U3LUC6SNS : Yes
U3SJEDR96 : can you test `yarn client`?
U3LUC6SNS : ```172-10-18-240:koko_client2 carlson$ yarn client
yarn client v0.27.5
$ webpack-dev-server --port 3000
sh: webpack-dev-server: command not found
error Command failed with exit code 127.
```

U3SJEDR96 : did you run `yarn` to install npm dependencies?
U3LUC6SNS : I don't believe so.  Doing that now  Lots of activity.
U3SJEDR96 : :thumbsup:
U0CLDU8UB : and if that doesn't get you any further, then maybe `yarn global add webpack-dev-server`?
U3LUC6SNS : All working!!! I had to kill some stray `webpack` processes, but then ... YAY!!  Thanks so much. Would never have been able to do this by myself.
U3SJEDR96 : <@U4BJ8UDCP> module names in elm match the filesystem 1-1: `src/Foo/Bar.elm` corresponds to `module Foo.Bar`. Case sensitivity is also important.
U3SJEDR96 : (the above assumes `"source-directories": ["src"]`)
U4BJ8UDCP : Hi there - I'm having some trouble with the module system - The compiler can't find one of my modules. in my ```elm-package.json``` my only source-directory is written as "source".

my folder hierarchy is:
```
/source
    /server
        Main.elm (named "Server.Main")
        Request.elm (named "Server.Request")
```
Server.Main fails to import Server.Request :disappointed:

U3SJEDR96 : Yeah, go for `/Server` instead :slightly_smiling_face:
U4BJ8UDCP : in the package.json?
U1P6FFJ64 : in the folder structure
U4BJ8UDCP : ah
U1P6FFJ64 : and the package too
U3SJEDR96 : the `elm-package.json` doesn't need any changes, but the directory needs to match the module-name, exactly
U4BJ8UDCP : the funny thing is - it was working great on my macbook, but now I'm on my ubuntu machine it isn't working :disappointed:
U1P6FFJ64 : Mac is case insensitive on the folders
U4BJ8UDCP : ooooooooooh :slightly_smiling_face: makes a lot of sense now
U4BJ8UDCP : thanks guys ^_^
U2D5SAEMN : <@U0CLDU8UB> I made `EverySet` today: <https://gist.github.com/leonderijke/18f04f991a5f1945876e285249f5c8ad> For now, only the functions I needed in the project, but is does work. Is this what you had in mind?
U0CLDU8UB : Yep, that's exactly it! :slightly_smiling_face:
U2D5SAEMN : Cool, thanks!
U3LT1UTPF : Hi! I have a type CycleList which contains List Cycle:
U3LT1UTPF : `type alias CycleList =    { cycleList : List Cycle }`
U3LT1UTPF : `type alias Cycle = { cycle : String , productList : List Product }`
U3LT1UTPF : Cycle.cycle is a date in form of a string 'yyymmdd'
U3LT1UTPF : What I want to do is find the latest Cycle (with the greatest string)
U3LT1UTPF : I tried to do it with foldl, but I don't know if it's the way to go
U2D5SAEMN : There's also `List.maximum` you could use to find the maximum element in a list: <http://package.elm-lang.org/packages/elm-lang/core/5.1.1/List#maximum>
U2D5SAEMN : Maybe you can express `yyyymmdd` as an Int instead of a String? That would make the comparison easier, I'd say.
U3SJEDR96 : strings are just as comparable, and `yyyymmdd` especially. As for getting the maximum; it's a little annoying that there is no `List.maximumBy : (a -&gt; comparable) -&gt; List a -&gt; Maybe a`
U3SJEDR96 : but as luck would have it, `List.Extra` has _exactly_ that :stuck_out_tongue:
```

U3SJEDR96 : <http://package.elm-lang.org/packages/elm-community/list-extra/6.1.0/List-Extra#maximumBy>
U3LT1UTPF : Then it would be `List.maximumBy cycleList.cycleList.cycle cycleList` ?
U62PV9CPN : So I have a validation function which checks the length of a String, what type would I use such that I could provide a type `a` which I know supported `length`? So my function could take more than `String` (say `List`, `Set`, etc)
U62PV9CPN : ```isLongerThan : Int -&gt; String -&gt; Bool
isLongerThan minLength subject =
    (String.length subject) &gt; minLength
```

U3SJEDR96 : <@U3LT1UTPF>  close - it would be `List.maximumBy .cycle cycleList.cycleList` - though if you name your type alias "somethingList", it would make senseif it were an alias for a list, not a record. You can use `type alias CycleList = List Cycle` and get rid of the extra step that way :slightly_smiling_face:
U3LT1UTPF : <@U3SJEDR96> Yes, I know. I did it so because it is info from a JSON decoding.
U3SJEDR96 : <@U62PV9CPN> unfortunately, there is no such thing - that would be a typeclass
U3SJEDR96 : <@U3LT1UTPF> Ah, okay. If you want, I'm pretty sure we can make your json-decoder return just that list, rather than a record, tho...
U3SJEDR96 : the nice thing about json decoders is that they can decouple representations
U3LT1UTPF : Ok, let's try
U3LT1UTPF : ```cycleListDecoder : JD.Decoder CycleList
cycleListDecoder =
    succeed
    CycleList
    |: (field "cycleList" (list cycleDecoder))

cycleDecoder : JD.Decoder Cycle
cycleDecoder =
    succeed
        Cycle
        |: (field "cycle" string)
        |: (field "productList" (list productDecoder))
```

U3SJEDR96 : ```cycleListDecoder : JD.Decoder (List Cycle)
cycleListDecoder =
  field "cycleList" (list cycleDecoder)
```

U3LT1UTPF : I don't know how to get that beautiful formatting...
```