

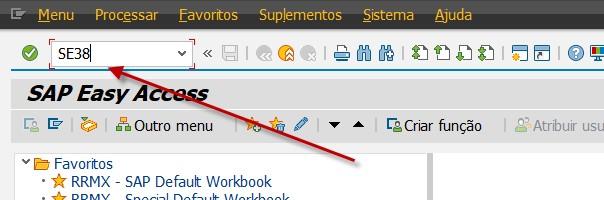
Academia ABAP With

Hana & Fiori

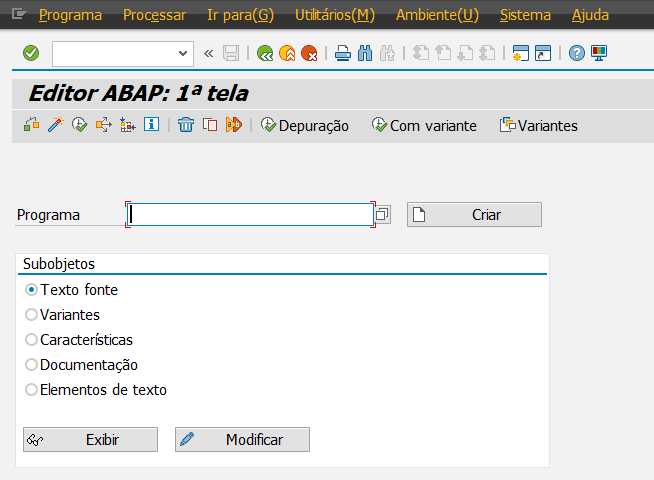


* **Criando seu primeiro programa em ABAP (SE38)**

Depois de toda a teoria, vamos agora começar a praticar nosso conhecimento em ABAP, para isso, vamos informar a transação SE38 no campo de comando, na parte superior esquerda da tela:



Será exibida a transação de edição de códigos do SAP:

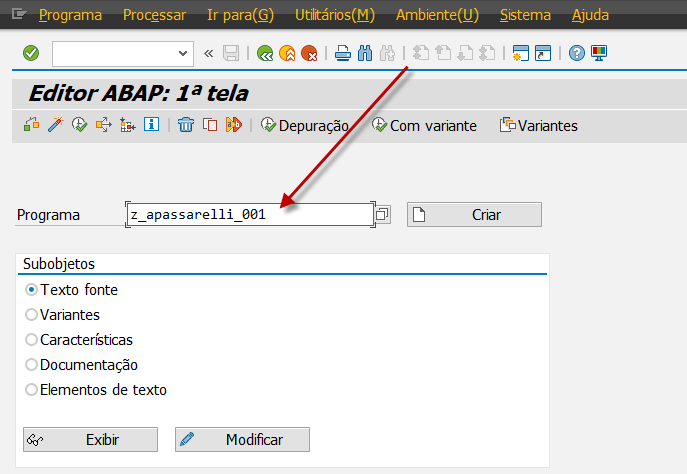


Nessa tela acima, você pode informar o nome de desenvolvimentos ABAP que já tenham sido criados, para exibir seu código fonte ou até mesmo editar o código, também é aqui que criaremos nosso primeiro desenvolvimento em ABAP, para

isso, vamos usar a letra Z ou Y para iniciar indicar o nome do nosso programa em seguida, informe seu usuário e o sequencial 001, por exemplo:

**Usuário**: APASSARELLI

**Programa**: ZAPASSARELLI001 ou Z\_APASSARELLI\_001



Após concluir esta etapa, clique no botão criar:

Conforme demonstrado na tela abaixo, informe a descrição do seu programa ABAP, essa descrição deve se referir ao tipo de processamento realizado no programa, como por exemplo: “Relatório de Faturas”, logo em seguida temos o tipo de programa, para começar a aula, usaremos o tipo “**1- Programa Executável**”, porém durante o curso estaremos trabalhando com os seguintes tipos:

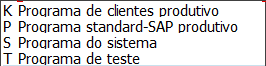
**F -** Grupo de Funções

**I -** Programa Include

**M -** Pool de módulos ou Module Pool

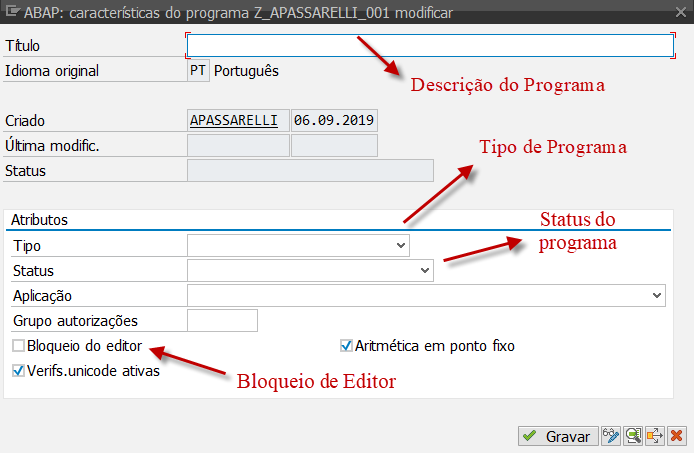
Logo abaixo do tipo do programa, encontramos a opção “Status”, que serve para

informar a classificação do programa, como por exemplo:

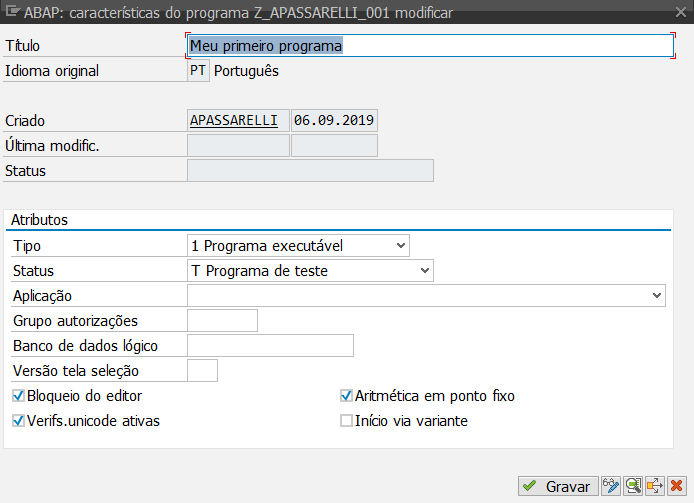


Esse é apenas um status informativo, para programas de teste utilizamos a opção “T” e quando criarmos um programa para Clientes, informamos que o programa é do tipo “K”.

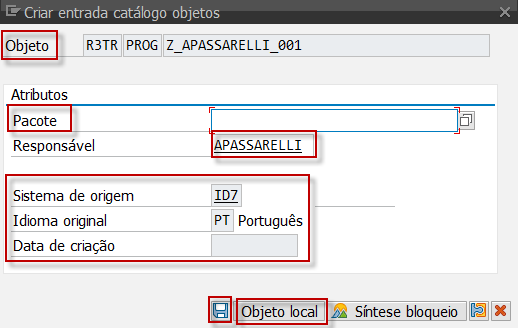
Já na última opção que vamos detalhar, você pode marcar o “Bloqueio de Editor” fazendo com que só você possa alterar ou eliminar seu programa, se alguma pessoa com usuário diferente do que foi criado o programa tentar fazer alguma dessas ações, essas serão bloqueadas.



A tela deve ficar conforme a imagem abaixo:



Após informada todas as opções, clique em: Será exibida uma tela como abaixo:



Vamos entender as opções destacadas na imagem anterior:

* + **Objeto:** São as classificações técnicas de cada objeto do SAP, todos os objetos são atribuídos a tipos específicos para serem organizados dentro do ambiente.
  + **Pacote:** Cada aplicação do SAP pode ser associada a um pacote, que agrupa todos os objetos de um projeto específico, ou de algum tipo de módulo, normalmente em clientes usamos os pacotes ZDEV para desenvolvimentos ABAP gerais, ou ZXX (XX = Módulo SAP), por exemplo ZMM para classificar todos os objetos de um mesmo módulo em um único ponto.
  + **Responsável:** É o usuário que está criando o objeto, essa informação fica registrada no log de sistemas.
  + **Sistema de Origem**: É o nome da máquina usada para hospedar o ambiente SAP.

