

## input[number] DIRECTIVE IN ANGULARJS

- **input [number]** is one of the AngularJS input directive in module ng.
- AngularJS directive **input [number]** is used to create an HTML text input with number validation and transformation.
- The input value must be a number otherwise error is thrown.
- This directive executes at priority level 0.

### Syntax for input [number] directive in AngularJs:

```
<input type="number"  
      ng-model="string"  
      [name="string"]  
      [min = "string"]  
      [max = "string"]  
      [ng-min="string "  
      [ng-max="string"]  
      [step="string"]  
      [ng-step="string"]  
      [required = "string"]  
      [ng-required = "string"]  
      [ng-minlength = "string"]
```



```
[ng-maxlength = "string"]
```

```
[pattern = "string"]
```

```
[ng-pattern = "string"]
```

```
[ng-change = "string"] >
```

### Parameter Values:

Parameters	Description
ngModel	Defines angular expression to data-bind to.
name (optional)	Name of the form under which the control is available.
min(optional)	To set the min validation error key if the value entered is shorter than min.
max (optional)	To set the max validation error key if the value entered is greater than max.
ngMin (optional)	To set the min validation error key if the value entered is shorter than ngMin. It does not initiate HTML5 native validation. To Proceeds an expression.
ngMax(optional)	To set the <b>max</b> validation error key if the value entered is greater than <b>ngMax</b> . It does not initiate HTML5 native validation. To Proceeds an expression.
step (optional)	To sets the <b>step</b> validation error key if the value entered does not proper to the <b>step</b> constraint.

ngStep(optional)	It same as the step. To sets the <b>step</b> validation error key if the value entered does not proper to the <b>ngStep</b> constraint. It does not initiate HTML5 native validation. To proceeds an expression.
required (optional)	Denotes the required validation error key if the value is not entered.
ngRequired (optional)	Sets the required attribute and required validation constraint to the element when the ngRequired expression sets to true.  Instead of required use ngRequired when we want data-bind to the required attribute.
ngMinlength (optional)	States the minlength validation error key if the value is shorter than minlength.
ngMaxlength (optional)	States the maxlength validation error key if the value is longer than maxlength.
pattern(optional)	It is similar to <b>ngPattern</b> except that the attribute value is the actual string. It contains the regular expression body that will be converted to a regular expression as in the <b>ngPattern</b> directive.
ngPattern (optional)	States pattern validation error key if the ng model value does not match RegExp found by evaluating the Angular expression given in the attribute value.  If the expression evaluates to a RegExp object, then this is used directly.  If the expression evaluates to a string, then it will be converted to a RegExp after wrapping it in ^ and \$ characters.

	For instance, "abc" will be converted to new RegExp ('^abc\$').
ngChange (optional)	An expression of Angular to be executed when input changes due to user interaction with the input element.

### Sample coding for input[number] directive in AngularJS

```
<!DOCTYPE html>
<html>
  <head>
    <title>Wikitechy AngularJS Tutorial</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.6/angular.min.js">
    </script>
  </head>
  <body>
    <form ng-app="myApp" name="Form" ng-controller="NumberCtrl">
      <h3>input[number] directive example</h3>
      Enter the Number:
      <input type="number" name="input" ng-model="value"
        min="0" max="99" required>
      <span ng-show="Form.input.$error.required">Required!</span>
      <span ng-show="Form.input.$error.number">Not valid
number!</span>
      <p>value = {{value}}</p>
      <p>Form.input.$valid = {{Form.input.$valid}}</p>
      <p>Form.input.$error = {{Form.input.$error}}</p>
      <p>Form.$valid = {{Form.$valid}}</p>
      <p>Form.$error.required = {{!!Form.$error.required}}</p>
    </form>
    <script>
      var app=angular.module('myApp', []);
      app.controller('NumberCtrl', ['$scope', function($scope) {
```



```
    });  
  </script>  
</body>  
</html>
```

