites Pro
	Versão Trial	Versão Pro		
Conversão de Registros	Sim	Não	1000 Registros Por Tabela. 5 Tabelas por Vez.	Não Há
Conversão Tabela	Sim	Não	5 Por Vez	Não Há
Criação Indices	Sim	Não	5 Por Vez	Não Há
Criação Foreign Key	Sim	Não	5 Por Vez	Não Há
Criação Projetos	Não	Não	Não Há	Não Há
De/Para	Não	Não	Não Há	Não Há
Bulk Insert	Sim	Não	0 Registros	Não Há
Agendamento	Não	Não	1000 Registros Por Tabela. 5 Tabelas por Vez.	Não Há
Gatilhos	Não	Não	Não Há	Não Há
Exportação para Script	Sim	Não	1000 Registros Por Tabela. 5 Tabelas por Vez.	Não Há