

Padrões de de Aplicações JAVA da UECE

24 de Junho

2021

Esse documento define os padrões de estilo de código para os programadores Java do Departamento de Informática da UECE, a estrutura base de uma aplicação e o critério de versionamento.

DETIIC /UECE



UNIVERSIDADE
ESTADUAL DO CEARÁ



GOVERNO DO
ESTADO DO CEARÁ

Histórico da Revisão

Descrição	Data	Versão	Autor
Criação do documento	16/02/2009	1.0	Pablo Nóbrega
Inclusão de capítulo sobre Git flow	24/06/2021	2.0	Marcos Eduardo
Inclusão do Capítulo Evidências	24/06/2021	2.1	Marcos Eduardo

Sumário

1. PACOTES	4
1.1. Declaração	4
1.2. Estrutura	4
2. CLASSES	4
2.1. Declaração	4
3. Métodos	5
3.1. Declaração	5
4. MAPEAMENTO OBJETO-RELACIONAL	6
4.1. Declaração de Campos na Tabela	6
4.1.1. Definição de chave primária	6
4.1.2. Campos não nulos	6
4.1.3. Campos Unique	6
4.1.4. Chave estrangeira	6
4.1.5. Campos com limitação de tamanho	6
4.1.6. Data/Hora	7
5. ESTRUTURA DAS APLICAÇÕES	7
5.1. Pastas	7
5.2. Páginas	8
6. VERSIONAMENTO DE APLICAÇÕES	8
6.1. Regras do Controle de Versões	8
6.1.1. Versão em produção	8
6.1.2. Fluxo de trabalho com GIT	9
6.2. Comentários da Versão	9
5. EVIDÊNCIAS DE IMPLEMENTAÇÃO	10

1. Pacotes

1.1. Declaração

Todos os pacotes e subpacotes devem seguir o padrão **br.uece.nome_aplicacao.tipos_de_classes**.

Exemplo: **br.uece.patrimonio.view.bean**

Classes abstratas e/ou interfaces devem estar no mesmo pacote das classes concretas. Exemplo: **br.uece.nome_aplicacao.dao** deve conter a classe **SimplesDao** (interface) e as classes concretas de acesso ao banco.

Não se deve utilizar nome de pacote com palavras no plural.

Exemplo: **br.uece.nome_aplicacao.view.helpers** (**ERRADO**).

1.2. Estrutura

Deve-se usar uma arquitetura de três camadas para as aplicações web consistindo de:

- Um pacote para as classes managed bean: **br.uece.nome_aplicacao.view.bean**
- Um pacote para as regras de negócio: **br.uece.nome_aplicacao.business**
- Um pacote para as classes de persistência: **br.uece.nome_aplicacao.persistence.entity**

Em relação aos outros pacotes deve ser seguido o padrão:

- Classes de Enumerations: **br.uece.nome_aplicacao.persistence.entity.enumeration**
- Classes DAO de acesso ao banco: **br.uece.nome_aplicacao.persistence.dao**
- Classes helpers de beans: **br.uece.nome_aplicacao.view.helper**
- Classes de validação **br.uece.nome_aplicacao.view.validator**
- Classes de conversão: **br.uece.nome_aplicacao.view.converter**
- Classes POJO para fins diversos: **br.uece.nome_aplicacao.util**
- Classes do Hibernate: **br.uece.nome_aplicacao.hibernate**
- Classes de exceção: **br.uece.nome_aplicacao.exception**

2. Classes

2.1. Declaração

As classes dos pacotes devem ser declaradas da seguinte forma:

Pacote	Nome	Exemplo
br.uece.nome_aplicacao.view.bean	Nome_Entidade_ou_FuncionalidadeBean	AlunoBean
br.uece.nome_aplicacao.view.helper	Nome_Entidade_ou_FuncionalidadeBeanHelper	AlunoBeanHelper
br.uece.nome_aplicacao.view.converter	Nome_Entidade_ou_FuncionalidadeConverter	DataConverter
br.uece.nome_aplicacao.view.validator	Nome_Entidade_ou_FuncionalidadeValidator	DataValidator

br.uece.nome_aplicacao.business	Nome_Entidade_ou_FuncionalidadeBO	AlunoBO
br.uece.nome_aplicacao.persistence.dao	Nome_Entidade_ou_FuncionalidadeDao	AlunoDAO
br.uece.nome_aplicacao.persistence.entity	Nome_Entidade	Aluno
br.uece.nome_aplicacao.exception	NomeExcecaoException	LoginException

3. Métodos

3.1. Declaração

Deve ser seguido o seguinte padrão em relação a métodos:

Tipo Classe	Funcionalidade	Nome	Exemplo
Persistência	Recuperar atributo boolean	isAtributo : boolean	isAtivo
	Recuperar outros atributos	getAtributo : Tipo_Atributo	getDataNascimento
Managed Bean	Salvar entidade	saveEntidade()	saveAluno()
	Editar entidade	updateEntidade()	updateAluno()
	Excluir entidade	deleteEntidade()	deleteAluno()
	Mostrar listagem na tela	showListaNome_Listagem	showListaAlunosAtivos
	Recuperar listagem de entidade com SelectItem	getListaEntidadeWithSelectItem	getListaAlunosAtivosWithSelectItem
	Chamar página de inclusão de entidade	showCadastroEntidade	showCadastroEstudante
	Chamar alteração de entidade	prepareUpdateEntidade	prepareUpdateEstudante
Business	Salvar entidade	saveEntidade(Entidade nome_atributo)	saveAluno(Aluno aluno)
	Atualizar entidade	updateEntidade (Entidade nome_atributo)	updateAluno(Aluno aluno)
	Excluir entidade	deleteEntidade (int codigoEntidade)	deleteAluno(int codigoAluno)
	Buscar um único objeto	findEntidadeById(int codigoEntidade)	findAlunoById(int codigoAluno)
	Listagem de entidade c/ característica especial	getListaEntidades	getListaProfessoresAtivos
Dao	Salvar entidade no BD	save(Entidade nome_atributo)	save (Aluno aluno)
	Atualizar entidade no BD	update(Entidade nome_atributo)	update(Aluno aluno)
	Excluir entidade do BD	delete(int codigoEntidade)	delete(int codigoAluno)
	Buscar um único objeto	findEntidadeById(int codigoEntidade)	findById(int codigoAluno)
	Listagem de entidade c/ característica especial	getListaEntidades	getListaProfessoresAtivos

Outros métodos possíveis:

Tipo de Classe	Funcionalidade	Nome	Exemplo
Business	Salvar em banco entidade especial	saveEntidade(Entidade entidade)	salvarAlunoNovato(Aluno aluno)
	Ajustar atributos de entidade antes de salvar no banco. Ex.: transformar String em branco por nulo	ajustaEntidade(Entidade entidade)	ajustaAluno(Aluno aluno)

Recuperar listagem com determinado filtro	<code>selecionarEntidadeTipo_Filtro()</code>	<code>selecionarAlunosSemProfessor()</code>
Recuperar listagem completa de determinada entidade	<code>selecionarTodosEntidade()</code>	<code>selecionarTodosAlunos()</code>

4. Mapeamento Objeto-Relacional (MOR)

4.1. Declaração de Campos na Tabela

Deve-se ter especial atenção ao declarar os atributos para que todas as constraints, o tamanho máximo dos campos e os tipos sejam gerados corretamente no banco de dados. Abaixo temos alguns exemplos:

4.1.1. Definição de chave primária

```
@Id
@Column(name="cd_aluno", nullable=false)
@SequenceGenerator(name = "sequence_aluno", sequenceName = "sequence_aluno")
@GeneratedValue(strategy = GenerationType.AUTO, generator = "sequence_aluno")
private Integer idAluno;
```

Para chaves primárias com geração de ID automática

4.1.2. Campos não nulos

Deve-se atentar para campos que não possam ficar nulos no banco, setando a propriedade nullable para false, como no exemplo abaixo.

```
@ManyToOne
@Column(name = "nome", nullable=false)
private String nome;
```

Lembrar sempre de colocar a propriedade

4.1.3. Campos Unique

```
@Column(name = "cpf", nullable=false, unique=true)
private String cpf;
```

Configura o campo cpf como unique

4.1.4. Chave estrangeira

```
@ManyToOne
@JoinColumn(name = "cd_professor", nullable=false)
private Professor professor;
```

Para chaves estrangeiras não nulas

4.1.5. Campos com limitação de tamanho

```
@Column(name="nome", nullable=false, length=120)
private String nomeEstudante;
```

Tamanho máximo de 120 caracteres

4.1.6.Data/Hora

Deve-se indicar se o campo possuirá somente data ou data e hora.

```
@Temporal(TemporalType.TIMESTAMP)  
@Column(name="dt_movimento", nullable=false)  
private Date dataMovimento;
```

Para campos com data
e hora

5. Estrutura das Aplicações

5.1. Pastas

As aplicações devem ter a estrutura representada pela figura. A seguir estão descritas informações sobre o que deve conter em cada uma das pastas:

- **script_banco***: script de criação do banco;
- **jrxml***: esboço dos relatórios salvos no iReport;
- **jasper***: relatório *jrxml* compilado pelo iReport;
- **css**: folhas de estilo do sistema;
- **imagens**: imagens utilizadas no sistema;
- **js**: arquivos de javascript;
- **paginas**: páginas jsp ou jspix;
- **pdf**: arquivo pdf gerado de um relatório;



*As pastas *jrxml*, *jasper* e *script_banco* não devem ter seus conteúdos disponibilizados externamente, portanto devem ficar fora de *WebContent*.

