WoodyValidationRuleReference

General

For each validation rule, the failmessage (i.e. the message displayed to the user in case the validation failed) can be overridden by specifying a child **wd:failmessage** element inside the validation rule element. The failmessage can contain mixed content. Example:

```
<wd:datatype base="string">
  <wd:validation>
    <wd:email>
      <wd:failmessage>Not a valid email address!</wd:failmessage>
    </wd:email>
  </wd:validation>
</wd:datatype>
```

To provide locale-dependent messages, use i18n tags in combination with the I18NTransformer.

Often the values that validation rules will check are specified as expressions. Woody uses for this the xReporter expression interpreter

Summary

Validator	Allowed datatypes
assert	all datatypes
email	string
javascript	all datatypes
length	string
mod10	string
range	integer, long, decimal
regexp	string
value-count	all datatypes as part of multivaluefield

Description

assert

Evaluates the expression specified in the "test" attribute. This expression should have a boolean result, it should evaluate to either true or false. Example: Suppose there are 2 fields widgets `password` and `confirmPassword`. We can use assert inside `confirmPassword` to check if is equals to `password` widget:

```
<wd:assert test="password = confirmPassword">
  <wd:failmessage>The two passwords are not equal.</wd:failmessage>
</wd:assert>
```

email

Checks that a value is a valid email address. Example:

```
<wd:email/>
```

javascript

Allows to write arbitrary validators via javascript source code. Example:

```
<wd:javascript>
  // widget is a reference to the current widget.
  // Note that a repeater can have validation rules,
  // but cannot itself display a validation error,
  // so any validation errors must be set on another
  // widget, such as on a field in the offending row.
  var success = true;
  // Note that special characters must be encoded in
  // xml, such as the less-than symble below:
  if (widget.value &lt; 1) {
    widget.setValidationError(new Packages.org.apache.cocoon.forms.validation.ValidationError("Trouble!",
false));
    success = false;
  }
  // Must return true or false to indicate whether the validation succeeded.
  return success;
</wd:javascript>
```

length

Checks the length of strings. This validation rule can take 3 attributes: min, max and exact. You can use either of these three separately or min and max together. The values of these attributes are expressions. Example:

```
<wd:length min="2" max="4"/>

Another example:

<wd:length exact="2*2">
   <wd:failmessage>Must be 4 characters long!</wd:failmessage>
</wd:length>
```

mod10

Uses the "mod10" algorithm used to check the validity of credit card numbers such as VISA. This validation rule does not require any additional attributes. Example:

```
<wd:mod10>
   <wd:failmessage>Invalid credit card number.</wd:failmessage>
</wd:mod10>
```

range

Checks the numeric range. This validation rule can take 3 attributes: min, max and exact. You can use either of these three separately or min and max together. The values of these attributes are expressions. Example:

```
<wd:range min="2" max="4"/>

Another example:

<wd:range exact="2*2"/>
```

regexp

Checks that a string matches a regular expression. It requires a "pattern" attribute specifying the regexp. The regular expression library used is Jakarta ORO, see ORO API docs for some information. Example:

```
<wd:regexp pattern="[a-z]{3,5}">
   <wd:failmessage>Invalid code!</wd:failmessage>
</wd:regexp>
```

value-count

Checks the number of items selected in a multivaluefield. Again works with min, max and exact attributes. Example:

```
<wd:value-count min="2" max="4"/>

Another example:

<wd:value-count exact="2"/>
```