Histórico de Versões

Data	Versão	Descrição	Autor	Revisor	Aprovado por
27/06/2011	0.1	Criação do documento para a versão 2.1.1 do *framework*	Cleverson Sacramento	<DES>	<LP>

Índice

Documento de Arquitetura de Software

1.Objetivo do Documento

O Documento de Arquitetura de Software provê uma visão geral da arquitetura através de diferentes tipos de visões para descrever os diferentes aspectos do sistema.

2.Objetivos e Restrições da Arquitetura

<Descreve os objetivos e as restrições para a definição da arquitetura, apresentando os principais aspectos da aplicação. Quando for pertinente, listar os Casos de Uso mais críticos para a arquitetura juntamente com os respectivos desafios.>

> **Documentação:**<Insira aqui as documentações e as imagens dos diagramas de caso de uso contidos na pasta *Elementos do Modelo Arquiteturalmente Significativos,* dentro do *Modelo de Requisitos,* que se encontra dentro de *Use Case View* >

3.Elementos Arquiteturalmente Significativos

<Descreve as principais interfaces e componentes da solução, registrando também a presença ou ausência de mecanismos arquiteturais inovadores (arquitetura crítica). Quando necessário, indicar necessidade de prototipação e/ou gestão de riscos arquiteturais.>

4.Descrição da Arquitetura

<Descreve as camadas, mecanismos e outras abstrações da arquitetura que se julguem necessárias.>

4.1. Módulo Core

Escrever

4.1.1. Controladores de camadas

O *framework* provê estereótipos para serem utilizados pelas classes das camadas das aplicações.

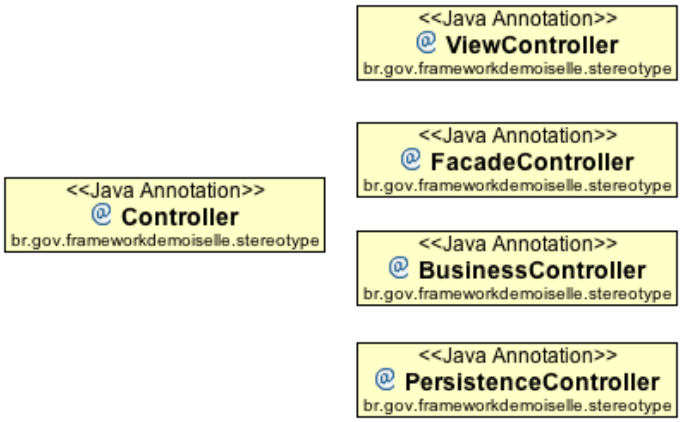

Figura 1: Controladores de camadas

Elemento	Descrição
Controller	Possibilita a criação de outros controladores de camada além dos pré-estabelecidos pelo *framework*.
ViewController	Define a classe como um controlador da camada de apresentação.
FacadeController	Define a classe como um controlador da fachada.
BusinessController	Define a classe como um controlador da camada de negócio.
PersistenceController	Define a classe como um controlador da camada de persistência.

4.1.2. Injeção

Possibilita a injeção de classes que não são nativas do CDI, garantindo a integração com outras bibliotecas.

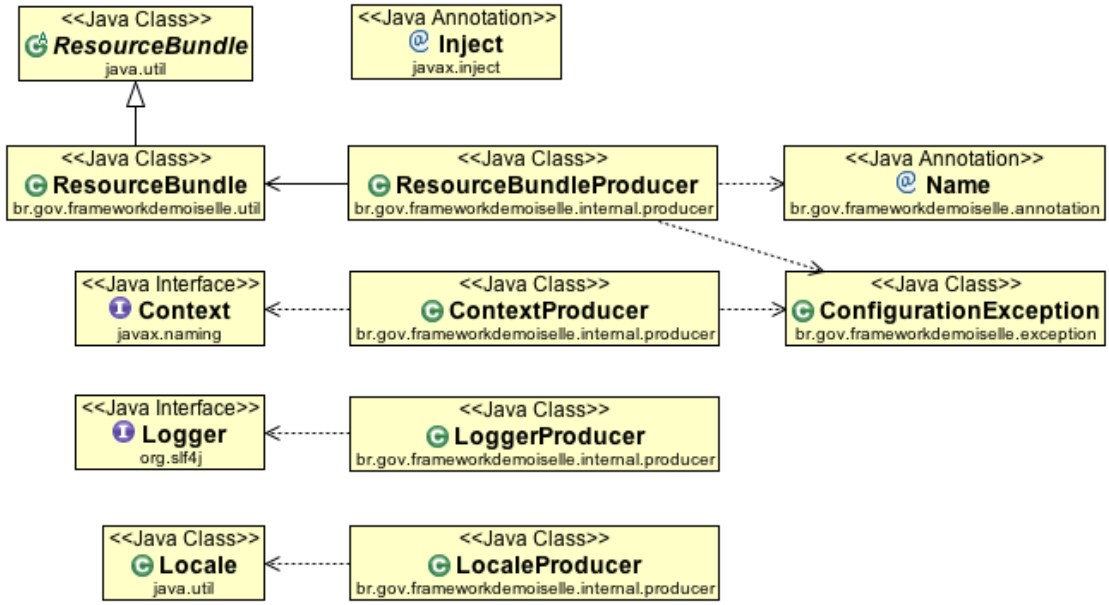

Figura 2: Produtores de injeção

Elemento	Descrição
Inject	Anotação do CDI que possibilita a injeção nos atributos.
ResourceBundleProducer	Fabrica e possibilita a injeção de ResourceBundle.
Name	Informa ao produtor de ResourceBundle o nome do arquivo de propriedades que contém as mensagens.
ResourceBundle	Classe nativa fabricada pelo produtor que possui um utilitário similar no *framework*.
ConfigurationException	Lançado pelos produtores caso haja má parametrização para o devido funcionamento.
ContextProducer	Fabrica e possibilita a injeção de Context.
Context	Classe nativa fabricada pelo produtor.
LoggerProducer	Fabrica e possibilita a injeção de Logger.
Logger	Interface para geração de log.

Elemento	Descrição
LocaleProducer	Fabrica e possibilita a injeção de Locale com base na instância da JVM.
Locale	Classe nativa fabricada pelo produtor.

Utiliza o recurso Portable Extensions para capturar eventos do CDI e gerar mensagens de *log* na inicialização e finalização da aplicação.

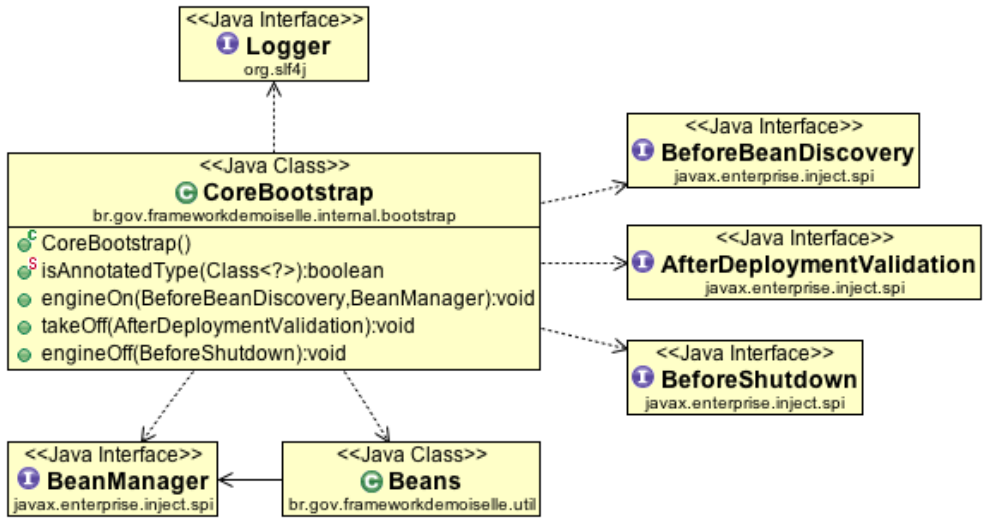

Figura 3: Bootstrap de injeção

Elemento	Descrição
CoreBootstrap	Observa os eventos lançados pelo CDI.
BeforeBeanDiscovery	Evento do CDI que, quando disparado, o CoreBootstrap gera *log* informando a versão do framework e guarda a instância do BeanManager.
AfterDeploymentValidation	Evento do CDI que, quando disparado, o CoreBootstrap gera *log* informando que a aplicação foi completamente carregada.
BeforeShutdown	Evento do CDI que, quando disparado, o CoreBootstrap gera *log* informando que a aplicação foi completamente finalizada.
Logger	Classe de log utilizada pelo CoreBootstrap.
Beans	Utilitário que facilita o acesso ao BeanManager.
BeanManager	Gerenciador de *beans* do CDI.

4.1.3. Inicializador e finalizador

Funcionalidade que permite que métodos sejam executados na inicialização da aplicação.

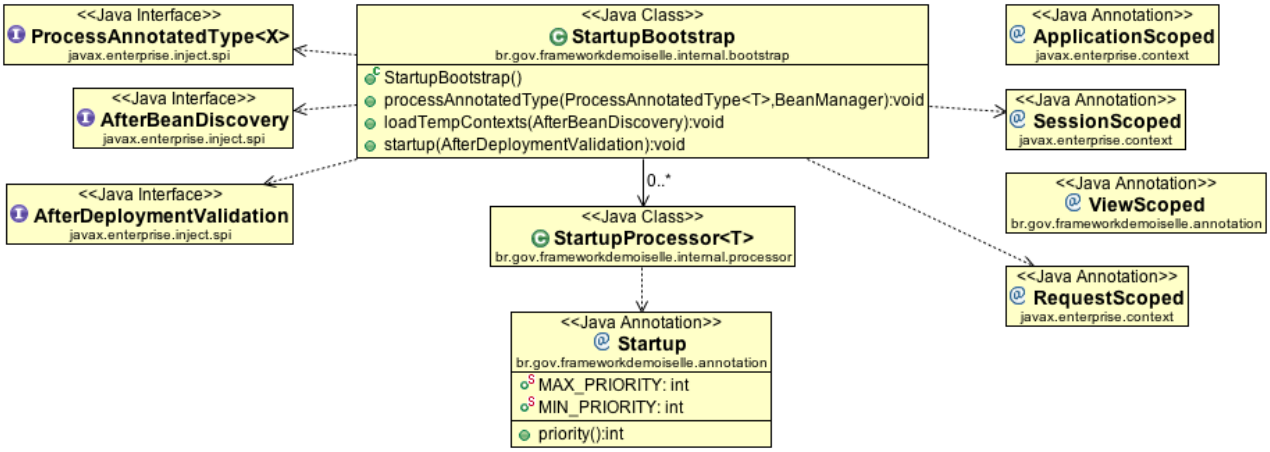

Figura 4: Inicializador

Elemento	Descrição
StartupBootstrap	Observa os eventos lançados pelo CDI.
ProcessAnnotedType	Evento do CDI que, quando disparado, o StartupBootstrap detecta todos os métodos anotados com @Startup e cria um processador para cada.
AfterBeanDiscovery	Evento do CDI que, quando disparado, o StartupBootstrap carrega pseudo-contextos para execução dos processadores.
AfterDeploymentValidation	Evento do CDI que, quando disparado, o StartupBootstrap executa todos os processadores. No final desregistra todos os pseudo-contextos para evitar conflitos.
StartupProcessor	Quando invocado, executa o método *startup* associado.
Startup	Marca um método como inicializador, definindo a prioridade se necessário.
ApplicationScoped	Escopo CDI registrado temporatiamente para possibilitar a execução dos processadores.
SessionScoped	Escopo CDI registrado temporatiamente para possibilitar a execução dos processadores.
ViewScoped	Escopo do *framework* registrado temporatiamente para possibilitar a execução dos processadores.
RequestScoped	Escopo CDI registrado temporatiamente para possibilitar a execução dos processadores.

