



---

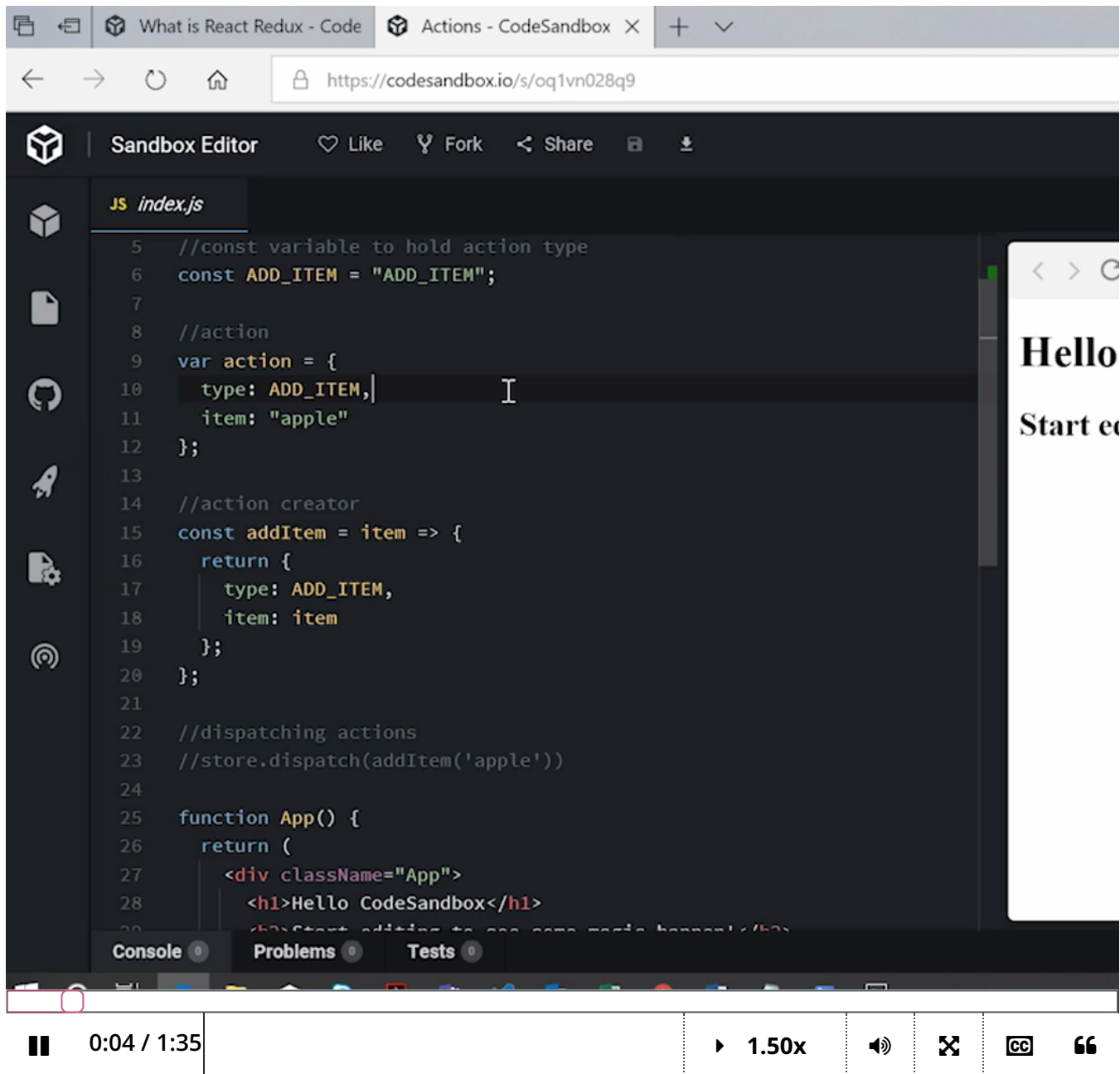
[Course](#) > [Redux](#) > [Actions](#) > Actions

---

**Audit Access Expires Jun 20, 2020**

You lose all access to this course, including your progress, on Jun 20, 2020.

## Actions



The screenshot shows a CodeSandbox editor with the following code in `index.js`:

```
5 //const variable to hold action type
6 const ADD_ITEM = "ADD_ITEM";
7
8 //action
9 var action = {
10   type: ADD_ITEM,
11   item: "apple"
12 };
13
14 //action creator
15 const addItem = item => {
16   return {
17     type: ADD_ITEM,
18     item: item
19   };
20 };
21
22 //dispatching actions
23 //store.dispatch(addItem('apple'))
24
25 function App() {
26   return (
27     <div className="App">
28       <h1>Hello CodeSandbox</h1>
29     </div>
30   );
31 }
```

The right sidebar shows a preview of the application with the text "Hello" and "Start ec". The bottom of the editor shows tabs for Console, Problems, and Tests. A video player interface is overlaid at the bottom of the screenshot, showing a play button, a progress bar at 0:04 / 1:35, a speed control at 1.50x, and icons for volume, full screen, and subtitles.

## Video

[Download video file](#)

## Transcripts

[Download SubRip \(.srt\) file](#)

[Download Text \(.txt\) file](#)

## Actions

## Actions

Actions are categorized events that trigger changes to your state. Actions have a type and contain some kind of payload information. Once actions are dispatched, the Reducer figures out what to do with the action type and payload and updates the state accordingly.

To implement an action, we can use a simple JavaScript object that contains a `type` and a `payload`. The `type` attribute is used to identify the action and is usually a capitalized string. It is a common practice to separate out the action type into a `const` variable. The `payload` attribute is used to attach any type of data (string, number, object, etc.) to the action, but it is recommended to send as little data as possible to get the job done. The `payload` attribute can have any name or format as long as you reference it accordingly in the Reducer.

Action:

```
const ADD_ITEM = 'ADD_ITEM' //const variable to hold action type

var action = {
  type: ADD_ITEM, //action type
  item: 'Apple' //payload
}
```

## Action Creators

Action creators are simply functions that return actions. They are used so you don't have to type out the entire action object every time you want to dispatch an action.

Action Creator:

```
const addItem = (item) => {  
  return {  
    type: ADD_TODO,  
    item: item  
  }  
}  
  
addItem('apple') // this creates an action object
```

## Dispatching Actions

To dispatch an action, simply use Redux's `store.dispatch(action)` method.

```
store.dispatch(addItem('apple'))
```

However, we can't dispatch actions without having a store yet. More information on this will be provided in later chapters.

Try out this live example on CodeSandbox: <https://codesandbox.io/s/oq1vn028q9>