challenge05.md

Procedure

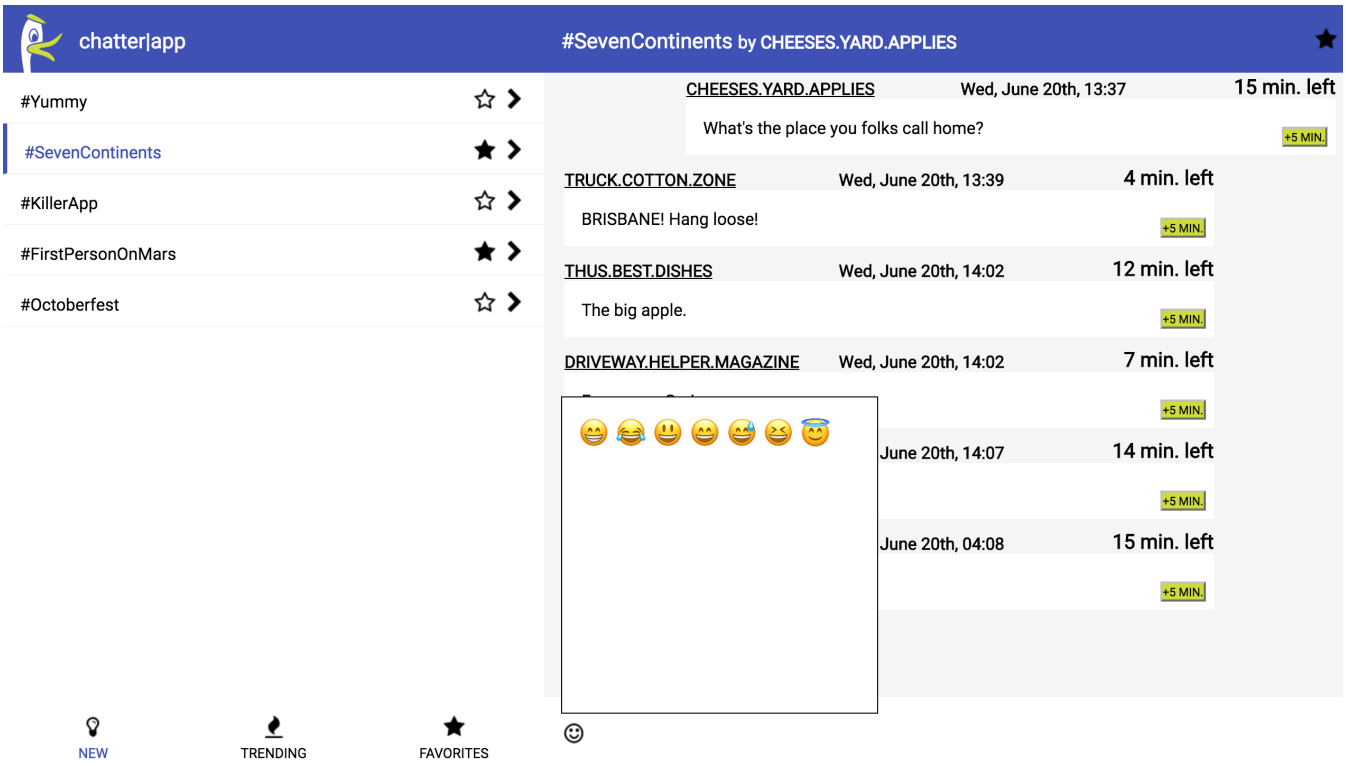
Please follow the instructions below and use the provided Github template for this week's assignment. You will need to upload your code after challenge 6.

You can download the instructions as pdf here.

There are different exercise types:

- These exercises are important, you should tackle them.
- (*) These asterisk-marked exercises are a bit more difficult and thus voluntary. They will improve your skills, tackle them only if you want an extra challenge. However, your app will also work without fulfilling the instruction.

Challenge Goal



Graded Criteria

#split: Channel area are positioned left & chat area right, as in the sample solution.

- 2 Pt. Channel area is on the left.
- 2 Pt. Chat area is on the right.
- 2 Pt. Both areas are full screen height.

