ioBroker.vis im Material Design Style

2017 by Uhula

Inhaltsverzeichnis

Anleitung zur Material Design CSS

Grundsätzlich wird die Visualisierung mit den bekannten basic und jqui Controls entworfen. Diese erhalten lediglich CSS-Klassenzuweisungen um im Browser im Material Design Style gerendert zu werden. Es gibt nicht für alle Controls entsprechende CSS-Klassen.

Die CSS-Anweisungen müssen dem Projekt auf der CSS-Registerseite unter „Projekt“ zugewiesen werden. Sie stehen dann sowohl zur Anzeige im Editor als auch zur Laufzeit zur Verfügung. Sie sind so gestaltet, dass sie sich im Editor nur auf das Editfenster auswirken und nicht auf den Rest des Editors.

Notwendige Einstellungen der basic- und jqui-Controls sind mit "MUSS" gekennzeichnet, optionale mit "KANN".

MUSS [Generell - CSS Klasse] **muss-Wert**

KANN [Generell - CSS Klasse] **kann-Wert**

Farben

Als Vorgabe ist der Style insgesamt in schwarz-weiß gehalten, es besteht aber die Möglichkeit über CSS Angaben die Darstellung der Text, Buttons usw. auch farbig zu erhalten. Hierzu stehen spezielle **mdui-color-(color), mdui-background-(color)** und **mdui-accent-(color)** CSS Klassen zur Verfügung. Verwendet werden die durch das Material Design vorgegebenen Farben.

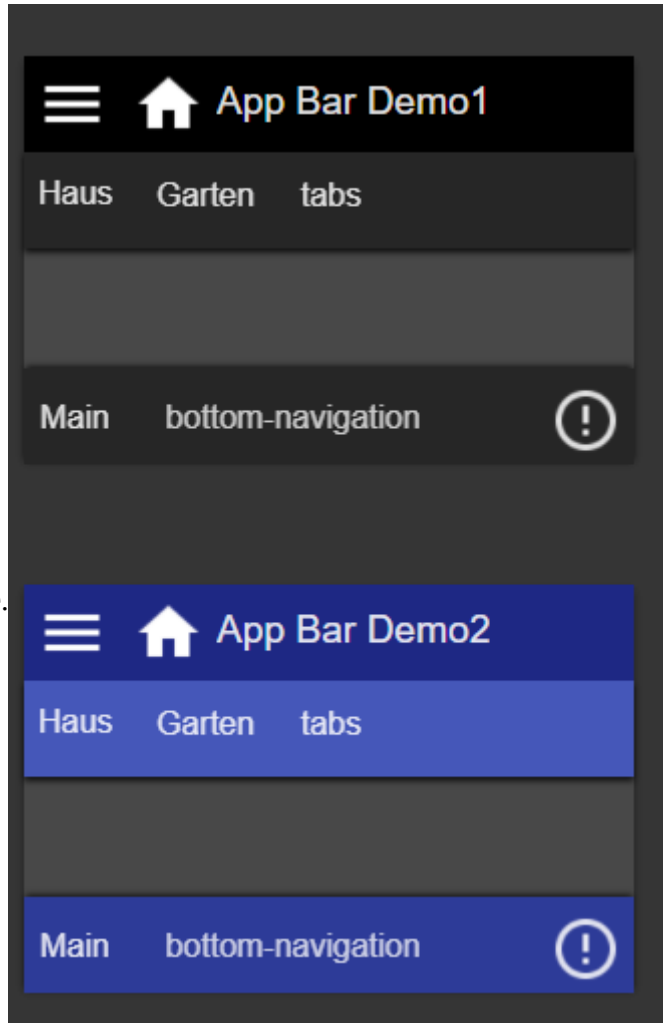
Die „color“ Angaben beziehen sich dabei auf die Schriftfarbe, die "background" Angaben auf den Hintergrund und die „accent“ Angaben auf Hervorhebungsfarben der einzelnen Controls.

Als Farben (color) stehen zur Verfügung:

teal	**teal**	**indigo**	**indigo**	**amber**	**amber**
lime	**lime**	**red**	**red**	green	green
yellow	yellow	brown	brown		

> **TIPP**
> Farben ziehen die Aufmerksamkeit des Betrachters auf sich. Damit sie diese Aufgabe gut erledigen können, dürfen sie nicht in einer Vielzahl von anderen Farben untergehen. Das Grundlayout sollte also bewusst einfarbig gestaltet sein, damit die Signalfarben dann um so deutlicher auffallen.

