Submitting the Form

It's time to create another action so that our screen can be updated with the new message, but is that all we need to do? Let's find out.

Right now, when you type a message and hit enter, it doesn't show up in the conversation list, and the page reloads.

Terrible!

Let's handle the form submission.

In `MessageInput.js`, add a `handleSubmit` event handler as shown below:

```
...
<form className="Message" onSubmit={handleSubmit}>
  ...
</form>
...
```

MessageInput.js

Think about it for a minute. To update the list of messages in the conversation...we need to dispatch an action!

This action needs to take the value in the input box, and add it to the messages of the active user.

Okay, so this looks like a good shape for the action:

```
{
  type: "SEND_MESSAGE",
    payload: {
      message,
      userId
    }
}
```

Got that?

Now, let's write the handleSubmit function:

```
//first retrieve the current state object
const state = store.getState();
const handleSubmit = e => {
  e.preventDefault();
  const { typing, activeUserId } = state;
  store.dispatch(sendMessage(typing, activeUserId));
};
```

MessageInput.js

Here's what is going on within the handleSubmit function:

With **e.preventDefault()**, I think you already know what that does. The typing value and activeUserId are fetched from the state since they'll both be used to create the dispatched action.

And finally, the action is dispatched with store.dispatch(sendMessage(typing, activeUserId)).

Oops, but with an action creator, sendMessage.

In `actions/index.js`, create the `sendMessage` action creator:

```
import {
  ...
  SEND_MESSAGE
} from "../constants/action-types";
export const sendMessage = (message, userId) => ({
  type: SEND_MESSAGE,
  payload: {
    message,
    userId }
})
```

actions/index.js

That also means the SEND_MESSAGE action type constant needs to be created in `constants/action-types.js`

```
export const SEND_MESSAGE = "SEND_MESSAGE";
```

Before testing the code, you should not forget to update the action creator imports in `MessageInput.js` to include sendMessage

```
import { setTypingValue, sendMessage } from "../actions";
```

So try it out. Does the code work?

Uh, No it doesn't.

The form is submitted, the page doesn't reload due to the form submission, the action is dispatched, but still no updates.

We've done nothing wrong, except that the action type hasn't been catered for in any of the reducers.

The reducers know nothing about this newly created action of type, `SEND_MESSAGE`

Let's fix that next.