Search..

Getting started Components

About Aiax Animation Visibility Effects Subtree functions Attributes Html, text and value Css and classes Position and size Manipulation

Containers

DataSet Utils

Grid system Typography Buttons Images

Checkbox File input Input Input material Keypad Rating Radio Slider Double Slider Spinner

App bar Bottom navigation

Tag input

Textarea

Bottom sheet Ribbon Menu Side bar Side navigation

Accordion

Badge Carousel Cards Cube

Charms Chat

Image compare Image magnifier Gravatar

Master

Progress & Activity

Stepper Splitter Tabs

NavView

Tabs material

Table Tiles TreeView

Wizard

Dialogs Info box

Hints Notify system

Popovers

Windows

Calendar Calendar picker Date picker

Media

Time picker Countdown

Progress & Activity



Progress and activity indicators are visual indications of an app loading content or process executed.

Activity indicators

Metro 4 implements several activity indicators, which you can use easily and simply. To create activity indicator add data-role="activity" activity indicators: ${\tt ring}$, ${\tt metro}$, ${\tt square}$, ${\tt cycle}$.

In addition you can define color subtype for activity indicator with attribute data-style and predefined values: light (default), dark and

Ring



Metro



Square

```
<div data-role="activity" data-type="square"></div>
                                                                                                                                                                                                                                                                 4
data role="activity" data-type="square" data-style="dark"></div>
<div data-role="activity" data-type="square" data-style="color"></div>
<div data-role="activity" data-type="square" data-style="color"></div></ti>
```

Cycle



Simple

```
<div data-role="activity" data-type="simple"></div>
<div data-role="activity" data-type="simple" data-style="dark"></div></div>
                                                                                                                                                                               4
<div data-role="activity" data-type="simple" data-style="color"></div</pre>
```

Activity overlay

This feature available from 4.1.20

Metro 4 contains object to create activity overlay. This functionality stored in object Metro.activity. Object contains two methods: open({...}) and close(activity) .

Options:

- type string, valid activity type
- style string, valid activity style
- autoHide integer, milliseconds
- overlavClickClose boolean, true or false
- overlayColor string, hex value, ex: #fffffff
- overlayAlpha float, from 0 to 1
- text additional text. New in 4.2.0



Table of contents

(B)

Progress & Activity Activity indicators Ring

Metro Square Cycle

Activity overlay Progress indicator

> Buffer Load

Line Events

Callbacks Observe Set and get value

```
Video playe
  Audio player
  Collapse
  Color module
  Form validator
  Hotkeys
  Micro templates
  Ripple
  Storage
  Session storage
  Sorter
  Touch and swipe
Utilities
Animations
Additional
```

```
<button class="button" onclick="</pre>
    Metro.activity.open({
        overlayClickClose: true
">Open</button>
<button class="button" onclick="</pre>
    var activity = Metro.activity.open({
    type: 'square',
        overlayColor: '#ffff',
        overlayAlpha: 1
    setTimeout(function(){
       Metro.activity.close(activity);
">Open</button>
<button class="button" onclick="</pre>
    Metro.activity.open({
        type: 'square',
        overlayColor: '#ffff',
        overlayAlpha: 1,
text: '<div class=\'mt-2 text-small\'>Please, wait...</div>',
        overlayClickClose: true
">Open</button>
```

Progress indicator

Metro 4 implements four types of progress indicator. To create progress indicator add data-role="progress" attribute to element and define additional options. The progress indicator type you can define with data-type=* attribute. To set value and buffer add attributes data-value or/and data-buffer.

Progress bar

```
<div data-role="progress" data-value="35"></div>
<div data-role="progress" data-value="35" data-small="true"></div>
```

Progress bar with buffer

```
<div data-role="progress" data-type="buffer"
   data-value="35" data-buffer="60"></div>
<div data-role="progress" data-type="buffer"
   data-value="35" data-buffer="60" data-small="true"></div>
```

Progress bar with buffer and load indicator

```
<div data-role="progress" data-type="load"
    data-value="35" data-buffer="75"></div>
</div data-role="progress" data-type="load"
    data-value="35" data-buffer="75" data-small="true"></div>
```

Progress line

```
<div data-role="progress" data-type="line" ></div>
<div data-role="progress" data-type="line" data-small="true" ></div>
```

Customize

To set your own custom color use $_{\tt data-cls-back}$, $_{\tt data-cls-bar}$ and $_{\tt data-cls-buffer}$ attributes.

```
<div data-role="progress"
   data-type="buffer"
   data-cls-bar="bg-plue"
   data-cls-bar="bg-blue"
   data-cls-buffer="bg-pink"
   data-value="25" data-buffer="65"></div>
```

Events

When the value or buffer changes in the progress, a change or/and buffer events is fired. You can use this events to observe progress value and buffer.

```
$("*progress").on("valuechange", function(val){
    console.log(val);
});

$("*progress").on("bufferchange", function(val){
    console.log(val);
```

});

Callbacks

Progress indicator implements several callbacks to respond to a change in the status of the indicator:

Function	Data-*	Desc
onValueChange(val, el)	data-on-value-change	Fired when value changes
onBufferChange(val, el)	data-on-buffer-change	Fired when buffer changes
onComplete(val, el)	data-on-complete	Fired when value is 100%
onBuffered(val, el)	data-on-buffered	Fired when buffer is 100%
onProgressCreate(el)	data-on-progress-create	Fired when element was created

```
<div data-role="progress" data-type="load"
    data-value="35" data-buffer="75"
    data-on-complete="alert('Complete!!!')"
    data-on-value-change="console.log(arguments)"
    data-on-buffer-change="console.log(arguments)">
</div>
```

Observe

If you change data-value or data-buffer attributes at runtime, your progress will be updated.

```
Value
Buffer
<div id="progress-observe"
                                                                                                                                                                         4
     data-role="progress"
data-type="buffer" class="mb-4"></div>
<div>
    <input id="progress-observe-value"</pre>
         class="w-100" type="range"
min="0" max="100" value="0">
     <input id="progress-observe-buffer"
  class="w-100" type="range"
  min="0" max="100" value="0">
</div>
<script>
     $(function(){
          $("#progress-observe-value").on("input change", function(){
                $("#progress-observe").attr('data-value', this.value);
          $("#progress-observe-buffer").on("input change", function(){
$("#progress-observe").attr('data-buffer', this.value);
     })
</script>
```

Set and get value

Component progress provides method to get and set value for progress. To set or get value use method val() . Also you can set and get values for buffer with method buff() .

```
Value

Buffer

0

Get value

var progress = $("#element").data("progress");
var progress_value;
var buffer_value;

// set value
progress.val(35);
progress.buff(65);

// get value
progress_value = progress.val();
buffer_value = progress.buff();
```