**flutter\_rating\_bar**

Le damos un mínimo, máximo y un valor inicial.

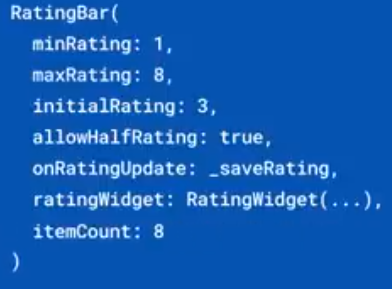
Con allowHalfRating se pueden hbilitar medias calificaciones.

onRatingUpdat es la función que se ejecuta al actualizar.

Se puede especificar cualquier widget como indicador (no solo la estrella)



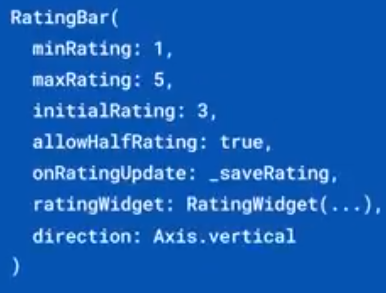
El default es 5 indicadores. Si se necesita otro valor usar itemCount:



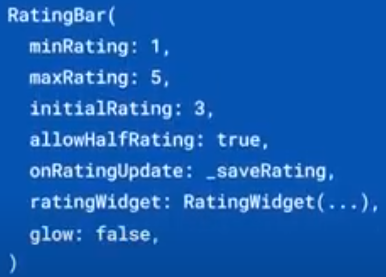
Se puede ajustar el relleno y tamaño del indicador con itemSize e itemPadding:



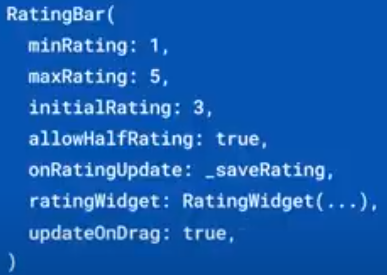
Se puede cambiar la orientación a vertical con direction:



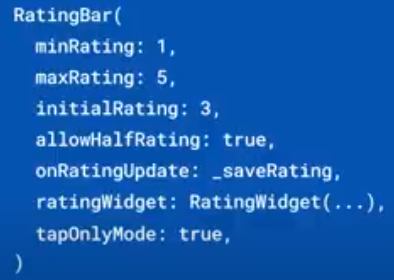
Viene un efecto de brillo (glowColor y glowRadius) que también se puede desactivar (glow:false):

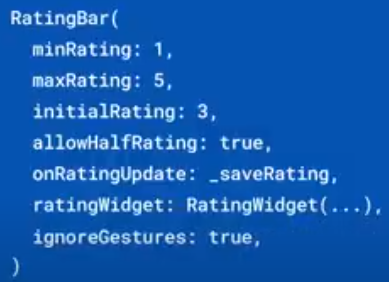
Para que la actualización del valor se produzca al arrastrar usar updateOnDrag:true:



