

# draggable\_customized\_btn\_navy\_bar

---

pub v1.0.1+2

Awesome Flutter

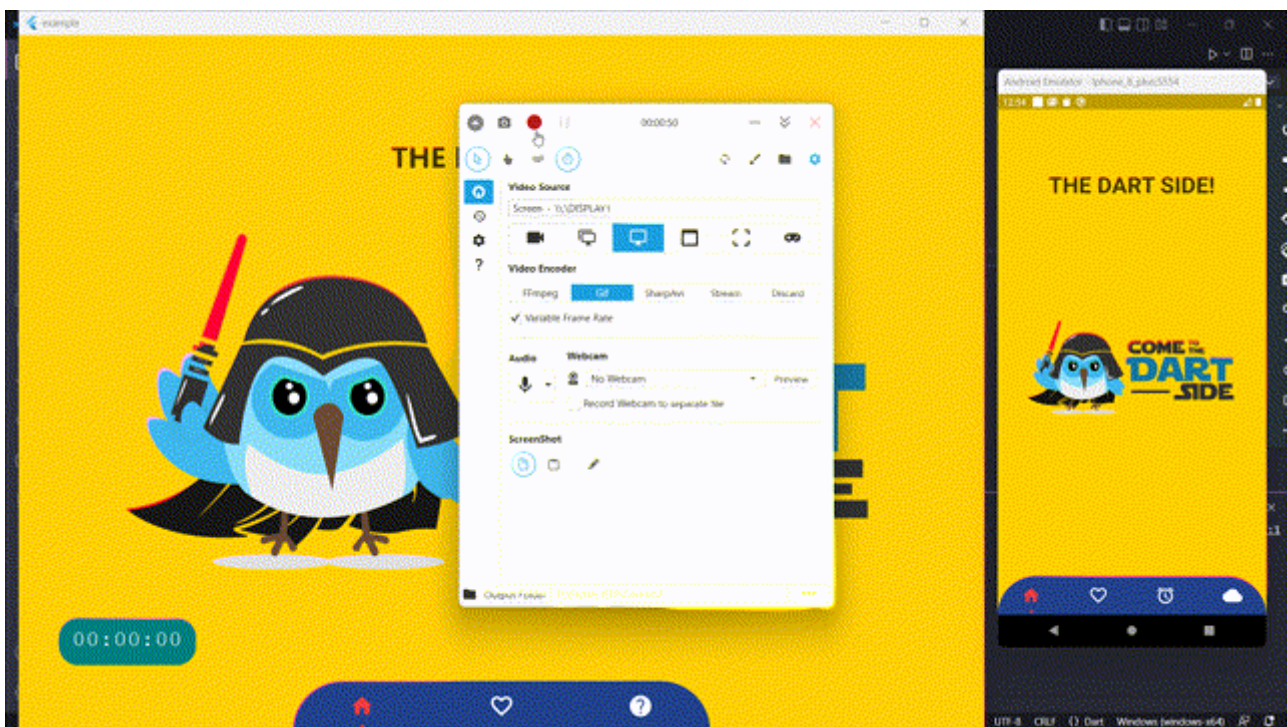
A bottom navigation bar that you can customize with the options you need, without any limits. You can also customize the appearance of the navigation bar.

The package is solving a problem with package `bottom_personalized_dot_bar` as it solves its issues and issues can be summarized in the following points:

- it's old and the owner didn't update it since 3 years.
- Some exception was caught by gestures.
- no clear definition for the max number and min number of displayed items.
- forget changes that take place in the displayed and hidden items in each run, so the user setup will be forgotten once the app is closed.
- not cross platform.

The Btn Nav bar is very beautiful and many people like it so much. For these reasons I built this package for public to help them to add this beautiful btn nav bar to their projects. I wish this will help you.