Idioma Original: É o idioma em que o programa foi criado, baseado no idioma da tela de seleção.

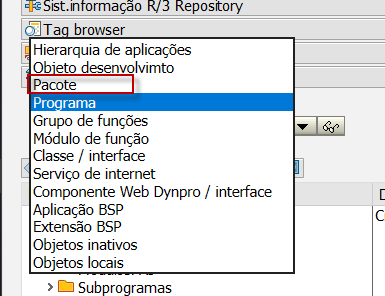
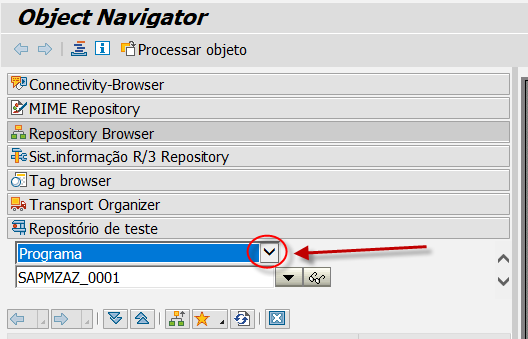
**Data de Criação**: É preenchida após o programa ser criado, para ficar registrada no log de sistemas

* + **Objeto Local**: Usamos para não associar o objeto a qualquer pacote, esses programas ficam gravados apenas dentro do ambiente e não geram request para o programa criado, vamos entender melhor o conceito de requests informando o pacote ZDEV em nosso primeiro programa.

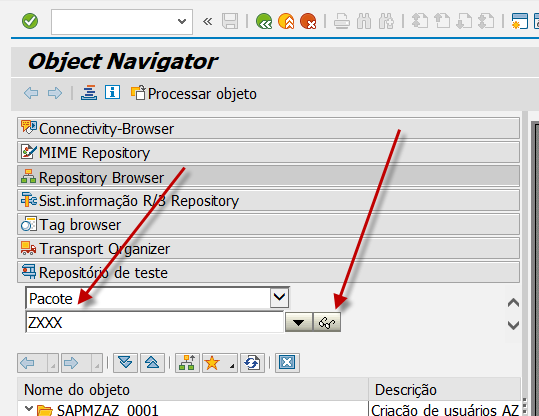


Para criar um pacote, podemos ir até a transação SE80, selecionar o objeto pacote e seguir os passos descritos abaixo:

### Transação:



Substitua o XXX pelas iniciais de seu nome e clique em **Exibir**:





As requests são os pacotes que transportam os objetos criados ou configurações entre os ambientes, como falamos na primeira parte da aula, existem vários ambientes SAP e cada um pode ser um computador diferente e para que os objetos sigam de um computador para outro, uma rota é criada através de algumas configurações, essa parte é feita pela equipe de BASIS que cuida de toda infraestrutura do SAP. Temos uma transação específica para gerenciar e criar “Requests” a transação **SE09** que veremos como funciona logo adiante, abaixo vamos conhecer os tipos de requests mais usados que existem no SAP.

### Workbench

A request “Workbench” é a mais comum dentre as requests, pois todos objetos criados como: dicionário de dados, códigos abap, formulários, etc., é a request que será mais usada por nós ABAPs, embora como veremos a seguir, existam ainda outros tipos de request envolvendo a área técnica, como veremos a seguir.

### Transporte de cópias

A request de “Transporte de Cópia” serve para tirarmos uma cópia de uma Request Workbench e transportar os objetos sem que seja necessário que a

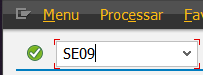
requests principal seja transportada, isso é usado para diminuir a margem de erros no transporte de requests para o ambiente “PRD” e garante que todos os objetos fiquem em apenas uma request no final.

