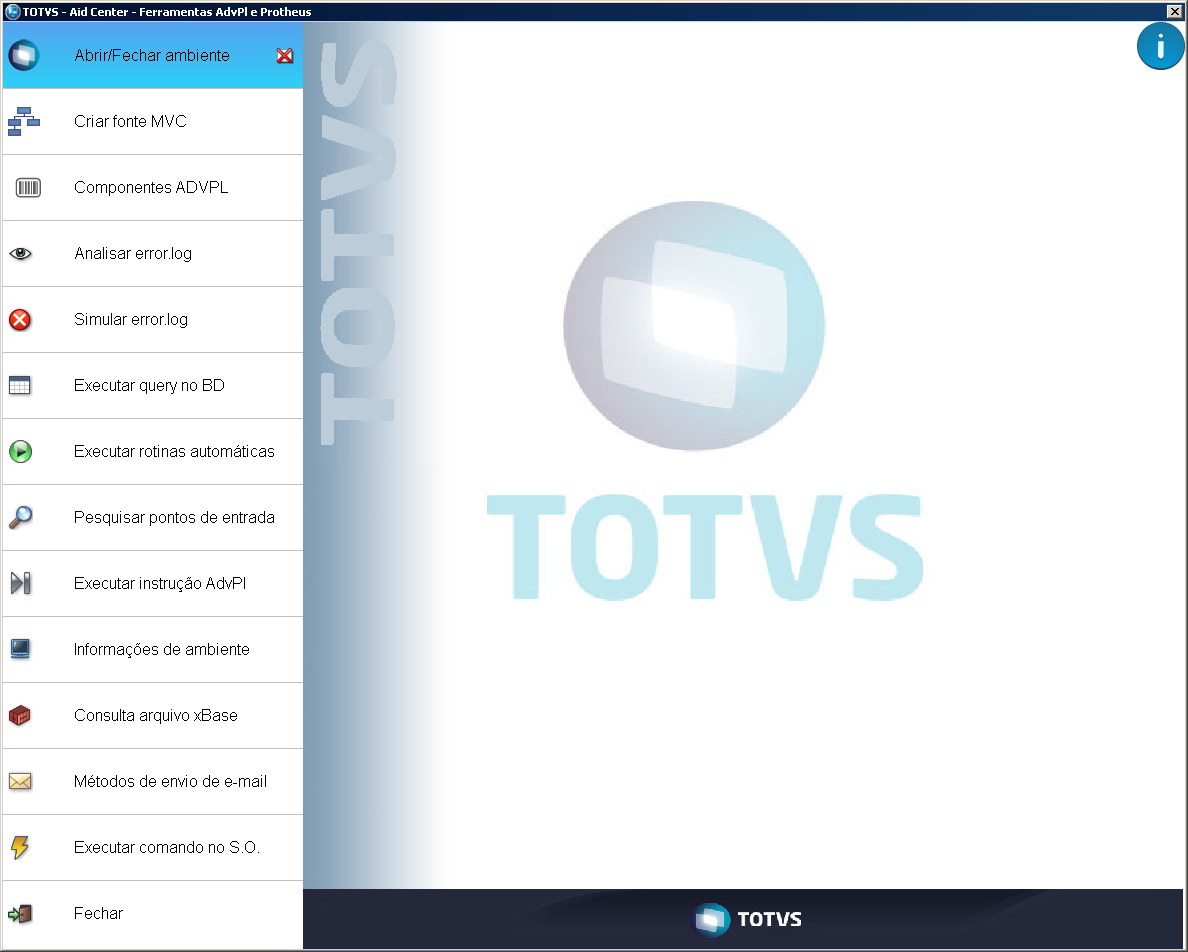
**Aid Center - Menu das utilidades**

Esse RdMake tem por finalidade auxiliar a utilização todas os demais RdMakes por mim feitos, possuindo um menu com a chamada de todos eles e também as opções de abrir o ambiente e executar comando no S.O..

