

mdSidenav

Hans Larsen, Jan. 2016

Goals

Sidenav is a core component of Angular Material Design. This documents the implementation of the `md-sidenav` component for Material 2.

References

- [Angular 1.x Material Sidenav Demo](#)
- [Angular 1.x Material Sidenav API](#)
- [Angular 1.x Material Sidenav Implementation](#)

Background

The current implementation of `md-sidenav` is separated into 2 directives: `md-sidenav` proper and `md-content`. The sidenav uses the Flexbox model to properly layout and `$mdTheming` to support themes. It provides a `$mdSidenav` service to open and close sidenav (by using `md-component-id` as a reference).

Requirements

The revised component should try to improve on accessibility and usability. Notably, the `md-content` directive is overloaded in material right now and should not be used. An element might be used, but it should not include side effects.

It should be relatively contained, ie. not have any requirement such as special CSS or styling for the parent/content.

The current implementation follows implicit async calls that are not easy to follow when reading the code, such as setting attributes and hoping by nextTick the proper calls have been made. We should make all async calls explicit. This will improve debugging and code readability greatly.

Promises should be provided by every methods of the API. This is already the case, and it would be a retro.

Accessibility should be a first class feature, and theming has to be supported by launch.

Mobile support should be a first class feature as well; supporting touch events and dragging the drawer in/out of view.

Support for the current two “modes”; floating and aside the page’s content. The “aside” mode is usable by using the `locked-open` attribute; it is the same as floating but the content of the page itself is still accessible and clicking outside the sidenav will not close it. It also doesn’t apply a shade over the content.

Case Studies

Insert stories from users using the current `mdSidenav`, if any are relevant.

Comparing Implementations

Bootstrap `sidebar-wrapper`

see <http://startbootstrap.com/template-overviews/simple-sidebar/>.

Example

```
<div id="wrapper">
  <!-- Sidebar -->
  <div id="sidebar-wrapper">
    Sidebar Content
  </div>
  <!-- Content -->
  Page content.
</div>
```

Details

This implementation uses two block containers, a wrapper and the sidenav proper.

The wrapper implement `padding` to move the content over, while the sidenav has a negative `margin` to move it over the padding. It applies a `position: fixed` to the sidebar proper, with a `top: 0; height: 100%`.

Analysis

- Using block elements makes it easier to style their content compared to, e.g. flex.
- Because the position needs to be `absolute`, it can’t be included in the middle of the page without a new stacking context.
- Fully CSS based.

Angular 1.x Material `md-sidenav`

see <https://material.angularjs.org/latest/api/directive/mdSidenav>

Example

```
<div layout="row">
  <md-sidenav md-component-id="myId" class="md-sidenav-left">
    Left
  </md-sidenav>
  <md-content>
    Page content.
  </md>
</div>
```

Details

This implementation is the most complex of the 3 listed here. It requires a `layout="row"` attribute to specify the flex layout, the `md-sidenav` directive, and an `md-content` which is also used as another directive. It uses `translate3d()` transforms to move the sidebar in/out.

Analysis

- Using translation transforms allow to use the GPU for better performance.
- Reliance on Flex means you need to also style the parent to create a flex container.
- Backdrop is created and inserted in the DOM.

Polymer <paper-drawer-panel>

see <https://elements.polymer-project.org/elements/paper-drawer-panel>.

Example

```
<paper-drawer-panel>
  <div drawer>
    Drawer panel
  </div>
  <div main>
    Main panel
  </div>
</paper-drawer-panel>
```

Details

In floating mode, Polymer uses `position: absolute` with a `left: 0` style. It then applies `transform: translateX(-100%)` similar to Angular 1.x Material.

Analysis

- The drawer is a self contained component.

Technical Considerations

We should use transform instead of left for GPU performance.

We should try to steer away from flex.

Suggested Implementation

Example

```
<md-sidenav-layout>
  <md-sidenav>
    Drawer panel
  </md-sidenav>
  <md-sidenav align="end">
    Right-side panel along the left-side sidenav.
  </md-sidenav>

  Main content panel
</md-sidenav-layout>
```

Details

The use of a container around both the sidenav and the content makes it easier to style, and as easy for developers to indicate structural intent. It also allows us to listen on touch and drag events.

The lack of a main content div makes it easier for developers to know they can put a container there to style how they want.

Implementation Details

The sidenav component is actually two components; the `md-sidenav-container` and 1 or 2 `md-sidenav`.

Proof of Concept

A PR has been submitted for review.

API

<md-sidenav-container>

Constraints

It is invalid to create an `<md-sidenav-container>` without any `md-sidenav`. An exception will be thrown during initialization of the component.

It is invalid to create an `<md-sidenav-container>` with more than one `md-sidenav` for the same alignment. An exception will be thrown during initialization of the component.

<md-sidenav>

Component

Inputs		
Name	Type	Description
align	"start" "end"	Optional, default = "start". The side the sidenav will appear.
opened	Boolean	Whether the sidenav is opened or closed.
mode	"over" "side"	Optional, default is "over". If "over", a backdrop will be visible and the content will not be pushed. If "side" the content will be pushed and still be accessible while the sidenav is open.
Outputs		
Name	Description	
open	Called when the <code>sidenav</code> is opened, after the animation finish.	
close	Called when the <code>sidenav</code> is closed, after the animation finish.	

Instance

Methods		
Signature		Description
<code>open(): Promise<void></code>		Open the sidenav, returning a <code>Promise</code> that will be resolved once the animation finish.
<code>close(): Promise<void></code>		Close the sidenav, returning a <code>Promise</code> that will be resolved once the animation finish.
<code>toggle(open?: Boolean): Promise<void></code>		Toggle the sidenav, returning a <code>Promise</code> that will be resolved once the animation finish. If <code>opened</code> is passed, this is equivalent to calling <code>open()</code> or <code>close()</code> , otherwise the default is to switch between the two states.
Properties		
Name	Type	Description
<code>container</code>	<code>MdSidenavContainer</code>	The <code><md-sidenav-container></code> surrounding this sidenav.

Accessibility

All accessibility attributes should transfer to the actual DOM, such as `role`, `aria-hidden` and `aria-labelledby` / `aria-describedby`.

Setting the focus when the sidenav open should be done by using the open `output`, like so:

```
<md-sidenav-layout>
  <md-sidenav (open)="i.focus()">
    Drawer panel
    <input #i value="Will be focused when open">
  </md-sidenav>

  Main content panel
</md-sidenav-layout>
```

Internationalization

The align attribute will swap depending on the direction of the component.

Mobile

The user should be able to half close the sidenav by holding his finger at the sidenav border and moving left halfway.

Releasing a finger should have a momentum feature where the sidenav closes if enough "force" is applied.

Performance Considerations

Performance impact of the sidenav is minimal, as the amount of sidenav in a single page would be very limited.

Graphical Performance Considerations

Every animation in the sidenav is the result of a CSS transition. The `translateX()` transform is used to make sure the browser use hardware acceleration where possible.

There's no other animation.

Migration from 1.x

Place all `<md-sidenav>` and `<md-content>` tags inside an `<md-sidenav-container>`. If a sidenav has a `md-sidenav-right` class to it, simply add an `align="end"` attribute to that one and remove all `md-*` classes. Although the `<md-content>` is not necessary anymore, it is okay to keep it for styling.

Addendum

Feature Request

A `md-side-navigation` directive simulating the navigation menu in the current angular.io sidebar. It is included here as a reminder and a soft TODO.