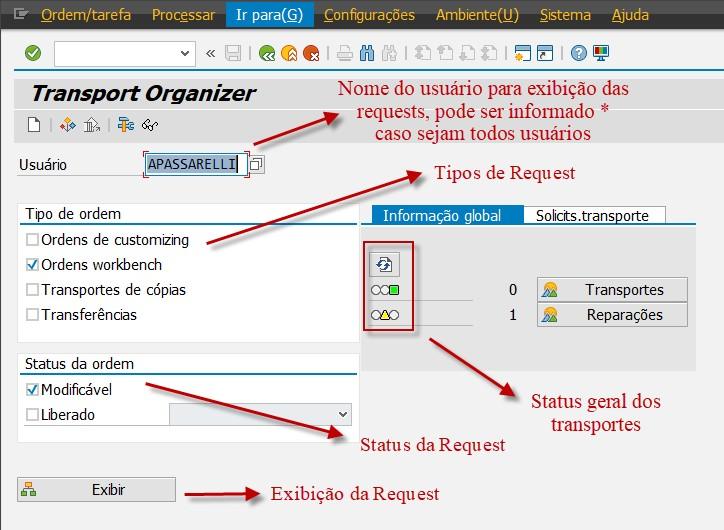
### Customizing

Essa request é usada para gravar as configurações realizadas pela equipe Funcional e também pode ser usada para diversas outras configurações de todos os módulos, inclusive os que envolvem ABAP, nela não existem códigos, ou objetos que pertençam a códigos, apenas dados de tabelas são transportados nesse tipo de request.

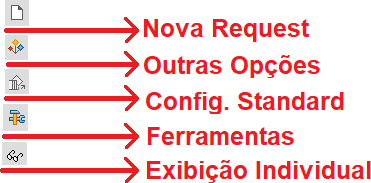
### Transações relacionadas a Request (SE03/SE09)

Vamos agora aprender como gerenciar as requests no SAP, para isso, informe a transação SE09 no “campo de comando”, conforme a imagem:



Será exibida a tela abaixo:

E também temos a “Barra de botões da tela”, com os seguintes campos:



Veremos durante o vídeo cada opção separadamente, assim como os detalhes e como funcionam.

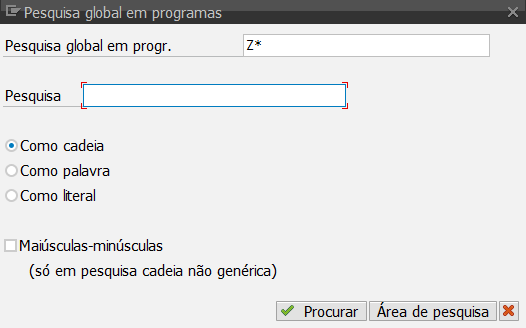


Sempre que um ABAP vai desenvolver um novo código, ou fazer uma lógica para ajustar os códigos já existentes é preciso pensar no que será feito, construir todas as partes do programa e como muitas lógicas podem ser reutilizadas, é muito bacana que busquemos códigos que já atendem ao que precisamos fazer, isso pode otimizar o nosso tempo durante o desenvolvimento de uma demanda além de nos apresentar outras ideias de outros programadores para o que estamos tentando fazer especificamente.

Para pesquisarmos o que precisamos em outras lógicas de outros programas, vamos conhecer duas formas três das diversas formas que você mesmo pode desenvolver para conseguir manter uma base de dados, como por exemplo, gravar todos os seus programas realizados em arquivos “TXT”, mantendo esses arquivos sempre com você, assim você poderá reutilizar no futuro, caso passe pelo mesmo cenário.

### EWK1

A transação EWK1 faz uma busca em todos os objetos que envolvam um código, procurando pela instrução específica informada pelo usuário, veja abaixo como funciona:



### Cadeia de pesquisa:

Na pesquisa global (ou seja, pesquisa no include geral ou programa) é possível uma entrada genérica com + e \*. Símbolo de escape é #, ou seja #+, #\* e ## para a pesquisa do caractere +, \* ou # . Não é permitida qualquer cadeia de pesquisa genérica para substituição e para pesquisa local.

Caso a cadeia de pesquisa tenha espaços em branco precedentes ou subsequentes, tem de estar incluída em caracteres especiais ( /-.,:;? ).

