challenge11.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 this challenge**.

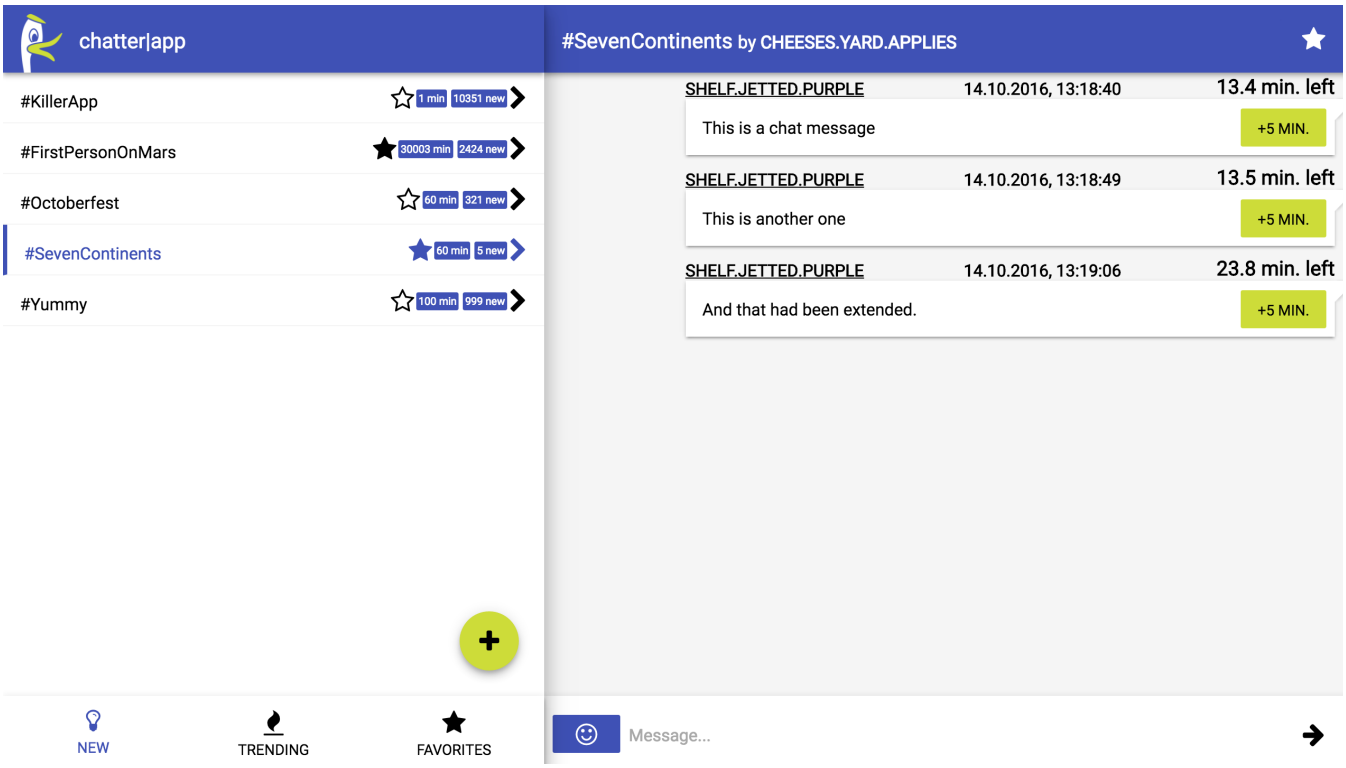
You can download the instructions as pdf here. There are different types of exercises:

- These exercises are important and 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 these instructions.

Prioritize the challenges. Don't spend too much time on the voluntary challenges!

Challenge Goal

This is how your app looks like after **challenge 11**.



Grading Criteria

1. A click event listener is programmatically attached to the channel and calls switchChannel() efficiently. `#channels#click` & `#channels#switch`
2. The app is now initialized via jQuery's ready function. The end of initialization is logged and functions are moved as instructed. `#initialize`
3. Messages inside the div are stored via createMessageElement() in its message object before returning the element. `#update#reference`

Instructions

Make the `#channels` work

We'll now get the channels switching back working. Let's start with the highlighting of the selected channel. First, we need to tell switchChannel which HTML element it should highlight.