## Live Preview



## Supported platforms

- Flutter Android
- Flutter iOS
- Flutter web

- Flutter desktop

## Package Structure

---

```
|-- lib
|   |-- draggable_customized_btn_navy_bar.dart
|   '-- src
|       |-- button_scale.dart
|       |-- colors.dart
|       |-- drag_item_update.dart
|       |-- draggable_customized_dot_bar_item.dart
|       |-- dragged_hidden_menu_option.dart
|       |-- dragged_menu_option.dart
|       |-- enums.dart
|       |-- hidden_menu_option.dart
|       |-- menu_option.dart
|       '-- parent.dart
```

## Null Safe!!

Hint: The package is supporting null safety



## Package overview

- ☒ Drag and Drop your options!
- ☒ Change icon , background , text , item , item background ..... colors

- ☒ Control the max number and minimum number of displayed items
- ☒ Listen to events 'Sort, Insert and Delete'
- ☒ Dynamically change the selected option
- ☒ Save the modified setup (changes in items and its indices) in shared preferences
- ☒ Custom options
- ☒ Unlimited options
- ☒ supports all the platforms [android- iOS - web - linux - mac - windows]
- ☒ And more ...

## Getting Started

First of all, Depend on the package.

```
dependencies:
  ...
  draggable_customized_btn_navy_bar: ^0.0.1 #Add the latest version
```

Then import the package in your dart file

```
import
'package:draggable_customized_btn_navy_bar/draggable_customized_btn_navy_bar.dart';
```

## Basic Setup

```
String _itemSelected = 'item-1';
DraggableCustomizedBtnNavyBar navyBarController =
DraggableCustomizedBtnNavyBar();
...
..
.
Scaffold(
  body: Stack(
    children: <Widget>[
      .... // Your App Home
      DraggableCustomizedBtnNavyBar(
        key : Key("myNav"), // configuration Scopes
        controller : navyBarController,
        keyItemSelected: _itemSelected,
        hiddenItems: <DraggableCustomizedDotBarItem>[
          DraggableCustomizedDotBarItem('item-4', icon: Icons.cloud,
name: 'Nube', onTap: (itemSelected) { /* event selected */ }),
          DraggableCustomizedDotBarItem('item-5', icon:
Icons.access_alarm, name: 'Alarma', onTap: (itemSelected) { /* event
selected */ }),
          DraggableCustomizedDotBarItem('item-6', icon:
```

```

Icons.message, name: 'Mensaje', onTap: (itemSelected) { /* event
selected */ }),
      DraggableCustomizedDotBarItem('item-7', icon:
Icons.notifications, name: 'Alerta', onTap: (itemSelected) { /* event
selected */ }),
      DraggableCustomizedDotBarItem('item-8', icon:
Icons.security, name: 'Seguridad', onTap: (itemSelected) { /* event
selected */ }),
      DraggableCustomizedDotBarItem('item-9', icon: Icons.help,
name: 'Ayuda', onTap: (itemSelected) { /* event selected */ }),
      DraggableCustomizedDotBarItem('item-10', icon:
Icons.settings, name: 'Config.', onTap: (itemSelected) { /* event
selected */ }),
    ],
    items: <DraggableCustomizedDotBarItem>[
      DraggableCustomizedDotBarItem('item-1', icon:
Icons.sentiment_very_satisfied, name: 'Flutter', onTap: (itemSelected) {
/* event selected */ }),
      DraggableCustomizedDotBarItem('item-2', icon:
Icons.favorite_border, name: 'Favorito', onTap: (itemSelected) { /*
event selected */ }),
      DraggableCustomizedDotBarItem('item-3', icon: Icons.face,
name: 'Perfil', onTap: (itemSelected) { /* event selected */ }),
    ],
  ),
],
),
);

```

## Customization

### *DraggableCustomizedBtnNavyBar*

Attribute	Description
<b>key</b>	Scope configuration name
<b>controller</b>	Controle Badges and check if settings is open
<b>items</b>	List of items to be displayed in the navigation bar
<b>hiddenItems</b>	List of items that will be hidden
<b>maximumNumberOfDisplayItems</b>	max number of displayed items - default:5
<b>onDisplayedStackOverflows</b>	function to be done if the user want to add item to the displayed items and he reaches to the max displayed number of items - default : showSnackBar. <b>onDisplayedStackOverflows: () { /* Your action */ }</b>
<b>minimumNumberOfDisplayedItems</b>	min number of displayed items - default:1