Para utilizar basta chamar a função U\_Dnl\_Util, será exibida a tela de menu de utilidades:

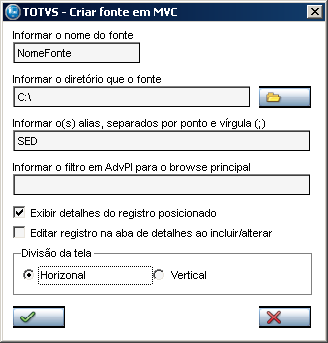


**Criação de fontes em MVC**

Esse RdMake tem por finalidade auxiliar a criação de um fonte MVC do zero, criando assim as estruturas básicas do mesmo:

* MenuDef
* ViewDef
* ModelDef
* BrowseDef

Para utilizar basta chamar a função DnlCrFlMVC, será exibida a tela de geração do fonte:

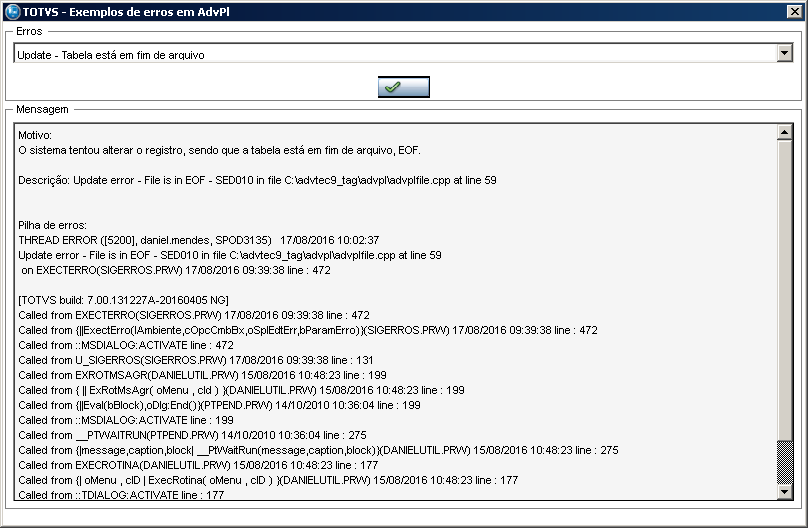


É necessário informar no mínimo o nome do fonte e o alias ( tabela ), sendo que o fonte será gerado no StartPath na ausência de um diretório.

**Simulação de error.log**

Esse RdMake tem por finalidade a simulação de erros do Protheus, apenas para estudo.

Para utilizar basta chamar a função DnlSigErros/DnlEnvSgEr, sendo que a diferença entre elas é que a função DnlEnvSgEr força a inicialização do sistema. Então será exibida a tela de simulação:



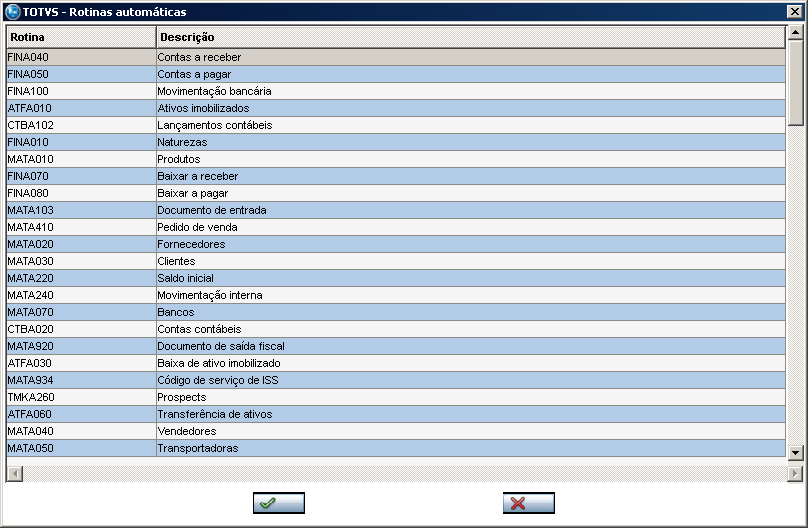
Para simular o erro, basta selecionar o erro na lista e confirmar no botão abaixo.

*Obs.:* Existe grande diferença entre os erros do P11 e P12, por exemplo a condição numérica onde no P11 gera erro e no P12 deixou de gerar.