#app bars: App bars are styled, comparable to solution, using Material Design as orientation.

- 1 Pt. Height of both app bars are comparable to solution.
- 1 Pt. Font size & weight are comparable to solution.
- 1 Pt. General inner spacings are comparable to solution.
- 1 Pt. Position of image (left app bar) is comparable to solution.
- 1 Pt. Position of text (left app bar) is comparable to solution.

#tab bar: Buttons are equally distributed and at the very bottom of the channel area.

- 2 Pt. Tab bar buttons are distributed equally.

- 1 Pt. Flexbox has been used to distribute buttons equally.
- 1 Pt. Tab bar is at the very bottom.
- 1 Pt. Tap bar takes up the entire width of channel area.

Your **#syntax** will be graded automatically. Overall, 16 Points (Pt.) can be achieved.

Instructions

1. Split the screen

Group the following "layout" styles in a new css "section". To keep everything ordered, as always, comment new stuff.

- Use absolute positioning to left-align the **channels** area, filling the full screen height. The area takes up 40% of the whole browser window.
  Tip: To find out if it worked you can apply a different background color, like red, for testing. Run it in your browser. You might want to change the color back again later.
- Do the same for the **chat** area. It should take up the rest of the browser window.

2. Work on the app bars

All app bar specs are explained in the Material Design guidelines. Use the ones which are necessary. If you have to choose, for instance, between mobile & desktop, go for desktop. Look up the correct styling of your font. You can convert the unit dp 1:1 to px. Material Design always recommends to apply style with padding.

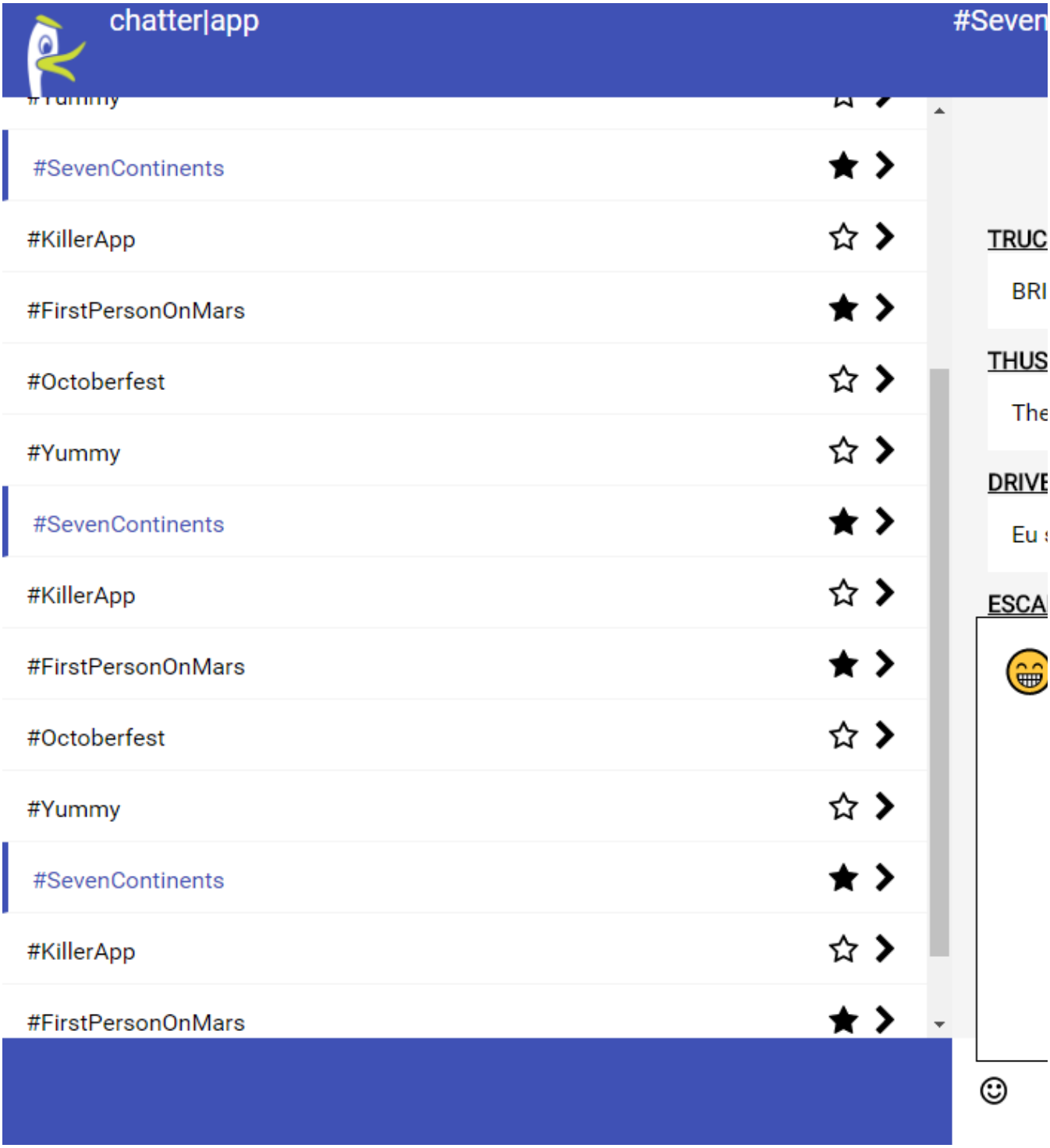
However, margin or position are there for a reason, so use them if you want to. To get Moocus to the right position, applying left: ? px is a lot easier than using padding for everything.