- Modify your `switchChannel()` to accept `channelElement` as a second parameter. This element will be a reference to the channel's HTML `<li>` element. `#element`
- Modify your `switchChannel()` to add the class `selected` to this `channelElement` (instead of the present content selector). Use the `$()` wrapper. `#select`

Secondly, we have to register a $.click() handler to get the channels selection back working.

- In `createChannelElement()`, attach a click event listener to your `<li>` in a programmatic way. Use an anonymous function. Inside the function, you have access to the `<li>` via the `this` reference. `#click`
- Within your anonymous listener function, call `switchChannel()`. Do not forget to provide both parameters (consider `this`). `#switch`
- Fix your `listChannels()`, so that it highlights the currently selected channel whenever the sort order changes. Use the `currentChannel` variable. `#sort`

Show `#messages` for the channels

- Whenever you switch a channel, you want to show the messages that 'belong' to the selected channel. Create a function `showMessages` that creates all messages that are saved in a channel's `messages` property and call it in the `switchChannel` function.
- Use the messages in your channels and $.each() for implementing this functionality. Note that this is a utility function for array operations. `#show`
- You will encounter the problem that there are no messages saved in the channels' `messages` property – not even if you 'create' some by selecting a channel and sending messages to it. This is because we push a new message to the `currentChannel.messages` array, however, we do not push it to the respective channel object in the global `channels` array.
- You might want to create dummy message objects in your channels in `channels.js` to test whether your function is working.

The `#chat` box

- (*) Add a `#counter` displaying the number of characters (e.g., 0/140) whenever the user types a letter. In total, not more than 140 characters should be accepted.
- (*) Add an `onclick` event to each `#smiley.` Clicking on a smiley will add it to the message input and adjust the character count accordingly.
- (*) Send the messages when the user hits the `#enter` button

`#Initialize` your app.

- Use one of jQuery's `#ready` functions to initialize your `#app`. Log "App is initialized" at the end of the initialization.
- Move the functions from the body `#onload` here (listChannels and loadEmojis).

`#Update` messages

Let's now implement the message update. Every ten seconds, all messages of the current channels shall be updated for three reasons:

1. the remaining time shall be updated
2. if the remaining time is small (e.g. < 5 minutes), the "5 min. left" text shall be in primary color
3. if there is no remaining time, delete both message object and message element.

You can achieve this in the following steps:

- `#Rework` your `createMessageElement()` to create the `<div>` as a jQuery object

- Once the messages are in the #messages div, you want to modify them on a regular basis, so you should keep a `#reference` to the element. Let `createMessageElement()` store the new message **element** (this can be the jQuery object) in its message **object**, before returning the element. Now you can modify the element based on the object.
- In your app initialisation, create the `#interval` which triggers every ten seconds, and let it log "Updating message elements..."
- During the timeout, update the remaining `#time` for all message elements (use `$.each()`) in the current channel (round it to one decimal to see an effect). Use jQuery find to access the time `em` within the `channelObject.element`
- Outsource this functionality into the Message object. Add an `update()` method in the `Message()` constructor function. Move the element's time modification to this method and call it in the interval.
- Also in the `update()`, mark the remaining time **primary colored** if it's shorter than five minutes `#dying`
- Register an event listener on each "+5 min." button, which has to `#extend` the message's lifetime by these 5 minutes. Don't forget to `update()` after the extension.
- (*) `#Delete` both element and object if a message's lifetime is expired. Use jQuery remove in the `update()` method. Use Array.splice in the interval function, because `$.each()`'s callback provides you the index of the object in the array. You can use return to tell the interval that it should delete the object. Don't forget do decrease the `messageCount` attribute of the channel.

`#cleanup`

- Structure and comment your CSS.
- Check code & syntax. Using W3C Validator helps.

Done?

Do not forget to **save** your work, **push** it to Github, and paste the URL in the submission mask.