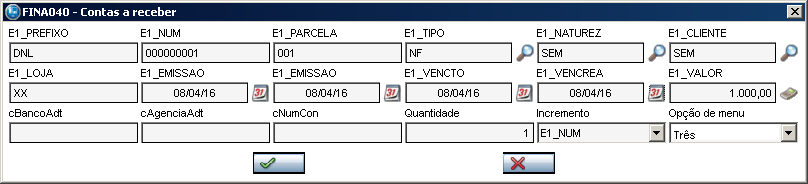
**Rotinas automáticas**

Essa rotina tem por finalidade auxiliar a popular a base de dados, podendo ser inclusas diversas movimentações de uma só vez conforme as rotinas já disponíveis e com isso por exemplo, podemos efetuar testes de performance de forma mais eficiente.

Para utilizar a rotina basta chamar a função DnlWMsExAu, será exibida a tela de seleção de rotinas:



Seleciona a rotina e então informe os dados para a inclusão:



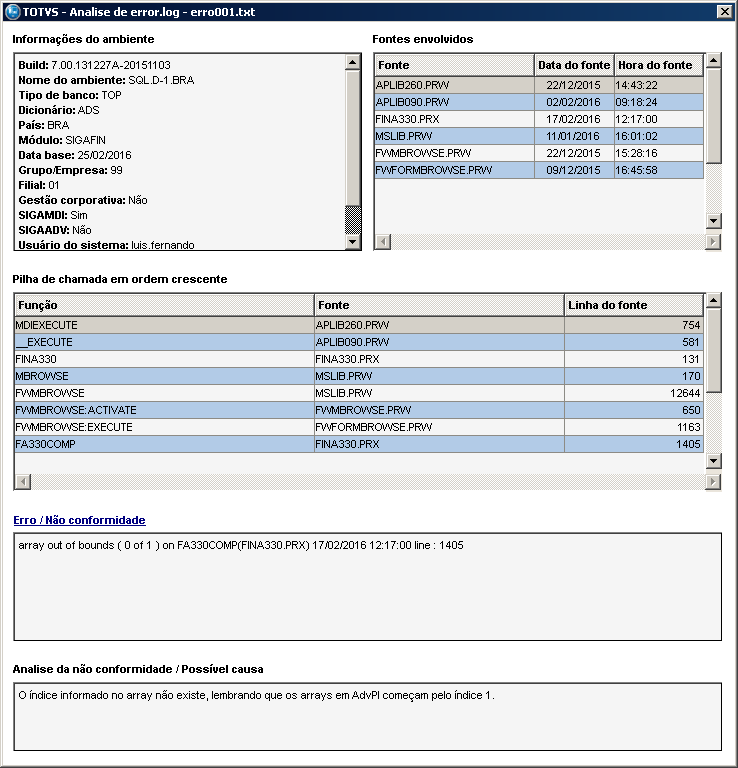
A última opção do Browse permite executar a rotina MVC, portanto é possível utilizar qualquer alias presente no SX3, existindo a possibilidade de erros no valid do modelo, pois é um modelo genérico.

*Obs.:* Os dados não são validados, caso estejam inválidos a rotina automática pode gerar erros e/ou não ser executada.

**Leitura de error.log**

Essa rotina tem por finalidade auxiliar a leitura de error.log.

Para utilizar basta chamar a função DnlRdErrLg, então será exibida uma tela de seleção de arquivos, confirmando a seleção do error.log será exibida a tela de leitura do mesmo:



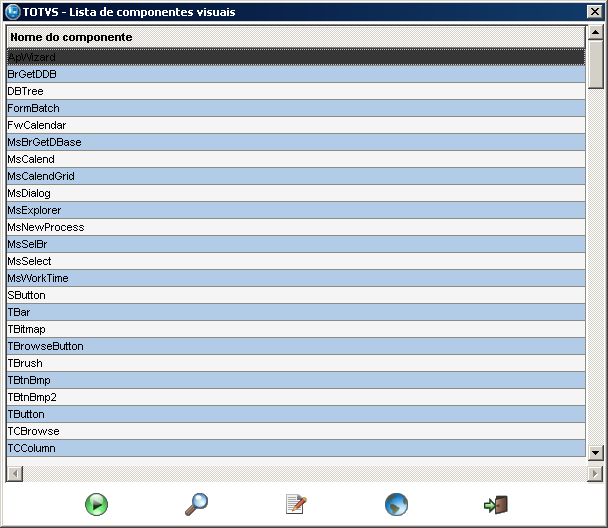
É importante que exista apenas um erro no arquivo aberto, caso contrário posso encontrar informações duplicadas e em caso de arquivos muito grandes, pode ser necessário criar a chave MAXSTRINGSIZE na General do AppServer, pois por padrão nossa string suporta 1MB.