Layout



Das normale Layout eines Screens besteht aus vier horizontalen Bereichen:

- application-bar,
- tab-navigation,
- content
- bottom-navigation

In der **application-bar** werden buttons und texte untergebracht, die permanent sichtbar sein sollen. Wie z.B. Menü- und Home-Buttons, Uhrzeit und Tagesanzeige. In der Regel hat ein Projekt genau eine application-bar, es sind aber auch mehrere möglich. Views, die als applicatiobn-bar genutzt werden, sollten mit "bar" beginnen, z.B. "barMain"

In der **tab-navigation** werden die buttons für die Navigation durch die einzelnen Views angezeigt. Je nach Navigationstiefe wird die tab-navigation angepasst. So besitzt z.B. der Haupt-Screen eine andere tab-navigation als z.B. der Haus-Screen, in welcher die Räume aufgeführt werden. Views, die als tab-navigation genutzt werden, sollten mit "tab" beginnen, z.B. "tabMain", "tabHaus".

Die **bottom-navigation** kann nun entweder wie die application-bar oder wie die tab-navigation genutzt werden. Views, die als bottom-navigation genutzt werden, sollten mit "bot" beginnen, z.B. "botMain".

Natürlich sind alle drei Komponenten optional und können je nach Screen auch weggelassen werden.

Content

Als Content wird der Screen-View selbst verwendet.

MUSS [Generell - CSS Klasse]
mdui-view

KANN [Generell - CSS Klasse]
mdui-background-(color)
Hintergrundfarbe (siehe dort), Vorgabe ist "#303030".

Bar, Tabs und Navigation

Da der Inhalt dieser drei Bereich auf mehreren Screens verwendet werden wird, werden diese nicht direkt auf den jeweiligen Screens angelegt, sondern es sind jeweils eigene Views, die dann über den **basic-view in widget Container** auf den Screens eingefügt werden. Sinnvollerweise benennt man diese Views mit "barXXX", "tabXXX" und "botXXX". Den Containern (nicht den Views!) in den Screens wird dann die zugehörige CSS Klasse zugewiesen, sie sollten alle eine Höhe von 48px haben.

MUSS [Generell - CSS Klasse]
mdui-application-bar

oder **mdui-tab-navigation**

oder **mdui-bottom-navigation**

KANN [Generell - CSS Klasse]
mdui-background-(color)
Hintergrundfarbe (siehe dort), Vorgaben sind #000000 und #212121.

KANN [Generell - CSS Klasse]
mdui-accent-(color)
Akzentfarbe um den aktuellen Navigations-Button zu kennzeichnen. Vorgabe ist #ffffff.

Navigation Buttons

Da die drei Bereiche hauptsächlich er Navigation dienen, werden in ihnen meistens Buttons eingefügt. Für Navigations-Buttons in den Bars, Tabs und Bottom-Navigations können verschiedene **jqui-Button Controls** verwendet werden.

MUSS [Generell - CSS Klasse]
mdui-navigation-button

KANN [Generell - CSS Klasse]
mdui-color-(color)
Schriftfarbe. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "weiß".

KANN [Generell - CSS Klasse]