- Both left and right app bar have some styles in common. Reuse previous styles which do not have to be changed according to Material Design. Select both app bars at once and define general styles:
  - reused styles
  - height
  - font size/ weight
  - spacing
- Use absolute positioning to bottom-align the icon. If the image has an absolute position, its position in the app bar should be where? Apply another position property!

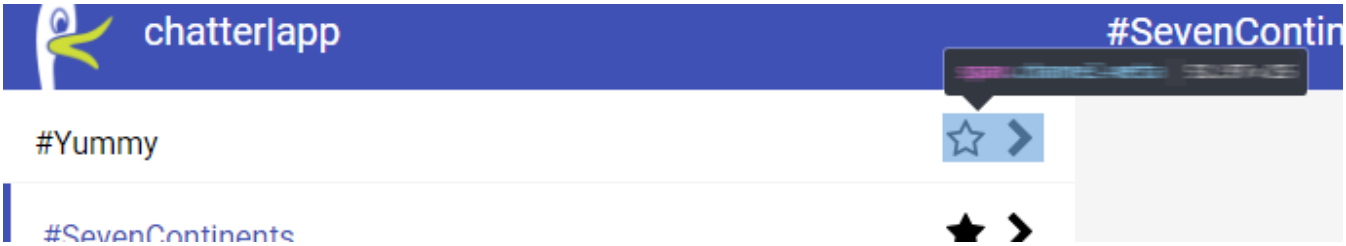
2. Canalize your channels

Decuple your channels - or in plain english: copy & paste them ten times.

- Now your entire "app" scrolls. That is, obviously, not the intended behavior! Position your channel list below the app bar and above the tab bar, using top and bottom. Now, only the channel list should scroll. Once this is done, remove the copies. You can still test the behavior by resizing your browser window. The following screenshot shows what we want, only the channel content scrolls. Before fixing your channel list, your entire browser window should have scrolled - different thing!



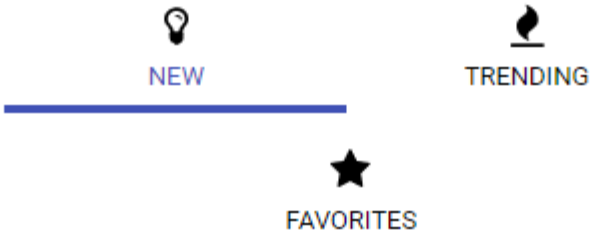
- Add the image to the channel tiles. Group each tile's image in an inline element and use the class channel-meta for selecting it. **Position** the icons right and vertically centered as shown here. **Tip:** The inspector will help. Either hover with your mouse over a rendered element (right-click inspect) or look it up via "Elements" in the tree.



