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

InfoBox

Metro 4 provides the ability to easily create information boxes. InfoBox can be predefined or created in runtime.

About

New in 4.2.7

InfoBox inform users about a specific task and may contain critical information. InfoBox has next structure:

×

What is Lorem Ipsum?

Lorem Ipsum is simply dummy text of the printing and typesetting industry.

```
<div class="info-box">
    <span class="button square closer"></span>
    <div class="info-box-content">
        ...
    </div>
</div>
```

Create InfoBox

Metro 4 provides two ways for create InfoBox:

- Predefined method
- Runtime method

Predefined method

First-off you must create `HTML` element with special structure. Then add any options over `data-*` attributes. And at the end add an attribute `data-role="infobox"` to your element for initialization `InfoBox` . After the initialization, the dialog will be hidden until it is called.

```
<div class="info-box" data-role="infobox">
    <span class="button square closer"></span>
    <div class="info-box-content">
        <h3>What is Lorem Ipsum?</h3>
        <p>Lorem Ipsum is simply dummy text of the printing and typesetting industry.</p>
    </div>
</div>
```

Open InfoBox

Create InfoBox at runtime

To create `InfoBox` at `runtime` you mus execute method `Metro.infobox.create` with required parameters and options.

Create InfoBox

```
var html_content =
    "<h3>What is Lorem Ipsum?</h3>" +
    "<p>Lorem Ipsum is simply dummy text...</p>";
Metro.infobox.create(html_content);
```

InfoBox types

You can create five type of InfoBoxes: `default` , `success` , `info` , `warning` and `alert` . To create it you must use attribute `data-type="..."` with a corresponding values for predefined InfoBoxes oe second parameter for `Metro.infobox.create` method.

```
<div class="info-box" data-role="infobox" data-type="alert">
    <span class="button square closer"></span>
    <div class="info-box-content">
        <h3>What is Lorem Ipsum?</h3>
        <p>Lorem Ipsum is simply dummy text of the printing and typesetting industry.</p>
    </div>
</div>
```

```
Metro.infobox.create("<p>Lorem Ipsum is simply dummy text...</p>", "alert");
```

Default Success Info Alert Warning

Options

You can set any options to define InfoBox:

Options	Data-*	Default	Description
type	`data-type`		Set InfoBox type: default, success, info, alert and warning

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

Options	Data-*	Default	Description
width	`data-width`	480	Set InfoBox width
height	`data-height`	auto	Set InfoBox height
height	`data-height`	auto	Set InfoBox height
overlay	`data-overlay`	true	Add overlay when InfoBox is opened
overlayColor	`data-overlay-color`	#000000	Overlay color. Can be hex color value or transparent
overlayAlpha	`data-overlay-alpha`	0.5	Overlay color alpha channel value.
autoHide	`data-auto-hide`	0	If this options > 0, InfoBox closed after this timeout in milliseconds
removeOnClose	`data-remove-on-close`	false	If this options is `true` InfoBox will be removed after closes from DOM
closeButton	`data-close-button`	true	Hide close InfoBox button
clsBox	`data-cls-box`		Additional class for InfoBox
clsBoxContent	`data-cls-box-content`		Additional class for InfoBox content block
clsOverlay	`data-cls-overlay`		Additional class for InfoBox overlay
onOpen	`data-on-open`	Metro.noop	This `callback` executed after InfoBox is open
onClose	`data-on-close`	Metro.noop	This `callback` executed after InfoBox is close
onInfoBoxCreate	`data-on-info-box-create`	Metro.noop	This `callback` executed after InfoBox is created

Events

When InfoBox works, it generated a number of events. You can use callbacks to these events to interact with it.

Events	Data-*	Desc
`onOpen(elem)`	`data-on-open`	Fired when InfoBox is open
`onClose(elem)`	`data-on-close`	Fired when InfoBox is close
`onInfoBoxCreate(elem)`	`data-on-info-box-create`	Fired when InfoBox was created

Open InfoBox

```
...html
<button class="button" onclick="infoBoxEventsExample1()">Open InfoBox</button>

...javascript
function infoBoxEventsExample1() {
    var el = Metro.infobox.create(
            "This is a simple InfoBox. Wait...",
            "",
            {
                closeButton: false,
                onOpen: function(){
                    var ib = $(this).data("infobox");
                    setTimeout(function(){
                        ib.setContent("Content can be changed at runtime. Wait...");
                        setTimeout(function(){
                            ib.setContent("Type also can be changed. Wait...");
                            ib.setType("alert");
                            setTimeout(function(){
                                ib.setType("info");
                                setTimeout(function(){
                                    ib.setType("warning");
                                    setTimeout(function(){
                                        ib.setContent("Say goodbye...");
                                        ib.setType("success");
                                        setTimeout(function(){
                                            ib.close();
                                        }, 2000)
                                    }, 2000)
                                }, 2000)
                            }, 2000)
                        }, 2000)
                    }, 2000)
                }
            }
    );
}
```

Methods

To interact with component you can use methods:

Method	Desc
`open()`	Use this method to open InfoBox
`close()`	Use this method to close InfoBox
`setContent(c)`	Use this method to change InfoBox content
`setType(t)`	Use this method to change InfoBox type
`isOpen()`	Use this method to check InfoBox state

InfoBox object

Metro 4 contains special object `Metro.infobox` to manipulate `InfoBoxes`. This object contains next methods:

- **create(text, type, options, not_open)** - create InfoBox
- **open(el, content, type)** - open InfoBox
- **close(el)** - close InfoBox
- **isInfoBox(el)** - check if element is InfoBox instance
- **isOpen(el)** - check if InfoBox is open
- **setContent(content)** - Set InfoBox content
- **setType(type)** - Set InfoBox type