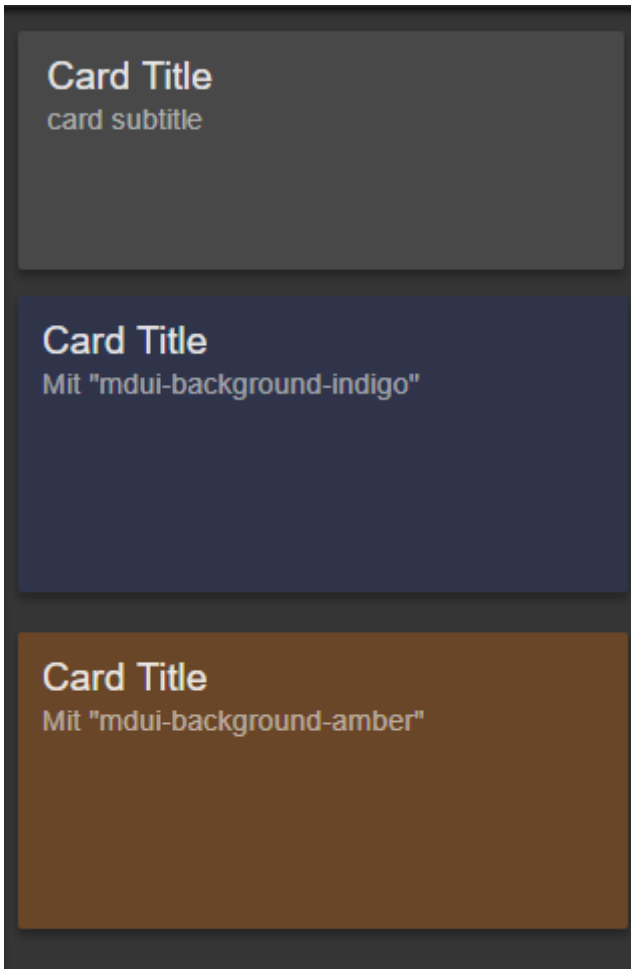
mdui-background-(color)

Hintergrundfarbe. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "transparent".

> **TIPP**
> Um Controls am rechten Rand so zu verankern, dass sie mit der Screenbreite skalieren, kann man bei der "left" Angabe "calc(100% - nnnpx)" angeben, mit nnn = Breite des Controls.

Cards

Cards können, müssen aber nicht verwendet werden. Mit Cards lassen sich Controls optisch gruppieren. Für eine Card wird das Control **basis-html** verwendet.



MUSS [Generell - CSS Klasse]
mdui-card

KANN [Generell - CSS Klasse]
mdui-background-(color)

Hintergrund der Card. (color) kann eine der Farben (siehe dort) sein. Vorgabe ist #404040

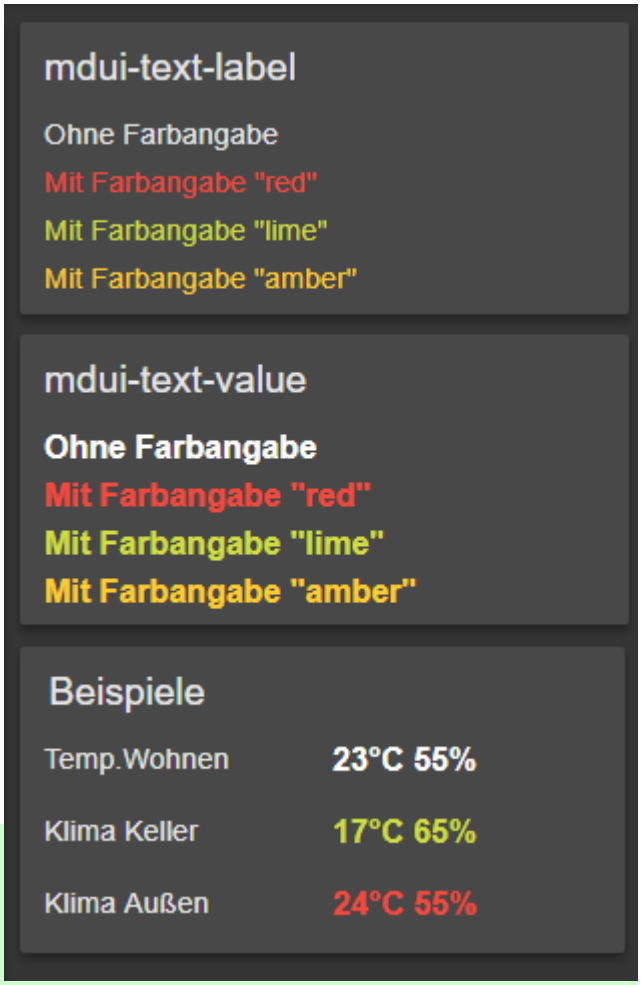
Cards können auch Titel und Untertitel haben. Auch hierfür wird das Control **basis-html** verwendet.