*Obs.:* Caso o AppServer já possua as funções de SQLite, as informações são gravadas em uma tabela chamada ERRORLOG, com isso é possível incluir as suas próprias verificações ou alterar as existentes.

**Exemplos de componentes visuais de AdvPl**

Essa rotina tem por finalidade demonstrar e exemplificar alguns dos componentes visuais do AdvPl.

Para utilizar basta chamar a função DnlTstCmpnt, será exibida a tela de seleção de componentes:

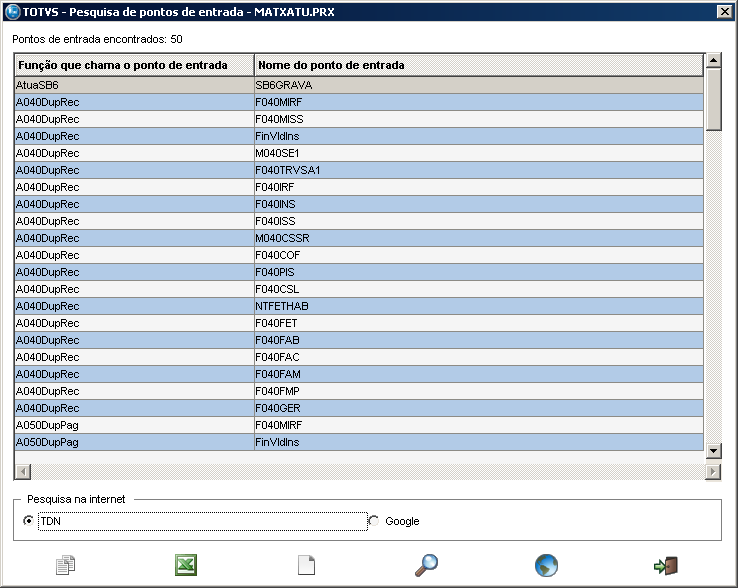


Nessa tela você possui a opção de executar o exemplo, buscar por um componente, ver o código fonte, pesquisar o componente no TDN e sair. Na opção de visualização do código é possível efetuar a impressão do mesmo.

**Pesquisa de ponto de entrada**

Essa rotina tem por finalidade auxiliar os analistas a buscar pontos de entrada em fontes AdvPl.

Para utilizar basta chamar a função DnlPEFont, será exibida uma tela de seleção de arquivos, ao confirmar a tela de seleção será exibida a listagem de pontos de entrada encontrados:

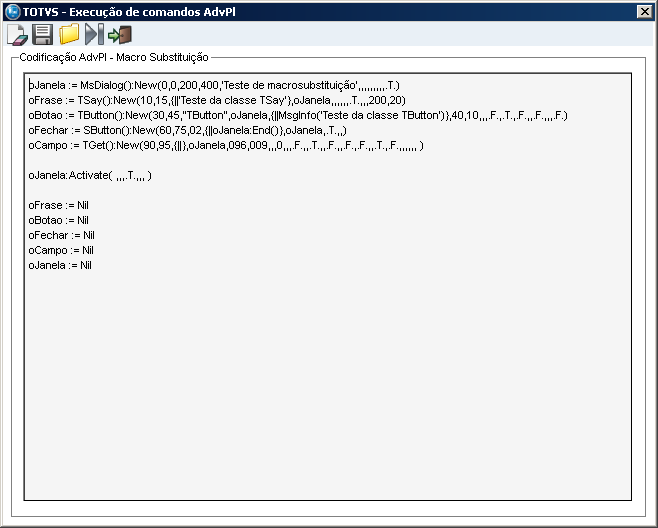


Nessa tela você possui a opção de copiar os pontos de entrada encontrados para a memória, exportá-los para .csv e .txt, pesquisar na listagem apresentada além de efetuar uma pesquisa do ponto de entrada selecionado na internet.