Disponibiliza facilidades para executar métodos na finalização da aplicação.

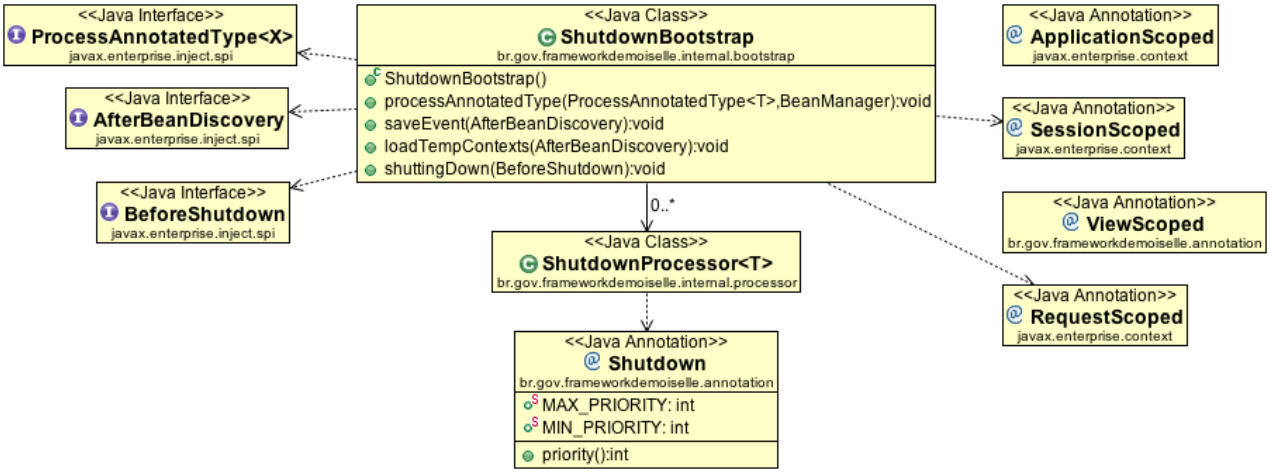

Figura 5: Finalizadores

Elemento	Descrição
ShutdownBootstrap	Observa os eventos lançados pelo CDI.
ProcessAnnotedType	Evento do CDI que, quando disparado, o ShutdownBootstrap detecta todos os métodos anotados com @Startup e cria um processador para cada.
AfterBeanDiscovery	Evento do CDI que, quando disparado, o ShutdownBootstrap carrega pseudo-contextos para execução dos processadores.
BeforeShutdown	Evento do CDI que, quando disparado, o ShutdownBootstrap executa todos os processadores.
ShutdownProcessor	Quando invocado, executa o método *shutdown* associado.
Shutdown	Marca um método como finalizador, definindo a prioridade se necessário.
ApplicationScoped	Escopo CDI registrado temporatiamente para possibilitar a execução dos processadores.
SessionScoped	Escopo CDI registrado temporatiamente para possibilitar a execução dos processadores.
ViewScoped	Escopo do *framework* registrado temporatiamente para possibilitar a execução dos processadores.
RequestScoped	Escopo CDI registrado temporatiamente para possibilitar a execução dos processadores.

4.1.4. Configuração

Mecanismo que oferece facilidades para leitura de arquivos de configuração.

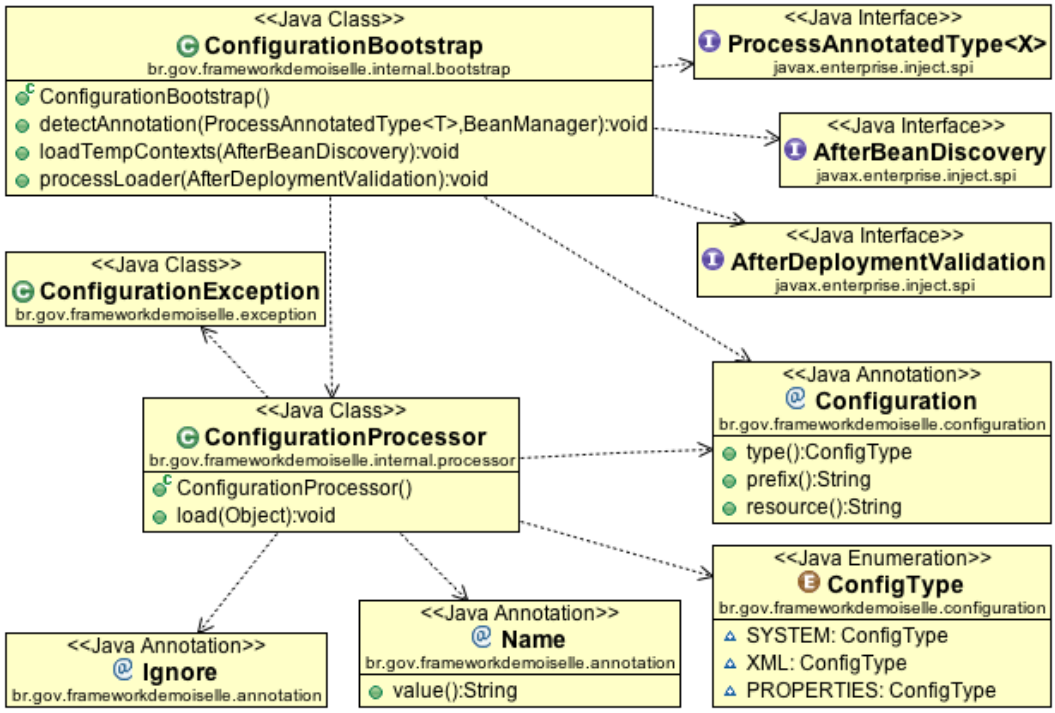

Figura 6: Configuração

Elemento	Descrição
ConfigurationBootstrap	Observa os eventos lançados pelo CDI.
ProcessAnnotedType	Evento do CDI que, quando disparado, o ConfigurationBootstrap detecta todos os métodos anotados com @Configuration para posterior processamento.
AfterBeanDiscovery	Evento do CDI que, quando disparado, o ShutdownBootstrap carrega pseudo-contextos para execução dos processadores.
AfterDeploymentValidation	Evento do CDI que, quando disparado, o ShutdownBootstrap invoca o processamento de todas as classes de configuração detectadas. No final desregistra todos os pseudo-contextos para evitar conflitos.
ConfigurationProcessor	Processa as classes de configuração carregando em cada atributo as informações contidas no arquivo.
Configuration	Marca uma classe para ser carregada do arquivo de configuração informado.
ConfigType	Indica o tipo do arquivo de configuração.
Name	Indica a chave no arquivo de configuração a ser carregada no atributo.
Ignore	Ignora o atributo no carragemento das configurações.
ConfigurationException	Exceção lançada caso haja alguma falha de parametrização da funcionalidade por parte da aplicação, conduzindo o usuário do *framework* para corrigir o erro.

4.1.5. Mensagem

Provê um mecanismo de armazenamento de mensagem para ser acessado a partir de qualquer camada.

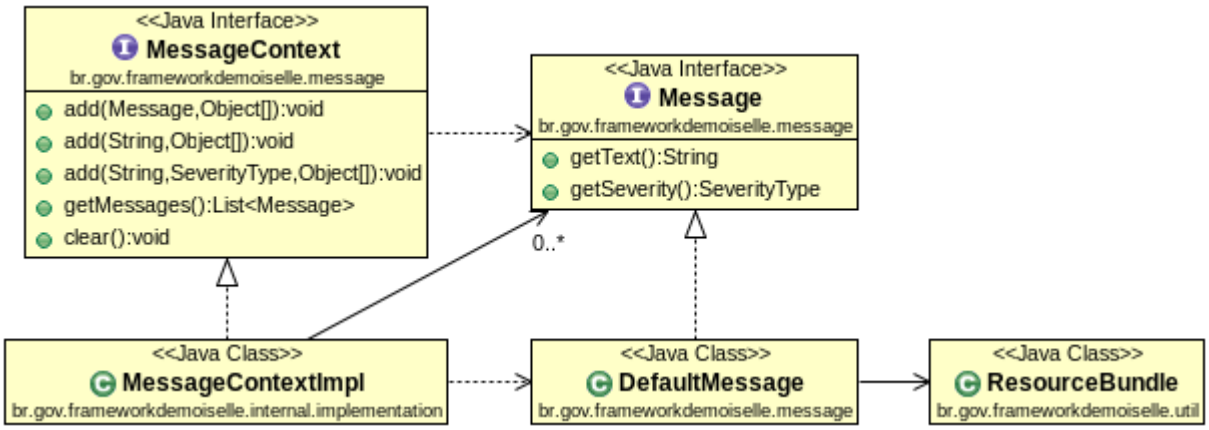

Figura 7: Mensagem

Elemento	Descrição
MessageContext	Representa a fachada que provê as funcionalidades para incluir, obter e limpar o *cache* de mensagens.
MessageContextImpl	Implementação do comportamento da fachada.
Message	Representa uma mensagem a ser armazenada.
DefaultMessage	Implementação padrão da interface Message que provê facilidades para internacionalização das mensagens.
ResourceBundle	Obtém as mensagens a partir do arquivos messages.properties.

4.1.6. Exceção

O mecanismo de tratamento de exceção baseado em anotação.

