U37HUSJ4R : so for example `paused` could be any field
U3SJEDR96 : There is no generic record-updater syntax; sadly
U37HUSJ4R : :cry:
U37HUSJ4R : So if I have `hold`, `pause`, `hangup` for example I need 3 functions and 24 lines of code :disappointed:
U3SJEDR96 : Not entirely true; but there's another thing... Can a call be simultaneously on hold, paused and hung up?
U37HUSJ4R : this is more state of the buttons
U37HUSJ4R : so if I call is on hold then the state is
U37HUSJ4R : ```paused : True,
hangup : True,
paused : False
```

U37HUSJ4R : but yes it could be all three
U37HUSJ4R : multiple different valid states here
U3SJEDR96 : I'd try to think of a better way to model those, though. In the meantime, you can use something like ```
updateControls : (Controls -&gt; Controls) -&gt; Call -&gt; Call
updateControls op call =
    { call | controls = Maybe.map op call.controls }
```

U37HUSJ4R : I'd LOVE a better way to model these
U37HUSJ4R : but really can't think of one :disappointed:
U3SJEDR96 : ```updatePaused : Bool -> Call -> Call
updatePaused newValue call =
    updateControls (\controls -> { controls | paused = newValue }) call
```

U3SJEDR96 : you'd still end up with 15 lines of code, including annotations, but it would be pretty clear what they all did. And if you decided that `call.controls` should be a `Result` rather than a `Maybe`, that's only a single line to change
U3SJEDR96 : as for your control-states.. I think having a `Status = Ongoing | Paused | OnHold | Hangup` (or something similar, depending on requirements) would make sense, with functions `canPause : Status -&gt; Bool` etc
U3SJEDR96 : Though it might make sense for a call to be both onhold and paused, in which case that would be a fifth case
U3SJEDR96 : The thing is that it might be possible to represent your possible states as a union type of possible states, and derive the available options in your view from that
U37HUSJ4R : I have thought about this, do you think its ok to end up with something like:`Status = Ongoing | Paused | OnHold | Hangup | OnHoldAndPaused | SomethingElseHere | SomeOtherStateAndSomethignElse`

U23SA861Y : seems like you want two types actually, one for when the phone is not active and then when it is active, what the sub state is
U3SJEDR96 : Yeah. Alternatively, there could be a "status on this end" and "status on the other end", with the possible actions derived from that combination
U37HUSJ4R : thinking out loud, maybe I want a `List Status`
U37HUSJ4R : so I have `[Hold, Hangup]`
```