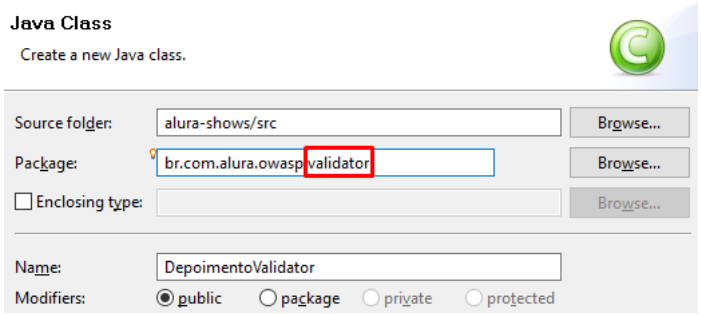
09

Mãos à obra: Criando a classe DepoimentoValidator

Vimos que a aplicação da Alura Shows não está fazendo nenhuma validação do que os usuários podem colocar no campo **Título** e **Mensagem** da página de depoimento. Com isso, precisamos criar uma validação para evitar que os usuário coloquem informações que serão prejudiciais para a nossa aplicação. Dessa forma, vamos criar a classe **DepoimentoValidator** dentro do pacote **validator** onde iremos configurar como será a validação que iremos realizar.



Vamos implementar a interface Validator do Spring para realizarmos a validação do que o usuário está passando no campo título e mensagem do depoimento. Ao fazermos isso, somos obrigado a implementar dois métodos **supports** e **validate**, o método supports recebe a classe do objeto que está querendo ser validado e retorna se o validador consegue lidar com ele. Coloque:

```
@Override
    public boolean supports(Class<?> clazz) {
        return Depoimento.class.isAssignableFrom(clazz);
}
```

Na sequência, no método validate, iremos realizar a configuração da validação do que queremos fazer. A nossa estratégia para evitar esse ataque vai ser verificar se esses campos possuem abertura ou fechamento de tag (< ou >), pois com isso, o usuário não conseguirá inserir a tag `<script>` para colocar o código Javascript do ataque que ele realizou anteriormente, caso os campos contenham essas tags, iremos mostrar uma mensagem de erro. Coloque:

```
@Override
public void validate(Object target, Errors errors) {
    Depoimento depoimento = (Depoimento) target;
    String titulo = depoimento.getTitulo();
    String mensagem = depoimento.getMensagem();

    if(titulo.contains("<") || titulo.contains(">")){
        errors.rejectValue("titulo", "errors.titulo");
    }

    if(mensagem.contains("<") || mensagem.contains(">")){
        errors.rejectValue("mensagem", "errors.mensagem");
    }
}
```

Feito isso, na próxima etapa iremos configurar o arquivo que irá conter as respectivas mensagens que serão disponibilizadas para o usuário caso ele insira abertura ou fechamento de tags.