Criação do BS

- Criar uma branch com o nome do seu serviço ex : feature/consulta-cadastro;

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

- Por padrão, um feature para cada serviço;

- De preferência, criar uma branch no git;

- Quando for criar a branch no GitLab, sempre definir o campo “Create from” como develop

- Criar os DTOS para request e o response, de acordo com BS

- No BS, é avaliado como criar o request e o response, utilizamos a referência do tipo da variável no arquivo .cls : ByVal são dados de entrada e ByRef são dados de saída.

Texto

Descrição gerada automaticamente

- Parâmetros de entrada são definidos no request == ByVal

- Parâmetros de saída são definidos no response == ByRef

- Estrutura do IBPJ

Texto

Descrição gerada automaticamente

**ATENÇÃO : TODA VEZ QUE FOR CRIAR UM REQUEST PARA O BS, EXTENDER PARA “BaseRequestDTO”, porque já tem vários dados dentro dessa classe**

**-** Análise o código legado e verique se os parâmetros já foram criados no BaseRequestDTO

**SUGESTÃO : OS NOVOS PROJETOS DA ARSENAL É POSSÍVEL UTILIZAR LOMBOK, PARA NÃO PRECISAR USAR GETTERS E SETTERS NA MÃO**

**RESUMO : O processo para criar um request no BS é analisar os parâmetros de entrada, avaliar se os parâmetros do ByRef é utilizado nos inputs do conector.**

- Já existe uma classe chamada IBPJ ERROS, dentro da biblioteca de utils. Os parâmetros de erros já foram criados na classe, não há necessidade de criar no response

- No arquivo .CLS, tudo que for DIM corresponde a declaração de variáveis.

Tela de computador com texto preto sobre fundo azul

Descrição gerada automaticamente

- Chamada para conector MAINFRAME

Interface gráfica do usuário

Descrição gerada automaticamente

- No .CLS , tudo que for “TYPE\_AREA\_DR84” não é necessário implementar no JAVA .

Tela de computador com texto preto sobre fundo azul

Descrição gerada automaticamente

- Os demais é necessário a implementação no BS

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

- No VB, toda chamada ao conector, existe dentro de um pacote.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

- A implementação em JAVA seguirá a mesma distribuição de pacotes

- O microserviço que será implementado não faz persistência direta ao banco de dados.

- A aplicação acessa o conector, o conector acessa o mainframe, o mainframe faz o trabalho de persistência de dados

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

- Existem dois tipos de serviço na arquitetura IBPJ : serviços que utilizam do conector Mainframe para fazer chamada de transação ou serviços que fazem acesso via procedure

- A procedure é responsável por fazer a persistência no banco de dados.

- Existem 3 modes do conector

Texto

Descrição gerada automaticamente

**ATENÇÃO : É NECESSÁRIO FAZER ANÁLISE DO CÓDIGO VB PARA REFERENCIAR O CONNECTOR NO MODE CORRETO**

**EXEMPLO:**

**Interface gráfica do usuário, Texto

Descrição gerada automaticamente**

**VB**

**Texto

Descrição gerada automaticamenteJAVA**

**- Ao analisar o método do conector, é possível abstrair quais serão a entrada e saida de dados.**

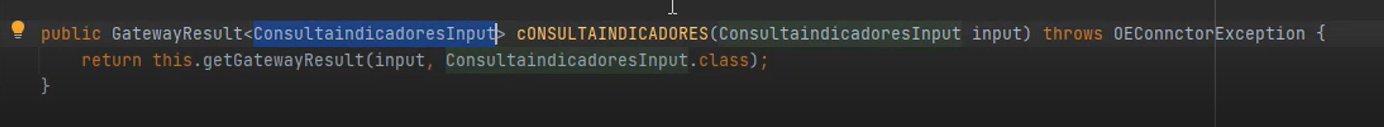
**Texto

Descrição gerada automaticamente**

**IMPORTANTE : EM ALGUMS MÉTODOS DO CONNECTOR, O MESMO TIPO DE ENTRADA VAI SER O MESMO TIPO DE SAIDA. ISSO ACONTECE, DEVIDO AO CÓDIGO LEGADO TRABALHAR COM PONTEIRO, ENTÃO AS VEZES A PRÓPRIA ENTRADA ERA USADA COMO SAIDA PORQUE**

**A VARIÁVEL ERA PASSADA COMO REFERENCIA E NÃO PELO VALOR EM SI, ESSA POLITICAS**

**SE REPLICOU PARA OS CONNECTORS EM JAVA**

****

**- O REQUEST E RESPONSE VÃO ESTÁ NO PACOTE DE DTO, no JAVA.**

**- O SERVICE É UTILIZADO O PADRÃO DE PROJETOS DE INTERFACE, OU SEJA, SEMPRE QUE FOR CRIADO UMA INTERFACE, É ADICIONADO O CONTRADO , E DEPOIS DISSO IMPLEMENTADO EM UMA CLASSE , OBRIGANDO A IMPLEMENTAÇÃO A CUMPRIR ESSE CONTRATO.**

**Interface gráfica do usuário, Texto

Descrição gerada automaticamente**

**- Padrão de nomenclatura : Consultar o nome da classe,métodos e pacotes que está no VB traduzir para o inglês para o Java.**

**- ERRO PADRAO DO IBPJ**

**Texto

Descrição gerada automaticamente**

**- TODO SERVICE PRECISA DAR UM THROWS EM IBPJ ERROS**

**Texto

Descrição gerada automaticamente**

**- Constantes no VB serão convertidas para variáveis estáticas no Java**

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**

****

**- Informações que estarão em todas as “Impls”**

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**

**- A consultar o arquivo . cls, desconsiderar o input Type\_AREA\_DR84**

**Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente**

**- Após o input Type\_AREA\_DR844, deverá ser considerado os inputs**

Texto

Descrição gerada automaticamente

**ATENÇÃO : 90% DOS PROBLEMAS VAI ESTÁ NA CHAMADA DO CONECTOR**

**Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente**

**MONTAR A CONSULTA AO CONNECTOR NO JAVA EXATAMENTE COMO ESTÁ NO VB, CONFERIR DIVERSAS VEZES.**

**Texto

Descrição gerada automaticamente**

**Texto

Descrição gerada automaticamente**

**O ‘Verifica Erros’ do VB foi substituída por uma classe no Java UtilService**

**Texto

Descrição gerada automaticamente**

**Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente**

**- Validação para verificar se houve erros**

**Texto

Descrição gerada automaticamente**

**- catch padrão do impl**

**Interface gráfica do usuário, Texto

Descrição gerada automaticamente**