useAddTodo Hook

In this lesson, we create a hook to add a new ToDo item.

WE'LL COVER THE FOLLOWING

useAddTodo

We create a hook that returns a callback function which allows us to add a new ToDo item.

```
// ./hooks/useAddTodo.js

import { useCallback } from 'react';
import { useSetDraft } from '../store';

let nextId = 100;

export const useAddTodo = () => {
  const setDraft = useSetDraft();
  return useCallback(
    title => {
      setDraft(draft => {
        draft.todos.push({ id: nextId++, title });
      });
    },
    [setDraft],
  );
};
```

With `setDraft` we can mutate the draft state. The `nextId` is defined outside of the hook to provide a unique ID every time the callback is called.

We explicitly use `useCallback` for returning the stable callback. It's not necessary if we are sure that stability is not important. But, we don’t know at

this point if the callback function will be passed for props of child components that are wrapped by React.memo. Hence, it’s a safe bet to always use `useCallback` in custom hooks.

Next

We continue to create another hook. In the next lesson, we learn to create `useDeleteTodo`.