## Code Explanation input[number] directive in AngularJS:

```
<!DOCTYPE html>  
<html>  
  <head>  
    <title>Wikitechy AngularJS Tutorial</title>  
    <script src="https://ajax.googleapis.com/ajax/libs/angularjs/  
      1.5.6/angular.min.js"></script>  
  </head>  
  <body>  
    <form ng-app="myApp" name="Form" ng-controller="NumberCtrl">  
      <h3>input[number] directive example</h3>  
      Enter the Number:  
      <input type="number" name="input" ng-model="value"  
        min="0" max="99" required>  
      <span ng-show="Form.input.$error.required">Required!</span>  
      <span ng-show="Form.input.$error.number">Not valid number!</span>  
      <p>value = {{value}}</p>  
      <p>Form.input.$valid = {{Form.input.$valid}}</p>  
      <p>Form.input.$error = {{Form.input.$error}}</p>  
      <p>Form.$valid = {{Form.$valid}}</p>  
      <p>Form.$error.required = {{!!Form.$error.required}}</p>  
    </form>  
    <script>  
      var app=angular.module('myApp', []);  
      app.controller('NumberCtrl', ['$scope', function($scope) {  
        }]);  
    </script>  
  </body>  
</html>
```

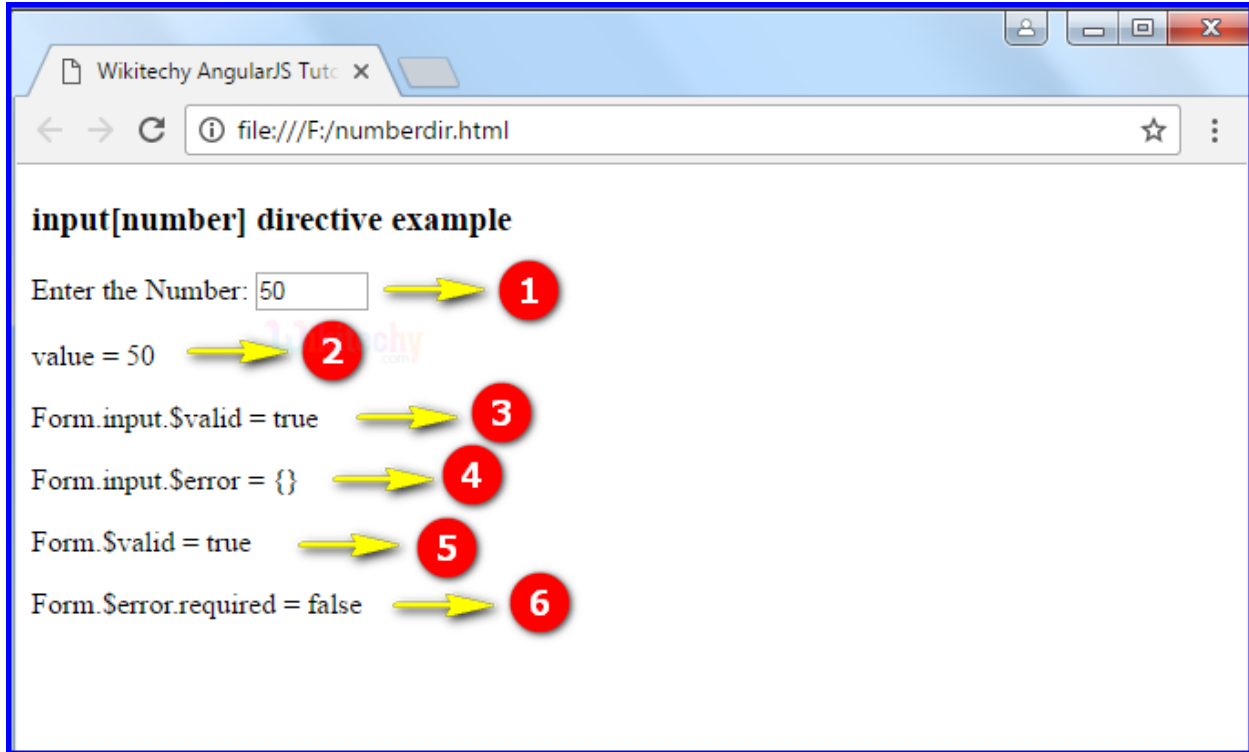
1. The **ng-app** specifies the root element ("myapp") to define AngularJS application.



