PROJETO FRAMEWORK - CELEPAR

CONTROLE DE EXCEÇÕES EM APLICAÇÕES JAVA WEB

Junho – 2007

Sumário de Informações do Documento

Tipo do Documento: Definição
Título do Documento: Controle de Exceções em Aplicações Java Web
Estado do Documento: EB (Elaboração)
Responsáveis: Cleverson Budel, Diego Pozzi, Fábio Sgoda, Filipe Lautert
Palavras-Chaves: exceção, java, controle de fluxo, projetos web
Resumo: Padronização na forma de se tratar erros nas aplicações web
Número de páginas: 7
Software utilizados:

Versão	Data	Mudanças
1.0	12/01/2005	Criação do documento
2.0	30/05/2007	Re-elaboração do documento

Sumário

1 CONTROLE DE EXCEÇÕES EM APLICAÇÕES JAVA WEB

1.1 Pré-requisitos

Para entendimento deste documento é necessário que se tenha conhecimento na linguagem Java e os conceitos do *Framework Struts*.

1.2 Introdução

Existem várias formas de se tratar exceções em aplicações *Java.* Este documento visa expor uma forma padronizada para esses tratamentos.

A forma proposta encontra-se compatível com a arquitetura das aplicações definida pela empresa e atende as principais necessidades dos projetos desenvolvidos pela Celepar.

O tratamento de exceções está fortemente ligado com a apresentação de mensagens, pois caso ocorra algum problema (exceção) na aplicação o usuário deve/pode ser informado.

Uma aplicação robusta deve ter seu tratamento de exceções muito bem feito, discriminado cada exceção de forma única, pois em caso de problemas o usuário receberá a mensagem tratada e saberá o que está acontecendo, reportando o problema de forma clara, conseqüentemente o desenvolvedor saberá exatamente o ponto onde o problema ocorreu. Essa é a importância de um bom tratamento de exceções.

1.3 Classes Envolvidas

As classes descritas a baixo encontram-se na na biblioteca de componentes do *Framework Pinhão Paraná*.

- **BaseDispatchAction**: Classe responsável por direcionar as requisições

do usuário especializando o tratamento de exceções. Os erros são logados através dessa classe por isso não se deve mais solicitar impressão de erro dentro da aplicação.

Todas as classes do tipo Action da aplicação devem herdar da BaseDispatchAction. Essa classe possue ainda alguns métodos para envio de mensagens em formato de *alert* (padrão JavaScript) ou escritas na própria tela corrente.

- **ApplicationException**: Responsável por manipular as exceções da aplicação. Possue construtores sobrecarregados onde recebendo a causa da exceção e/ou a chave da mensagem e/ou ícone da mensagem que será mostrada ao usuário.

- **Mensagem**: Fornece métodos para manipular a classe de mensagens utilizada e mantê-la única em todo o sistema. Podem existir várias formas de recuperar as mensagens do sistema, exemplo: mensagens gravadas em base de dados, em arquivo texto, em xml, etc. Existe a interface chamada MensagemInterface que define as regras para todas essas variações de recuperação de mensagens. Já estão implementadas as seguintes classes de acesso: MensagemPr (default: realiza busca a partir de um arquivo de propriedades) e MensagemBD (realiza busca a partir do banco de dados). Podem ser implementas outras formas de acesso desde que implementem a interface MensagemInterface.

  Desta forma sempre estaremos trabalhando com a classe Mensagem não importando qual forma de acesso está sendo utilizada.

1.4 Forma de Tratamento

O tratamento das exceções deve se dar da seguinte forma: primeiramente identificando todos os blocos de código com possibilidade de ocorrer exceções. Após a identificação deve-se tratar as exceções inserindo *try-catch* nestes blocos. O erro sempre deve ser tratado no momento em que ocorre, para isso deve-se transformar a exceção ocorrida numa ApplicationException e levantar (*throws*) a exceção tratada (ApplicationException) para a classe chamadora. A classe chamadora deve verificar

se a exceção já esta tratada, se estiver apenas lança para a sua classe superior, senão deve-se fazer o tratamento e lança-la para sua classe superior. Para poder levantar a exceção tratada todos os métodos devem lançar ApplicationException em sua assinatura (*throws ApplicationException*).

Algumas exceções que fazem parte do negócio da aplicação devem ser previstas e especificadas pelo analista projetista. As exceções não especificadas devem ser tratadas pelo desenvolvedor, onde este também poderá fazer cadastrado das mensagens que serão apresentadas aos usuários quando o problema ocorrer. Em caso de dúvida na mensagem a ser exibida o desenvolvedor deve conversar com o analista projetista.

