6.2.4 Handle Form Submission

Now that we have a form for users to use, let's make it functional. Think back to when we've done this before—it usually involved the following actions in JavaScript:

- Adding an event listener to the `<form>` element to execute a function on submission
- Capturing the form's input data to use elsewhere in the app

Luckily, we'll follow the same basic process for this form. The only difference is what we're doing with the input data. Let's get started!

To begin, let's navigate to `homepage.js`. Create two new variables to store a reference to the `<form>` element with an `id` of `user-form` and to the `<input>` element with an `id` of `username`:

```
var userFormEl = document.querySelector("#user-form");
var nameInputEl = document.querySelector("#username");
```

Now that we have variables for the two form elements, let's create a function called `formSubmitHandler` to be executed upon a form

submission browser event.

Add the following code to `homepage.js`:

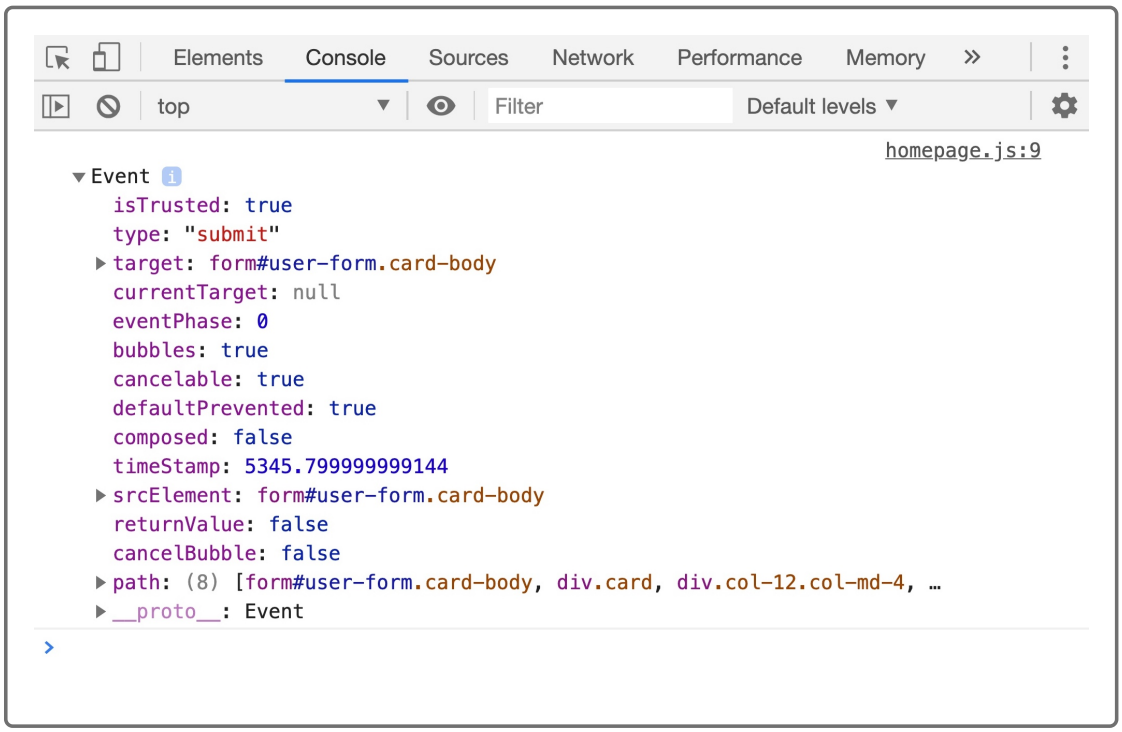
```
var formSubmitHandler = function(event) {
  event.preventDefault();
  console.log(event);
};
```

Lastly, let's add the submit event listener to the `userFormEl`. Because we don't need to execute the `getUserRepos()` function at the bottom of the `homepage.js` file, remove that line of code and replace it with this:

```
userFormEl.addEventListener("submit", formSubmitHandler);
```

Before we add any more functionality to `formSubmitHandler()`, let's make sure the event listener is working correctly. We should always test functionality like this before we write too much code, because if there's an issue, we'll be able to spot it faster and fix it before moving on.

