

Search...

Getting started
Components

- M4Q
- About
- Population
- Constructor
- Ajax
- Animation
- Loops
- Visibility
- Effects
- Subtree functions
- Attributes
- Html, text and value
- Css and classes
- Position and size
- Manipulation
- DataSet
- Events
- Utils
- Base
- Containers
- Grid system
- Typography
- Tables
- Forms
- Buttons
- Images
- Figures
- Lists
- Form components
- Checkbox
- File input
- Input
- Input material
- Keypad
- Rating
- Radio
- Select
- Slider
- Double Slider
- Spinner
- Switch
- Tag input
- Textarea
- Menus
- App bar
- Bottom navigation
- Bottom sheet
- Menu
- Ribbon Menu
- Side bar
- Side navigation
- Controls
- Accordion
- Badge
- Carousel
- Cards
- Cube
- Counter
- Charms
- Chat
- Donut
- Image compare
- Image magnifier
- Gravatar
- List
- ListView
- Master
- NavView
- Panels
- Progress & Activity
- Streamer
- Stepper
- Splitter
- Tabs
- Tabs material
- Table
- Tiles
- TreeView
- Wizard
- Information
- Dialogs
- Info box
- Hints
- Notify system
- Popovers
- Toasts
- Windows
- Date & time
- Calendar
- Calendar picker
- Date picker
- Time picker
- Countdown
- Formatting date
- Media

Slider

Let the user specify a numeric value with slider component.

About slider

Component `slider` let the user specify a numeric value which must be no less than a given value, and no more than another given value. To create `slider` add attribute `data-role="slider"` to element.

```
<input data-role="slider">
```

Metro 4 support two type of slider: `vertical` and `horizontal` (default). To create `vertical` slider, add attribute `data-vertical="true"` to element. To set specific size of the `slider`, use attribute `data-size`. He sets the `width` for `horizontal` and `height` for the `vertical` slider.

```
<input data-role="slider" data-vertical="true" data-size="200">
```

Slider value

The slider in Metro 4 can return **two** types of values: the `actual value` and its equivalent in `percent`. How the value will be returned determines the attribute `data-return-type`, it can take two values: `value` and `percent`. By default slider return `actual value`. To return value in percent equivalent, set this attribute to `data-return-type="percent"`.

Return actual value

-4040

Return percent

-4040

```
<input data-role="slider" data-return-type="value">
<input data-role="slider" data-return-type="percent">
```

Accuracy

You can set `accuracy` for slider with attribute `data-accuracy`.

0

```
<input data-role="slider" data-accuracy="5" data-hint-always="true">
```

Additional target

You can put slider value to additional targets. To set it, add attribute `data-target="..."` to element. Value for this attribute must be selector specific string.

0

```
<input data-role="slider" data-target="#slider-target1, #slider-target2">
<div class="p-2 bg-cyan fg-white text-center" id="slider-target"></div>
<input type="text" id="slider-target2">
```

Buffer

Very often the slider is used to create a control in the media player where it is necessary to show the buffering process. Slider Metro 4 provides such a possibility "out of the box." To set `buffer` use attribute `data-buffer` or special methods (see below).

```
<input data-role="slider" data-buffer="60" data-value="20">
```


Important! For buffer you must use **percent** value!

Hint

Table of contents


- Slider
- About slider
- Slider value
- Accuracy
- Additional target
- Buffer
- Hint
- Position
- Value
- Options
- Events
- Methods
- Observe
- Customize

You can enable `hint` for slider. To enable `hint` add attribute `data-hint="true"` to element.

A horizontal slider with a green track and a grey track. A black marker is positioned at the 50% mark. A small grey hint box is visible above the marker.

```
<input data-role="slider" data-hint="true">
```

Also you can make the hint as permanent. To set `hint` in `permanent` mode add attribute `data-hint-always="true"`.

A horizontal slider with a green track and a grey track. A black marker is positioned at the 50% mark. A small grey hint box is visible above the marker.

```
<input data-role="slider" data-hint="true" data-hint-always="true">
```

Hint position


You can set `hint position` with attribute `data-hint-position`. To set specific position use next values for this attribute: `top`, `bottom`, `left` and `right`.

Four horizontal sliders with green tracks and grey tracks. Each has a black marker and a red hint box. The hint boxes are positioned at the top, bottom, left, and right of the markers respectively.

```
<input data-role="slider" data-hint-position="top">
<input data-role="slider" data-hint-position="bottom">
<input data-role="slider" data-hint-position="left">
<input data-role="slider" data-hint-position="right">
```

Hint value

You can use `template` for hint value with two variables `$1` and `$2`. First - actual value, second - percent value.

