U29JSAR9S : ah, of course
U23SA861Y : the one thing it does to would be return 0 for an empty list as opposed to say Nothing
U48AEBJQ3 : How about```
maybeList : List (Maybe a) -> Maybe (List a)
maybeList =
    List.foldr (Maybe.map2 (::)) (Just [])
```

U23SA861Y : hmm, yes I think that would do what you want
U29JSAR9S : cheers guys, been interesting working through this with you - my understanding of how to handle Maybe's has definitely come along this evening!
U23SA861Y : no problem, and good luck
U23SA861Y : in your further learnins for great profit
U29JSAR9S : last one for you, for the road:```
maybeTuple3 : ( Maybe a, Maybe b, Maybe c ) -&gt; Maybe ( a, b, c )
maybeTuple3 tuple =
    case tuple of
        ( Just a, Just b, Just c ) -&gt;
            Just ( a, b, c )

        _ -&gt;
            Nothing
```

Trying to figure out how to do this with a Maybe.mapX

U23SA861Y : I think you probably want curry3 or something like that
U23SA861Y : mmm there is only curry and uncurry no uncurry 3
U23SA861Y : but perhaps you could write it```
uncurry3 : (a -> b -> c -> d) -> (a,b,c) -> d
uncurry3 f (a,b,c) = f a b c
```

U23SA861Y : then it would go together something like `maybeTuple3 = uncurry3 (Maybe.map3 (,,))`
U23SA861Y : I think, this is me coding without checking if it compiles
U29JSAR9S : this works :```
maybeTuple3 : ( Maybe a, Maybe b, Maybe c ) -&gt; Maybe ( a, b, c )
maybeTuple3 ( a, b, c ) =
    Maybe.map3 (,,) a b c
```

U29JSAR9S : I didn't realise you could do the (,,) thing
U23SA861Y : well there you go
U29JSAR9S : cheers :slightly_smiling_face:
U29JSAR9S : again
U29JSAR9S : :beers:
U23SA861Y : :beers: :beers:
U5V0HVAKB : hello all
U5V0HVAKB : i am just starting to learn elm
U5V0HVAKB : and i follow the official guide and try to add helpers to my form