O tipo da pesquisa depende do método de pesquisa selecionado:

* **como cadeia** = cadeia de pesquisa é procurada no texto como uma sequência parcial qualquer
* **como palavra** = cadeia de pesquisa é procurada como palavra no sentido do léxico ABAP.

Caso as maiúsculas/minúsculas estejam marcadas, a cadeia de pesquisa é procurada na ortografia entrada. De outro modo, é procurada independentemente das maiúsculas/minúsculas.

| Na cadeia de pesquisa | genérica, | a | procura | é | sempre efetuada |
| --- | --- | --- | --- | --- | --- |
| independentemente |  | da |  |  | ortografia. |

No caso da pesquisa por palavras, não se pode distinguir entre maiúsculas/minúsculas.

### o Área de pesquisa:

A área de pesquisa pode ainda ser processada. O ícone "Seleção múltipla" no popup de pesquisa junto ao nome do programa/classe conduz a um outro popup com os componentes da área de pesquisa. Os componentes serão retirados da

área de pesquisa, quando a sua marcação for anulada, possibilitando a inserção de novos componentes.

Na pesquisa em classes, o botão "Objetos herdados" no popup da área de pesquisa permite exibir todos os componentes herdados e transferi-los para a área de pesquisa. São apenas oferecidos os componentes válidos na classe a ser pesquisada, ou seja, componentes em excesso após a redefinição serão ocultados.

### Utilização

Entrada genérica com + e \*, mas não para a substituição. O símbolo de escape é #, portanto #+ e #\* para a pesquisa do caractere + ou \*.

**O tipo de pesquisa depende do método de pesquisa selecionado:**

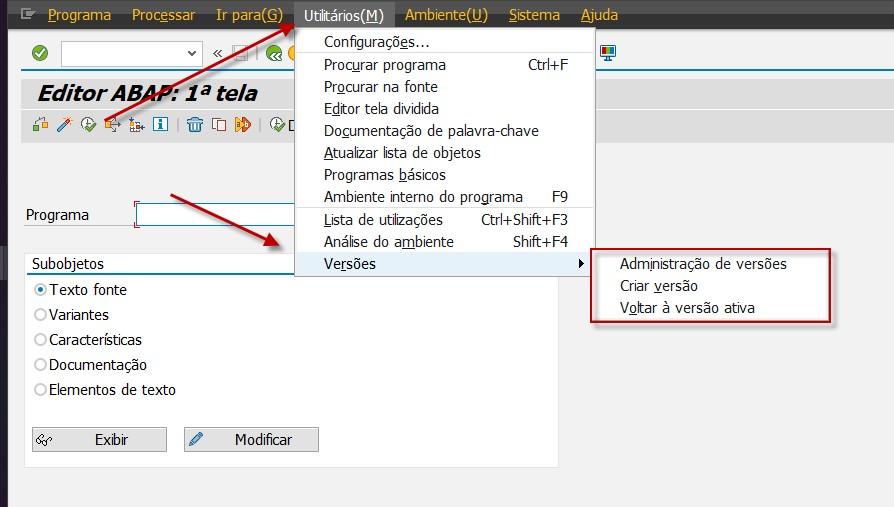
* **como cadeia** = cadeia de pesquisa, é procurada como sequência parcial no texto
* **como palavra**= cadeia de pesquisa, é procurada no sentido do léxico ABAP/4/

Se estiver marcado **Maiúsculas/Minúsculas**, a cadeia de pesquisa é procurada na ortografia. Caso contrário, a pesquisa é efetuada independentemente das maiúsculas e das minúsculas. Com uma cadeia de pesquisa genérica, a pesquisa é sempre efetuada independentemente da ortografia.