2. Ng-controller specifies the application controller in AngularJS the controller value is given as **“NumberCtrl”**.
3. The **ng-model** bind an input field value to AngularJS application variable (**“value”**).
4. **min** is used to set the minimum number starts from 0.
5. **max** parameter is used to declare the maximum number is 99.
6. If the number is not entered into the text field then the **required** parameter is used.
7. Ng-show directive is used to hides HTML elements. If the required directive shows an error the content is displayed like **“Required”**
8. Ng-show directive is used to hides HTML elements. If the required directive shows an error the content is displayed like **“Not valid number”**
9. **Form.input.\$valid** to checks the valid number or not. If the number placed between 0 to 99 then the output displays true otherwise false.
10. **Form.input.\$error** to check whether the valid number or not .If the number specified in error it through the exception (Like “required”=“true) otherwise it is an empty curly braces ( { } ).
11. **Form.\$valid** is used to check whether the form is valid or not and output will be displayed in <p> tag.
12. **Form.error.\$required** is used to number is required or not. If the number is required and output will be displayed as false otherwise true.



## Sample Output input[number] directive in AngularJS:



**input[number] directive example**

Enter the Number:  → 1

value = 50 → 2

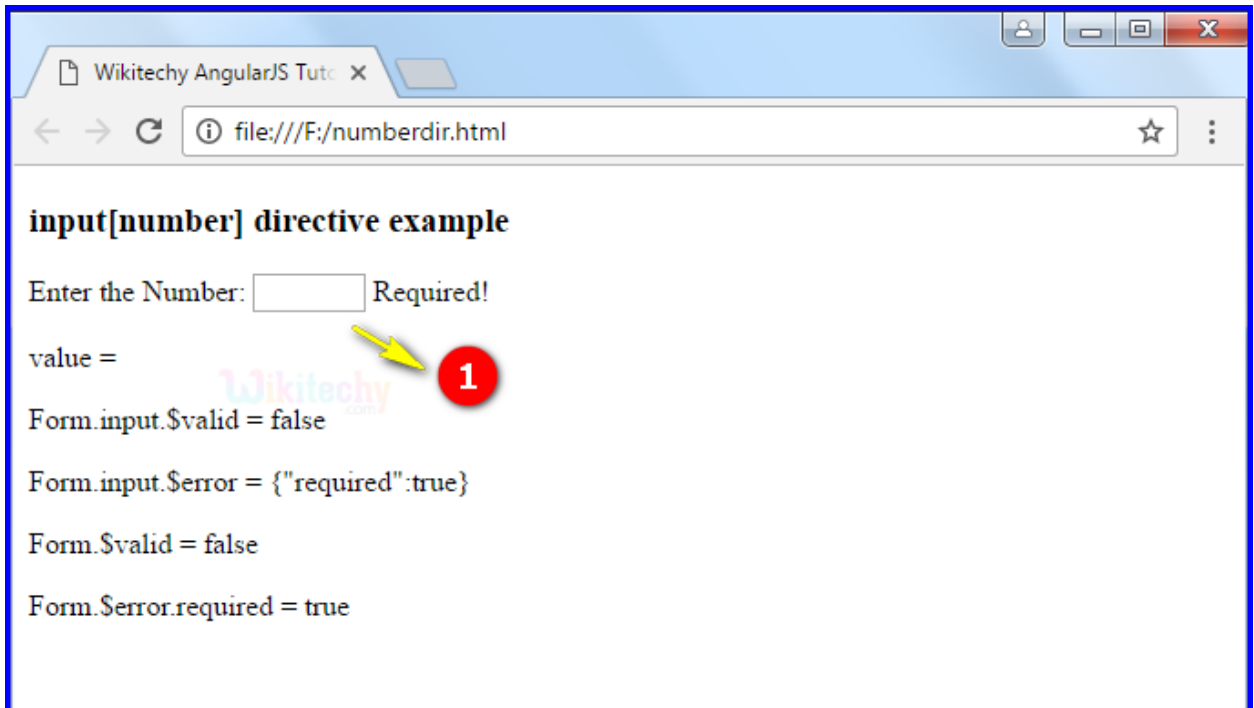
Form.input.\$valid = true → 3

Form.input.\$error = {} → 4

Form.\$valid = true → 5

Form.\$error.required = false → 6

1. If the user scroll the number in the input field.
2. The output will be displays as **value=50**.
3. The output displays true because it is consider as a number from 0-99.
4. The output displays empty curly braces it means does not thrown any error.
5. The output displays true it is valid number.
6. If the text box does not empty so the output displays as false.



1. If user does not specify any number in the input field and output displays as "Required".





**input[number] directive example**

Enter the Number: 100|  

value = 

```
Form.input.$valid = false
Form.input.$error = {"max":true}
Form.$valid = false
Form.$error.required = false
```

1. If user specify the number is 100 in the input text field. (The value starts from 0-99)
2. The output displays value is an empty.