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

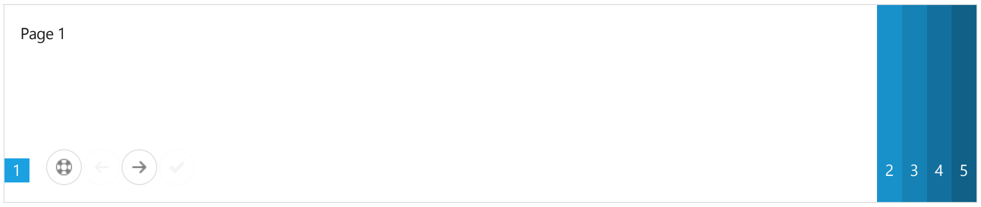
Date & time
Calendar
Calendar picker
Date picker
Time picker
Countdown
Formatting date

Wizard

Create user friendly wizard easy with Metro 4 Wizard component.

About

With the `wizard` control, you can easily generate `multi-step` wizard dialogs. To create `wizard` create wizard structure and add role `wizard` with attribute `data-role="wizard"` to element. Each page a `section` element with sub block element with class `.page-content`.



```
<div data-role="wizard">
    <section><div class="page-content">Page 1</div></section>
    <section><div class="page-content">Page 2</div></section>
    <section><div class="page-content">Page 3</div></section>
    <section><div class="page-content">Page 4</div></section>
    <section><div class="page-content">Page 5</div></section>
</div>
```

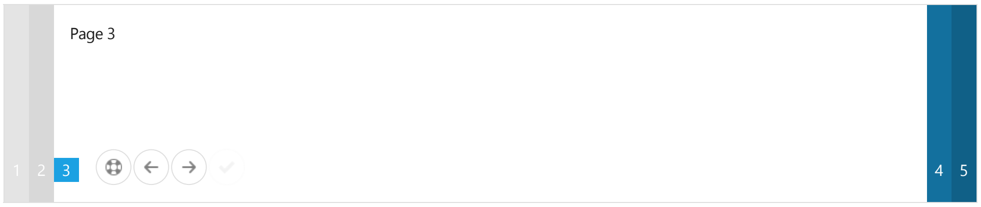
Controls

When Metro 4 create a `wizard` component, it adds buttons to navigate on wizard steps: `help`, `prev`, `next` and `finish`. For each button you can use callback to interact with wizard:

Event	Data-*
onHelpClick()	`data-on-help-click`
onPrevClick()	`data-on-prev-click`
onNextClick()	`data-on-next-click`
onFinishClick()	`data-on-finish-click`

Start page

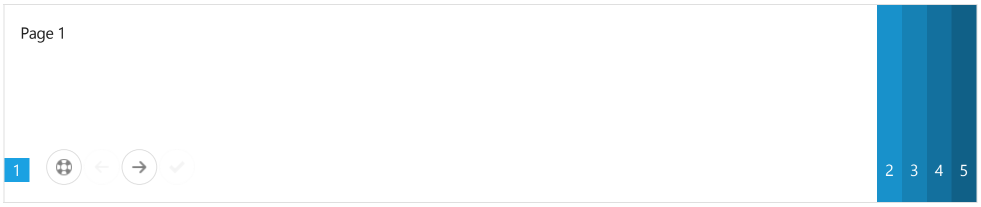
You can specify page number fro start wizard with attribute `data-start="..."`. Value for this attribute must be page index from `1` to `pages count`.



```
<div data-role="wizard" data-start="3">
    <section><div class="page-content">Page 1</div></section>
    <section><div class="page-content">Page 2</div></section>
    <section><div class="page-content">Page 3</div></section>
    <section><div class="page-content">Page 4</div></section>
    <section><div class="page-content">Page 5</div></section>
</div>
```

Finish page

You can specify page, when finish button cam be enabled. To set it, use attribute `data-finish="..."`. Value for this attribute must be page index from `1` to `pages count`.



```
<div data-role="wizard" data-finish="3">
    <section><div class="page-content">Page 1</div></section>
    <section><div class="page-content">Page 2</div></section>
    <section><div class="page-content">Page 3</div></section>
    <section><div class="page-content">Page 4</div></section>
    <section><div class="page-content">Page 5</div></section>
</div>
```

Table of contents

Events

You can use additional events to interact with wizard:

Event	Data-*	Desc
onPage(index, page, wizard)	`data-on-page`	Fired when page changed
onBeforePrev(index, page, element)	`data-on-before-prev`	When this function return false, page not changed
onBeforeNext(index, page, element)	`data-on-before-next`	When this function return false, page not changed

Methods

You can use `wizard` methods to interact with component.

