05

Criando um conversor para nossa data

O converter que estamos usando tem alguns problemas. O primeiro deles é que ele está gerando uma data com valor errado se não usarmos o *TimeZone* correto, e o segundo é o tipo com que ele trabalha. Como vimos no exercício anterior ele trabalha com `java.util.Date` mas nós estamos usando um `Calendar` e queremos trabalhar em cima de `Calendar`.

Para resolver esse dois problemas, podemos criar o nosso próprio converter. Crie a classe `CalendarConverter` no pacote `br.com.casadocodigo.loja.converters`.

Dentro de nossa classe temos que implementar a interface `javax.faces.convert.Converter`, a qual nos obrigará a implementar os métodos `getAsObject(FacesContext context, UIComponent component, String dataTexto)` e o `getAsString(FacesContext context, UIComponent component, Object dataObject)`.

Vamos fazer uso do conversor de `Date` que temos pronto, para nos ajudar. Crie um atributo de instância chamado `converter` do tipo `DateTimeConverter`, e já instancie a classe ai mesmo.

No construtor de nosso `CalendarConverter`, vamos informar o padrão do nosso sistema para o formato da data, e também já deixar o `TimeZone` setado para `TimeZone.getTimeZone("America/Sao_Paulo")`.

Nos métodos, o código será bem simples, no método `getAsObject` basta chamar o próprio converter que temos para `Date` que recebemos um `Date`, o que fica fácil transformar em `Calendar`.

```
Date data = (Date) converter.getAsObject(context, component, dataTexto);
```

E no método `getAsString`, devemos usar o `Calendar` que recebemos como objeto e passar como `Date` para nosso converter. Nesse método, precisamos evitar o `NullPointerException`, então verifique antes de realizar o *casting* se o objeto não é nulo, caso seja, retorne `null`. Após obter o `Calendar`, só precisamos chamar nosso converter para retornar o valor.

```
return converter.getAsString(context, component, calendar.getTime());
```

Agora não precisamos mais nos preocupar com instanciar o atributo `dataPublicacao` na classe `Livro`, assim, remova o `Calendar.getInstance()` que fizemos.

E para finalizar, abra o formulário `livros/form.xhtml` e remova o `<f:convertDateTime ... />` bem como retire o `.time` do `value` do `input`. Ficando simplesmente assim:

```
<h:inputText value="#{adminLivrosBean.livro.dataPublicacao}" id="dataPublicacao" />
<h:message for="dataPublicacao" />
```

Agora temos um converter que já tem o padrão pré-definido e bem como o `TimeZone` e o **JSF** identifica automaticamente o tipo que precisa para realizar a conversão sem que seja necessário informar na tag `input` um converter específico.

Realize um *Full Publish* e teste nossa aplicação.