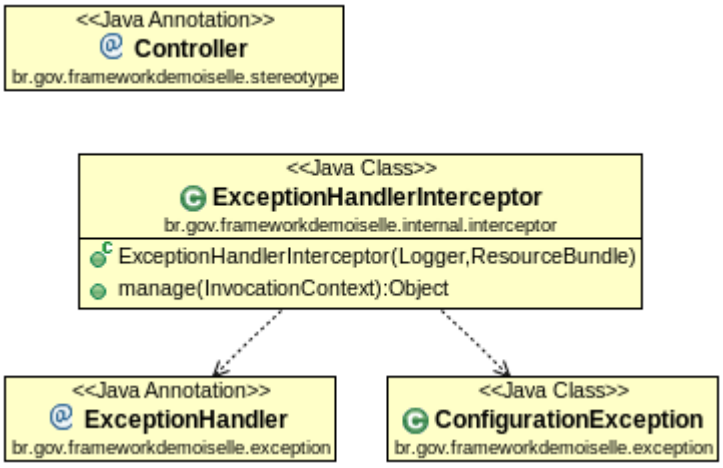

Figura 8: Exceção

Elemento	*Descrição*
Controller	Apenas as classes controladoras de camada poderão fazer uso do mecenismo de tratamento de exceções.

Elemento	*Descrição*
ExceptionHandler	Marca o método tratador de exceção, o qual deve receber como parâmetro o tipo de exceção que será tratada.
ExceptionHandlerInterceptor	Intercepta as chamadas aos métodos dos controladores de camadas e delega o tratamento da exceção para o método adequado da própria classe, caso o tipo de exceção seja compatível.
ConfigurationException	Caso o método tratador seja ambíguo ou defina mais de um parâmetro, esta exceção é lançada.

4.1.7. Transação

O mecanismo de controle de transação oferece elementos para marcar métodos transacionais, que serão tratados pelo interceptador com base nas estratégias de transação.

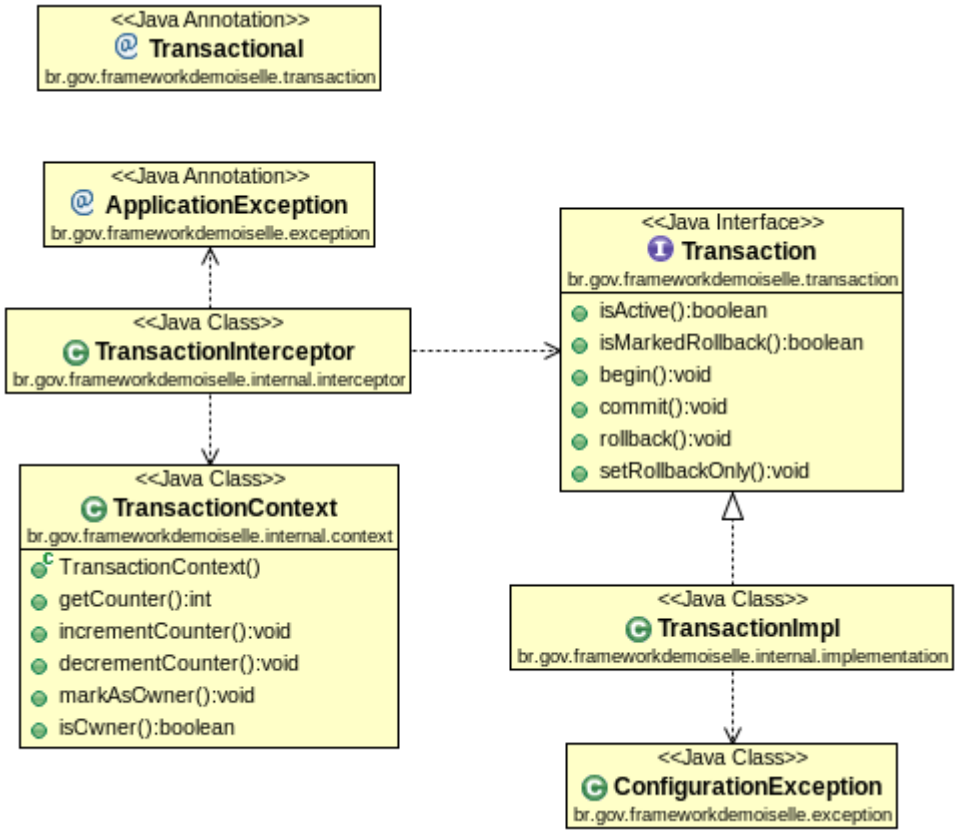

Figura 10: Transação

Elemento	**Descrição**
Transactional	Marca o método transacional que deve ser gerenciado pelo mecanismo de controle de transação.
TransactionInterceptor	Intercepta as invocações aos métodos transacionais e executa o gerenciamento da transação, delegando para as tarefas específicas para o elemento Transaction. É responsável por iniciar, completar ou desfazer sessões transacionais.
ApplicationException	Caso a invocação do método concreto lance uma exceção estereotipada como ApplicationException, o mecanismo verificará a necessidade ou não de desfazer a transação tomando como base as informações definidas na anotação. Se a exceção não for de aplicação, a sessão transacional será desfeita.

Elemento	Descrição
TransactionContext	Guarda informações referentes à sessão transacional, possibilitando o controle de métodos transacionais aninhados, garantindo a conclusão da sessão no último nível de desembpilhamento.
Transaction	Interface que estabelece o contrato para implementação das estratégias de transação.
TransactionImpl	Caso nenhuma estratégia de transação seja definida, esta implementação assume o controle e lança exceções sempre que métodos transacionais são invocados.
ConfigurationException	São lançadas pelo TransactionImpl para orientar o usuário do *framework* definir estratégias concretas.

4.1.8. Template

Oferece moldes para facilitar a criação de classes padronizadas, as quais devem implementar, herdar ou modificar os *templates*.

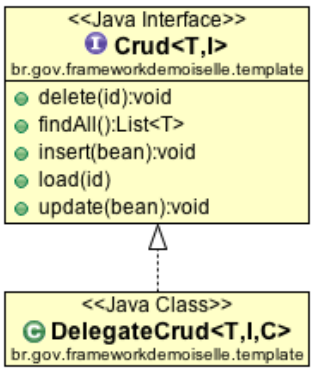

Figura 12: Template

Elemento	Descrição
Crud	Interface que estabelece o contrato para classes CRUD.
DelegateCrud	Delegação das chamadas para outra classe CRUD definida via *generics*.

4.1.9. Paginação

Oferece mecanismos para armazenamento de informações referentes à paginação, provendo insumos para funcionalidades das extensões.

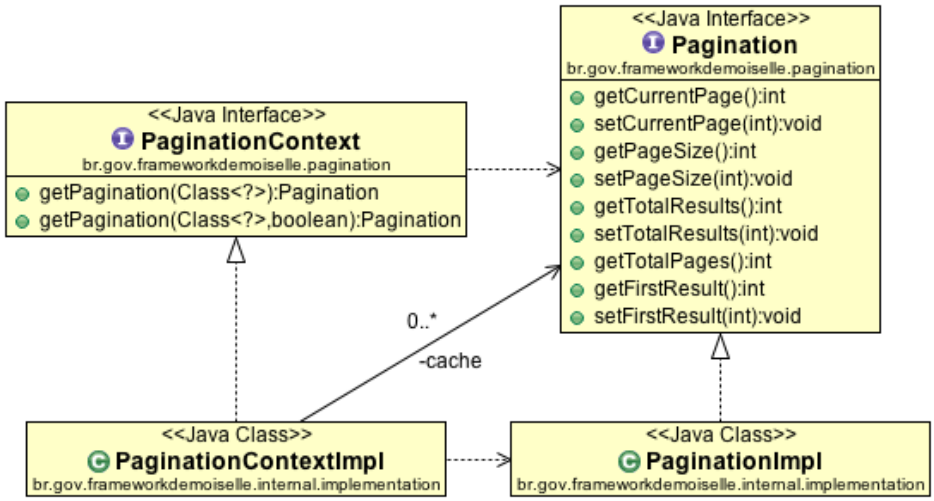

Figura 13: Paginação

Elemento	Descrição
PaginationContext	Representa a fachada que provê as funcionalidades para obter objetos do tipo Pagination.
PaginationContextImpl	Implementação do comportamento da fachada de paginação.
Pagination	Contém todas as informações necessárias para representar uma página.
PaginationImpl	Implementação padrão que guarda o estado de cada propriedade definida em Pagination.

4.1.10. Segurança

O mecanismo de segurança está subdividido em duas funcionalidades: autenticação e autorização.

A funcionalidade de autenticação agrupa elementos para promover a extensibilidade da solução, possibilitando definir estratégias. O *core* define o orquestrador do mecanismo, os pontos de extensão e lança eventos relacionados ao processo de autenticação.

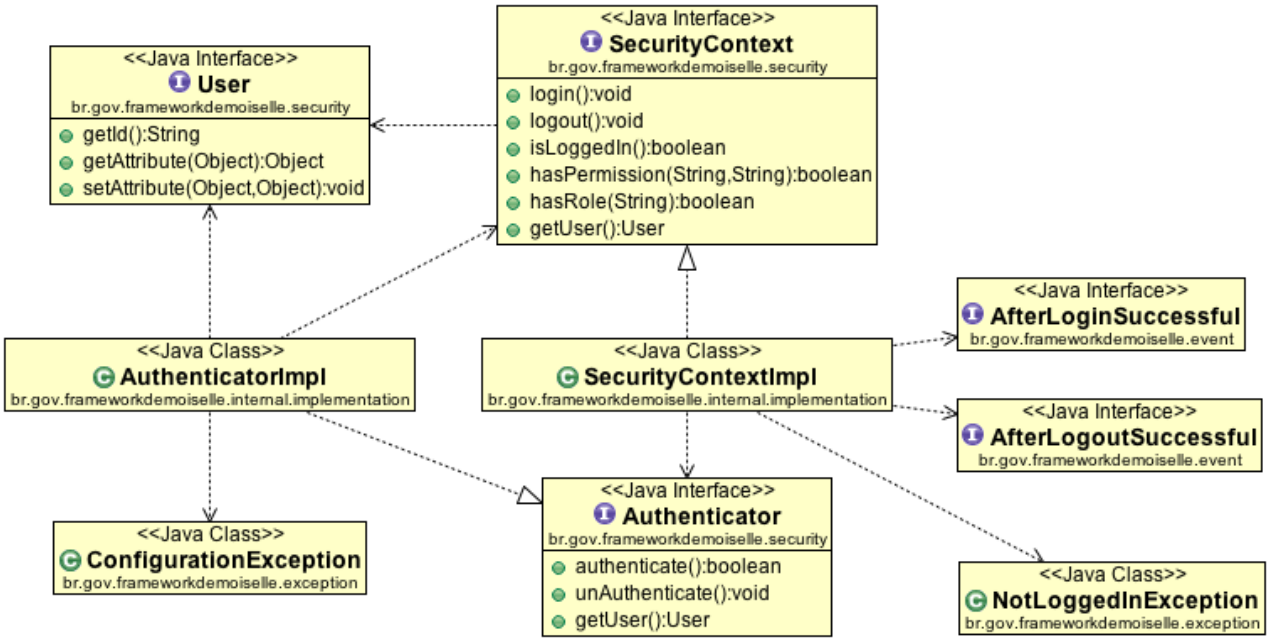

Figura 14: Autenticação

Elemento	Descrição
SecurityContext	Representa a fachada que orquestra o processo de autenticação, tais como efetuar *login* e *logout*, além de verificar e obter o usuário autenticado.
SecurityContextImpl	Implementação do comportamento da fachada de segurança que delega a responsabilidade das tarefas de autenticação sem ter conhecimento da estratégia escolhida. Dispara eventos relacionados ao processo de autenticação.
User	Interface que representa o usuário autenticado.
Authenticator	Interface que estabelece o contrato para implementação das estratégias de autenticação.
AuthenticatorImpl	Caso nenhuma estratégia de autenticação seja definida, esta implementação assume o controle e lança exceções sempre que tarefas de autenticação forem invocadas.
ConfigurationException	São lançadas pelo AuthenticatorImpl para orientar o usuário do *framework* definir estratégias concretas.
AfterLoginSuccessful	Evento disparado após *login* bem sucedido.
AfterLogoutSuccessful	Evento disparado após *logout* bem sucedido.
NotLoggedInException	Lançada sempre que uma tarefa de autenticação dependente de uma sessão autenticada é invocada sem que haja a sessão, por exemplo: *logout*.

