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

Table of contents

Countdown

Create a countdown element in Metro 4 is very simple with the use of a special countdown component.

About

The goal of the `countdown` component is to create an element that can count down time from the specified time point. To create `countdown` add `data-role="countdown"` attribute to element and `setup options`.

03 . 17 . 59 . 54
DAYS HOURS MINS SECS

03 . 17 . 59 . 54
DAYS HOURS MINS SECS

03 . 17 . 59 . 54
DAYS HOURS MINS SECS

03 . 17 . 59 . 55
DAYS HOURS MINS SECS

```
<div data-role="countdown" data-hours="90"></div>
```

Options

You can set `options` to set times and visual styles for the `countdown` element.

Options	Data-*	Default	Desc
days	`data-days`	0	Add days to interval
hours	`data-hours`	0	Add hours to interval
minutes	`data-minutes`	0	Add minutes to interval
seconds	`data-seconds`	0	Add seconds to interval
date	`data-date`	Now	Set start point interval
daysLabel	`data-days-label`	days	Set label for days
hoursLabel	`data-hours-label`	hours	Set label for hours
minutesLabel	`data-minutes-label`	mins	Set label for minutes
secondsLabel	`data-seconds-label`	secs	Set label for seconds
clsZero	`data-cls-zero`		Set additional class for zero value
clsAlarm	`data-cls-alarm`		Set additional class for alarm state
clsDays	`data-cls-days`		Set additional class for days
clsHours	`data-cls-hours`		Set additional class for hours
clsMinutes	`data-cls-minutes`		Set additional class for minutes
clsSeconds	`data-cls-seconds`		Set additional class for seconds
onCountdownCreate	`data-on-countdown-create`	Metro.noop	Fired when component is created
onAlarm	`data-on-alarm`	Metro.noop	Fired when countdown is done
onTick	`data-on-tick`	Metro.noop	Fired when countdown is ticks
onZero	`data-on-zero`	Metro.noop	Fired when part is sets to zero

Setup interval

By default the `start point` for interval is `today`. To this point you can add `days`, `hours`, `minutes` and `seconds`.

01 . 02 . 02 . 58
DAYS HOURS MINS SECS

```
<div data-role="countdown"
     data-days="1"
     data-hours="2"
     data-minutes="3"
     data-seconds="4"></div>
```

Also you can set start point manually. To set start point add `data-date` attribute to element. Value for this attribute can be **VALID** date string.

00 . 00 . 00 . 00
DAYS HOURS MINS SECS

```
<div data-role="countdown"  data-date="01/01/2018"
     data-days="1"
     data-hours="2"
     data-minutes="3"
     data-seconds="4"></div>
```

Media
Video player
Audio player
Tools
Collapse
Color module
Draggable
Dropdown
Form validator
Hotkeys
Micro templates
Ripple
Storage
Session storage
Sorter
Touch and swipe
Utilities
Animations
Additional

Customize

You can customize `countdown` control with special attributes: `data-cls-zero` , `data-cls-alarm` , `data-cls-days` , `data-cls-hours` , `data-cls-minutes` , `data-cls-seconds` . With this attributes you can define classes to customize visual style for `countdown` .

01 00 00 09
DAYS HOURS MINS SECS

Start

```
<div data-role="countdown"
     data-start="false"
     data-days="1"
     data-seconds="10"
     data-cls-part="no-divider"
     data-cls-days="bg-orange fg-white"
     data-cls-hours="bg-red fg-white"
     data-cls-minutes="bg-green fg-white"
     data-cls-seconds="bg-blue fg-white"
     data-cls-zero="bg-light fg-lightGray"
></div>
```

Events

When the `countdown` is running, it generates various `events` that you can use. How to define Metro 4 components events see Events rules.

- **onCountdownCreate(el)** - fired when component created
- **onAlarm(now, el) - fired when countdown is done**
- **onTick({d, h, m, s}, el) - fired every tick (one second)**
- **onZero(part, el) - fired when part sets to zero**

01 . 00 . 00 . 09
DAYS HOURS MINS SECS

Start

```
<div id="countdown_events" data-role="countdown"
     data-start="false"
     data-days="1"
     data-seconds="10"
     data-on-countdown-create="console.log('Countdown was created!')"
></div>
<button class="button mt-2"
    onclick="$('#countdown_events').data('countdown').start()">Start</button>
```

Methods

Each `countdown` component contains same methods:

- **start()** - Start countdown
- **stop()** - Stop countdown
- **pause()** - Pause countdown
- **isPaused()** - Return true, if countdown is paused
- **getTimepoint(asDate)** - Return countdown start point
- **getBreakpoint(asDate)** - Return countdown break point
- **getLeft()** - Return object. It contains days, hours, minutes and seconds to stop

01 . 00 . 00 . 09
DAYS HOURS MINS SECS

Start Stop Pause Breakpoint Minutes left

```
<div id="countdown_methods" data-role="countdown"
     data-start="false"
     data-days="1"
     data-seconds="10"
></div>

<button class="button mt-2"
    onclick="$('#countdown_methods').data('countdown').start()">Start</button>

<button class="button mt-2"
    onclick="$('#countdown_methods').data('countdown').stop()">Stop</button>

<button class="button mt-2"
    onclick="$('#countdown_methods').data('countdown').pause()">Pause</button>

<button class="button mt-2"
    onclick="alert($('#countdown_methods').data('countdown').getBreakpoint(true))">Breakpoint</button>

<button class="button mt-2"
    onclick="alert('Minutes left: ' + $('#countdown_methods').data('countdown').getLeft().minutes)">Minutes left</button>
```

i18n

To change `language` for `countdown` use attribute `data-locale` . All predefined language files stored in `i18n` folder. You can add own internationalisation file to `i18n` folder with same structure and use it. For more information about `i18n` see this page.

03 . 17 . 59 . 54
TAGE UHR MIN SEK

```
<div data-role="countdown" data-hours="90" data-locale="de-DE"></div>
```

You can change locale in runtime. To change locale, update value for attribute `data-locale` .

en-US de-DE hu-HU uk-UA ru-RU

03 . 17 . 59 . 54
DAYS HOURS MINS SECS

```
<div class="d-flex flex-justify-center mb-4">
    <button class="button m-1" onclick="changeLocale($(this).text())">en-US</button>
    <button class="button m-1" onclick="changeLocale($(this).text())">de-DE</button>
    <button class="button m-1" onclick="changeLocale($(this).text())">hu-HU</button>
    <button class="button m-1" onclick="changeLocale($(this).text())">uk-UA</button>
    <button class="button m-1" onclick="changeLocale($(this).text())">ru-RU</button>
</div>

<div id="countdown_locale"
    data-role="countdown"
    data-hours="90"
    style="font-size: 36px"></div>

<script>
    function changeLocale(locale){
        $('#countdown_locale').attr('data-locale', locale);
    }
</script>
```