U37HUSJ4R : my thinking is certainly around impossible states
U37HUSJ4R : for example I don't want paused to be true if user isn't on a call
U3SJEDR96 : that imaginary `fun` above can _only_ touch `pauzed`, and is not aware of anything else in your record. So that basically creates the same guarantee as nesting it
U37HUSJ4R : so I have something like
U37HUSJ4R : ```type Status
    = Available
    | Wrap
    | OnCall Call
```

U3SJEDR96 : That might be extended to `Status = ... | OnCall Bool Call` though
U3SJEDR96 : which guarantees that you only have that bool if you're actually in a call
U3SJEDR96 : come to think of it `OnCall Bool Call` is basically `OnCall Call | Paused Call` anyway
U37HUSJ4R : true, but this is just a simple example
U37HUSJ4R : i also have transfer, hold etc
U6FFD2QG0 : Hi everyone, I'm running into something that seems like it should have a simple solution, but I can't figure out what that is. I need to construct an instance of `Cmd msg` as an alternative to `Cmd.none` in an if branch. I'm not using any outside effects or anything. Here's the relevant code snippet:```
update : Msg -&gt; Model -&gt; (Model, Cmd Msg)
update msg model =
  case msg of
    Tick newTime -&gt;
      let
        newTime = decTimer model
        newCmd = if newTime.activeTimer &gt; 0
                   then Cmd.none
                   else Cmd.Cmd TimerDone -- help!
      in
        (newTime, newCmd)
```

U0LPMPL2U : Is this to prevent duplication between the `Tick` and `TimerDone` branches of your `update` ?
U6FFD2QG0 : yeah, basically
U3SJEDR96 : I would suggest taking the contents of your `TimeDone` branch, putting it into a separate function, and replace that with```
TimerDone -&gt;
    timerDone model
Tick newTime -&gt;
    if .activeTime (decTimer model) &gt; 0 then (newTime, Cmd.none) else timerDone model
````

U0LPMPL2U : I'd suggest extracting the common logic to a helper function and calling that from both branches instead of trying to send a `Msg`
U6FFD2QG0 : fair enough. Is this just not a typical way that a Cmd would be used?
U0LPMPL2U : You almost never want to just send Msg to yourself
U0LPMPL2U : Msg is meant to represent events from the outside world
U3SJEDR96 : No, a `Cmd msg` represents something for the runtime to execute asynchronously, after which it can call your `update` with the resulting `Msg`
U6FFD2QG0 : ok, that's good to know
U6FFD2QG0 : thanks!
U3SJEDR96 : you don't really need the runtime in order to call a function, though, so using a function to abstract the behaviour of "what should happen when your timer is done" is definitely the recommended approach
U37HUSJ4R : If I wanted to make this more generic:```
updatePaused : Bool -&gt; Call -&gt; Call
updatePaused newValue ({ controls } as call) =
    { call
        | controls =
            Maybe.map
                (\controls -&gt; { controls | paused = newValue })
````

~~~
            call.controls
   }
```
~~~