Da mesma forma, a funcionalidade de autorização possibilita a definição de estratégias. O *core* define o orquestrador do mecanismo, os interceptadores e os pontos de extensão.

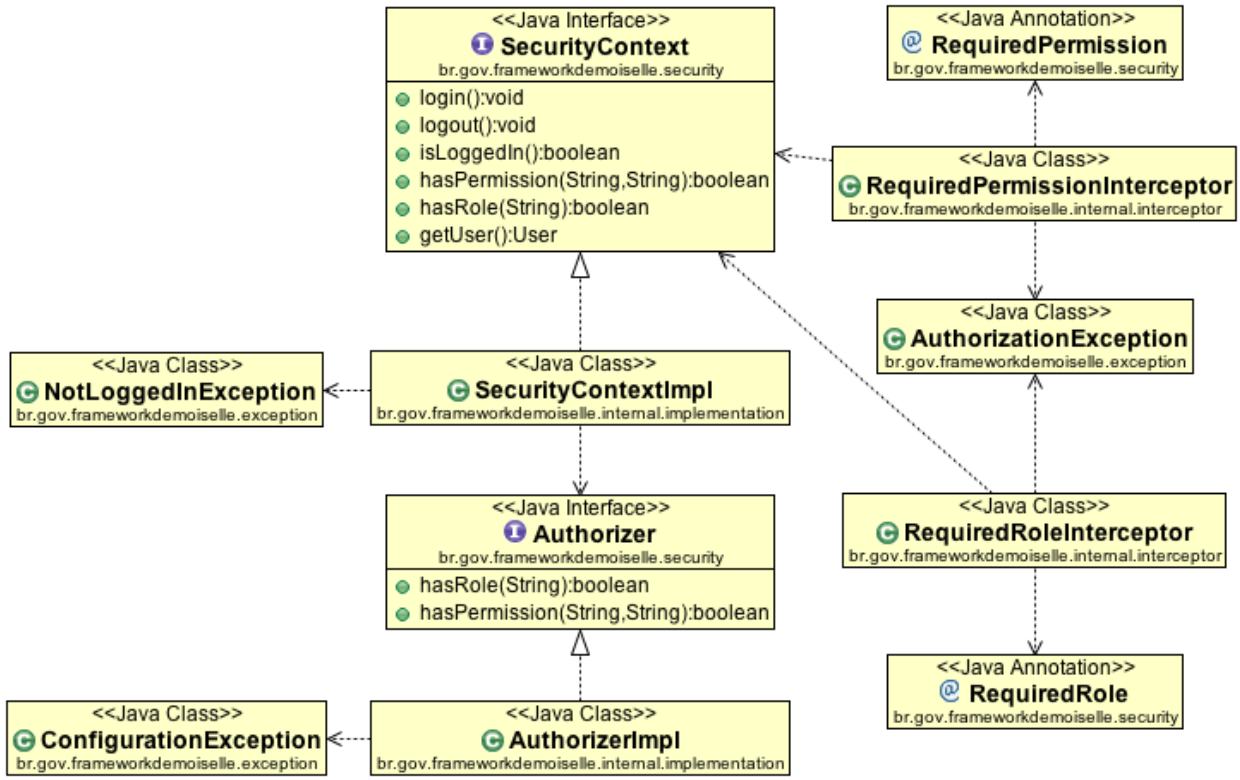

Figura 15: Autorização

Elemento	Descrição
SecurityContext	Representa a fachada que orquestra o processo de autorização, tal como verificar permissões.
SecurityContextImpl	Implementação do comportamento da fachada de segurança que delega a responsabilidade das tarefas de autorização sem ter conhecimento da estratégia escolhida.
NotLoggedInException	Todas as tarefas de autorização dependem de uma sessão autenticada, caso contrário esta exceção é lançada.
Authorizer	Interface que estabelece o contrato para implementação das estratégias de autorização.
AuthorizerImpl	Caso nenhuma estratégia de autorização seja definida, esta implementação assume o controle e lança exceções sempre que tarefas de autorização forem invocadas.
ConfigurationException	São lançadas pelo AuthorizerImpl para orientar o usuário do *framework* definir estratégias concretas.
RequiredPermission	Marca os métodos que devem ser interceptados para verificação de segurança baseada em recursos e operações, conforme o padrão ANSI RBAC.
RequiredPermissionInterceptor	Executa as tarefas de gerenciamento de segurança, verficando se o usuário autenticado possui permissão para invocar o método, delegando a decisão para o método hasPermission do orquestrador.
AuthorizationException	Caso o usuário autenticado não possua permissão para executar determinada tarefa, esta exceção é lançada.
RequiredRole	Marca os métodos que devem ser interceptados para verificação de segurança

Elemento	Descrição
	baseada nos papéis do usuário, conforme o padrão ANSI RBAC.
RequiredRoleInterceptor	Executa as tarefas de gerenciamento de segurança, verficando se o usuário autenticado possui o papel adequado para invocar o método, delegando a decisão para o método hasRole do orquestrador.

4.1.11. Utilitário

Provê algumas classes utilitárias que facilitam a execução de tarefas cotidianas.

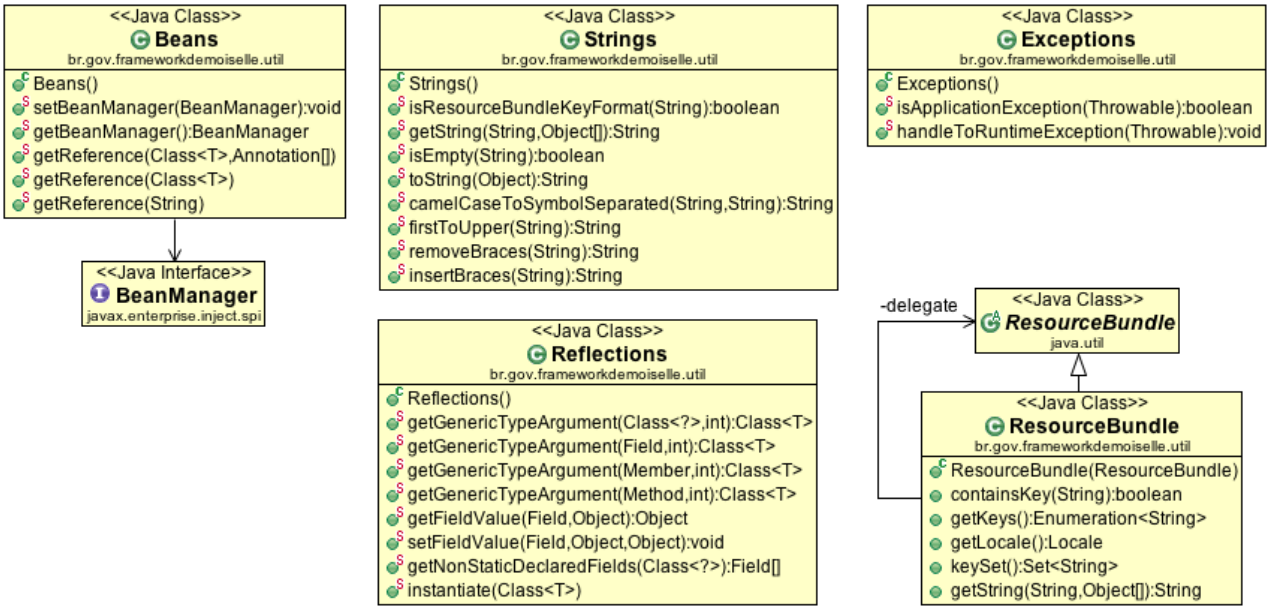

Figura 16: Utilitários

Elemento	Descrição
Beans	Oferece facilidades para manipulação do BeanManager e maneiras alternativas ao @Inject para obtenção de objetos.
BeanManager	Gerenciador de *beans* da especificação CDI.
Strings	Oferece facilidades para manipulação de *strings*.
Exceptions	Oferece facilidades para manipulação de exceções.
Reflections	Oferece facilidades para fazer reflexão.
ResourceBundle	Delega as chamadas para o ResourceBundle do Java acrescentando facilidades para manipulação de mensagens parametrizadas.

4.2. Módulo de Extensão JPA

Extensão do Módulo Core que disponibiliza funcionalidades referentes ao JPA.

4.2.1. Injeção

Possibilita a injeção de classes que não são nativas do CDI.

