Event Handling: Form Validation

Everyone has submitted a form at some point. Form submissions are another common action users take on a website. We've all seen what happens if we put in values that aren't accepted on a form -- frustrating errors! Those errors prompt the user to input accepted form values before submission and are the first check to ensure valid data gets stored in the database.

Learning how to implement front-end validation before a user submits a form is an important skill for developers. In this reading, we'll learn how to check whether two password values on a form are equal and prevent the user from submitting the form if they're not.

Validate passwords before submitting a form

In order to validate passwords, we need a form with two password fields: a password field and a confirmation field. We'll also include two other fields that are common on a signup page: a name field and an email field. See the example below:

HTML

```
<!DOCTYPE html>
<html>
  <head>
    <script src="script.js">
  </head>
  <body>
    <form class="form form--signup" id="signup-form">
      <input class="form__field" id="name" type="text" placeholder="Name" style
      <input class="form__field" id="email" type="text" placeholder="Email" sty
      <input class="form__field" id="password" type="text" placeholder="Passwor
      <input class="form__field" id="confirm-password" type="text" placeholder=
```

```
      <button class="form__submit" id="submit" type="submit">Submit</button>
    </form>
  </body>
</html>
```

Now, we'll set up our `script.js`file with code that will:

- Listen for a form submission event
- Get the values of the two password fields and check for a match
- Alert the user if there's not a match, or submit the form

Javascript

```
// script.js
window.addEventListener("DOMContentLoaded", event => {
  // get the form element
  const form = document.getElementById("signup-form");

  const checkPasswordMatch = event => {
    // get the values of the pw field and pw confirm field
    const passwordValue = document.getElementById("password").value;
    const passwordConfirmValue = document.getElementById("confirm-password")
      .value;
    // if the values are not equal, alert the user
    // otherwise, submit the form
    if (passwordValue !== passwordConfirmValue) {
      // prevent the default submission behavior
      event.preventDefault();
      alert("Passwords must match!");
    } else {
      alert("The form was submitted!");
    }
  };
  // listen for submit event and run password check
  form.addEventListener("submit", checkPasswordMatch);
});
```

In the code above, we prevented the form submission if the passwords don't match using Event.preventDefault(). This method stops the default action of an event if the event is not explicitly handled. We then alerted the user that the form submission was prevented.

What we learned:

- Front-end form validation prevents invalid data from being recorded in the database.
- We use `Event.preventDefault()` to stop form submission.
- Users are typically notified when default behavior is prevented.