Se puede deshabilitar la entrada de arrastre con tapOnlyMode:true



Para desactivar la entrada de gestos por completo usar ignoreGestures:true



|  |  |
| --- | --- |
| **Código** | **Comentarios** |
| import 'package:flutter/material.dart';  import 'package:flutter\_rating\_bar/flutter\_rating\_bar.dart';  class FlutterRatingBarScreen extends StatefulWidget {    const FlutterRatingBarScreen({Key? key}) : super(key: key);    @override    State<FlutterRatingBarScreen> createState() => \_FlutterRatingBarScreenState();  }  class \_FlutterRatingBarScreenState extends State<FlutterRatingBarScreen> {    late final \_ratingController;    late double \_rating;    double \_userRating = 3.0;    int \_ratingBarMode = 1;    double \_initialRating = 2.0;    bool \_isRTLMode = false;    bool \_isVertical = false;    IconData? \_selectedIcon;    //-------------------------------------------------------------    @override    void initState() {      super.initState();      \_ratingController = TextEditingController(text: '3.0');      \_rating = \_initialRating;    }    //-------------------------------------------------------------    @override    Widget build(BuildContext context) {      return MaterialApp(        debugShowCheckedModeBanner: false,        theme: ThemeData(          primarySwatch: Colors.amber,          appBarTheme: AppBarTheme(            titleTextStyle: Theme.of(context)                .textTheme                .titleLarge                ?.copyWith(color: Colors.white),          ),        ),        home: Builder(          builder: (context) => Scaffold(            appBar: AppBar(              title: const Text('Flutter Rating Bar'),              actions: [                IconButton(                  icon: const Icon(Icons.settings),                  color: Colors.white,                  onPressed: () async {                    \_selectedIcon = await showDialog<IconData>(                      context: context,                      builder: (context) => IconAlert(),                    );                    \_ratingBarMode = 1;                    setState(() {});                  },                ),              ],            ),            body: Directionality(              textDirection: \_isRTLMode ? TextDirection.rtl : TextDirection.ltr,              child: SingleChildScrollView(                child: Column(                  crossAxisAlignment: CrossAxisAlignment.center,                  mainAxisSize: MainAxisSize.min,                  children: <Widget>[                    const SizedBox(                      height: 40.0,                    ),                    \_heading('Rating Bar'),                    \_ratingBar(\_ratingBarMode),                    const SizedBox(height: 20.0),                    Text(                      'Rating: $\_rating',                      style: const TextStyle(fontWeight: FontWeight.bold),                    ),                    const SizedBox(height: 40.0),                    \_heading('Rating Indicator'),                    RatingBarIndicator(                      rating: \_userRating,                      itemBuilder: (context, index) => Icon(                        \_selectedIcon ?? Icons.star,                        color: Colors.amber,                      ),                      itemCount: 5,                      itemSize: 50.0,                      unratedColor: Colors.amber.withAlpha(50),                      direction: \_isVertical ? Axis.vertical : Axis.horizontal,                    ),                    const SizedBox(height: 20.0),                    Padding(                      padding: const EdgeInsets.symmetric(horizontal: 16.0),                      child: TextFormField(                        controller: \_ratingController,                        keyboardType: TextInputType.number,                        decoration: InputDecoration(                          border: const OutlineInputBorder(),                          hintText: 'Enter rating',                          labelText: 'Enter rating',                          suffixIcon: MaterialButton(                            onPressed: () {                              \_userRating =                                  double.parse(\_ratingController.text ?? '0.0');                              setState(() {});                            },                            child: const Text('Rate'),                          ),                        ),                      ),                    ),                    const SizedBox(height: 40.0),                    \_heading('Scrollable Rating Indicator'),                    RatingBarIndicator(                      rating: 8.2,                      itemCount: 20,                      itemSize: 30.0,                      physics: const BouncingScrollPhysics(),                      itemBuilder: (context, \_) => const Icon(                        Icons.star,                        color: Colors.amber,                      ),                    ),                    const SizedBox(height: 20.0),                    const Text(                      'Rating Bar Modes',                      style: TextStyle(fontWeight: FontWeight.w300),                    ),                    Row(                      children: [                        \_radio(1),                        \_radio(2),                        \_radio(3),                      ],                    ),                    Row(                      mainAxisAlignment: MainAxisAlignment.center,                      children: <Widget>[                        const Text(                          'Switch to Vertical Bar',                          style: TextStyle(fontWeight: FontWeight.w300),                        ),                        Switch(                          value: \_isVertical,                          onChanged: (value) {                            setState(() {                              \_isVertical = value;                            });                          },                          activeColor: Colors.amber,                        ),                      ],                    ),                    Row(                      mainAxisAlignment: MainAxisAlignment.center,                      children: <Widget>[                        const Text(                          'Switch to RTL Mode',                          style: TextStyle(fontWeight: FontWeight.w300),                        ),                        Switch(                          value: \_isRTLMode,                          onChanged: (value) {                            setState(() {                              \_isRTLMode = value;                            });                          },                          activeColor: Colors.amber,                        ),                      ],                    ),                  ],                ),              ),            ),          ),        ),      );    }    //------------------------------------------------------------    Widget \_radio(int value) {      return Expanded(        child: RadioListTile<int>(          value: value,          groupValue: \_ratingBarMode,          dense: true,          title: Text(            'Mode $value',            style: const TextStyle(              fontWeight: FontWeight.w300,              fontSize: 12.0,            ),          ),          onChanged: (value) {            setState(() {              \_ratingBarMode = value!;            });          },        ),      );    }    //--------------------------------------------------------------    Widget \_ratingBar(int mode) {      switch (mode) {        case 1:          return RatingBar.builder(            initialRating: \_initialRating,            minRating: 1,            direction: \_isVertical ? Axis.vertical : Axis.horizontal,            allowHalfRating: true,            unratedColor: Colors.amber.withAlpha(50),            itemCount: 5,            itemSize: 50.0,            itemPadding: const EdgeInsets.symmetric(horizontal: 4.0),            itemBuilder: (context, \_) => Icon(              \_selectedIcon ?? Icons.star,              color: Colors.amber,            ),            onRatingUpdate: (rating) {              setState(() {                \_rating = rating;              });            },            updateOnDrag: true,          );        case 2:          return RatingBar(            initialRating: \_initialRating,            direction: \_isVertical ? Axis.vertical : Axis.horizontal,            allowHalfRating: true,            itemCount: 5,            ratingWidget: RatingWidget(              full: \_image('assets/heart.png'),              half: \_image('assets/heart\_half.png'),              empty: \_image('assets/heart\_border.png'),            ),            itemPadding: const EdgeInsets.symmetric(horizontal: 4.0),            onRatingUpdate: (rating) {              setState(() {                \_rating = rating;              });            },            updateOnDrag: true,          );        case 3:          return RatingBar.builder(            initialRating: \_initialRating,            direction: \_isVertical ? Axis.vertical : Axis.horizontal,            itemCount: 5,            itemPadding: const EdgeInsets.symmetric(horizontal: 4.0),            itemBuilder: (context, index) {              switch (index) {                case 0:                  return const Icon(                    Icons.sentiment\_very\_dissatisfied,                    color: Colors.red,                  );                case 1:                  return const Icon(                    Icons.sentiment\_dissatisfied,                    color: Colors.redAccent,                  );                case 2:                  return const Icon(                    Icons.sentiment\_neutral,                    color: Colors.amber,                  );                case 3:                  return const Icon(                    Icons.sentiment\_satisfied,                    color: Colors.lightGreen,                  );                case 4:                  return const Icon(                    Icons.sentiment\_very\_satisfied,                    color: Colors.green,                  );                default:                  return Container();              }            },            onRatingUpdate: (rating) {              setState(() {                \_rating = rating;              });            },            updateOnDrag: true,          );        default:          return Container();      }    }    //------------------------------------------------------------------    Widget \_image(String asset) {      return Image.asset(        asset,        height: 30.0,        width: 30.0,        color: Colors.amber,      );    }    //------------------------------------------------------------------    Widget \_heading(String text) => Column(          children: [            Text(              text,              style: const TextStyle(                fontWeight: FontWeight.w300,                fontSize: 24.0,              ),            ),            const SizedBox(              height: 20.0,            ),          ],        );  }  //---------------------------------------------------------------  class IconAlert extends StatelessWidget {    @override    Widget build(BuildContext context) {      return AlertDialog(        title: const Text(          'Select Icon',          style: TextStyle(            fontWeight: FontWeight.w300,          ),        ),        shape: RoundedRectangleBorder(          borderRadius: BorderRadius.circular(10.0),        ),        titlePadding: const EdgeInsets.all(12.0),        contentPadding: const EdgeInsets.all(0),        content: Wrap(          children: [            \_iconButton(context, Icons.home),            \_iconButton(context, Icons.airplanemode\_active),            \_iconButton(context, Icons.euro\_symbol),            \_iconButton(context, Icons.beach\_access),            \_iconButton(context, Icons.attach\_money),            \_iconButton(context, Icons.music\_note),            \_iconButton(context, Icons.android),            \_iconButton(context, Icons.toys),            \_iconButton(context, Icons.language),            \_iconButton(context, Icons.landscape),            \_iconButton(context, Icons.ac\_unit),            \_iconButton(context, Icons.star),          ],        ),      );    }    //--------------------------------------------------------    Widget \_iconButton(BuildContext context, IconData icon) => IconButton(          icon: Icon(icon),          onPressed: () => Navigator.pop(context, icon),          splashColor: Colors.amberAccent,          color: Colors.amber,        );  } |  |

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente Imagen que contiene Escala de tiempo

Descripción generada automáticamente

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -