U0FP80EKB : I generally call it a parameterized type
U6DQCTZT2 : <@U6EAT2Z37> I meant the pattern of using current and other, I know there's ziplists but they are prev, curr, remaining
U6EAT2Z37 : Oh. Sorry I'm not sure about that. I think that that (the thing you're referring to that has list then an element, then a list) is called a Zipper, not a ZipList, by the way. A ZipList is something else.
U3LUC6SNS : I have the code```
renderedContent model =
  Html.div [(HA.property "innerHTML" <| Json.Encode.string "This is a Test!")] []
```
which is called in the (`style-elements`) view by
```
named "content" (html (Debug.log "renderedContent" (renderedContent model)))
```
Oddly, it leads to a stack overflow.  Where should I look to find the trouble?

U6DQCTZT2 : <@U6EAT2Z37> <https://www.youtube.com/watch?v=IcgmSRJHu_8&amp;amp;t=894s> using Zip List here, but he might be wrong.
U6FFMA51S : I'm reducing a commutative function (application order doesn't matter). Should I use `foldl` or `foldr`?
U2J1FUQTZ : How do I insert a `&amp;lt;br&amp;gt;` tag in a text? Something like `text "one&amp;lt;br&amp;gt;two&amp;gt;` does not work
U6FFMA51S : (Beginner myself) but I think maybe something like `Html.div [Html.text "one", <http://Html.br|Html.br>, Html.text "two"]` would work
U6FFMA51S : I don't see `br` defined, so maybe that should be modified to `Html.div [Html.text "one", Html.node "br" [] [], Html.text "two"]`
U6FFMA51S : Scratch that, I found it <http://package.elm-lang.org/packages/elm-lang/html/latest/Html#br>
U6GGSMDFZ : (also beginner) A little more defined that will compile would be ```Html.div [] [ Html.text "one" <http://Html.br|Html.br> [][] Html.text "two"]```
U6GGSMDFZ : or drop the Html bit if you are ```import Html exposing (..)``` (all)
U6GGSMDFZ : <@U6FFMA51S> in your examples the first list passed to div is for is attributes, the second list is it's contents - ie ```div [attrs][contents]```
U3SJEDR96 : <@U6FFMA51S> `foldl`
U3SJEDR96 : <https://ellie-app.com/3SwJD9vjQ59a1/0> &amp;lt;- though you should only bother if you're writing library code; it's unlikely to matter much compared to the overhead of rendering stuff. If it might matter, benchmark. :slightly_smiling_face:
U6FFMA51S : I didn't know about the benchmark package.
I was just asking to develop "good habits"

U3SJEDR96 : :thumbsup:
U6GB56346 : ```neverText : Html Never
neverText = text "never dispatch message"
```

This can be compiled.

```
htmlNever : Html msg -&amp;gt; Html Never
htmlNever elem = elem
```

But this cannot. Both `Html msg` to `Html Never`.
Why?

U2ABT6UKF : I am a beginner. I understand basic programming concepts. I am having trouble finding a resource to learn elm. One that will in the end show practicle examples to use moving forward.
U6FFMA51S : <@U6GB56346> I note that the following also compiles: ```htmlNever : Html msg -> Html Never
htmlNever elem = text "never dispatch message"
```

U6GB56346 : <@U6FFMA51S> Yep. My understanding is that `text` is `Html msg` which contains placeholder type, so it is inferred to `Html Never`. But why `elem` is not inferred to the final type?
```