Simulation Storyboard
Use JavaScript to add interactivity

1. In the VS Code web tool, you are starting with the empty JavaScript file called **script.js** to add interactivity to your web page. Select the **Next arrow** to continue.

2. First, you add a comment. Next, you create two variables to hold the input elements and the element that will display tasks. Under **script.js**, select the field next to 1. This is the first line of your code. Type one line of code in each field exactly as it displays in the Code pad. Then, press **Enter**.

   Code pad code:

   ```
   //constants declared for input button and task list area
   const taskInput = document.querySelector("#newtask input");
   const taskSection = document.querySelector('.tasks');
   ```

3. Notice your comment is about what this section of code does. And you are using a **querySelector** so JavaScript can find HTML elements and modify them. Select the **Next arrow** to continue.

4. Per the requirements, users must be able to add a task by pressing **Enter** on their keyboard. To code this, you can add an event listener for the **Enter** key. Select the **Next arrow** to continue.

5. Type one line of code in each field exactly as it displays in the Code pad. Then, press **Enter**.

   Code pad code:

   ```
   //listener for the Enter key. Used to add a new task.
   taskInput.addEventListener("keyup", (e) => {
   if (e.key == "Enter") {createTask(); } } );
   ```

6. Notice that the **event listener** is aware of *every* keystroke, but it *only* calls "createTask", the function that the handler you just added (keyup) executes, when users press the **Enter** key. Select the **Next arrow** to continue.

7. Per the requirements, your web page must function on both desktop and mobile devices. You must include an **Add** button so users can add tasks without a keyboard. To code this, you can add an **onclick event**. Select the **Next arrow** to continue.

8. Type one line of code in each field exactly as it displays in the Code pad. Then, press **Enter**.

Code pad code:

```
//the onclick event for the 'Add' button
document.querySelector('#push').onclick = function () {
createTask(); }
```

9. Notice that the **onclick event** calls the same function as the **event listener** that you added in the previous step for the **Add** button. Check out the Preview pane to see how your web page has progressed! Select the **Next arrow** to continue.

10. Most of your JavaScript file contains code that adds tasks to the task area on your page. This function runs when a user presses **Enter** or selects the **Add** button. But this function must first check to make sure the user typed in a task. Select the **Next arrow** to continue.

11. Type one line of code in each field exactly as it displays in the Code pad. Then, press **Enter**.

    Code pad code:

```
//the function that creates a task
function createTask() {
if (taskInput.value.length == 0) {
alert("The task field is blank. Enter a task name and try again."); } }
```

12. Notice that the code you just wrote has an **if statement** to check whether the task entry field has a task in it. If it doesn't, then your page will alert the user. And you haven't closed your createTask function yet. You'll do that later. Select the **Next arrow** to continue.

13. Now it's time to write code to add the task the user enters in the field. For this, use an **else statement**. Under the code you just typed, type one line of code in each field exactly as it displays in the Code pad. Then, press **Enter**.

    Code pad code:

```
else {
 }
```

14. Between the last curly braces, you have a section for your **else statement**. To create a new task, you can write code that will use JavaScript to write HTML! The JavaScript code places the new HTML code in the tasks section of your web page. Select the **Next arrow** to continue.

15. You have a lengthy set of code to type. Between the curly braces for the else statement, type one line of code in each field exactly as it displays in the Code pad. Then, press **Enter**.

    Code pad code:

    //this block inserts HTML that creates each task into the task area div element
    taskSection.innerHTML +=
    **` **
    <label id="taskname">
    <input onclick="updateTask(this)" type="checkbox" id="check-task">
    ** ${document.querySelector('#newtask input').value} **
    </label>
    ** **
    ** `;**

16. Notice one of the div elements the JavaScript code creates for the task item is a **Delete** button for each task, allowing users to delete a task. Select the **Next arrow** to continue.

17. Next, you need to add code for the **Delete** button to function. Under the last closing div tag and inside the closing curly brace for the **else statement**, type one line of code in each field exactly as it displays in the Code pad. Then, press **Enter**.

    Code pad code:

    var current_tasks = document.querySelectorAll(".delete");
    for (var i = 0; i < current_tasks.length; i++) {
    current_tasks[i].onclick = function () {
    this.parentNode.remove(); } }

18. You need to add code that will show and remove the scrollbar. Next, see what that looks like. Select the **Next arrow** to continue.

    Continue adding interactivity to the web page with JavaScript

    For the purpose of your project scenario, you add code to adjust a scrollbar. The code will show a scrollbar if the area in your task list is larger than 300 pixels. If the area is less than 300 pixels, the code will remove the scrollbar.

    You can see what this code looks like in your **script.js** file.

    Select **X** to close this window and continue.

19. Notice the code you added. This code adds a scrollbar if the area is larger than 300 pixels and removes the scrollbar if the area is less than 300 pixels. Select the **Next arrow** to continue.

20. To complete the code, you apply a strikethrough style to tasks that users check to mark complete and remove it for tasks that users do not check. Under the createTask function, type one line of code in each field exactly as it displays in the Code pad. Then, press **Enter**.

    Code pad code:

    ```
    function updateTask(task) {
    let taskItem = task.parentElement.lastElementChild;
    if (task.checked) {
    taskItem.classList.add("checked");}
    else {taskItem.classList.remove("checked"); } }
    ```

21. Take a moment to review what you've accomplished for your interactive task list web page! Select the **Next arrow** to continue.

You successfully created your JavaScript file in VS Code. Your web page is now interactive so your users can use the task list with a variety of functions.