MUSS [Generell - CSS Klasse]
mdui-card-title
oder **mdui-card-subtitle**

KANN [Generell - CSS Klasse]
mdui-color-(color)
Hiermit kann festgelegt werden, welche Farbe der Text haben soll. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "weiß".

Labels

Für ein Label (Text) wird das Control **basis-html** verwendet. Es kann sowohl für die Darstellung von Labels als auch von Werten verwendet werden. Labels werden etwas dunkler und kleiner als Werte dargestellt. Natürlich lassen sich auch die unter Cards beschriebenen card-title bzw. card-subtitle für normale Texte verwenden.

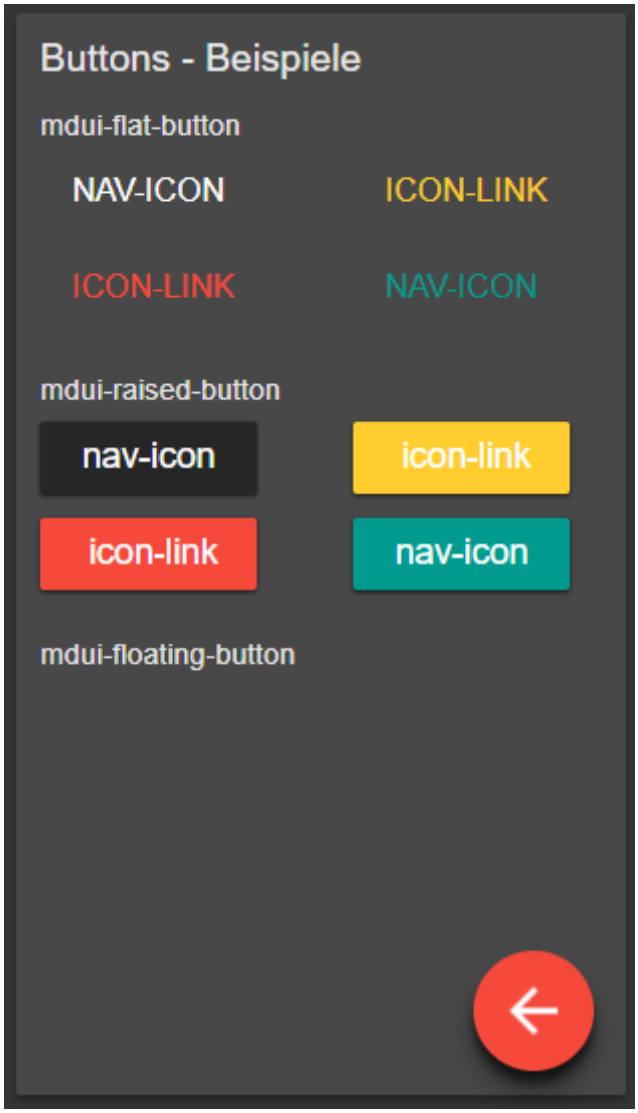


MUSS [Generell - CSS Klasse]
mdui-text-label
oder **mdui-text-value**

KANN [Generell - CSS Klasse]
mdui-color-(color)
Schriftfarbe. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "weiß".

> **TIPP**
> Da bei jedem Attribut auch auf ioBroker Variable zugriffen werden kann, ist auch eine Farbsteuerung über Variable möglich. Hierzu eine Zeichenvariable deklarieren und ihr den gewünschten Farbwert "red", "green", ... zuweisen. In der CSS Klasse dann **mdui-color-{meine Variable}** verwenden.
>
> Bsp: **mdui-color-{javascript.0.farbwert-temperatur}**

Buttons



Für einen Button können verschiedene **jqui-Button Controls** verwendet werden. Bei den meisten funktioniert die Darstellung korrekt, wenn eine der Button CSS Klassen zugewiesen wird. Im Material Design gibt es drei Button Arten: flat-button, raised-button und floating-button.

MUSS [Generell - CSS Klasse]
mdui-flat-button
oder **mdui-raised-button**
oder **mdui-floating-button**

KANN [Generell - CSS Klasse]
mdui-color-(color)
Hiermit kann festgelegt werden, welche Farbe für den Text verwendet werden soll. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "weiß". Primär bei flat-button

KANN [Generell - CSS Klasse]
mdui-background-(color)
Hiermit kann festgelegt werden, welche Farbe für den Hintergrund verwendet werden soll. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "schwarz". Primär bei raised- und floating-button

Radio Buttons

Für Radio-Eingaben wird eines der **jqui-Radio... Controls** verwendet.

