

```
1  module Main exposing (main)
2
3  import Browser
4  import DoubleSlider as DoubleSlider exposing (..)
5  import Html exposing (Html, div)
6  import SingleSlider exposing (..)
7
8
9  main : Program Flags Model Msg
10 main =
11     Browser.element { init = init, update = update, view = view, subscriptions = subscriptions }
12
13
14
15 -- MODEL
16
17
18 type alias Model =
19     { singleSlider : SingleSlider.SingleSlider Msg
20     , doubleSlider : DoubleSlider.DoubleSlider Msg
21     }
22
23
24 type alias Flags =
25     {}
26
27
28 init : Flags -> ( Model, Cmd Msg )
29 init flags =
30     let
31         minFormatter =
32             \value -> String.fromFloat value
33
34         model =
35             { singleSlider =
36                 SingleSlider.init
37                     { min = 0
38                     , max = 1000
39                     , value = 500
40                     , step = 50
41                     , onChange = SingleSliderChange
42                     }
43             |> SingleSlider.withMinFormatter minFormatter
```

```

44         , doubleSlider =
45             DoubleSlider.init
46                 { min = 0
47                   , max = 1000
48                   , lowValue = 500
49                   , highValue = 750
50                   , step = 50
51                   , onLowChange = DoubleSliderLowChange
52                   , onHighChange = DoubleSliderHighChange
53                 }
54     }
55     in
56     ( model, Cmd.none )
57
58
59
60 -- UPDATE
61
62
63 type Msg
64     = DoubleSliderLowChange Float
65     | DoubleSliderHighChange Float
66     | SingleSliderChange Float
67
68
69 update : Msg -> Model -> ( Model, Cmd Msg )
70 update msg model =
71     case msg of
72         DoubleSliderLowChange str ->
73             let
74                 newSlider =
75                     DoubleSlider.updateLowValue str model.doubleSlider
76             in
77             ( { model | doubleSlider = newSlider }, Cmd.none )
78
79         DoubleSliderHighChange str ->
80             let
81                 newSlider =
82                     DoubleSlider.updateHighValue str model.doubleSlider
83             in
84             ( { model | doubleSlider = newSlider }, Cmd.none )
85
86         SingleSliderChange str ->

```

```
87         let
88             newSlider =
89                 SingleSlider.update str model.singleSlider
90         in
91         ( { model | singleSlider = newSlider }, Cmd.none )
92
93
94
95 -- VIEW
96
97
98 view : Model -> Html Msg
99 view model =
100     div []
101         [ div [] [ DoubleSlider.view model.doubleSlider ]
102           , div [] [ SingleSlider.view model.singleSlider ]
103         ]
104
105
106 subscriptions : Model -> Sub msg
107 subscriptions model =
108     Sub.none
109
```