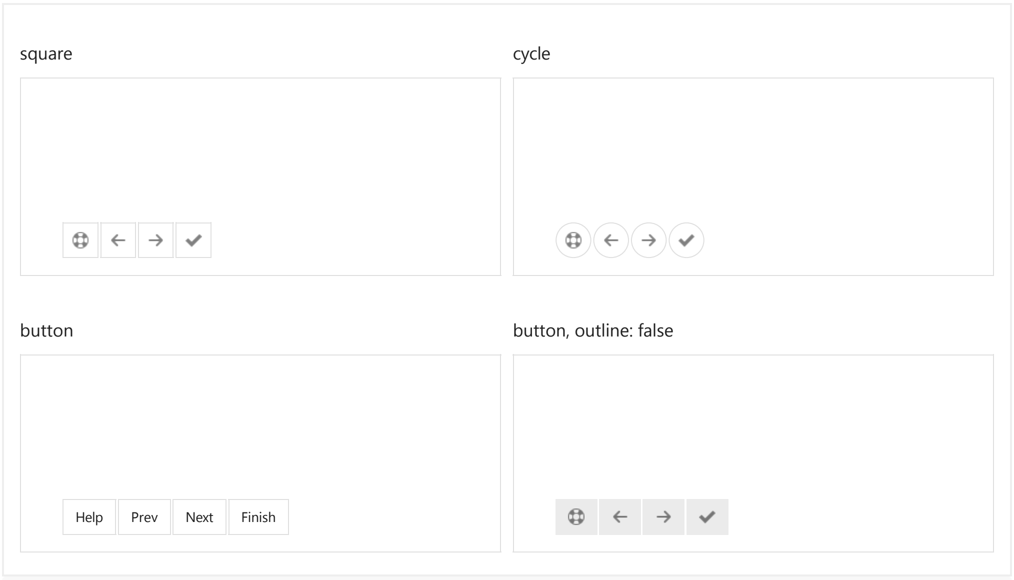
- **next()** - switch to next page
- **prev()** - switch to prev page
- **first()** - switch to first page
- **last()** - switch to last page
- **toPage(p)** - switch to specified page (p - page index from 1)

Customize

You can change a style for the `wizard` and it controls and elements with special `data attributes`.

Controls style

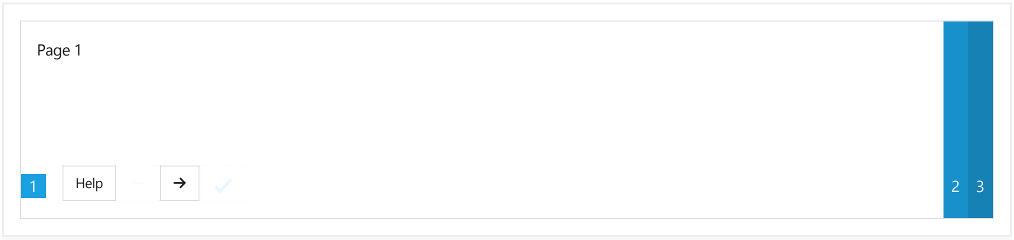
Two data attributes `data-button-mode` and `data-button-outline` are responsible for controls buttons style. With attribute `data-button-mode` you can set three modes for buttons: `button`, `cycle` and `square`. With attribute `data-button-outline` you can set buttons to `outline` style (default: true). See more about Buttons.



```
<div data-role="wizard" data-button-mode="square"></div>

<div data-role="wizard" data-button-mode="cycle"></div>

<div data-role="wizard" data-button-mode="button"
     data-icon-help="<span>Help</span>"
     data-icon-prev="<span>Prev</span>"
     data-icon-next="<span>Next</span>"
     data-icon-finish="<span>Finish</span>"></div>

<div data-role="wizard" data-button-mode="button" data-button-outline="false"></div>
```

