

The `validity` property of input fields

The `validity` property of input fields helps to get error details when the field is *invalid*. This property tests the different types of validation error.

Here is how to get the `validity` property of an input field:

```
var input =document.getElementById('IdOfField');  
var validityState_object =input.validity;
```

The possible values for the `validity` property are:

- `valueMissing`
- `typeMismatch`
- `patternMismatch`
- `tooLong`
- `rangeUnderflow`
- `rangeOverflow`
- `stepMismatch`
- `valid`
- `customError`

Here is [an example at JS Bin that shows how to test the different types of validation errors](#), or you may try it here in your browser (enter bad values, too big, too small, enter invalid characters, etc.):

Enter a value between 10 and 20

Submit Query

Note that testing it in Chrome/Opera/Firefox does not produce the same results. So far Opera has the most advanced implementations, however, entering "21" for example in the `<input type="number" max="20"/>` input field may yield some unexpected results depending on the browser. Test it yourself.

Source code:

```
<!DOCTYPE html>
<html>
...
<body>
  <script>
    function validate() {
      var input =document.getElementById('b');
      var validityState_object =input.validity;

      if(validityState_object.valueMissing) {
12.      input.setCustomValidity('Please set an age
(required)');
      } else if(validityState_object.rangeUnderflow) {
      input.setCustomValidity('Your value is too low');
      } else if(validityState_object.rangeOverflow) {
      input.setCustomValidity('Your value is too high');
      } else if(validityState_object.typeMismatch) {
      input.setCustomValidity('Type mismatch');
      } else if(validityState_object.tooLong) {
      input.setCustomValidity('Too long');
      } else if(validityState_object.stepMismatch) {
22.      input.setCustomValidity('stepMismatch');
      } else if(validityState_object.patternMismatch) {
      input.setCustomValidity('patternMismatch');
      } else {
      input.setCustomValidity('');
    }
  }
```

```
    }  
  </script>  
30. <form class="myForm">  
    <label for="b">Enter a value between 10 and 20: </label>  
    <input type="number" name="text" id="b" min="10" max="20"  
        required oninput='validate();' />  
    <button>Submit</button>  
  </form>  
  </body>  
39. </html>
```

THE VALIDATIONMESSAGE PROPERTY

It is also possible to get the validation error message, using the `validationMessage` property of input fields.

```
var input = document.getElementById('b');  
console.log("Validation message =  
" + input.validationMessage);
```

This will be useful for making custom error messages. More about this topic in the next section of the course.