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 View-Editfenster auswirken und nicht auf den Rest des Editors.

In der folgenden Beschreibung sind notwendige Einstellungen der basic- und jqui-Controls sind mit "MUSS" gekennzeichnet, optionale mit "KANN".

Muss?	Eigenschaft	Werte	Beschreibung
Ja	CSS Klasse	...	Der Wert MUSS gesetzt werden
	CSS Klasse	...	Optional

Bei allen Views kann eine Designhilfe eingeschaltet werden um die Umrisse der Widgets besser erkennen zu können:

Muss?	Eigenschaft	Werte	Beschreibung
	CSS Klasse	mdui-design	Sorgt im View-Editfenster für eine andere Darstellung der Widgets um z.B. deren Umrisse dauerhaft sehen zu können.

Hinweis
Nach dem Setzen einer CSS-Klasse für ein View muss der View im Editor aktualisiert werden, entweder durch kurzes Umschalten zu einem anderen View oder durch [F5] Browser-Update. ioBroker.vis wendet die neuen CSS-Klassen nicht sofort an.

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 umso 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 application-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 kann die tab-navigation angepasst werden, in dem in den Screens einfach andere tab-navigations eingebunden werden. 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".

Alle drei genannten Komponenten werden als eigene Views angelegt und dann in den Screens nur referenziert (**basic-view in widget Container**).

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

Content

Als Content wird der Screen-View selbst verwendet. Dass es sich um einen Material Design Style Screen handelt wird durch die Zuweisung der CSS-Klasse mdui-view festgelegt.

Muss?	Eigenschaft	Werte	Beschreibung
Ja	CSS Klasse	mdui-view	
	CSS Klasse	mdui-background-(color)	Hintergrundfarbe. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "#303030".

Hinweis
Nach dem Setzen einer CSS-Klasse für ein View muss der View im Editor aktualisiert werden, entweder durch kurzes Umschalten zu einem anderen View oder durch [F5] Browser-Update. ioBroker.vis wendet die neuen CSS-Klassen nicht sofort an.

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 das Widget **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?	Eigenschaft	Werte	Beschreibung
Ja	CSS Klasse	mdui-application-bar oder mdui-tab-navigation oder mdui-bottom-navigation	
	CSS Klasse	mdui-background-(color)	Hintergrundfarbe (siehe dort), Vorgaben sind #000000 und #212121.
	CSS Klasse	mdui-accent-(color)	Akzentfarbe um den aktuellen Navigations-Button zu kennzeichnen. Vorgabe ist #ffffff.

Navigation Buttons

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

Muss?	Eigenschaft	Werte	Beschreibung
Ja	CSS Klasse	mdui-navigation-button	
	CSS Klasse	mdui-color-(color)	Schriftfarbe. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "weiß".
	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 Screen-Breite 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 Widgets optisch gruppieren. Für eine Card wird das Widget **basis-html** verwendet.

Sollen Cards Titel und Untertitel haben, so können hierfür die unter „Labels“ beschriebenen CSS-Styles verwendet werden.

Muss?	Eigenschaft	Werte	Beschreibung
Ja	CSS Klasse	mdui-card	
	CSS Klasse	mdui-background-(color)	Hintergrundfarbe. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "#404040".

Labels

Für ein Label (Text) wird das **basis-html** Widget verwendet. Es kann sowohl für die Darstellung von Titeln, Untertiteln, Labels als auch von Werten verwendet werden. Labels werden etwas dunkler und kleiner als Werte dargestellt.

Muss?	Eigenschaft	Werte	Beschreibung
Ja	CSS Klasse	mdui-label oder mdui-value oder mdui-title oder mdui-subtitle	Label-Text Wert Titel, Überschrift Untertitel
	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** Widgets 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?	Eigenschaft	Werte	Beschreibung
Ja	CSS Klasse	mdui-flat-button oder mdui-raised-button oder mdui-floating-button	
	CSS Klasse	mdui-color-(color)	Schriftfarbe. (color) kann eine der Farben (siehe dort) sein, Vorgabe für flat ist „blau“, für die anderen "weiß".
	CSS Klasse	mdui-background-(color)	Hintergrundfarbe. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "blau".

Radio Buttons

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

Muss?	Eigenschaft	Werte	Beschreibung
Ja	CSS Klasse	mdui-radio	
	CSS Klasse	mdui-color-(color)	Schriftfarbe. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "weiß".
	CSS Klasse	mdui-background-(color)	Hintergrundfarbe. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "transparent".
	CSS Klasse	mdui-accent-(color)	Akzentfarbe um die aktuelle Auswahl zu kennzeichnen. Vorgabe ist „blau“.

Inputs

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

Muss?	Eigenschaft	Werte	Beschreibung
Ja	CSS Klasse	mdui-input	
	CSS Klasse	mdui-color-(color)	Schriftfarbe. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "weiß".
	CSS Klasse	mdui-background-(color)	Hintergrundfarbe. (color) kann eine der Farben (siehe dort) sein, Vorgabe ist "transparent".

Switches

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

Muss?	Eigenschaft	Werte	Beschreibung
Ja	CSS Klasse	mdui-switch	
Ja	HTML anhängen	<label for="(id)_checkbox"> </label>	**Wichtig!** Wobei (id) durch die ID des Controls selbst ersetzt werden muss, Bsp: "w00033_checkbox"
	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.

...