Controls icons

You can change controls icons with attributes: `data-icon-help`, `data-icon-prev`, `data-icon-next` and `data-icon-finish`. Value for these attributes must be valid **html tag** or **path to image**.



```
<div data-role="wizard"
     data-icon-help="<span>Help</span>"
     data-icon-prev="<span class='mif-arrow-left'></span>"
     data-icon-next="<span class='mif-arrow-right'></span>"
     data-icon-finish="images/checkmark.png">
    <section><div class="page-content">Page 1</div></section>
    <section><div class="page-content">Page 2</div></section>
    <section><div class="page-content">Page 3</div></section>
</div>
```

Custom classes

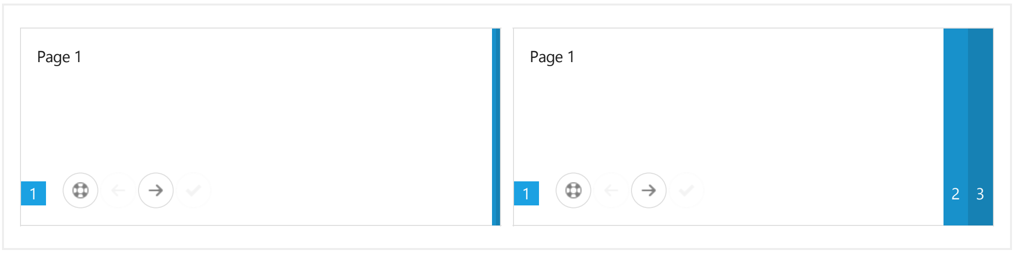
Also you can use special data attributes to define your own classes for wizard and controls: `data-cls-wizard`, `data-cls-actions`, `data-cls-help`, `data-cls-prev`, `data-cls-next`, `data-cls-finish`.

If you need to change colors for feature and complete section, you must define your own style for this with next patterns:

```
Less preprocessor code:

.wizard {
    & > section {
        &:nth-child(1) {border-color: @cyan;}
        &:nth-child(2) {border-color: darken(@cyan, 5%);}
        &:nth-child(3) {border-color: darken(@cyan, 10%);}
        &:nth-child(4) {border-color: darken(@cyan, 15%);}
        &:nth-child(5) {border-color: darken(@cyan, 20%);}
        &:nth-child(6) {border-color: darken(@cyan, 25%);}
        &:nth-child(7) {border-color: darken(@cyan, 30%);}
        &:nth-child(8) {border-color: darken(@cyan, 35%);}
        &:nth-child(9) {border-color: darken(@cyan, 40%);}
        &:nth-child(10) {border-color: darken(@cyan, 50%);}

        &.complete {
            &:nth-child(1) {border-color: @lightGray;}
            &:nth-child(2) {border-color: darken(@lightGray, 5%);}
            &:nth-child(3) {border-color: darken(@lightGray, 10%);}
            &:nth-child(4) {border-color: darken(@lightGray, 15%);}
            &:nth-child(5) {border-color: darken(@lightGray, 20%);}
            &:nth-child(6) {border-color: darken(@lightGray, 25%);}
            &:nth-child(7) {border-color: darken(@lightGray, 30%);}
            &:nth-child(8) {border-color: darken(@lightGray, 35%);}
            &:nth-child(9) {border-color: darken(@lightGray, 40%);}
            &:nth-child(10) {border-color: darken(@lightGray, 50%);}
        }
    }
}
```

Responsive

By default wizard displayed in mobile mode wit shrinks feature and complete sections. To expand wizard use special classes: `.wizard-wide-*`. Where asterisk is one of: `sm`, `md`, `lg`, `xl` or `xxl`. If you need expand wizard always use class `.wizard-wide-fs`. See more information about Media breakpoints.


Page 1
1
Page 1
1
2
3