5.2. Páginas

Todas as páginas, imagens, css e arquivos de script devem possuir o nome completo, minúsculo e interligado por “_” (underline) entre cada palavra. Exemplo: *lista_alunos.jspx*, *desativar_usuario.png*, etc.

Deve ser seguido o seguinte padrão para os nomes de tipos de páginas:

Tipo de Página	Padrão	Exemplo	Pasta no Projeto
Listagem de entidade	<i>lista_entidades.jspx</i>	<i>lista_alunos.jspx</i>	<i>listagem</i>
Listagem de entidade com característica especial	<i>lista_entidades_característica.jspx</i>	<i>lista_alunos_ativos.jspx</i>	<i>listagem</i>
Cadastrar ou editar entidade	<i>cadastro_entidade.jspx</i>	<i>Cadastro_aluno.jspx</i>	<i>cadastro</i>
Filtro de relatórios e página de exibição do relatório	<i>filtro_relatorio_nomerelatorio.jspx</i>	<i>filtro_relatorio_alunos_ativos.jspx</i>	<i>relatorio</i>

6. Versionamento de Aplicações

6.1. Regras do Controle de Versões

Deve-se utilizar o padrão de três dígitos para as versões. Exemplo: **1.0.2**.

Ao se gerar o primeiro release, a versão do sistema passa a ser 1.0.0. A partir daí, caso o sistema sofra alterações, cada desenvolvedor que for realizar as alterações deve criar sua própria branch e trabalhar diretamente nela.

Ao terminar o desenvolvimento da branch, o desenvolvedor deverá efetuar um merge com o trunk.

6.1.1. Versão em produção

Para versões que estejam em produção. Ela só deve ser criada quando as alterações forem finalizadas e transformadas em um release do produto. A primeira TAG criada deve ter a numeração 1.0.0.

Caso o sistema sofra alterações profundas, devem ser zerados os dois segundos dígitos da versão. Exemplo: a versão anterior do sistema era 1.2.5, mas foi trocada a interface gráfica de richfaces para icefaces. Nesse caso, muda-se a versão do sistema diretamente para 2.0.0, sugerindo mudanças de impacto.

No caso de correção de falhas da versão em produção, altera-se o terceiro dígito. Exemplo: 1.0.3, 1.0.4, etc. Ao incluir novas funcionalidades, alterar o segundo dígito. Exemplo: 1.1.3, 1.2.2, etc.

6.1.2. Fluxo de trabalho com GIT

O fluxo de trabalho deve ser baseado no **GitFlow**. Este auxilia o desenvolvimento contínuo de software e a implementação de práticas de DevOps.

O fluxo de trabalho do **GitFlow** é distribuído em 5 branches (ou ramos) divididas em dois grupos de:

Branchs Principais (ou fixas):

- Branch “**master**” para as entregas oficiais
- Branch “**develop**” para integração entre as branches

Obs.: A partir delas que são originadas as novas implementações.

Branchs Secundárias (ou temporárias):

- Branch “ **feature/*** ” para novas implementações
- Branch “ **hotfixes/*** ” para correções de bugs urgentes
- Branch “ **releases/*** ” para novas versões do sistema

As branches “**bugfix/xxxx**” também são branch temporárias. São branches que contém uma **correção não emergencial** que pode ser levada para a próxima release. Seu ciclo de vida é semelhante ao da feature, Somente muda o nome por respeito à nomenclatura da classificação da implementação.

6.2. Comentários da Versão

Sempre incluir uma mensagem explicativa da versão criada indicando o que foi modificado no sistema.

Exemplo de comentário na correção de uma BRANCH de correção:

Criação BRANCH: 1.0.1

- * Correção da funcionalidade xxx para realizar a inclusão yyy;
- * Correção do somatório de pontos do relatório zzzz.

Obs.: após realizar qualquer inclusão com sucesso, deve ser feito um “merge” com a “TRUNK”.

Exemplo de comentário de criação de TAG:

Criação TAG: 2.0.5

- * Versão em produção lançada em 01/01/2009

Obs.: TAG e TRUNK devem estar sempre iguais.

7. Evidências de Implementação

Para exemplificar evidências de implementação, este Capítulo apresenta algumas telas de um dos sistemas da UECE que foi codificado com base nas principais recomendações descritas neste documento.

7.1. Sispessoal

O Sispessoal é o Sistema de Administração de Recursos Humanos da Universidade Estadual do Ceará. Este é um aplicativo web utilizado no gerenciamento e recuperação das informações dos servidores docentes e técnico-administrativos.