So to test the event listener, save `homepage.js`, refresh the page in the browser, and try to search for a user. This won't do much, but we can see if it works by opening the Chrome DevTools console. We should see something like this image after we submit the form:



Now that we know the form submit functionality is working, let's make it actually do something!

PAUSE

What does `event.preventDefault()` do?

It stops the browser from performing the default action the event wants it to do. In the case of submitting a form, it prevents the browser from sending the form's input data to a URL, as we'll handle what happens with the form input data ourselves in JavaScript.

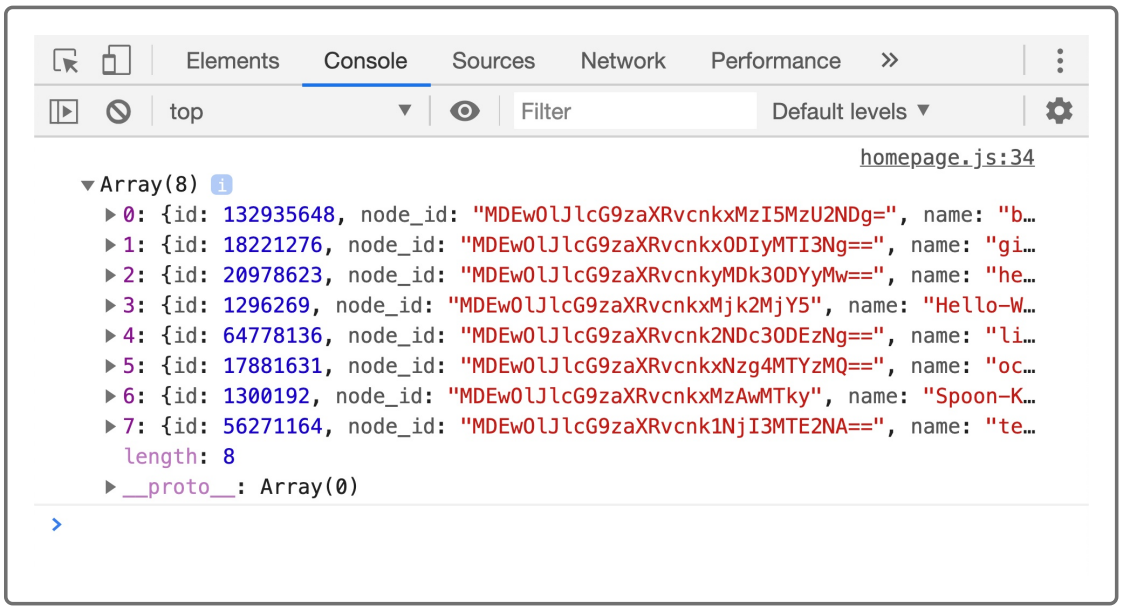
Hide Answer

Let's update the `formSubmitHandler()` function to get the value of the form `<input>` element and send it over to `getUserRepos()`. Put this code in the function underneath the `event.preventDefault();` line:

```
// get value from input element
var username = nameInputEl.value.trim();

if (username) {
  getUserRepos(username);
  nameInputEl.value = "";
} else {
  alert("Please enter a GitHub username");
}
```

Okay, you should be all set up. Save `homepage.js`, refresh the page, and try searching for a GitHub username, like your own username or "facebook". The result in the console should look like the following image:



Great! Now we can use `fetch()` to search for any GitHub user whenever we fill out the form. Let's quickly review the code we just placed into `formSubmitHandler()`, and then we'll get into displaying this data in the page.

When we submit the form, we get the value from the `<input>` element via the `nameInputEl` DOM variable and store the value in its own variable called `username`. Note the `.trim()` at the end: this piece is useful if we

accidentally leave a leading or trailing space in the `<input>` element, such as " octocat" or "octocat ".

Then we check that there's a value in that `username` variable. We wouldn't want to make an HTTP request without a username, if we accidentally left the `<input>` field blank! If there is in fact a value to `username`, we pass that data to `getUserRepos()` as an argument. Then, to clear the form, we clear out the `<input>` element's value.

At this point, we can use a form to search for a GitHub user's repositories, and we're getting a list of those repositories in the response. Next we need to display them on the page.