Expresso São Miguel - Setor de T.I

Integração com API de Configurações

11 de novembro de 2019

VISÃO GERAL

Biblioteca Java responsável pela integração de aplicações baseadas Java com a API de Controle de Configurações.

OBJETIVOS

1. Facilitar a integração das aplicações desenvolvidas em java com a api de configurações.
2. Disponibilizar ao programador uma ponte entre as aplicações e suas configurações.
3. Centralizar funções de conexão com a api e facilitar a manutenção e disponibilização de futuras melhorias.

ESPECIFICAÇÕES

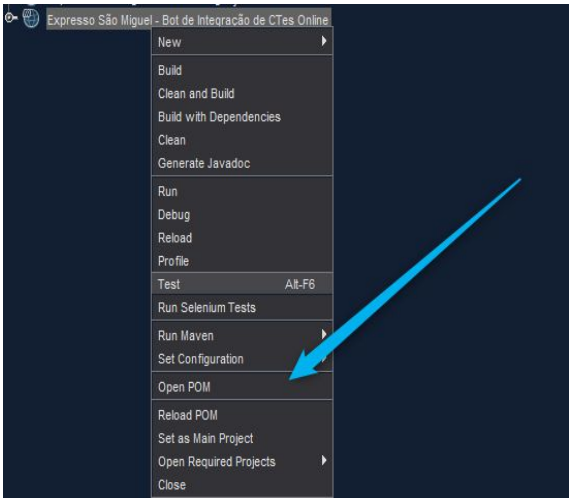
Biblioteca desenvolvida na linguagem de programação Java baseada em Maven. Funciona tanto em projetos Java baseados em Maven através do pom.xml, como em projetos Java comuns através da importação do .jar da biblioteca.

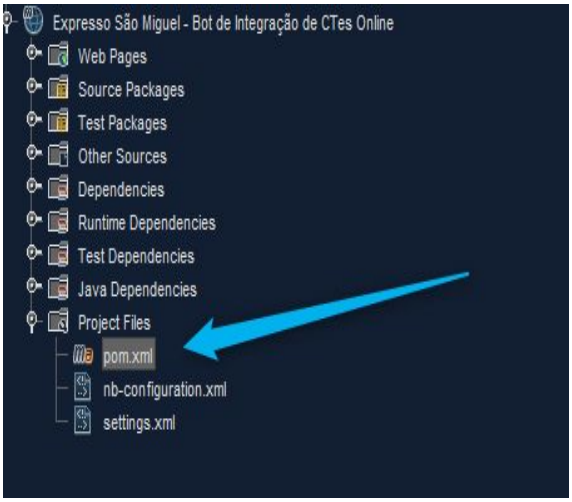
UTILIZAÇÃO

Projetos baseados em Maven

Para utilizar a biblioteca em projetos Java baseados em Maven deve-se configurar no arquivo pom.xml as seguintes propriedades:

Acessando pom.xml





Configurando o repositório Maven Expresso São Miguel

```
<distributionManagement>
    <snapshotRepository>
        <id>nexus-snapshots</id>
        <url>http://172.16.0.205:8081/nexus/content/repositories/snapshots/</url>
    </snapshotRepository>
    <repository>
        <id>nexus-releases</id>
        <url>http://172.16.0.205:8081/nexus/content/repositories/releases/</url>
    </repository>
</distributionManagement>

<repositories>
    <repository>
        <id>exp</id>
        <name>Exp Releases repository</name>
        <url>http://172.16.0.205:8081/nexus/content/groups/public/</url>
        <layout>default</layout>
    </repository>
</repositories>
```

Adicionando a dependência no pom.xml

```
<dependencies>
    <dependency>
        <groupId>br.com.esm</groupId>
        <artifactId>configurations</artifactId>
        <version>2.0.0.RELEASE</version>
    </dependency>
```