**Macro substituição de comandos AdvPl**

Essa rotina tem por finalidade auxiliar em testes simples como já fazemos com a rotina de fórmulas do Protheus, sendo possível macro substituir qualquer comando em sequência, possuindo as limitações de não utilizar laços, condições, variáveis locais etc... Porém é possível usar variáveis private, IIF, aEval entre outras formas para conseguir efetuar laços e condições.

Para utilizar basta chamar a função DnlExcAvpl, será exibida uma tela com um extenso campo de edição:

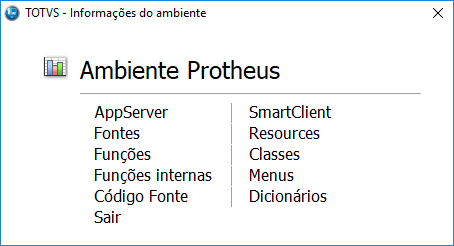


Nessa tela é possível abrir arquivos dos comandos AdvPl, salvar os comandos, executar etc.

**Informações do Ambiente Protheus**

Essa rotina tem por finalidade a análise do ambiente Protheus, sendo possível ver algumas informações em relação ao AppServer, SmartClient e também a extração das funções, fontes, resources.

Para utilizar basta chamar a função DnlRdEnrmnt, será exibida uma tela com as opções disponíveis na rotina:



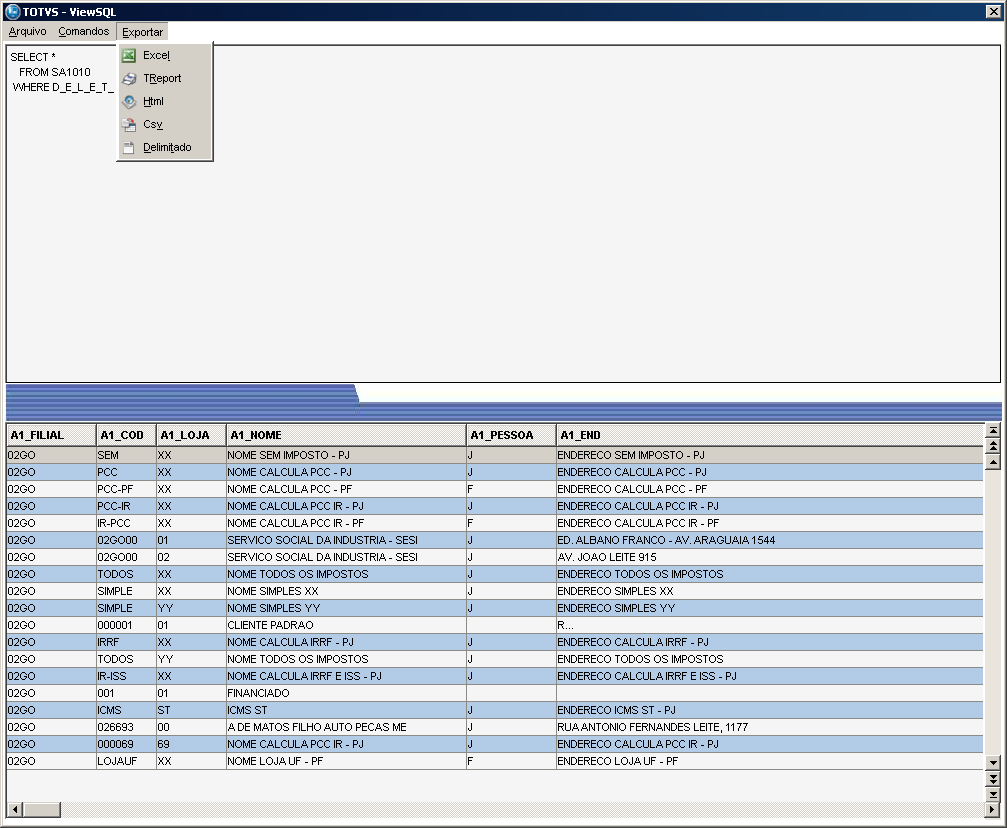
Nessa tela é possível consultar algumas informações do ambiente como:

* Informações do AppServer
* Informações do SmartClient
* Fontes presentes no RPO
* Resources presentes no RPO
* Funções presentes no RPO
* Funções internas ( binário )
* Verificação de menu
* Verificação de código fonte Advpl ( é necessário possuir o fonte/arquivo )
* Verificação de dicionário, onde inconsistências são analisadas

**ViewSQL**

Essa rotina tem por finalidade a consulta de querys no banco de dados que está configurado no Protheus, sendo possível a consulta e a exportação da mesma.