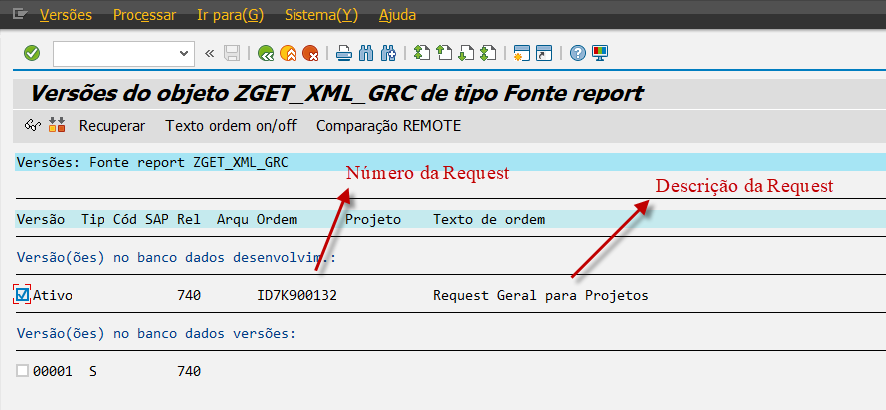


A cada request gerada e transportada para um objeto do SAP, uma nova versão daquele objeto é criada, assim, caso alguma coisa aconteça de errado, você pode voltar a versão anterior para ajustar o que deu errado.

O versionamento de programas fica no “Menu Superior” em cada objeto, navegando até a opção “Versão”, conforme a imagem abaixo:



Nessa tela podemos verificar todas as versões transportadas ou ativas de um objeto SAP, durante o curso vamos explorar essas opções mais detalhadamente, além da barra de botões da tela, com a explicação de cada campo:





Para codificar em ABAP, assim como em outras linguagens de programação, temos que respeitar algumas regras durante o desenvolvimento dos nossos programas, pois todo código tem sua estrutura genérica que não pode ser alterada, pois a ordem de cada lógica tem que ser encaixada perfeitamente em seu lugar, assim como veremos nos exemplos abaixo.

Ao criarmos um programa em ABAP, nenhum código além do nome do programa é informado no editor, todo o código é de responsabilidade do programador ABAP que ao conhecer as “Sintaxes” presentes na linguagem, consegue converter as solicitações dos clientes em códigos, para atender todas as áreas do SAP.

Programa criado:

Como vemos na imagem acima, somente o cabeçalho do programa é informado ao programa ser criado, precisamos entender que todos os objetos que farão parte de todo desenvolvimento devem e ficar na parte superior, pois o conceito

de um objeto criado é que sua utilização seja a partir de sua declaração, por exemplo:

**Declaração de** Campo de Valor **do tipo** Valor. Campo de valor = **10,00**.

Se usarmos a lógica da seguinte forma, não será possível ativar o código, pois o

objeto ainda não foi criado onde está sendo “**alimentado com o valor de 10,00**”.

Campo de valor = **10,00**

**Declaração de** Campo de Valor **do tipo** Valor. (Criação do objeto usado acima)

A lógica deve ser respeitada, ou seja, para colocarmos uma informação em algum lugar, esse lugar primeiramente precisa existir e só então poderemos trabalhar com ele, entendendo esse conceito, vemos que a maioria dos objetos de um programa será declarada no topo, logo abaixo do nome do programa, pois eles serão úteis em todo o código, os tornando “**Globais**”.

Logo abaixo, estaremos conhecendo os objetos existentes no SAP, assim como sua utilização em exemplos de códigos realizados durante a aula, mas antes, vamos entender o porque a lógica acima deve ser respeitada.



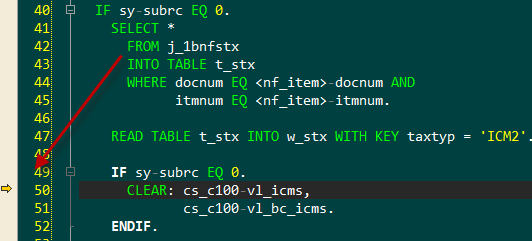
Um programa ABAP sempre está lendo todas as linhas codificadas e seguindo com o fluxo desenvolvido, em programas procedurais eles partem de um ponto e avançam até o final, ou até alguma instrução que o faça parar, esse

conceito de seguir o fluxo sempre para “baixo” é o que caracteriza uma

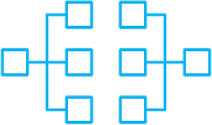
programação procedural.