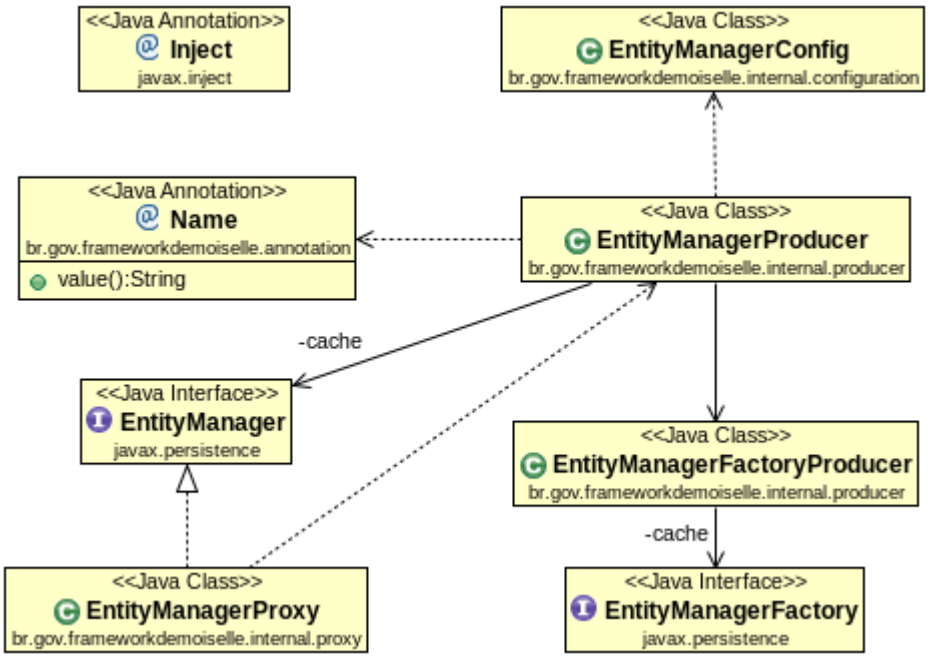

Figura 17: Injeção

Elemento	Descrição
Inject	Anotação do CDI que possibilita a injeção nos atributos.
EntityManagerFactoryProducer	Ativa a injeção de EntityManagerFactory.
EntityManagerFactory	Classe da especificação JPA.
EntityManagerProducer	Ativa a injeção de EntityManager.
Name	Para os casos de injeção de EntityManager anotado com @Name, o produtor considera o valor definido na anotação como o nome da unidade de persistência desejada.
EntityManagerConfig	Possui parametrizações para o funcionamento do produtor.
EntityManagerProxy	Classe produzida pelo EntityManagerProducer que delega para o EntityManager, instanciando-o sob demanda, e implementa a serialização.
EntityManager	Classe da especificação JPA.

4.2.2. Transação

Estratégia de transação JPA que delega as invocações para o mecanismo de transação do EntityManager.

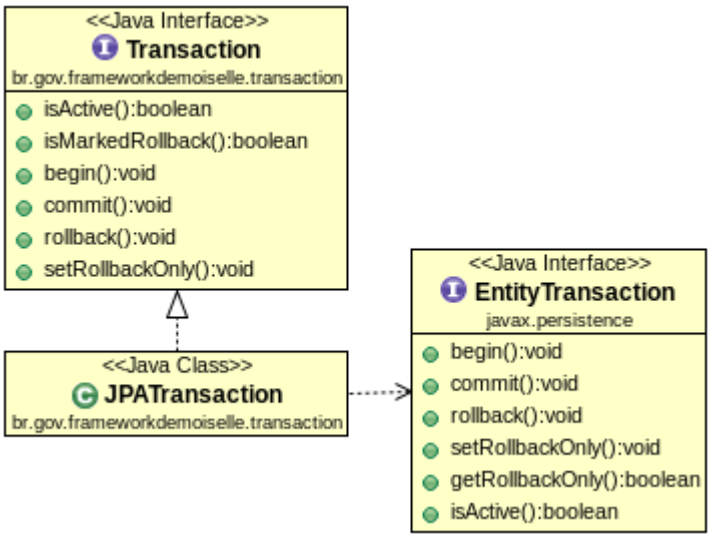

Figura 18: Transação

Elemento	Descrição
Transaction	Interface do módulo Core que define o contrato para estratégias de controle de transação.
JPATransaction	Implementação da estratégia de transação que delega as chamadas para o EntityTransaction.
EntityTransaction	Classe nativa JPA que representa uma sessão transacional.

4.2.3. Template

Oferece moldes para facilitar a criação de classes padronizadas, as quais devem implementar, herdar ou modificar os *templates*.

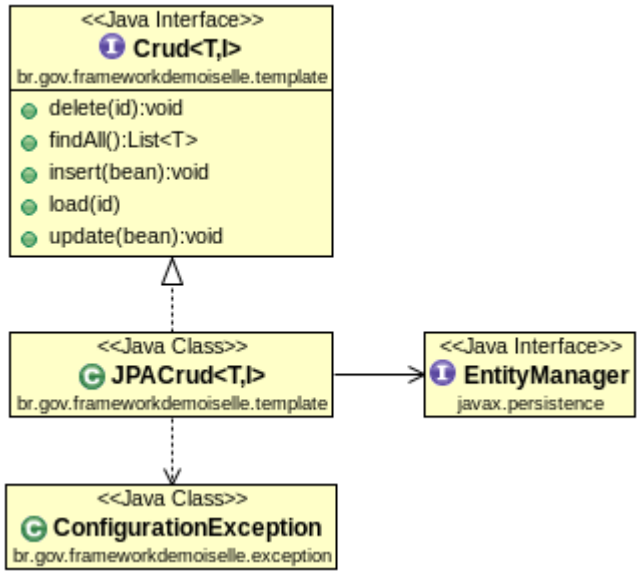

Figura 19: Template

Elemento	Descrição
Crud	Template do módulo Core.
JPACrud	Implementa o template Crud delegando o funcionamento das operações para o EntityManager.
EntityManager	Classe nativa JPA que representa uma sessão transacional.
ConfigurationException	Em caso de falhas na parametrização do template, esta exceção será lançada.

4.2.4. Paginação

Integra com o mecanismo de transação definido no módulo Core para possibilitar a paginação dos resultados do banco de dados.

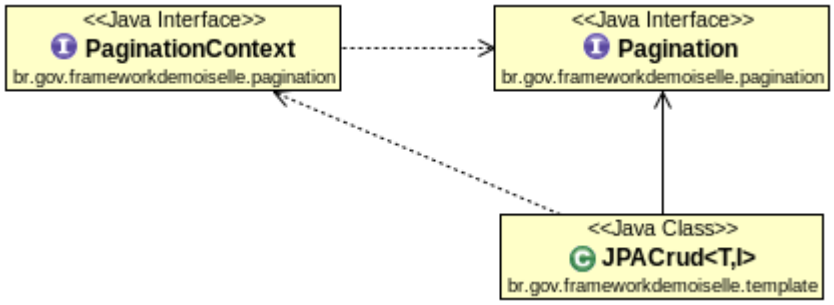

Figura 20: Paginação

Elemento	Descrição
JPACrud	Obtém e define as informações do contexto para possibilitar a paginação dos resultados.
PaginationContext	Interface definida no módulo Core.
Pagination	Interface definida no módulo Core.

4.3. Módulo de Extensão JTA

Extensão do Módulo Core que disponibiliza funcionalidades referentes ao JTA.

4.3.1. Transação

Estratégia de transação JTA que delega as invocações para o UserTransacion da especificação JTA.

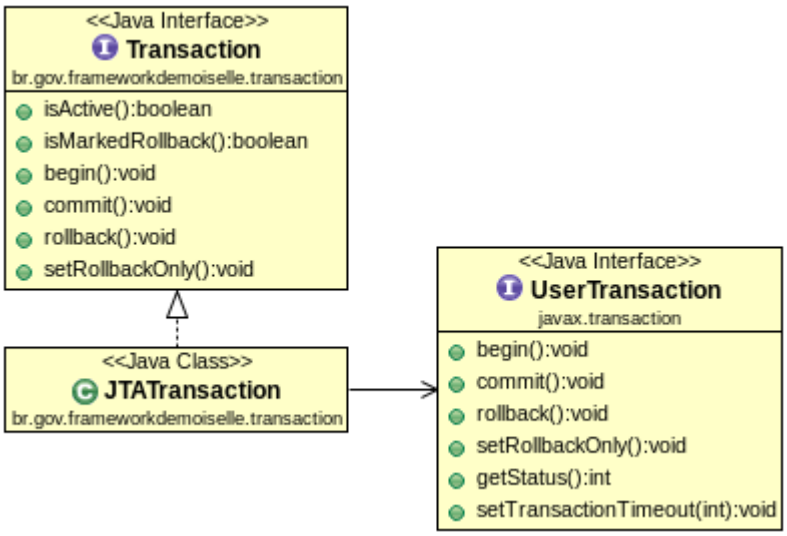

Figura 21: Transação

Elemento	Descrição
Transaction	Interface do módulo Core que define o contrato para estratégias de controle de transação.
JTATransaction	Implementação da estratégia de transação que delega as chamadas para o UserTransaction.
UserTransaction	Classe nativa que representa uma sessão transacional.

4.4. Módulo de Extensão JUnit

Extensão do Módulo Core que disponibiliza funcionalidades referentes ao JUnit.

4.4.1. Utilitário

Implementação de *runner* para JUnit que ativa de forma prática a injeção em casos de teste. Esta extensão faz uso da extensão SE.

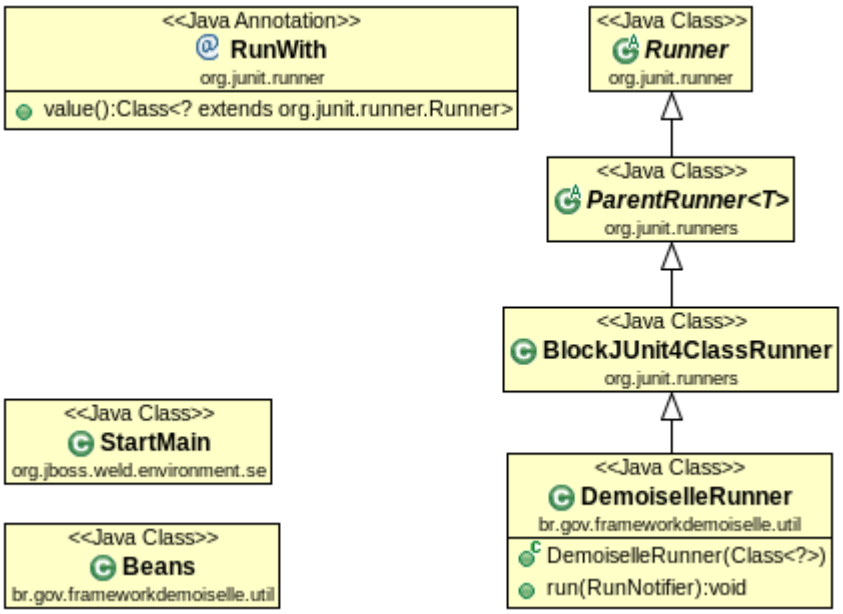

Figura 22: Utilitário

Elemento	Descrição
RunWith	Anotação JUnit para redefinir o *runner*.
DemoiselleRunnner	Implementação que ativa a injeção nos casos de teste com auxílio das classes StartMain e Beans.
BlockJUnit4ClassRunner	Classe nativa do JUnit.
ParentRunner	Classe nativa do JUnit.
Runner	Classe nativa do JUnit.
StartMain	Classe nativa do Weld que inicia um *container* CDI em ambiente SE.
Beans	Classe utilitária do framework que possibilita acesso aos *beans*.