Para utilizar basta chamar a função DnlSgFzQry, será exibida uma tela com as opções necessárias para efetuar as consultas e exportá-las:



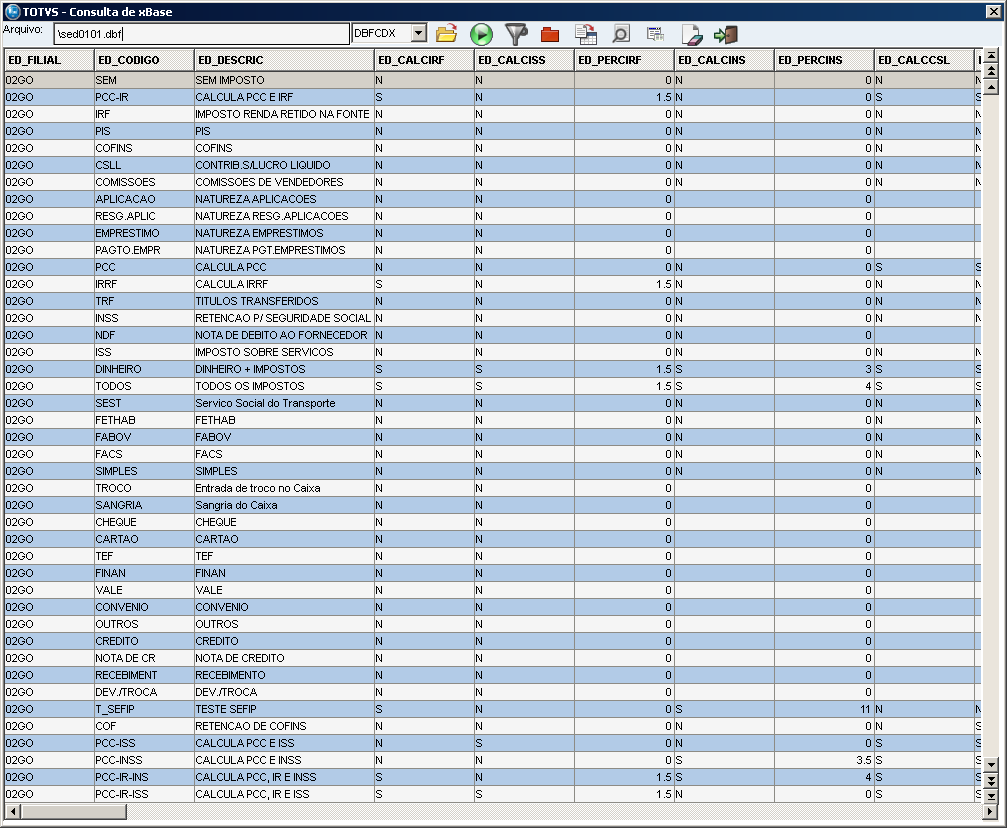
Nessa tela é possível efetuar consultas SQL, updates, deletes entre demais comandos SQL. Ao efetuar as consultas é possível exportá-las para CSV, HTML, XML, TXT ( Delimitado ) e também TReport.

*Obs.:* A tela possui uma opção de ‘Driver’, permitindo efetuar querys no banco de dados via DBAccess ou querys no SQLite local do servidor ( físico, memória ou temporário ).

**Consulta de arquivo xBase**

Essa rotina tem por finalidade consultar os arquivos do tipo xBase que normalmente o Protheus utiliza como dicionário, abrindo arquivos do tipo DBF e DTC ( C-Tree ).

Para utilizar basta chamar a função DnlSgOpnDb, será exibida uma tela onde é possível informar o arquivo xBase, abrir e consultar o mesmo assim como exportá-lo para outros formatos.