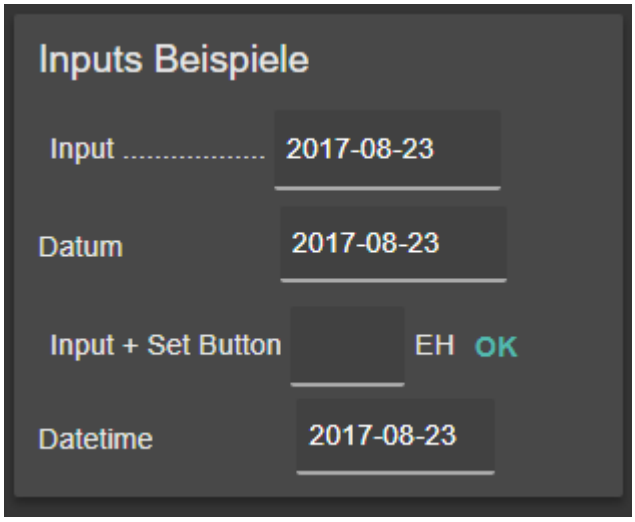
MUSS [Generell - CSS Klasse]
mdui-radio

KANN [Generell - CSS Klasse]
mdui-background-(color)
Hiermit kann die Hintergrundfarbe festgelegt werden. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "transparent".

KANN [Generell - CSS Klasse]
mdui-color-(color)
Hiermit kann die Schriftfarbe festgelegt werden. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "weiß".

KANN [Generell - CSS Klasse]
mdui-accent-(color)
Akzentfarbe um die aktuelle Auswahl zu kennzeichnen. Vorgabe ist #ffffff.

Inputs



Für eine Input können verschiedene **jqui-Input Controls** verwendet werden. Bei den meisten funktioniert die Darstellung korrekt. Material Design typisch werden Inputs nur durch eine untere Rahmenlienie angezeigt, die beim Fokus farbig wird.

MUSS [Generell - CSS Klasse]
mdui-input

KANN [Generell - CSS Klasse]
mdui-color-(color)
Hiermit kann festgelegt werden, welche Farbe ein Label vor der Eingabe hat (wenn vorhanden). (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "weiß".

KANN [Generell - CSS Klasse]
mdui-background-(color)
Hiermit kann festgelegt werden, welche Farbe für den Hintergrund verwendet werden soll. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "transparent".

Switches

Für einen Switch wird das Control **basis-bool checkbox** verwendet.

MUSS [Generell - CSS Klasse]
mdui-switch

MUSS [Allgemein - HTML anhängen]
<label for="(id)_checkbox"></label>
Wichtig! Wobei (id) durch die ID des Controls selbst ersetzt werden muss, Bsp: "w00033_checkbox"

KANN [Generell - CSS Klasse]
mdui-accent-(color)
Hiermit kann festgelegt werden, welche Farbe im EIN Zustand verwendet werden soll. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "weiß".

Demo-Projekt

Mit auf GitHub befindet sich auch ein Demo-Projekt, welches eine ioBroker.vis Visualisierung ähnlich einer Android-App mit Application-bar, Tab-Navigation und Bottom-Navigation bietet. Die Controls dieses Demo-Projektes sind nicht mit echten Instanzen verbunden. Die verwendeten Symbole (Icons) befinden sich alle im Projektunterordner „images".

Das Demo-Projekt, also die ZIP-Datei, kann in ioBroker.vis unter Projekt importieren eingelesen werden.

Das Demo-Projekt enthält die folgenden Views:

appMain
Application-Bar, wird auf allen screen-Views über einen **View in Container** in diese eingebunden. Enthält Buttons um das Main-Menü und das Funktions-Menü zu öffnen. Weiterhin Uhrzeit- und Datumsanzeige.

botDemo
Bottom-Navigation, die auf den beiden Demo-Screens über einen **View in Container** in diese eingebunden wird.

botMain
Bottom-Navigation, die auf allen Sceens (außer Demo) über einen **View in Container** in diese eingebunden wird.

...