4.5. Módulo de Extensão SE

Extensão do Módulo Core que ativa um container CDI baseado no Weld para ambiente SE.

4.5.1. Injeção

Utiliza o recurso Portable Extensions para capturar eventos do CDI e registrar pseudo-contextos para o ambiente SE.

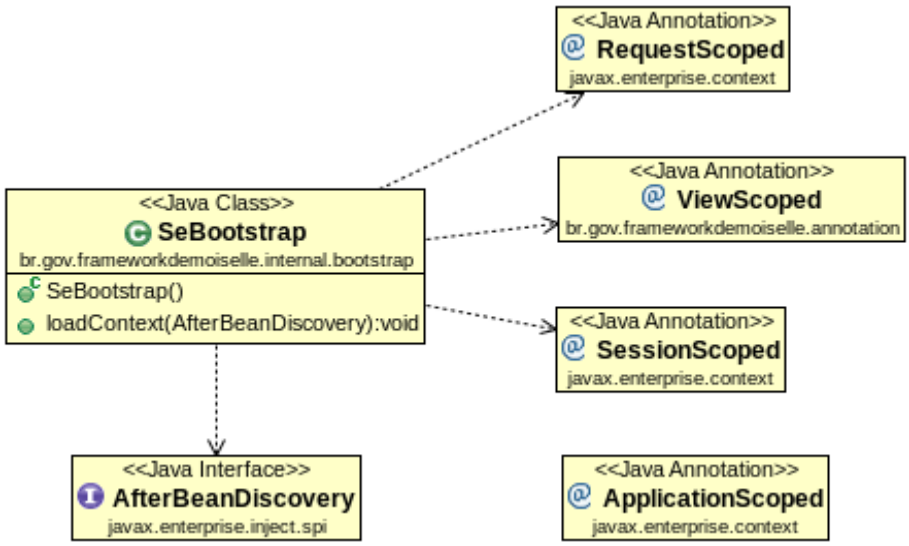

Figura 23: Bootstrap

Elemento	Descrição
SeBootstrap	Observa os eventos lançados pelo CDI.
AfterBeanDiscovery	Evento do CDI que, quando disparado, o SeBootstrap registra os pseudo-escopos.
ApplicationScoped	Escopo CDI registrado para possibilitar a execução da aplicação SE.
SessionScoped	Escopo CDI registrado para possibilitar a execução da aplicação SE.
ViewScoped	Escopo CDI registrado para possibilitar a execução da aplicação SE.
RequestScoped	Escopo CDI registrado para possibilitar a execução da aplicação SE.

4.6. Módulo de Extensão JSF

Extensão do Módulo Core que disponibiliza funcionalidades referentes ao JSF.

4.6.1. Injeção

Possibilita a injeção de classes que não são nativas do CDI.

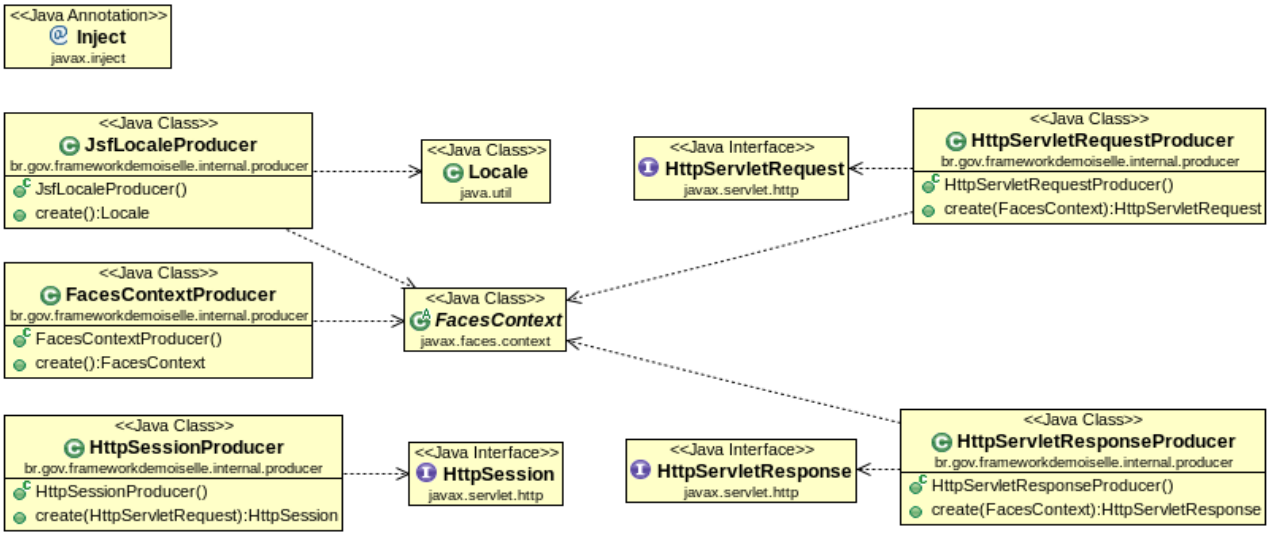

Figura 24: Produtor

Elemento	Descrição
Inject	Anotação do CDI que possibilita a injeção nos atributos.
JsfLocaleProducer	Possibilita a injeção de Locale com base no FacesContext.
Locale	Classe nativa Java.
FacesContextProducer	Possibilita a injeção de FacesContext.
FacesContext	Classe nativa JSF.
HttpSessionProducer	Possibilita a injeção de HttpSession com base no FacesContext.
HttpSession	Classe nativa JEE.
HttpServletRequestProducer	Possibilita a injeção de HttpServletRequest com base no FacesContext.
HttpServletRequest	Classe nativa JEE.
HttpServletResponseProducer	Possibilita a injeção de HttpServletResponse com base no FacesContext.
HttpServletResponse	Classe nativa JEE.

Utiliza o recurso Portable Extensions para capturar eventos do CDI e registrar novos contextos.

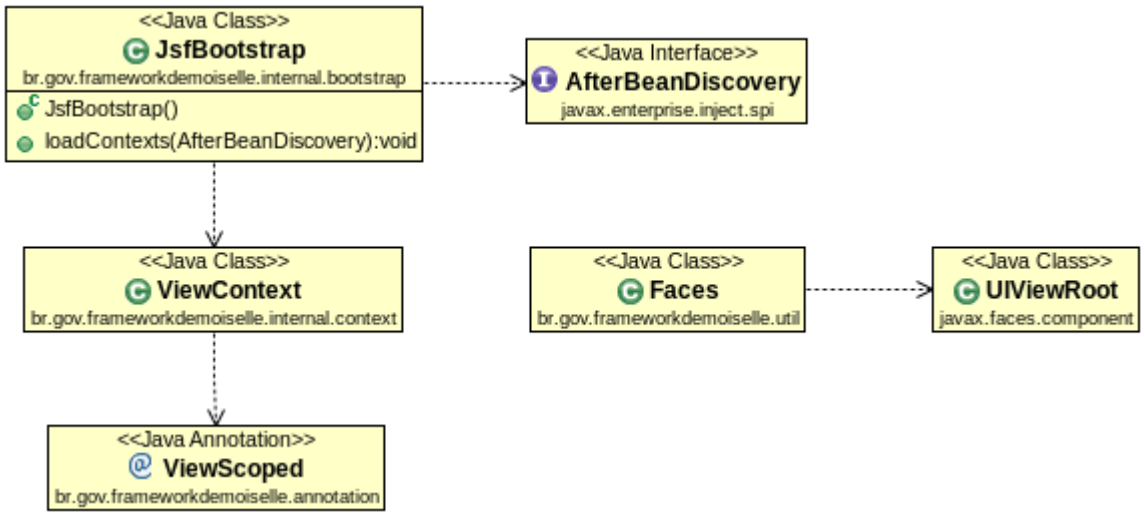

Figura 25: Bootstrap

Elemento	Descrição
JsfBootstrap	Observa os eventos lançados pelo CDI.
AfterBeanDiscovery	Evento do CDI que, quando disparado, o JsfBootstrap registra o ViewScoped
ViewContext	Implementação do escopo customizado que toma como base o UIViewRoot obtido através do utilitário Faces.
ViewScoped	Escopo de visão definido no módulo Core.
Faces	Classe utilitária.
UIViewRoot	Classe nativa JSF.

4.6.2. Mensagem

Mecanismo que converte as mensagens do contexto em mensagens JSF.

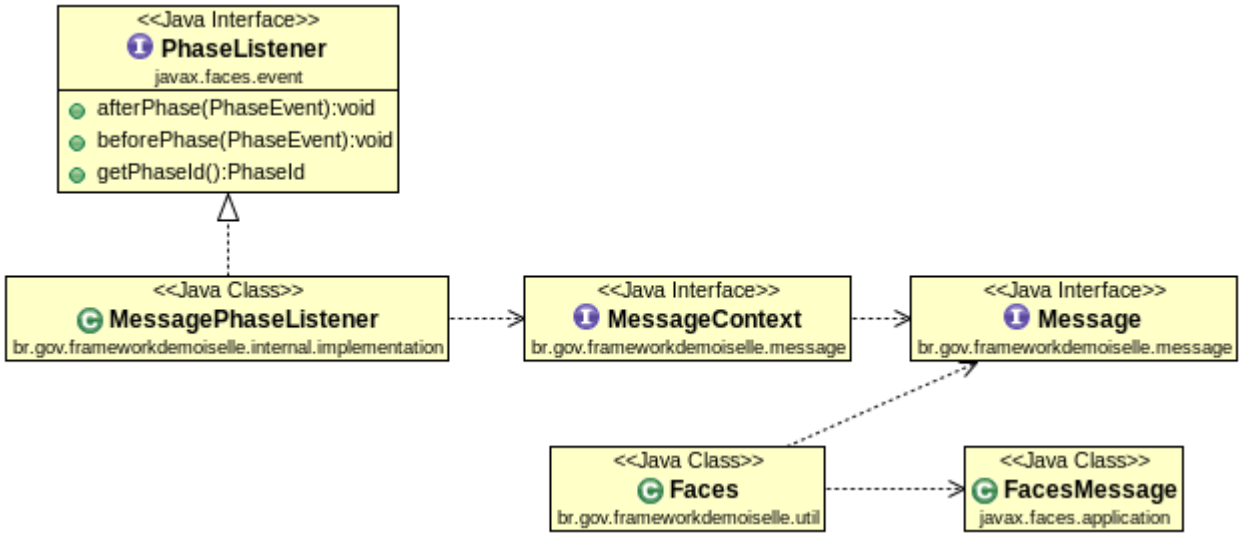

Figura 26: Mensagem

Elemento	Descrição
PhaseListener	Interface nativa JSF.
MessagePhaseListener	Observador da fase RENDER_RESPONSE que lê as mensagens do contexto, converte em mensagens JSF por intermédio da classe utilitária Faces e limpa o contexto de mensagens.
MessageContext	Fachada para as funcionalidade de manipulação de mensagens. Interface definida no módulo Core.
Message	Representa uma mensagem. Interface definida no módulo Core.
Faces	Classe utilitária que converte Message em FacesMessage.
FacesMessage	Classe nativa JSF.

4.6.3. Exceção

Escrever.

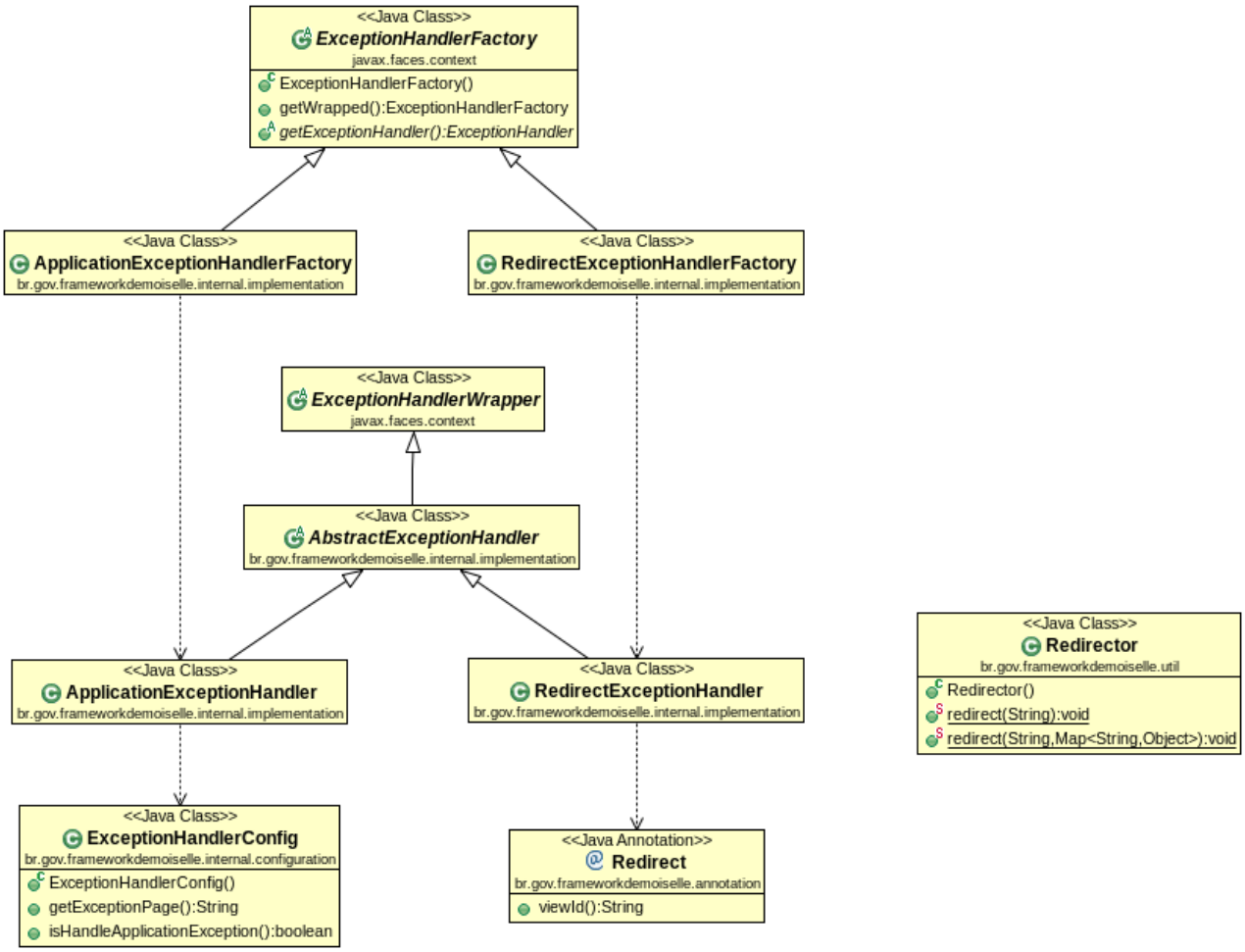

Figura 27: Exceção

Elemento	Descrição
ApplicationExceptionHandlerFactory	Fabrica tratadores de exceção de aplicação.
ApplicationExceptionHandler	Trata exceções de aplicação.

Elemento	Descrição
RedirectExceptionHandlerFactory	Fabrica tratadores de exceção com redirecionamento de tela.
RedirectExceptionHandler	Trata exceções com redirecionamento de tela.
Redirect	Marca uma exceção indicando a tela de destino.
Redirector	Redireciona para a tela com base na anotação Redirect.
ExceptionHandlerConfig	Parametrização do mecanismo de tratamento de exceções.
AbstractExceptionHandler	Implementações padronizadas para tratamento de exceções.
ExceptionHandlerWrapper	Classe nativa JSF.
ExceptionHandlerFactory	Classe nativa JSF.

4.6.4. Template

Oferece um contrato padronizado para criação de classes da camada de apresentação com JSF.

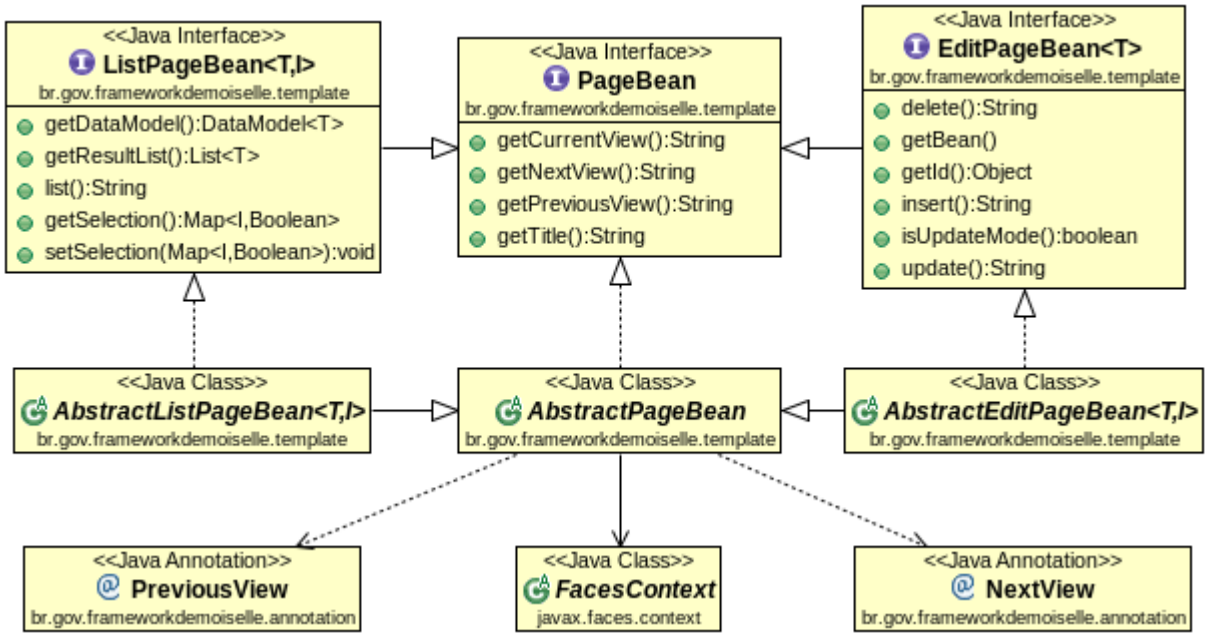

Figura 28: Template

Elemento	Descrição
PageBean	Define os métodos básicos que toda página pode necessitar.
AbstractPageBean	Implementação padrão do PageBean que define o comportamento dos métodos tela anterior, tela atual e tela seguinte com base no FacesContext e nas anotações PreviousView e NextView.
PreviousView	Define a tela anterior.
NextView	Define a tela seguinte.
FacesContext	Classe nativa JSF.
ListPageBean	Define os métodos básicos para telas de listagem.
AbstractListPageBean	Implementação padrão do ListPageBean.
EditPageBean	Define os métodos básicos para telas de edição.

Elemento	Descrição
AbstractEditPageBean	Implementação padrão do EditPageBean.

4.6.5. Segurança

Mecanismo que observa os eventos de segurança.

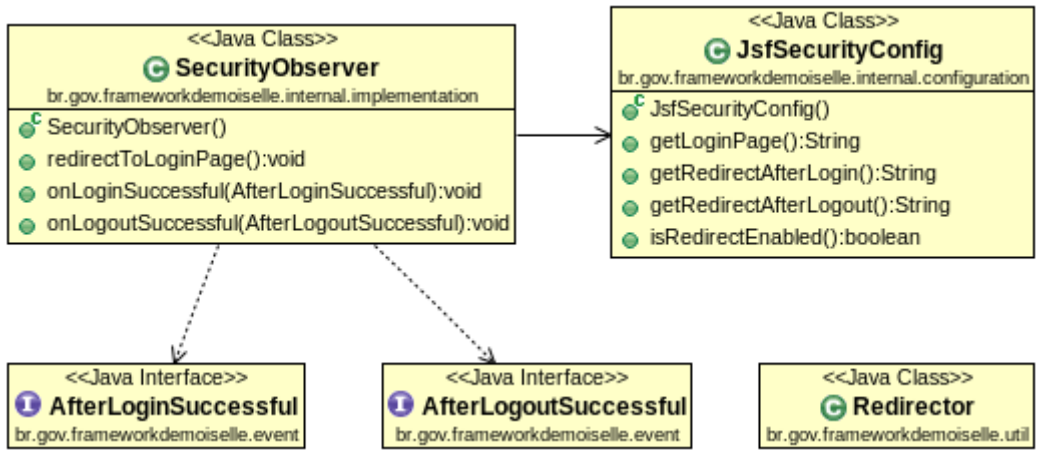

Figura 29: Observador dos eventos de segurança

Elemento	Descrição
SecurityObserver	Observa os eventos de segurança e dispara o redirecionamento de telas com base na parametrização configurada.
AfterLoginSuccessful	Evento definido no módulo Core.
AfterLogoutSuccessful	Evento definido no módulo Core.
Redirector	Classe utilitária.
JsfSecurityConfig	Representa as parametrizações de segurança referentes ao JSF.

4.6.6. Utilitário

Oferece facilidades para manipulação de classes JSF.

