

U5QJW0DDE : <@U4JT89FGB> i'm sorry you don't understand the issue, but I think some of the others have

U5QJW0DDE : it's not complicated.

U5QJW0DDE : it's just about separating model state from view heirachies

U0LPMPL2U : If I understand correctly, you'd like to be able to pull some sort of "global" value directly from within deeply nested child functions?

U5QJW0DDE : that would be one solution, yes

U5QJW0DDE : or, to "subscribe" to a particular part of the model for a child

U5QJW0DDE : i will post to the mailing list

U663M2MB7 : Is there any equivalent to ``\$` from Haskell in Elm?

U0CL0AS3V : <@U663M2MB7> `<`]

U663M2MB7 : Thank you! :slightly_smiling_face:

U64FYS317 : Can anyone help me troubleshoot this? I'm getting a `cannot find variable: Mdl` in the following code, although it's defined just a few lines before.``

module Lib.Layout.View exposing (..)

```
import Types exposing (Model)
import Html exposing (..)
import Html.Attributes exposing (..)
import Material.Layout as Layout
import Material
```

```
type alias Mdl =
    Material.Model
```

```
view : (Model -&gt; Html msg) -&gt; Model -&gt; Html msg
```

```
view viewFn model =
```

```
-- Cannot find variable `Mdl`
```

```
Layout.render Mdl
```

```
    model.mdl
```

```
    [ Layout.fixedHeader
```

```
    ]
```

```
    { header = [ h1 [ style [ ( "padding", "2rem" ) ] ] [ text "counter" ] ]
```

```
    , drawer = []
```

```
    , tabs = ( [], [] )
```

```
    , main = [ viewFn model ]
```

```
    }
```

```
``
```

U64FYS317 : I'm confused at the possibility of such a thing

U300LJUAK : I think your confusing the meaning of this part.

```
``type alias Mdl =
```

```
    Material.Model``
```

This only defines `Mdl` as an alias of the type `Material.Model`, so you can use it in function definition. It does not copy `Material.Model`'s constructor into `Mdl`.

U300LJUAK : ``type alias Foo = { rawr : String }

```
type alias Bar =
    Foo
```

```
someFunc : Bar -&gt; String
```

```
someFunc { rawr } =
```

```
    rawr``
```

In this case I could do `Foo "someString"` and get a value of type `Foo`, but I can't do `Bar "someString"`. I can, on the

other hand, pass a value of type `Foo` into `someFunc`, because `Bar` is an alias of `Foo`.

U64FYS317 : <@U300LJUAK> Thanks. I'm trying to reconcile this with some examples I've seen

U300LJUAK : No problem, I doubt I can really help you any further than that with elm-mdl unfortunately. You can try <#C12KMAYJX|elm-mdl>, too. Perhaps someone on there will have a better answer than me. I'm just not very familiar with it.

U300LJUAK : <@U64FYS317> From what I understand from the docs though, the first argument of Layout.render (the `Mdl` you're trying to pass) should be a message that's local to your app and receives a value of type Material.Msg. See this example:

```
```import Material.Layout as Layout
import Material
```

```
type alias Model =
{ ...
 , mdl : Material.Model -- Boilerplate
}
```

```
type Msg =
...
| Mdl Material.Msg -- Boilerplate
```

...

```
App.program
{ init = (model, Layout.sub0 Mdl)
 , view = view
 , subscriptions = Layout.subs Mdl model
 , update = update
}```
```

U64FYS317 : <@U300LJUAK> Lol I've struggled a lot with this one. Was actually just coming to that conclusion myself (albeit 100x slower than you)

U64FYS317 : I was trying to figure out how they used Mdl as a seeming constructor in the `Layout.render` function

U64FYS317 : I assume creating the ```

```
type Msg =
...
| Mdl (Material.Msg MyMsg)
```
```

actually does let us use the `Mdl` name as a constructor

U64FYS317 : as it's then defined as a component of a union type?

U64FYS317 : and thus fits neatly into the layout.render fn