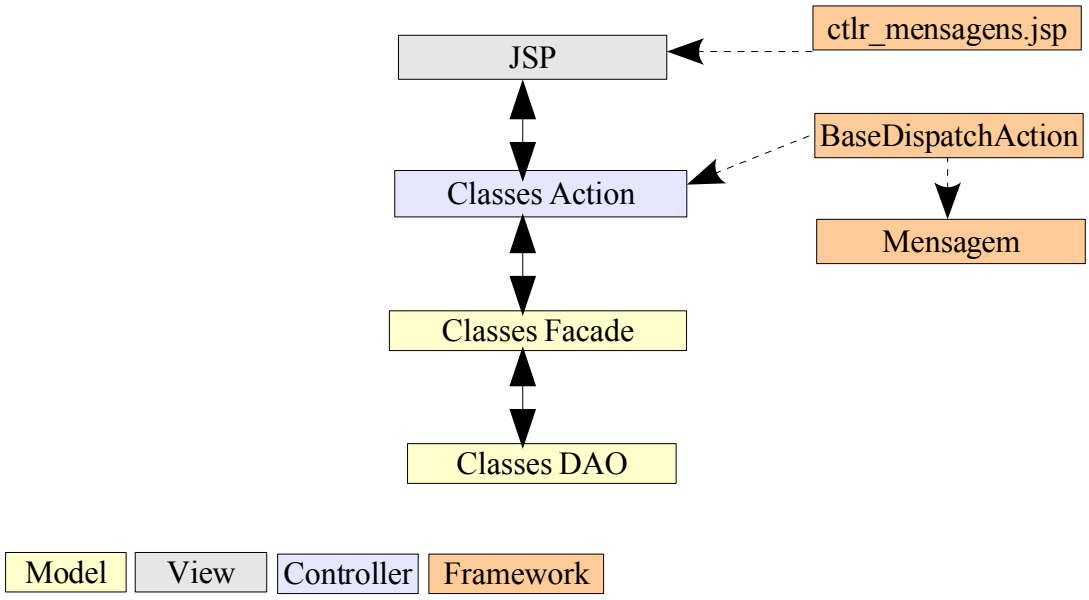
Abaixo temos um exemplo de método realizando tratamento de possíveis exceções:

```
/**
 * Remove objeto Aluno.
 * @param Aluno a ser removido.
 * @throws ApplicationException.
 */
public void excluirAluno(Aluno aluno) throws ApplicationException {

	try {
		this.validarStatusAluno(aluno); //Verifica se o aluno pode ser excluído
	} catch (ApplicationException appEx) {
		throw appEx;
	} catch (Exception ex) {
		throw new ApplicationException("mensagem.erro.matricula.statusAluno", ex);
	}

	DAOFactory hibernateFactory = DAOFactory.getDAOFactory(DAOFactory.HIBERNATE);
	try {
		HibernateUtil.currentTransaction(); //Abre sessão e transação

		hibernateFactory.getAlunoDAO().excluirAluno(aluno); //Exclui o aluno

		HibernateUtil.commitTransaction(); //Fecha sessão e transação

	} catch (ApplicationException appEx) {
		HibernateUtil.rollbackTransaction();
		throw appEx;
	} catch (Exception ex) {
		HibernateUtil.rollbackTransaction();
		throw new ApplicationException("mensagem.erro.matricula.excluirAluno", ex);
	}

}
```

*Observe que os tratamentos (mensagens) são diferentes para cada tipo de problema.

1.4.1 Fluxo da Aplicação

A figura abaixo descreve o fluxo básico da aplicação segundo a arquitetura da *Plataforma Pinhão Paraná*:



As exceções são tratadas e propagadas para a camada superior até serem apresentadas aos usuários.

A JSP **ctlr_mensagens** possui o *layout* para apresentar as mensagens ao usuário. Todas as páginas devem importar essa JSP pois é através dela que são apresentadas as exceções e as mensagens aos usuários. O desenvolvedor deve utilizar o framework **Tiles** que gerenciar o *layout* da aplicação e suprimir a declaração de importação da JSP **ctlr_mensagem** em todas as páginas. Para mais informações sobre o Tiles consultar o documento “Gerenciamento de Layout com Tiles”.

A classe **BaseDispatchAction** é a última a receber a exceção. Ela abre a exceção tratada (ApplicationException) e recupera o código da mensagem. Através da classe **Mensagem** recupera a informação a ser apresentada ao usuário. Caso chegue a classe BaseDispatchAction uma exceção não tratada será apresentada uma mensagem genérica ao usuário (a mensagem genérica não dá subsídios para o usuário e desenvolvedor saberem qual o real problema ocorrido, por isso deve ser feito o tratamento adequado de todas as possíveis exceções).

1.4.2 Configurações Complementares

Existem outras configurações que devem ser inseridas na aplicação para se tentar tratar todos os possíveis pontos de erros.

Deve ser inserido no arquivo *web.xml* das aplicações a tag *<error-page>* que redireciona a requisição do usuário para uma página de tratamento de erros dependendo da resposta do servidor. Exemplo, erro 404 (recurso não encontrado).

Os arquivos JSP também devem inserir a tag **<% page errorPage="***caminho para página de tratamento de erro***" %>** para tratar possíveis erros que aconteçam na página.

1.5 Conclusão

O modelo para tratamento de exceções proposto consiste em tratar e propagar a exceção já tratada até a camada que fará a apresentação de mensagens amigáveis aos usuários da aplicação. Essa abordagem é simples e tenta onerar minimamente o tempo de desenvolvimento.