Software House

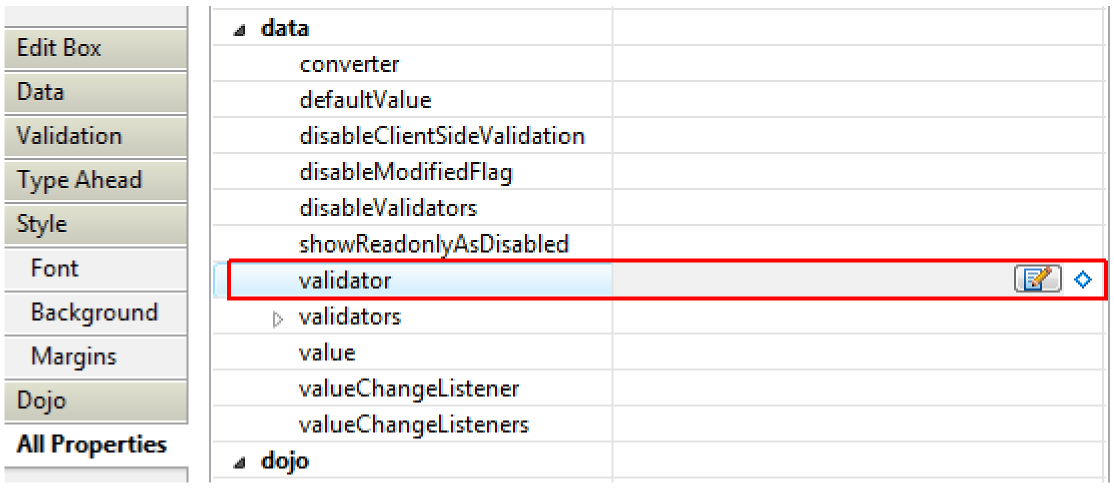
SPECIALIST IN SAGE AND NOTES/DOMINO DEVELOPMENT

Using beans as validators in XPages

June 23, 2013

This article describes how you can create your custom validators by implementing *javax.faces.validator.Validator* interface. But one problem with this is that you need to create a separate Java class for each validation, for e.g. you would have individual Java classes for validating email, credit card number and so on. And for each Java class you have to add *<validator>…</validator>* entry in *faces-config.xml*.

This article on JSF conversion and validation describes how a bean can be used for validation keeping all the validation methods in a single Java class. To do that you use the *validator* attribute under *data* section of control properties.



First create a Java class where all your validations would be stored, say *AcmeValidators*.

```
package uk.co.pipalia;

import java.io.Serializable;

public class AcmeValidators implements Serializable {
    // We will define its methods in a moment
}
```

Now we define its methods which would be used for validation. There are some prerequisites on how the these methods should be defined:

1. The method should be *public*
2. Its signature should take parameters of *FacesContext*, *UIComponent* and *Object*
3. Its return type should be *void*

You can name the method anything you want. So our Java class would now look something like this (if we are validating for email):

Update 23-June-2013: Removed line of code *facesContext.addMessage(component.getClientId(facesContext), message);* as it causes error message to be displayed twice if "Display Errors" control is used. Thanks Oliver.

```
package uk.co.pipalia;

import java.io.Serializable;
import javax.faces.application.FacesMessage;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.validator.ValidatorException;

public class AcmeValidators implements Serializable {

   private static final long serialVersionUID = 7157377379640149444L;

   // The method we would be using to validate email
   public void validateEmail(FacesContext facesContext, UIComponent component, Object value) {
      // Check if email is valid or not
      if (value.toString().indexOf('@') == -1) {
         // Create message saying that email is invalid
         FacesMessage message = new FacesMessage("Email is invalid.");
         // Throw exception so that it prevents document from being saved
         throw new ValidatorException(message);
      }
   }

}
```

Now create a *managed-bean* entry for the above class in *faces-config.xml*. Keep the scope to *view*.

```


   acmeValidators
   uk.co.pipalia.AcmeValidators
   view

```

To call the validation method set the validator property for control. For e.g.

One caveat here is that our validator will not fire if the field value is blank. So now you can put all your validation methods in a single class and then call on your controls.

Bonus tip: You would probably like to look at Apache Commons Validator which provides validations for email, credit card and more.

June 23, 2013 [http://www.pipalia.co.uk/using-validator-property-to-keep-all-your-validations-in-a-single-java-class/] | by Naveen Maurya | in Bean, faces-config.xml, java, jsf, Validation, xpages .

9 Comments

9 thoughts on "Using beans as validators in XPages"

Oliver Busse
June 23, 2013 at 3:52 pm

Nice one!
But if you add the message to the FacesContext AND throw a new ValidatorException you will get the message twice if you use the xp:messages control. With the single message control bound to the field everything works fine. So I omitted the addMessage method to get this working even with the messages control.

Naveen Maurya Post author
June 23, 2013 at 4:38 pm

Thanks Oliver. You were right. In "Display Errors" control it does show the error message twice. I have updated my code.

Fernando Rodriguez
August 9, 2013 at 2:17 pm

Hi,
excellent post!!

I had to add:
import javax.faces.validator.ValidatorException;
to throw the exception

Naveen Maurya Post author
August 9, 2013 at 6:12 pm

Thanks Fernando. I have updated my code.

Fernando Rodriguez
August 20, 2013 at 3:04 pm

Thanks for sharing!!

How could I compare with another field to validate?
I need to compare value with another field in the xpage, but I can´t get the value of other field

kind regards

Naveen Maurya Post author
August 21, 2013 at 2:18 pm

Have a look at this answer on StackOverflow – http://stackoverflow.com/a/14527326/1047998. It describes how you can get the value of a field from your Java Code. I have used same technique and it works in validators.

Martin Rolph
November 2, 2013 at 8:26 pm

Great technique!

Is there a way to customise the error message per control? e.g. if you have two URL fields the error message would just be URL is Invalid so you wouldn't know which one

---

Naveen Maurya Post author
November 3, 2013 at 11:24 am

Thanks Martin. As of now I haven't been able to find a way to attach custom error messages to a field. If I do I surely post an update here.

Samir Pipalia
November 6, 2013 at 10:55 pm

Hi Martin, thanks for your question. In order to get around this issue you can use component.getClientId(facesContext) and then based on the clientid create custom error message accordingly. Hope this helps.