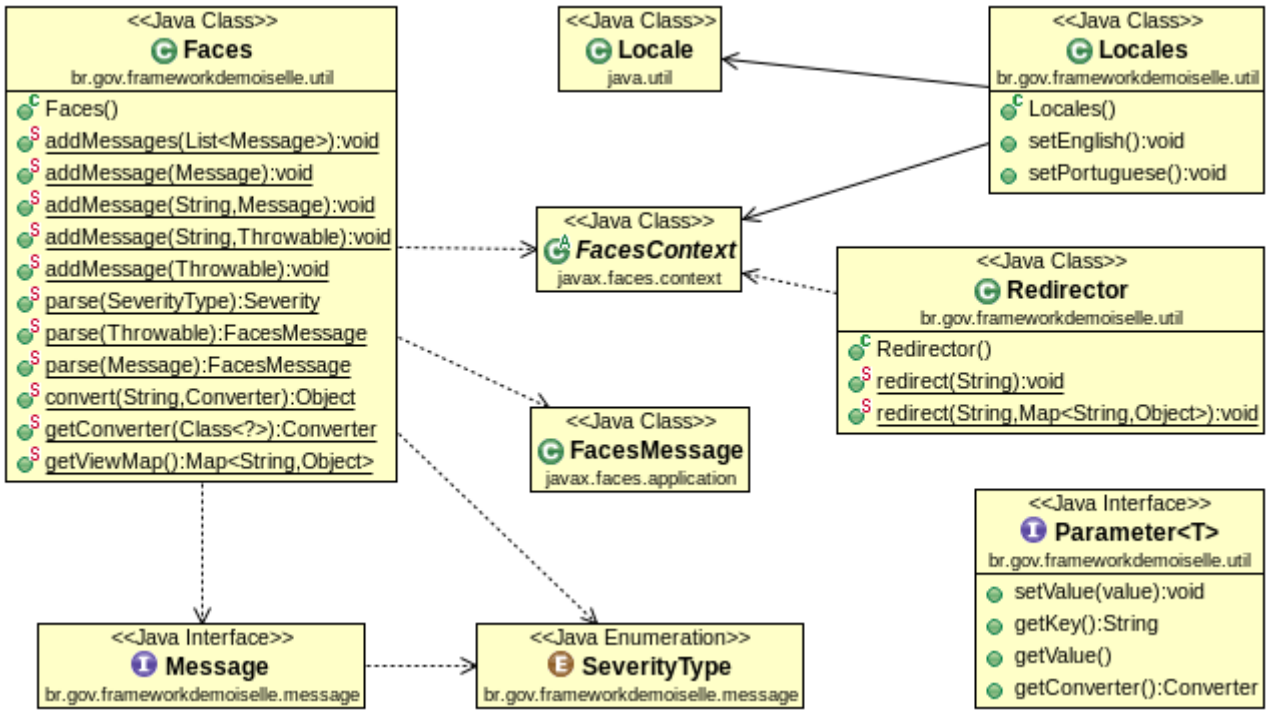

Figura 30: Utilitário

Elemento	Descrição
Faces	Oferece facilidades para acesso e manipulação do FacesContext, tal como conversão de mensagens.
FacesContext	Classe nativa JSF.
Message	Classe definida no módulo Core.
SeverityType	Classe definida no módulo Core.
FacesMessage	Classe nativa JSF.
Locales	Oferece facilidades para manipulação de Localea partir do JSF.
Locale	Classe nativa Java.
Redirector	Oferece facilidades para redirecionamento de telas.
Parameter	Oferece facilidades para manipulação de parâmetros HTTP no JSF.

4.7. Módulo de Extensão Shiro

Extensão do Módulo Core que disponibiliza funcionalidades referentes ao Apache Shiro.

4.7.1. Injeção

Possibilita a injeção de classes nativas do Shiro.

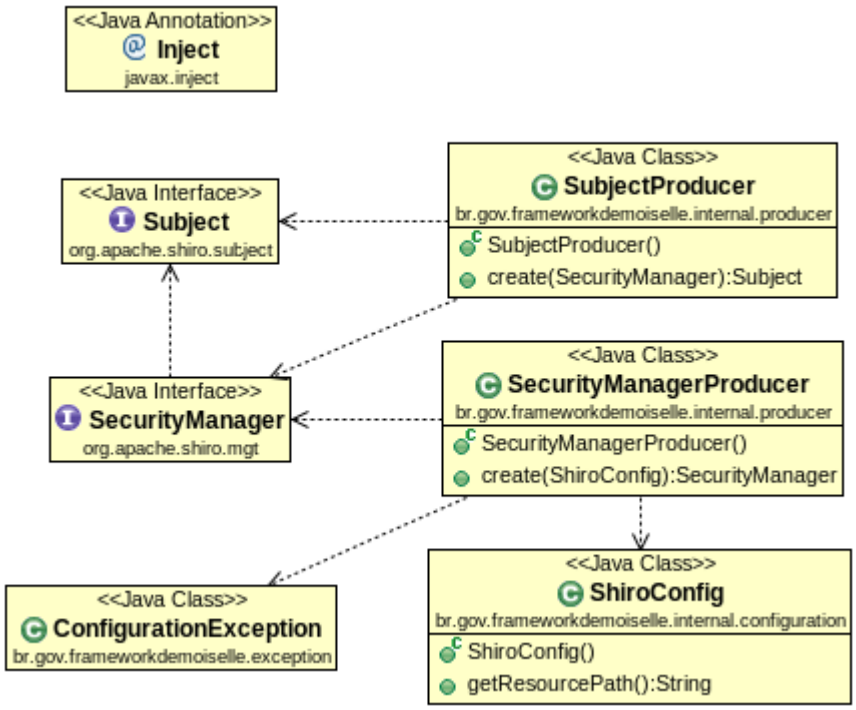

Figura 31: Produtor

Elemento	Descrição
Inject	Anotação do CDI que possibilita a injeção nos atributos.
SubjectProducer	Fabrica e possibilita a injeção de Subject.
Subject	Classe nativa do Shiro.
SecurityManagerProducer	Fabrica e possibilita a injeção de SecurityManager.
SecurityManager	Classe nativa do Shiro.
ShiroConfig	Parametrização do SecurityManagerProducer.
ConfigurationException	Caso haja falha na configuração do produtor, esta exceção será lançada.

4.7.2. Segurança

Estratégia de autenticação que delega as invocações para o mecanismo do Shiro.

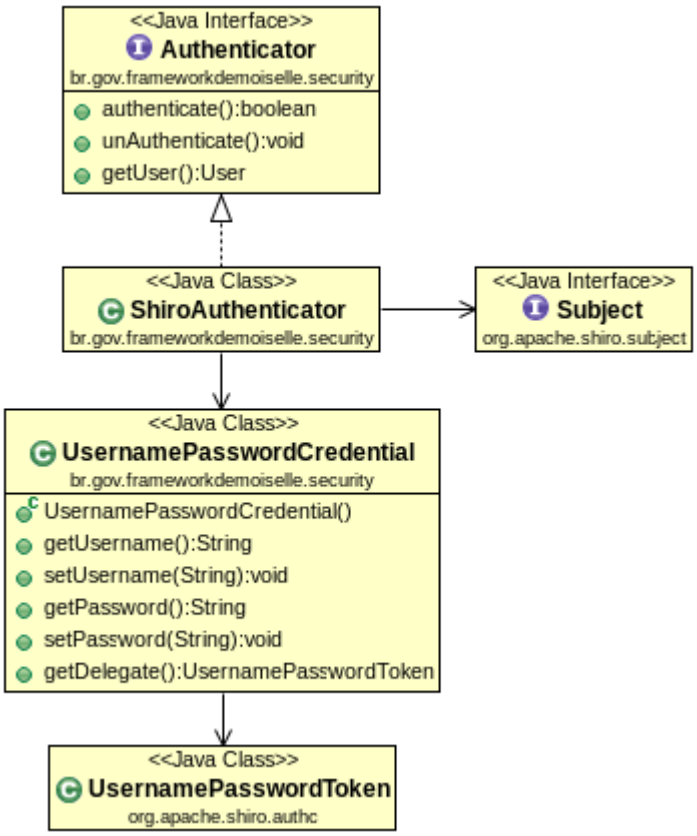

Figura 32: Autenticação

Elemento	Descrição
Authenticator	Interface do módulo Core que define o contrato para estratégias de autenticação.
ShirtoAuthenticator	Implementação da estratégia de autenticação que delega as chamadas para o Subject.
UsernamePasswordCredential	Transporta as credenciais do usuário e está disponível via injeção, delegando as operações para o UsernamePasswordToken.
UsernamePasswordToken	Classe nativa do Shiro.

Estratégia de autorização que delega as invocações para o mecanismo do Shiro..

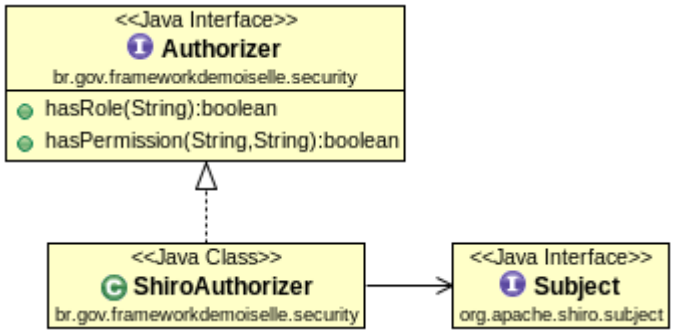

Figura 33: Autorização

Elemento	Descrição
Authorizer	Interface do módulo Core que define o contrato para estratégias de autorização.
ShiroAuthorizer	Implementação da estratégia de autorização que delega as chamadas para o Subject.
Subject	Classe nativa do Shiro.

4.8.Padrões e Mecanismos Arquiteturais

<Apresenta os mecanismos de análise e projeto, regras de modelagem, estereótipos e padrões comportamentais da arquitetura>

Documentação:<Insira aqui as documentações e as imagens dos diagramas de interação contidos nas pastas *Padrões Comportamentais Importantes*. Essas pastas se encontram situadas debaixo das pastas *04 - Elementos do Modelo Arquiteturalmente Significativos*, dentro do *Modelo de Projeto* e/ou do *Modelo de Análise*>

4.9.Topologia

<Apresenta o diagrama geral de implantação.>

Documentação:<Insira aqui a documentação e a imagem do diagrama de implantação (*deployment diagram*)>

4.10.<Visão de Implementação>

<Esta seção é opcional e descreve os principais componentes do software a serem implantados e suas relações de dependência.>

Documentação:<Insira aqui a documentação e a imagem do diagrama de componentes>

4.11.Outras Visões

<Apresenta outras abstrações da arquitetura que se julguem necessárias, como visão de processos concorrentes, visão de dados, etc>

5.Decisões e Justificativas

<Descreve as principais decisões tomadas para a arquitetura e as justificativas para as escolhas dentre as alternativas consideradas.>