Existe um indicador que vai

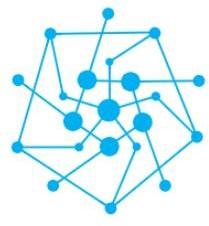
interpretando cada linha do código e as convertendo em  informações





Quando falamos de orientação a objetos, fugimos totalmente do conceito de começo, meio e fim, pois toda aplicação depende de uma série de métodos e pode englobar diversos processos, como por exemplo, existe uma

classe no SAP criada para calcular os impostos, ela se chama “CL\_TAX\_CALC\_BR”, nela cada método corresponde, por exemplo, a um tipo de imposto, onde o cálculo é feito apenas nesse método e todas as lógicas necessárias, isso facilita na hora de construir aplicações

complexas, que dependam desses

tipos de processos. Como veremos

abaixo, existem diversos métodos,

um para calcular o ICMS, outro IPI o

outro o ISS e assim por diante, dessa

forma, é como se vários mini

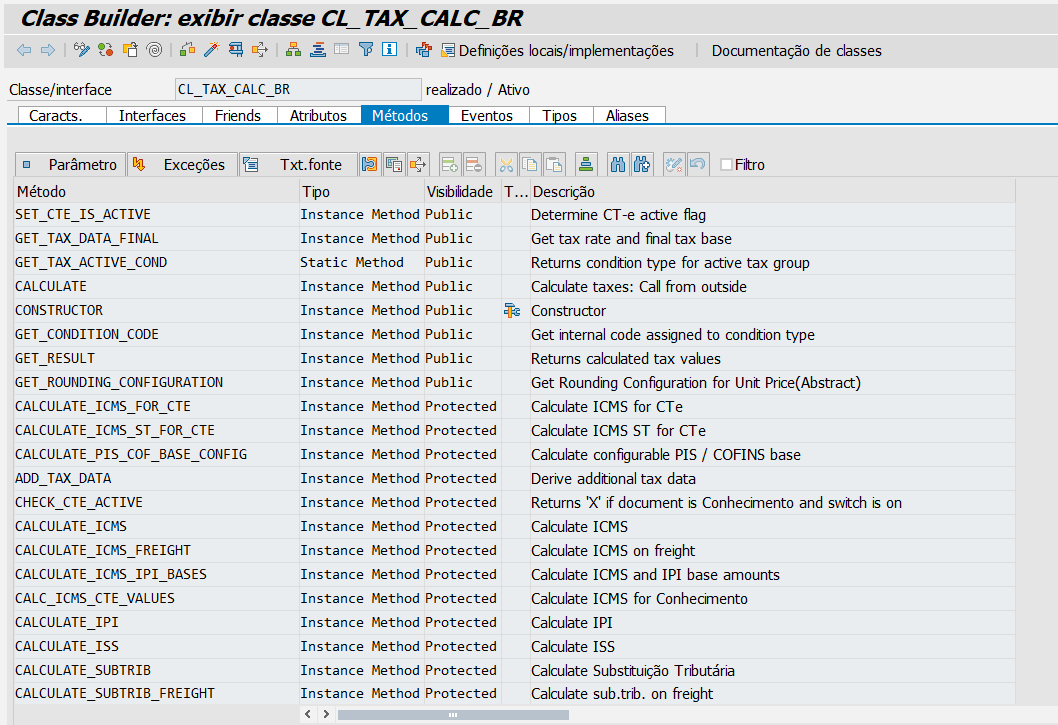
programas fossem construídos a cada

método, os métodos possuem

campos de importação e

exportação para enviar ou receber

dados de dentro dos métodos, essa configuração permite que todos os métodos que precisem se comunicar diretamente tenham a possibilidade de transferir dados através desse campos, uma outra vantagem é ter todos os objetos dentro de um só local, assim se for necessário usar alguns desses objetos em diversos locais essas opções de parâmetros podem ajudar com essa questão.

**Classe: CL\_TAX\_CALC\_BR**

**Método para calcular o ICMS: CALCULATE\_ICMS**