3. Pick up the tab bar

- Bottom-align the tab bar.

Your tab bar is currently a bit messy:



- This is where flexbox comes into play. Flexify your tab bar, by distributing the remaining white-space equally between buttons. Use Flexbox efficiently! There is a very simple way, which requires only few additions to your code. If you do not know what to do, google something like 'flexbox equal width' or 'flexbox increase to equal width'.

4. Shoot the (messages) messenger

- Reposition the "+5 min" buttons inside your messages. They should get 8px distance to each bottom and the right side.
- Distribute the meta-data's contents evenly and bottom-align it using flex.
  Tipp: you do not have to change your HTML for this! A few CSS tweaks would be enough. You have to use the justify-content and align-items property. Go to MDN if you need further information .
- (*) Let all dates appear in one column. You would have to select the HTML elements which surround dates and give them fixed width. Thus you force the dates to stay in-between .

5. Add some smileys

- Insert the following message bar **after your last chat message** and format the code nicely. If the unicode characters are not displayed as smileys in your browser, try typing them in by hand instead of using copy&paste.

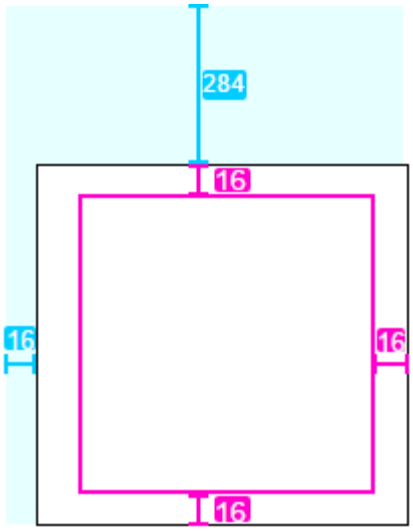
```
<div id="chat-bar"><button id="emojis-button"><img
src="http://ip.lfe.mw.tum.de/sections/smile-o.png">
</button><div id="emojis">&x1F601; &x1F602; &x1F603;
 &x1F604; &x1F605;</div></div>
```

- Style it somewhat like your tab bar, so it looks later like the provided sample solution.
  Tip: Considering that coders are lazy and this message bar looks a lot like the tab bar (the background is also white, it's bottom-aligned...), you could minimize your code by joining chat bar and tab bar selectors. You would only need to add a selector for the chat bar and the chat bar buttons. If you forgot the syntax for CSS Selectors look it up on W3C.
- While buttons are distributed evenly in the tab bar, in the chat bar only one button exists and it should be on the very left. So override the tab bar's button styling! You would need two further styling properties for the chat bar's button, one of which resets a flexbox-property from the tab bar buttons.

  Tip: Consider carefully if you should put it above or below the other styles for tab and chat bar. Remember that cascading order is important!

6. Set the style

- Now, you need a new style for the emoji box. The div should appear as a menu above the chat (see sample solution above). Apply the styles shown in the model below. That is
  - white background color and thin black borders,
  - a square div,
  - font about 28px, depending on the div's size,
  - left-aligned text, and
  - scrolling enabled when there are too many emojis



- (*) We used unicode description for displaying emojis. Use Unicode Hexadecimal Code from: Unicode Emojis. If you want to use other smileys feel free todo so. You want to know what that Unicode is? Check out W3C!

After you're finished the chat bar and emojis div should be comparable to the provided sample solution.

7. Scroll through the messages

- Enable scrolling like in your channels. Use the same procedure as before. Again, the user should only be able to scroll between the app and chat bar. It should not be possible to scroll below your app.

8. Clean up

- Structure and comment your CSS. Add headlines to different sections in your CSS. Structure these sections into styles for specific components of your app (i.e. all styles for message get one section in CSS). You can also insert comments on which styles belong to which step in the instructions. This will make it a lot easier for your fellow students to assess your work!
- Check code & syntax. Using W3C Validator helps.

9. Save your code

Do not forget to save your code! You will build upon your work in the next implementation challenge.