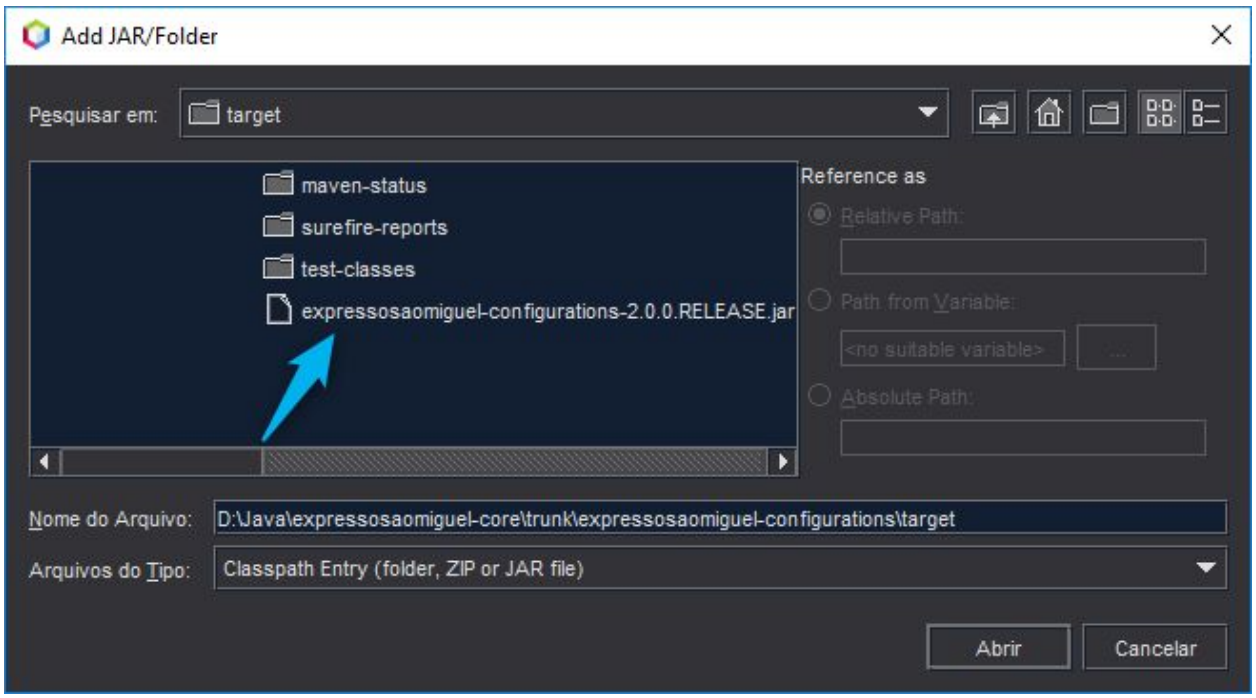
Projetos não baseados em Maven

Para utilizar a biblioteca em projetos Java que não são baseados no Maven, é necessário ter o .jar da biblioteca e importa-lo manualmente no projeto.

Adicionando arquivo .jar as dependências do projeto

No seu projeto, clica com o botão direito sobre “Libraries” e clique em “Add JAR/Folder”.

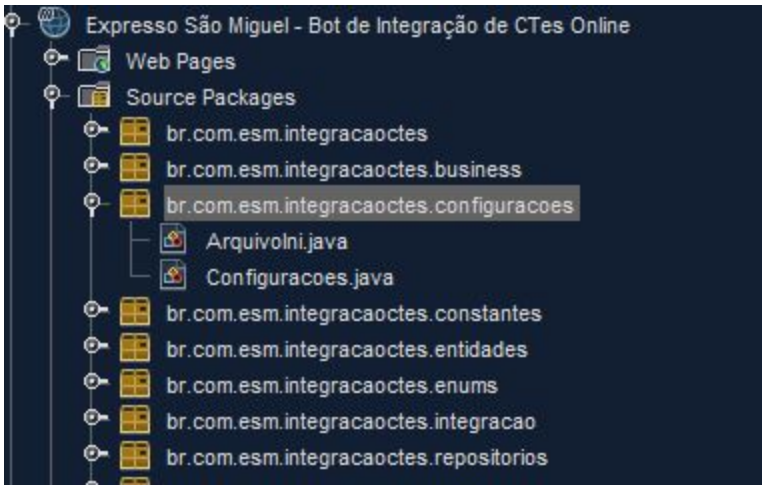
Após isso, navegue até onde o .jar se encontra em seu computador e clique em “Abrir”.



Após essas duas etapas, o projeto já estará configurado para utilizar a biblioteca.