7.2. Principais Funcionalidades

O sistema conta com as seguintes funcionalidades principais: frequência dos professores, afastamento, ascensão, atividade extra, averbação, cargo e estrutura, controle de portaria, controle de resolução, qualificação, processos e protocolos, listagem de servidores baseada em filtro, bem como outros cadastros básicos.

A figura 1 ilustra a tela Principal do sistemas, com menu e atalhos para as principais funcionalidades do sistema.

Figura 1 - Tela Principal



As figuras a seguir ilustram, respectivamente, três exemplos de funcionalidades utilizadas no Sistema. São elas: Frequência de Professores (Figura 2), Afastamentos (Figura 3) e Controle de Ascensão Funcional de servidores da UECE (Figura 4).

Figura 2 - Análise de Frequência de Professores

Sistema

Cadastros Básicos

Cadastros Funcionais

Importação de servidores

Censo / Ranking THE

Usuário: marcos.eduardo

Sessão 50m31s

SAIR

Frequência de Professores

Voltar

Gerar Frequência

Listar Frequências Geradas

Histórico

Home

Observações:

1) As horas de atividades de ensino de graduação são preenchidas automaticamente com base no registro de aulas da caderneta eletrônica. Assim, é necessário que o professor informe, na caderneta eletrônica, a data que realmente ministrou a aula para que seja considerada na frequência.

2) As demais atividades são preenchidas com base nas horas semanais do PAD do professor.

Neste caso, é levado em consideração o limite de horas estabelecido na resolução que regulamenta o PAD.

Na frequência, as horas das atividades de ensino são multiplicadas por 2.

Centro/Facul./Inst.: Centro de Ciências e Tecnologia - CCT

Curso: Ciência da Computação (014)

Mês/ano: 04/2019

Status: Em Análise pelo DEPEs

Fator Multiplicador (horas, Grad.): 2

Frequência do Curso: Ciência da Computação (014) - Mês: 04/2019

Dados referentes a: 04/06/2019 13:34:02

Validado	Professor	Atividades de Ensino	Outras Atividades	Total	Carga Horária	Validado	Afastamento	Observação
1	<div></div> 009642-1-9 - ANA LUIZA BESSA DE PAULA BARROS	<div>Graduação32h</div> <div>Pós-Graduação0h</div> <div>Subtotal32h/mês</div>	<div>Orientação0h</div> <div>Pesquisa0h</div> <div>Extensão0h</div> <div>Administração120h</div> <div>Subtotal120h/mês</div>	<div>Ensino32h</div> <div>Outras Ativ.120h</div> <div>Total160h/mês</div>	<div>Regime (DE) 40h</div> <div>x4</div> <div>CH Mensal160h</div>	<div></div> Validado pelo coordenador em 04/06/2019 13:34:02		Faltando contabilizar 4h semanais de NDE.
2	<div></div> 009930-1-4 - ANDRE LUIZ MOURA DOS SANTOS	<div>Graduação128h</div> <div>Pós-Graduação0h</div> <div>Subtotal128h/mês</div>	<div>Orientação0h</div> <div>Pesquisa32h</div> <div>Extensão0h</div> <div>Administração0h</div> <div>Subtotal32h/mês</div>	<div>Ensino128h</div> <div>Outras Ativ.32h</div> <div>Total160h/mês</div>	<div>Regime (DE) 40h</div> <div>x4</div> <div>CH Mensal160h</div>	<div></div> Validado pelo coordenador em 04/06/2019 13:34:02		
		<div>Graduação64h</div> <div>Pós-Graduação0h</div> <div>Subtotal64h/mês</div>	<div>Orientação0h</div> <div>Pesquisa0h</div> <div>Extensão0h</div> <div>Administração0h</div> <div>Subtotal0h/mês</div>	<div>Ensino64h</div> <div>Outras Ativ.0h</div> <div>Total64h/mês</div>	<div>Regime40h</div> <div>x1</div> <div>CH Mensal40h</div>	<div></div> Validado pelo coordenador em 04/06/2019 13:34:02		

Figura 3 - Afastamentos

SISTEMA DE PESSOAL - 1.66.4

Sistema

Cadastros Básicos

Cadastros Funcionais

Importação de servidores

Censo / Ranking THE

Usuário: marcos.eduardo

Sessão 50m31s

SAIR

Listagem de Afastamentos

Gerar Planilha

Imprimir

Emitir Declaração

Solicitar

Home

Figura 4 - Ascensão Funcional de servidores da UECE

UNIVERSIDADE

ESTADUAL DO CEARÁ

Sistema

Cadastros Básicos

Cadastros Funcionais

Importação de servidores

Censo / Ranking THE

Usuário: marcos.eduardo

Sessão 50m31s

X

SAIR

Listagem de Ascensões, Estágios Probatórios e Incentivos

Listas das ascensões solicitadas e/ou autorizadas (aprovadas).