A horizontal slider with a green track and a grey track. A black marker is positioned at the 50% mark. A blue hint box is visible above the marker, displaying 'V: 0, 50%'.

```
<input data-role="slider"
  data-hint-mask="V: $1, $2%"
  data-hint-always="true"
  data-value="0"
  data-show-min-max="true"
  data-min="-40" data-max="40">
```

Options

Options	Data-*	Default	Desc
min	<code>data-min</code>	0	Min slider value
max	<code>data-max</code>	100	Max slider value
showMinMax	<code>data-show-min-max</code>	false	When true, additional block will be created and inserted before slider
value	<code>data-value</code>	0	Initial slider value
accuracy	<code>data-accuracy</code>	0	Slider accuracy
buffer	<code>data-buffer</code>	0	Initial slider buffer value
hint	<code>data-hint</code>	false	Show slider hint
hintAlways	<code>data-hint-always</code>	false	Show slider hint permanent
hintPosition	<code>data-hint-position</code>	top	Hint position. Can be: top, left, right, bottom
hintMask	<code>data-hint-mask</code>	<code>\$1</code>	Hint output mask (template)
vertical	<code>data-vertical</code>	false	Vertical slider orientation
target	<code>data-target</code>		Additional targets for slider value
returnType	<code>data-return-type</code>		How value will be returned: value or percent
size	<code>data-size</code>	0	Slider specific size

Events

When slider works, it generated the numbers of events. You can use callback for this event to behavior with slider.

Event	Data-*	Desc
<code>onStart(val, percent, slider)</code>	<code>data-on-start</code>	Fired when start sliding
<code>onStop(val, percent, slider)</code>	<code>data-on-stop</code>	Fired when stop sliding
<code>onMove(val, percent, slider)</code>	<code>data-on-move</code>	Fired when user move slider marker
<code>onSliderClick(val, percent, slider)</code>	<code>data-on-slider-click</code>	Fired when user clicked on slider
<code>onChangeValue(val, percent, slider)</code>	<code>data-on-change-value</code>	Fired when slider value was changed
<code>onChangeBuffer(val, slider)</code>	<code>data-on-change-buffer</code>	Fired when slider buffer value was changed
<code>onFocus(val, percent, slider)</code>	<code>data-on-focus</code>	Fired when slider marker get focus

Event	Data-*	Desc
onBlur(val, percent, slider)	<code>data-on-blur</code>	Fired when slider marker loses focus
onSliderCreate(slider)	<code>data-on-slider-create</code>	Fired when slider was created

Also you can use standard `onchange` event for `input` with `data-role="slider"` .

-100100

```
<input data-role="slider"
  data-show-min-max="true"
  data-min="-100" data-max="100"
  data-on-change="${'#event-receiver'}.val('Value: '+arguments[0]+'', '+arguments[1]+'%')">

<input type="text" id="event-receiver">
```

Methods

In additional, You can use slider methods to interact with the component.

- **val()** - get value
- **val(v)** - set value
- **buff()** - get buffer value
- **buff(v)** - set buffer value

```
<div class="row">
  <div class="cell-md-6">
    <input type="number" min="0" max="100"
      oninput="${'#slider-methods'}.data('slider').val(this.value)">
  </div>
  <div class="cell-md-6">
    <input data-role="slider" data-accuracy="10" id="slider-methods">
  </div>
</div>
```

Observe

You can change attributes `data-value` , `data-buffer` at runtime and slider will be updated.

Change value in input and observe how value changed in slider

0

```
<div class="row">
  <div class="cell-md-6">
    <input type="number" min="0" max="100" id="inp-obs" value="0">
  </div>
  <div class="cell-md-6">
    <input data-role="slider" id="slider-obs">
  </div>
</div>
<script>
  $('#inp-obs').on("keyup input change paste propertychange", function() {
    $('#slider-obs').attr('data-value', this.value);
  })
</script>
```

Customize

If you need to customize the slider, you can use next options:

Option	Data-*	Desc
clsSlider	<code>data-cls-slider</code>	Additional class for slider
clsBackside	<code>data-cls-backside</code>	Additional class for slider backside
clsComplete	<code>data-cls-complete</code>	Additional class for slider complete
clsBuffer	<code>data-cls-buffer</code>	Additional class for slider buffer
clsMarker	<code>data-cls-marker</code>	Additional class for slider marker
clsHint	<code>data-cls-hint</code>	Additional class for slider hint
clsMinMax	<code>data-cls-min-max</code>	Additional class for min-max info block
clsMin	<code>data-cls-min</code>	Additional class for min value for info block
clsMax	<code>data-cls-max</code>	Additional class for max value for info block

0100

```
<input data-role="slider"
  data-value="50">
```

```
data-buffer="75"  
data-hint-always="true"  
data-hint-position="bottom"  
data-show-min-max="true"  
data-cls-backside="bg-dark"  
data-cls-marker="bg-blue border-50 custom-marker"  
data-cls-hint="bg-cyan custom-marker shadow-2"  
data-cls-complete="bg-red"  
data-cls-buffer="bg-yellow"  
data-cls-min-max="bg-green fg-white p-2 mb-3-minus">
```

Secret classes :)

■

■

●

```
<input data-role="slider" class="thin">  
<input data-role="slider" class="ultra-thin">  
<input data-role="slider" class="ultra-thin cycle-marker">
```