× Go to Dashboard

Immutable Update Patterns

Section 14, Lecture 256

Immutable Update Patterns on reduxjs.org: http://redux.js.org/docs/recipes/reducers/ImmutableUpdatePatterns.html (http://redux.js.org/docs/recipes/reducers/ImmutableUpdatePatterns.html)

Updating Nested Objects

The key to updating nested data is **that *every* level of nesting must be copied and updated appropriately**. This is often a difficult concept for those learning Redux, and there are some specific problems that frequently occur when trying to update nested objects. These lead to accidental direct mutation, and should be avoided.

Common Mistake #1: New variables that point to the same objects

Defining a new variable does *not* create a new actual object - it only creates another reference to the same object. An example of this error would be:

```
function updateNestedState(state, action) {
    let nestedState = state.nestedState;
    // ERROR: this directly modifies the existing object reference - don't do this!
    nestedState.nestedField = action.data;

    return {
        ...state,
        nestedState
    };
}
```

This function does correctly return a shallow copy of the top-level state object, but because the `nestedState` variable was still pointing at the existing object, the state was directly mutated.

Common Mistake #2: Only making a shallow copy of one level

Another common version of this error looks like this:

```
function updateNestedState(state, action) {
    // Problem: this only does a shallow copy!
    let newState = {...state};

    // ERROR: nestedState is still the same object!
    newState.nestedState.nestedField = action.data;

    return newState;
}
```

Doing a shallow copy of the top level is *not* sufficient - the `nestedState` object should be copied as well.

Correct Approach: Copying All Levels of Nested Data

Unfortunately, the process of correctly applying immutable updates to deeply nested state can easily become verbose and hard to read. Here's what an example of updating `state.first.second[someId].fourth` might look like:

Browse Q&A Continue