CLASSE DE CONFIGURAÇÕES

Foi convencionado que os projetos Expresso São Miguel deverão ter um pacote exclusivo para a classe de configurações, baseados nos projetos da empresa deve-se seguir a nomenclatura “br.com.esm.nome_do_projeto.configuracoes”. Dentro do pacote uma classe chamada “Configuracoes” deve ser criada.



Dicas para facilitar o uso

Para ter mais facilidade e para que o código fique mais claro e limpo convém optar por mais uma dependência chamada “Lombok” que torna o uso de *Getters* e *Setters* dispensável, além do mais construtores vazios ou completos com todos os argumentos da classe podem ser criados através de anotações de classe.

Implementando a classe de Configurações

A classe “Configuracao” previamente criada deve estender da “ExpConfiguracoesBase”, cuja qual é uma classe abstrata que tem por função buscar e alimentar as propriedades de configurações existentes nesta classe.

Ao estender a classe base, dois métodos abstratos são obrigatórios de serem implementados.

Método “getChaveConfiguracoes()”

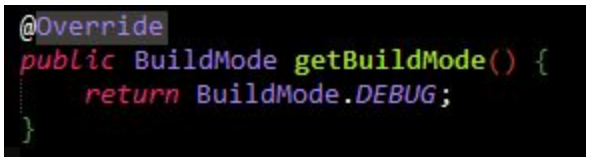
Deve retornar a chave da aplicação que está cadastrada na tabela “T_APLICACAO_CONFIGURACAO” no banco SistemaExp. Essa chave é um UUID gerado no cadastro das configurações e é único para cada aplicação.

A aplicação deverá receber a mesma chave em ambiente de produção e homologação.

Método “getBuildMode()”

Este método tem por objetivo identificar em qual ambiente as configurações devem ser carregadas.

- BuildMode.DEBUG = Ambiente de Homologação, Api está no servidor de homologação conectada ao banco de dados SistemaExp do servidor 174.
- BuildMode.RELEASE = Ambiente de Produção, Api está no servidor de produção e conectada ao banco de dados SistemaExp do servidor 161.



DEVE-SE TER MUITO CUIDADO AO UTILIZAR. NUNCA UTILIZAR CONFIGURAÇÃO DE PRODUÇÃO EM HOMOLOGAÇÃO E VICE-VERSA. SEMPRE LEMBRAR DE MUDAR EM CASO DE UTILIZAÇÃO DO PROJETO EM AMBIENTE DE HOMOLOGAÇÃO E POSTERIORMENTE MUDAR PARA SUBIR EM PRODUÇÃO.

Criação das propriedades referentes às configurações

As propriedades da classe que são referentes a configurações devem ser anotadas com “@Configuration”.

A anotação tem os seguintes parâmetros:

- name = Chave da aplicação que está na tabela T_APLICACAO_CONFIGURACAO_VALOR.
- defaultValue = Atribua um valor default caso a propriedade não seja encontrada no banco de dados.
- trueValue = Para propriedades booleanas utilize este parâmetro para definir qual é o valor que deverá retornar *true* para a propriedade.
- falseValue = Para propriedades booleanas utilize este parâmetro para definir qual é o valor que deverá retornar *false* para a propriedade.

```
@Configuration(name = "MAXIMO_TENTATIVAS_ENVIO", defaultValue = "3")
private Integer maximoTentativasIntegracao;

@Configuration(name = "INTERVALO_EXECUCAO_INTEGRACAO", defaultValue = "10")
private Integer intervaloTempoIntegracao;

@Configuration(name = "ACRESCIMO_TEMPO_TENTATIVA", defaultValue = "300")
private Long acrescimoParaTentarNovamente;

@Configuration(name = "API_INTEGRACAO_BASE_URL")
private String wsIntegracaoBaseURL;
```

Carregamento das configurações

Ao criar a classe, seja utilizando o SpringBoot por meio das anotações @Component na classe e @Autowired para fazer a injeção da classe de Configurações ou através do método tradicional Configuracoes configuracoes = new Configuracoes(), as propriedades serão carregadas automaticamente.

Utilize Singleton para que as propriedades não sejam carregadas toda vez que utilizar a classe.