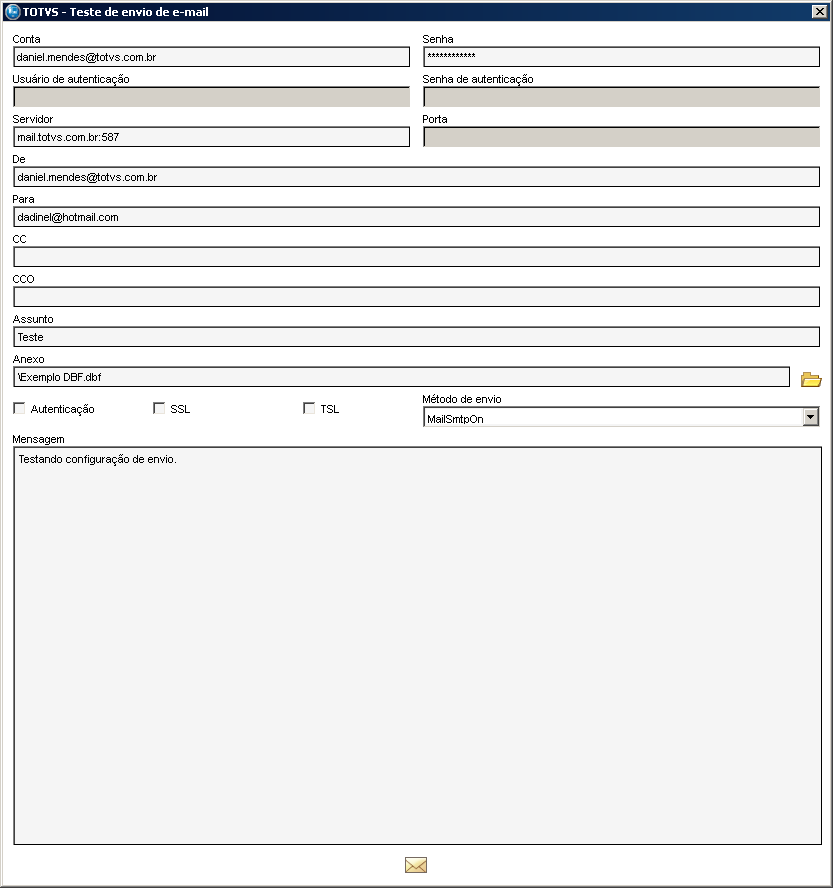


Nessa tela é possível consultar o arquivo, filtrar os registros, efetuar o pack do arquivo assim como também o zap e exportá-los, sendo que a exportação obedecerá o filtro informado. A exportação pode ser efetuada para XML, HTML, TXT, CSV, DBF e DTC.

**Métodos de envio de e-mail**

Essa rotina tem por finalidade testar as configurações de envio de e-mail no Protheus com os dois atuais métodos de envio.