Imprimir

Enviar Declaração

Solicitar

Home

Legenda: (X) - Autorizar ou Indeferir; (↶) - Reverter; (✎) - Editar; (⊖) - Excluir; (📄) - Declaração; (📄) - Indeferido; (✓) - Autorizado

Pesquisar Ascensões (Clique aqui para visualizar filtro)

Critério de Pesquisa: NOME (Funcionário)

Centro / Departamento

Nº SPU / VÍPROC:

Curso / Lotação

Status:

Função Geral

Período: a

Função Pretendida *

Regime de Trabalho

Tipo de solicitação:

Pesquisar

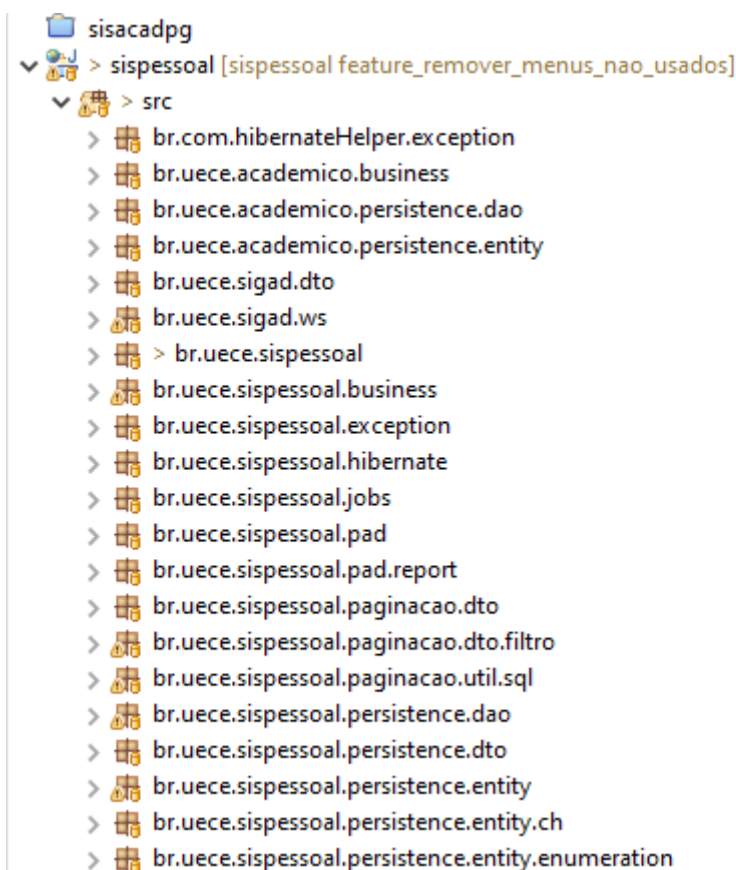
Limpar

Código	Funcionário / Regime / Titulação	Função e Nível Pretendidos	Titulação Pretendida	PROCESSO / PORTARIA / DOE	Datas	Curso (Centro)	Tipo de solicitação	Situação					
10724	006800-1-X - ANA CRISTINA DE MORAES (Regime: DE) Titulação Atual: DOUTOR Função Atual: Professor Adjunto - J	Professor Assistente (Nível: E)		SPU / VÍPROC: 09172317-5 PORTARIA: 26/2010 Motivo: PROGRESSÃO	Iniciado em: 07/11/2008 DOE: 19/05/2010	Pedagogia (FACEDI)	PROGRESSÃO	AUTORIZADA					
10725	006752-1-9 - ARICLECIO CUNHA DE OLIVEIRA (Regime: DE) Titulação Atual: POS-DOUTOR Função Atual: Professor Adjunto - I	Professor Assistente (Nível: F)		SPU / VÍPROC: 09171537-7 PORTARIA: 68/2010 Motivo: PROGRESSÃO	Iniciado em: 25/05/2009 DOE: 19/05/2010	Educação Física (CCS)	PROGRESSÃO	AUTORIZADA					
10726	006854-1-9 - JOUBERTH MAX MARANHÃO PIORSKY AIRES (Regime: DE) Titulação Atual: POS-DOUTOR Função Atual: Professor Adjunto - J	Professor Adjunto (Nível: J)		SPU / VÍPROC: 09657392-9 PORTARIA: 489/2010 Motivo: PROGRESSÃO	Iniciado em: 26/12/2008 DOE: 19/05/2010	Ciências Sociais (CH)	PROGRESSÃO	AUTORIZADA					
	006722-1-1 - SARAH DIVA DA SILVA IPIRANGA (Regime: DE)			SPU / VÍPROC: 10129196-5	Período:								

7.3. Pacotes

A Figura 4 exibe a parte da Estrutura de Pacotes criado para agrupar as classes Java do Sistema, conforme recomendações deste documento.

