Homework 3: User Models and JavaScript

Warm up: due Friday 2/2 @ 11:59pm on Courseworks
Main: due Tuesday 2/6 @ 11:59pm on Courseworks.

Warm-up:

What to submit:
2 files with the following titles:
- js_warmup_screenshot.pdf
- js_warmup.html

Submit them separately to Courseworks, do **not** zip them up.

Problems:
1. Fill out this short form about your TA section preferences:
   a. https://forms.gle/NhGpBaXZyYGFq7Pr7
2. Create a webpage with a single button that has the text "counter" on the button. When the user presses it, two things happen (in this order):
   a. The JavaScript console has a message in it that says "I was clicked " and has a random number.
   b. An alert box appears with some text. You can use any text except "foo".
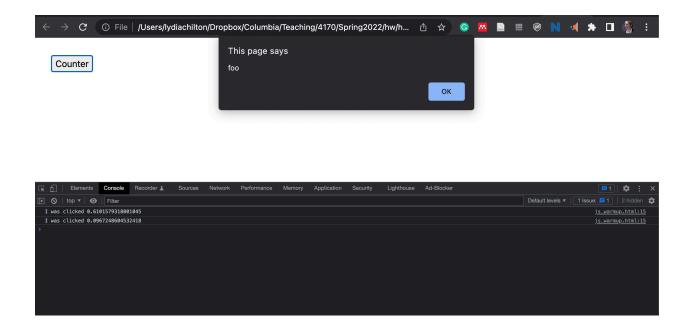
You must use JQuery (not plain JavaScript or HTML) to register the click handler after the page has loaded.

Use a single HTML file that has all your HTML, and JavaScript code (and CSS if you use any).

Use the following snippet to include JQuery:

```
<script src=https://code.jquery.com/jquery-3.3.1.min.js crossorigin="anonymous"></script>
```

3. Submit a screenshot showing the alert and at least two console statements. It should look like the following:

Main Assignment

What to submit:
Two files

- User_model_questions.pdf
- tweet_post.html

Problem 1. User Models

1. One of the primary goals of Courseworks is to allow students to upload assignments. Starting from https://courseworks2.columbia.edu/ what are the subgoals needed to complete that goal. For each stage, take a screen shot, give the stage a name that describes that subgoal, and name one or two low-level actions that are critical to completing that subgoal.
   a. When the final subgoal is complete, what feedback is given to let you know you executed the goal correctly?
   b. Note: there is no exact number of subgoals. As a general heuristic, usually each page has one subgoal. But sometimes a single page has two subgoals. Use your best judgement. Talk to a TA if you're not sure.
2. Many mobile devices (phones, tablets) have menus that appear when you swipe from the top, rather than having to click a start button or a menu button. What advantage does it have from the perspective of low-level interactions? Why?
3. What major feature did every major streaming service (YouTube, Netflix, Hulu, etc.) introduce to save users one low-level interaction? Why did the introduce it? (You may answer from the perspective of users or the service owners, or both? What do you think? Do you use this feature or did you disable it? Why or why not?

Problem 2. Recreate functionality to post a tweet.
Create a website with HTML, CSS, JavaScript and JQuery where a user can write a tweet and see it appear below in a list of their own tweet. This is similar to the functionality of twitter described in class regarding execution and evaluation. Don't replicate the Twitter design, just the basic functionality of writing and posting a tweet with a **max of 30 characters** and providing feedback to help users evaluate the goal of posting a tweet and the subgoal of writing a tweet that within the max number of characters. The following is the functionality you need to replicate:

1. There needs to be place to write the tweet and a button to press to post the tweet.
2. There needs to be a number that shows how many characters the user has remaining. The max length can be hard coded in JavaScript (but not in HTML).
3. As the user types their tweet text, the following must happen:
    a. The characters remaining number must change to reflect the current number of characters left.
    b. If the user exceeds the number of characters, a negative number must be shown for characters remaining.
    c. Negative characters remaining numbers must be red and bold
    d. Non-negative characters remaining numbers must be lightgrey and normal font weight.
    e. If the number of characters remaining is negative, the "post tweet" button must not be clickable (disabled).
    f. If the user presses "enter" the "post tweet" action is taken.
4. When the user presses the "post tweet" button the following things must happen:
    a. The post must appear with the poster's username. The username can be hard coded in JavaScript (but not in HTML).
    b. New posts must appear at the top of the list of posts, so that users see the latest tweets at the top of the list.
    c. The text in the text input element must disappear.
    d. The number that counts the number of characters remaining must return to the maximum number.
    e. The cursor must return to the box where the user writes tweets (so that they can immediately start writing another tweet!)
    f. If there is no text in the text input element, or all the text is blank space, do not post the tweet.

Obviously, "posting a tweet" will only work locally on your computer. It will not post anything to the internet. Also, the posted tweets with not persist if the close the page and open it again.

For this problem you only need to turn in your code, there is no write up.

You do not need bootstrap, but it you want to use it, please use the same link from previous assignments.

This assignment is about functionality and not design. As long as we can see and read all the elements and they have appropriate sizes and positions, the design is fine. Below is a screenshot of a reference solution. You do not need to copy it, it is only here to convey that having very basic design is fine. The following screenshots show examples of what the app looks like in various stages. You may copy this basic design if you like. Again, it's not about design, it's about functionality.

When the text is post-able:

Write a tweet

Call me Ishmael.

13   Post Tweet

POSTS

chilton	Third post
chilton	Second post
chilton	First post

When the text Is not post-able: (the text continues past where you can read it)

Write a tweet

Call me Ishmael. Some years ago- never

-12 Post Tweet

POSTS

chilton	Third post
chilton	Second post
chilton	First post