EPROCESSOS: Um Sistema Editor de Processos de Software

Sávio Geraldo Ferreira Fraga

Universidade Federal de Ouro Preto

GUIA DE USO DO CÓDIGO FONTE

Será apresentado aqui um tutorial para o uso do código fonte do sistema desenvolvido neste trabalho. Primeiramente serão mostradas as ferramentas necessárias para a edição do código e a seguir será mostrada a descrição do projeto do sistema.

1. Requisitos para Edição

O projeto do sistema está no padrão do MAVEN. Sendo assim, para a edição do código é necessária a instalação do Apache Maven 2. Abaixo segue um guia de instalação e uso do Maven.

1.1 Instalação e Uso do Apache Maven

O primeiro passo é fazer o download no site <*http://maven.apache.org*>.
Em seguida, descompactar os arquivos em algum diretório a sua escolha, que será referenciado aqui como $MAVEN.
Adicionar o diretório $MAVEN/bin à variável PATH.
Para verificar a instalação, execute no prompt/shell de comando o seguinte comando:

```
mvn –help
```

Para criar o projeto do EPROCESSOS no Eclipse, copie a pasta "**eprocessos**" que se encontra disponível dentro da pasta "**src**" do CD de instalação do sistema e cole esta pasta em algum diretório do seu computador.
Dentro desta pasta encontra-se um arquivo "**pom.xml**" e uma pasta "**eprocessos**". O arquivo "**pom.xml**" possui as informações básicas de um projeto, bem como as diretivas de como o artefato final deste projeto deve ser construído. Na pasta "**eprocessos**" encontra-se o código fonte do sistema.
Agora acesse via prompt/Shell a pasta "**eprocessos**" que você colocou no seu computador e digite o seguinte comando:

```
mvn eclipse:eclipse
```

Agora o projeto no formato do Eclipse já está criado e pode ser acessado por este para fazer a edição do projeto.

Para "rodar" o código acesse via prompt/Shell a pasta do seu projeto e digite o seguinte o comando:

```
mvn tomcat:run
```

Após executar este comando o sistema estará acessível no *browser* no endereço "**http://localhost:8080/eprocessos**".

2. Estrutura do Projeto

O código fonte do projeto encontra-se dentro do diretório "**src/eprocessos/src/main**". Dentro deste diretório encontram-se os diretórios: "**bd**", "**java"** e "**webapp**". Nas seções seguintes será explicado o conteúdo de cada um destes diretórios.

2.1 Diretório "bd"

Neste diretório temos o arquivo "**eprocessos.xml**" e o diretório "**postgre**".

O arquivo "**eprocessos.xml**" contém o projeto do *DBDesigner*, com o qual pode ser visto a modelagem do banco de dados da aplicação de forma gráfica.

O diretório "**postgre**" contem os scripts SQL para a geração do banco de dados da aplicação. Estes scripts podem ser gerados através do *DBDesigner*.

2.2 Diretório "java"

Neste diretório temos outros dois diretórios "**META-INF**" e "**br/ufop/eprocessos/**".

2.2.1 "META-INF"

A especificação Java Persistence API requer um arquivo "**META-INF/persistence.xml**" para configuração. Você precisa ter o arquivo "**persistence.xml**" dentro do diretório "**META-INF**", que deve estar no seu *classpath*. Detalhes sobre o "**persistence.xml**" podem ser encontrados na documentação da JPA.

2.2.1 “br/ufop/eprocessos”

Dentro deste diretório temos os diretórios “**dao**”, ”**entity**”, ”**enumerations**”, ”**service**”, ”**util**” e “**web**”.

O diretório “**dao”** contém as classes Java responsáveis pela comunicação com o SGBD, ou seja, todas as funcionalidades de bancos de dados, tais como obter as conexões e executar comandos SQL.

O diretório “**entity**” contém as classes Java responsáveis por mapear objetos do modelo de dados, sendo que foi gerada uma classe para cada tabela com suas propriedades.

O diretório “**enumerations**” contém apenas uma classe, “*PerfilUsuario.java*”. A funcionalidade principal é agrupar valores com o mesmo sentido dentro de uma única estrutura, neste caso, seriam os perfis dos usuários cadastrados no sistema. Embora sua funcionalidade é totalmente diferente, a criação de uma estrutura enumerada é parecida com uma classe. Basta trocarmos a palavra chave “**class**” pela palavra chave “**enum**”. Esta classe ainda não está sendo utilizada na aplicação, uma vez que a aplicação conta atualmente com apenas um usuário cadastrado. Portanto esta classe será útil para trabalhos futuros na aplicação.

O diretório “**service**” contém as classes Java responsáveis pelas regras de negócio da aplicação, regras tais como listar, adicionar, remover e incluir um registro no banco de dados. Essas classes acessam e utilizam métodos implementados nas classes dos pacotes “**dao**” e “**entity**” para disponibilizar serviços da aplicação.

O diretório “**util**” contém algumas classes com métodos úteis para a aplicação, tais como, listagem de alguns itens e questões de segurança. As classes deste pacote são pouco utilizadas na aplicação.

O diretório “**web**” contém as classes responsáveis por publicar os serviços e o modelo de dados, os quais estão representados nos pacotes “**service**” e “**entity**” para a camada de apresentação. As classes desse diretório usam a injeção de dependência do *Guice* para criar seus objetos remotos e configurar o *DWR* para públicar os serviços da aplicação disponíveis no pacote “**service**”.

2.4 Diretório “webapp”

- **css:** esta pasta possui apenas o arquivo “**app.css**”. Este é um arquivo de folhas de estilo em cascata (*Cascading Style Sheets*), ele é um mecanismo simples para adicionar estilos (fontes, cores, espaçamentos) em documentos Web.
- **img:** esta pasta possui as imagens do sistema, tais como, os ícones e a logomarca do sistema.
- **js:** dentro desta pasta há outras duas pastas “**ext**” e “**application**”. Na pasta “**ext**” esta a instalação do *framework EXTJS.* Na pasta “**application**” estão os arquivos com os códigos *JavaScripts* responsáveis por criar as interfaces da aplicação. A interface é construída através de componentes do *EXTJS.*
- **META-INF:** contém as informações credenciais do banco de dados para conexão com o PostgreSQL.
- **pages:** inclui o arquivo “**main.jsp**” que é o arquivo com toda a interface do sistema. É neste arquivo que são criados os componentes para exibição ao usuário logado no sistema. Nesta pasta também se encontra a pasta “**security**” que possui páginas JSP para as quais os usuários são redirecionados caso algum erro de autenticação ou permissão aconteça.
- **WEB-INF:** inclui o arquivo “**web.xml**” descritor do contexto de aplicação web, segundo a especificação Java Servlet/J2EE. As informações nele contidas são as configurações específicas da aplicação.
- **index.jsp:** este arquivo é acessado após o usuário ter feito o login. Este arquivo apenas redireciona para a página “**/pages/main.jsp**”.