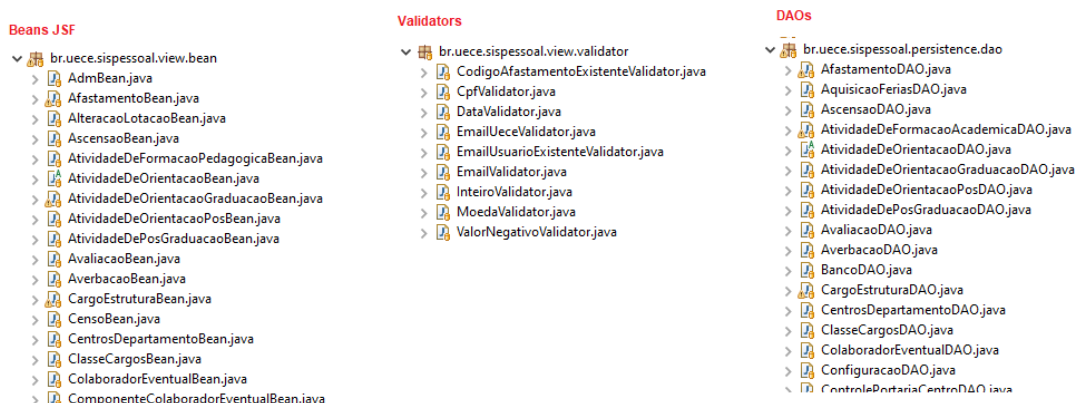
Figura 5 - Estrutura de Pacotes do Sispessoal



7.4. Classes

A Figura 6 exibe algumas classes criadas com nomes específicos sugeridos nesse documento.

Figura 6 - Exemplos de Classes



7.5. Métodos

A Figura 7 exibe três métodos criados com base no padrão estabelecido para nomenclatura de métodos deste documento.

Figura 7 - Métodos

```
30 import static br.uece.sispeessoal.view.helper.DeclaracaoBeanHelper.getDeclaracaoBean;
81
82 public class AscensaoBean {
83
84     private List<SelectItem> funcaoEspecificaselectItem;
85     private List<SelectItem> funcaoGeraisselectItem;
86     private List<SistemaProtocoloUnico> spuList;
87     private List<SelectItem> funcaoSelectItem;
88     private List<Funcionario> funcionarioList;
89     private List<SelectItem> instrucaoSelectItem;
90
91     public String prepareUpdateAscensao() {}
106
107     public void updateAscensao() {}
136
137     public void reverterParaSolicitado() {}
175
176     private void validarprepareAutorizacaoAscensao() throws AscensaoNaoAutorizada {}
186
187 }
```

