Project: Event Handling With Mr. Spud Face

Mr. Spud Face is a charming potato with an inexplicably handsome face. He has recently moved to your state and needs to get an updated driver's license with his current information, so that he can do nifty things like drive to the grocery store, vote, and take his spud spouse on special dates.

Use Javascript to create a driver's license for Mr. Spud Face. If you want to go the extra mile, create a Mr. Spud Face drag-and-drop game as a bonus.

Now that you know how to handle page events with Javascript, put that knowledge into use on this project!

In this project, you will:

- Write Javascript to handle common form events
- Handle input, checkbox, and button click events
- Grab form values and update elements with those values
- Utilize bubbling and event delegation
- Use the HTML Drag-and-Drop API (if you dare!)

Project overview

Use what you've learned about event handling to complete this project. Demonstrate that you can use event listeners on page elements and event handlers.

We have set up a project folder for you to use inside this folder called `spud-face-project.zip` with an HTML file, CSS file, and Javascript file. Use this folder to complete your project.

In phases 1-4, you will write Javascript to grab the driver's license form values and update Mr. Spud Face's license, as well as handle other form events on inputs and button clicks. To get a good understanding of the HTML Drag-and-Drop API, complete the bonus section by making a drag-and-drop spud game.

Phase 1: Create a spud driver's license

In your `spud-face-project` folder, open up your `spud-face.html` file. Open it up in a browser to see what the page looks like.

We've filled it with elements, chief of which are a `<form>` element and a `<div>` depicting a driver's license. Use the form to build out the driver's license information.

Phase 1A: Get form values and display on driver's license

The form values on the left should update the driver's license information on the right. Set up event listeners on the form whenever the user inputs a value into a form input. Get the value of that form input and update the corresponding information on the driver's license. You might want to use the following:

- document.getElementById()
- eventTarget.addEventListener()
- HTMLElement: input event
- checkbox.checked attribute

Phase 1B: Refactor to use event delegation and event.target

In phase 1A, you might have set up event listeners on each form input. While that does work, it would be ideal to make use of event delegation and attach a single listener to our form.

1. Set up a single event listener on the form to listen for an input change.
2. Write some logic, in the form of an `if` statement or `switch` case statement, to update the `innerHTML` of the driver's license elements that correspond with `event.target`.
3. You may want to use `event.target.id` and `event.target.value`.
4. Make sure your script runs after the DOM has loaded.

Phase 2: Add focus and blur events to form inputs

Jazz up your form inputs by adding a quick color change on `focus`, and removing it on `blur`. Give active inputs a background color of `lightgreen` and no background color (initial state) when inactive. Use the following to do so:

- `Element: focus event`
- `Element: blur event`

Phase 3: Check that license numbers match

Check that the numbers entered by the user on the license number fields match. In your HTML file, these are represented by the inputs with the IDs of `input-license-num` and `input-license-num-confirm`.

If the numbers don't match, then change the background color of both inputs to be `lightcoral`.

Again, you'll use `event.target.value` here. You might want to use `setTimeout()` to give the user some time to fill out the form.

Phase 4: Update submit button click count

Since this isn't a real form that actually submits the driver's license info anywhere, you won't need to make a server or API request. Instead, write a function to increment the click count every time the submit button is clicked.

Listen for a `click` event on the button. Then, update the click count inside of the button.

Bonus: Mr. Spud Face Drag-and-Drop Game

Use the HTML Drag-and-Drop API to create a Mr. Spud Face drag-and-drop game inside of your `spud-face.html` file.

Add the images inside of the project's `images` folder to your HTML file, and write Javascript that will let the user drag the spud body parts and drop them onto the spud body. Set up handlers for these drag events:

- `dragStart`
- `drag` or `dragOver`
- `dragEnd` or `drop`