Attribute	Description
<code>onDisplayedStackIsEmpty</code>	function to be done if the user want to remove item from the displayed items and he reaches to the min displayed number of items - default : <code>showSnackBar</code> . <code>onDisplayedStackIsEmpty: () { /* Your action */ }</code>
<code>keyItemSelected</code>	Item key that is selected
<code>width</code>	Navigation bar width
<code>height</code>	Navigation bar height
<code>borderRadius</code>	Navigation bar radius
<code>selectedColorIcon</code>	Selected Icon color
<code>unSelectedColorIcon</code>	Unselected Icon color
<code>navigatorBackground</code>	Navigator Container Background color
<code>settingBackground</code>	Setting Container Background color (Hidden items)
<code>iconSetting</code>	Settings button icon
<code>iconSettingColor</code>	Settings button icon color
<code>badgeColor</code>	Badge Indicator color
<code>badgeTextColor</code>	Badge Text color
<code>settingTitleText</code>	Setting Title Text
<code>settingTitleColor</code>	Setting Title color
<code>settingSubTitleText</code>	Setting Sub-Title Text
<code>settingSubTitleColor</code>	Setting Sub-Title color
<code>doneText</code>	Done button Text
<code>textDoneColor</code>	Text Done Color
<code>buttonDoneColor</code>	Button done color
<code>hiddenItemBackground</code>	Background of hidden item
<code>iconHiddenColor</code>	Icon Hidden Color
<code>textHiddenColor</code>	Text Hidden Color
<code>dotColor</code>	Selection Indicator Color (Dot
<code>boxShadow</code>	Shadow of container

Attribute	Description
<code>onOrderHideItems</code>	Event when you sort the hidden options, this has as parameter the list of hidden options with the new order. <code>onOrderHideItems:</code> <code>(List&lt;BottomPersonalizedDotBarItem&gt; hiddenItems) { /* Your action */ }</code>
<code>onOrderItems</code>	Event when ordering browser options, this has as parameter the list of options with the new order <code>onOrderItems:</code> <code>(List&lt;BottomPersonalizedDotBarItem&gt; items) { /* Your action */ }</code>
<code>onAddItem</code>	Event when you add a new option to the navigation bar, this has as parameters the item you add and the list of options. <code>onAddItem:</code> <code>(BottomPersonalizedDotBarItem itemAdd, List&lt;BottomPersonalizedDotBarItem&gt; items) { /* Your action */ }</code>
<code>onRemoveItem</code>	Event when you delete an option from the navigation bar, this has as parameters the element to delete and the list of hidden options. <code>onRemoveItem:</code> <code>(BottomPersonalizedDotBarItem itemRemove, List&lt;BottomPersonalizedDotBarItem&gt; hiddenItems) { /* Your action */ }</code>

### *DraggableCustomizedBtnNavyBar*

Attribute	Description
<code>settingsIsOpen</code>	Check if settings is Open <code>bool settingsIsOpen</code>
<code>hasBadge</code>	Check if badge exist. <code>bool hasBadge(String keyItem)</code>
<code>getBadge</code>	Get badge value return <code>null</code> if badge not found. <code>String? getBadge(String keyItem)</code>
<code>setBadge</code>	Set badge value. <code>void setBadge(String keyItem, dynamic badge)</code>

### *DraggableCustomizedDotBarItem*

Attribute	Description
<code>keyItem</code>	Unique key
<code>icon</code>	Item icon
<code>name</code>	Item name

Attribute	Description
<code>badge</code>	Initial Item badge value
<code>onTap</code>	Event with you press the item. <code>onTap: (String keyItem) { /* Your action */ }</code>

## License & Accesability

- This is the first version of my package so if you see any problems you're free to open an issue.



## Contributing

Pull requests are welcome. For major changes, please open an issue first to discuss what you would like to change.