7.6. Mapeamento Objeto-Relacional

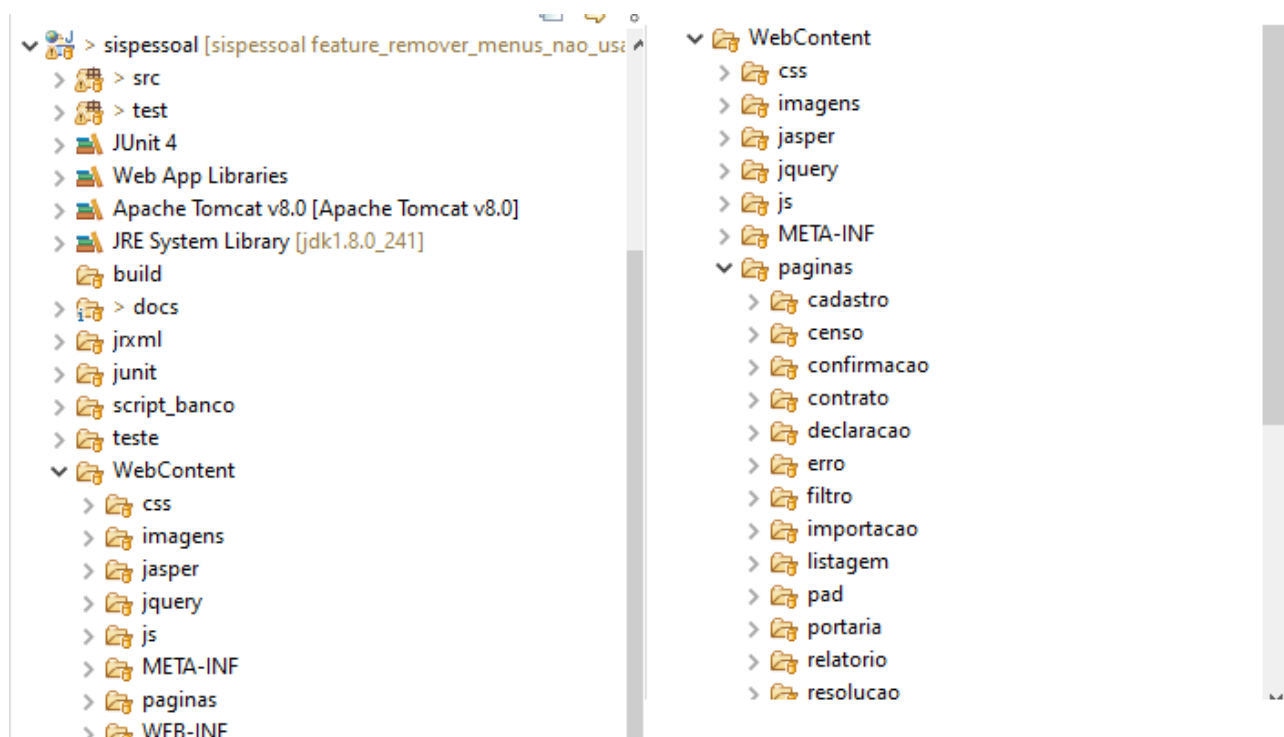
Esta seção exibe algumas exemplos de mapeamento de entidades no Sistema Sispeessoal. A Figura 8 exibe o mapeamento da chave primária criado para entidade Ascensão utilizada na funcionalidade de Ascensão Funcional dos servidores da UECE, bem como ilustra o mapeamento de uma atributo do tipo data que não permite inclusão de valores nulos

Figura 8 - Mapeamento da entidade “Ascensao.java” com a tabela “pessoal.ascensao”

```
42 @Entity
43 @Table(name = "ascensao", schema = "pessoal")
44 public class Ascensao implements Serializable {
45
46     private static final long serialVersionUID = 1L;
47
48     @Id
49     @Basic
50     @Column(name = "id_ascensao", nullable = false)
51     @SequenceGenerator(allocationSize = 1, sequenceName = "seq_ascensao", name = "seq_ascensao")
52     @GeneratedValue(generator = "seq_ascensao", strategy = GenerationType.SEQUENCE)
53     private Integer idAscensao;
54
55     @Column(name = "dt_inicio", nullable = false)
56     @Temporal(TemporalType.DATE)
57     private Date dtInicio;
58
59     @Column(name = "dt_fim", nullable = true)
60     @Temporal(TemporalType.DATE)
61     private Date dtFim;
62
63     @ManyToOne
64     private Funcao funcao;
65 }
```

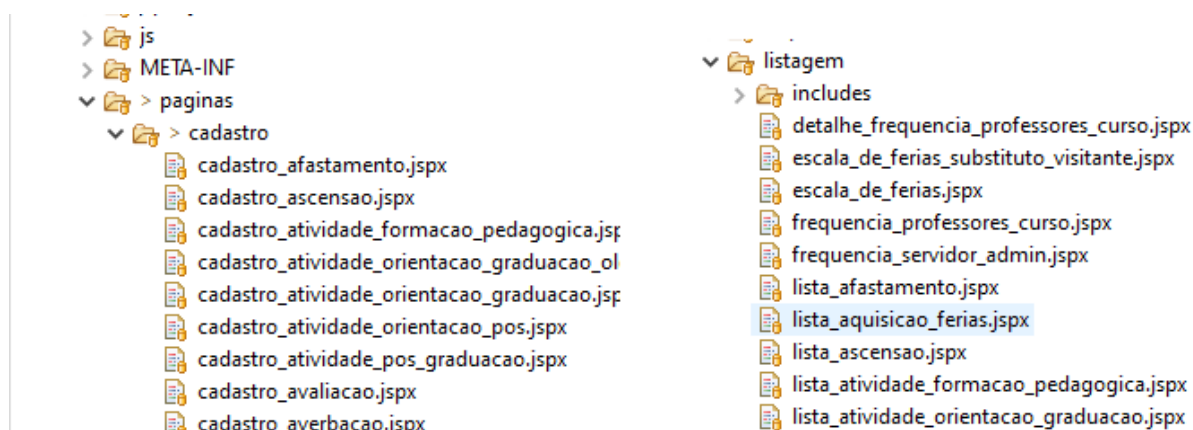
7.7. Pastas

A figura a seguir ilustra o padrão de pastas nomeadas conforme estabelecido neste documento:



7.8. Páginas

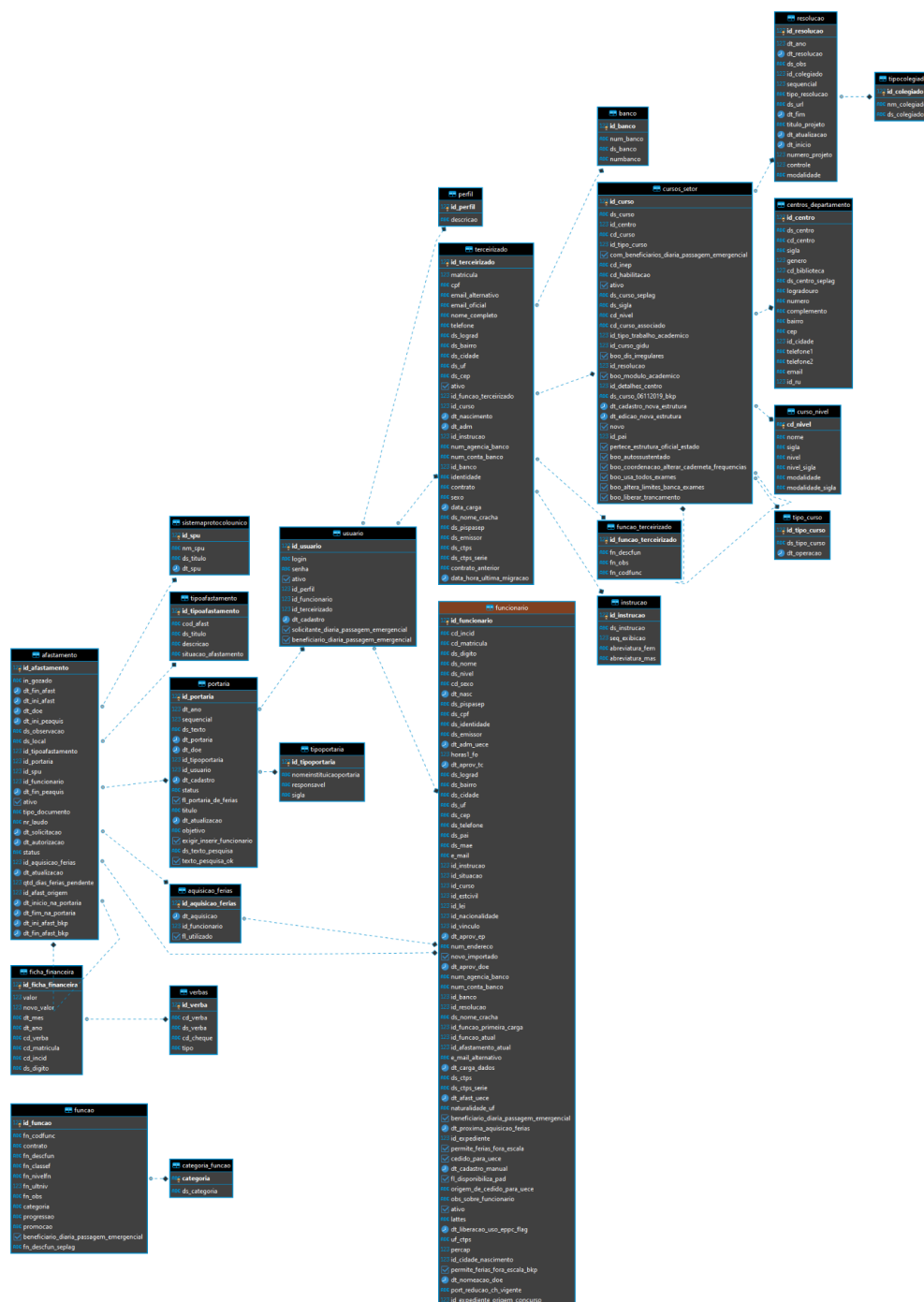
Todas ou a maior parte das páginas, imagens, css e arquivos de script possuem o mesmo padrão definido neste documento, com o nome completo, minúsculo e interligado por “_” (underline) entre cada palavra, conforme estabelecido neste documento. A figura a seguir ilustra dois exemplos desse padrão utilizado:



7.9. Diagrama Entidade-Relacionamento do Sispessoal.

A figura a seguir ilustra o DER com as principais tabelas da base de dados do sistema Sispessoal. Caso não seja possível visualizar, clique no link a seguir para visualizá-lo em formato PDF.

Link par o PDF: [Diagrama Entidade-Relacionamento - Sispeessoal.pdf](#)



Obs.: O Dbeaver foi utilizado como ferramenta de engenharia reversa para gerar este Diagrama Entidade-Relacionamento - DER.