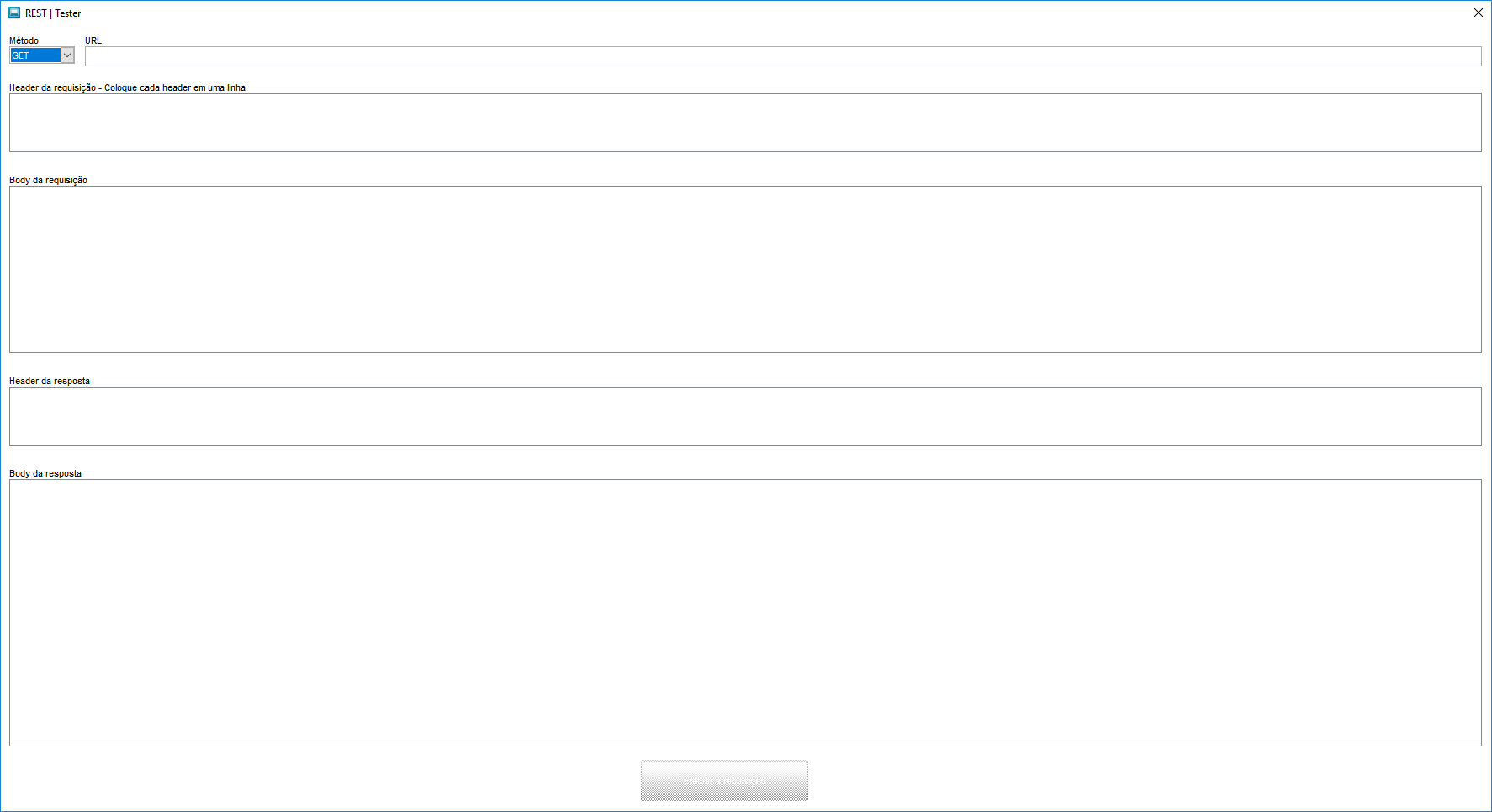
Para utilizar basta chamar a função DnlSndEMal, será exibida uma tela onde é possível informar o e-mail, senha, servidor, porta entre outras informações necessárias para o envio de e-mail.



Aqui é possível enviar o e-mail com autenticação no servidor, segurança SSL e TSL, anexo etc.

**Consumo de REST**

Essa tela tem o intuito de efetuar testes de consumo de webservices, como por exemplo REST, porém utilizando as funções de HTTP do binário.



É possível escolher entre o consumo via métodos: GET, POST, PUT e DELETE.

O timeout não é configurável, portanto segue o padrão do binário, 120s.

**Navegador WEB**

Essa tela é apenas um navegador WEB criado com base no componente TWebEngine do binário, que permite navegar no Chromium.



Existe a opção de salvar a página aberta em PDF.

**Validação/Permissão de acesso ao Aid Center**  
Existe uma user function que pode ser compilada para impedir ou criar regras de acesso a ferramenta, abaixo um exemplo de implementação:

#include "protheus.ch"

//------------------------------------------------------------------------------------------------

/\*/{Protheus.doc} Dnl\_XVld

Permite bloquear o acesso as ferramentas do AidCenter

@param nType Tipo do processo

Os valores do nType podem ser:

01 - DNL\_SQL

02 - DNL\_ADVPL

03 - DNL\_COMPONENT

04 - DNL\_EMAIL

05 - DNL\_ERRORLOG

06 - DNL\_MVC

07 - DNL\_P\_ENTRADA

08 - DNL\_READ\_ERROR

09 - DNL\_EXECAUTO

10 - DNL\_ENVIRONMENT

11 - DNL\_BROWSER

12 - DNL\_REST

13 - DNL\_ISAM

14 - DNL\_AIDCENTER

@return Boolean conforme a permissão de acesso

@author Daniel.Mendes

@since 25/04/2019

@version 1.0

/\*/

//------------------------------------------------------------------------------------------------

user function Dnl\_XVld(nType)

//Minha regra aqui...

/\*if nType == 1

endif\*/

return .F.