U5ABF3BH7 : I have, it was good, I should have asked then.
U5ABF3BH7 : I went to the last 2
U23SA861Y : You could hop on flight up to vancouver, Canada. :stuck_out_tongue:
U40QW928G : oh ok interesting
U5ABF3BH7 : <@U23SA861Y> :slightly_smiling_face:
U5LFUHH19 : As I transition to Elm from Perl, I'm finding my JSON encoding is wacked. From my existing Perl JSON API, what should be Ints have `""`s around them, so they look like strings. Is there a way to decode them with a `String.toInt` kind of thing so I can match the JSON to my (proper) objects?
U0LPMPL2U : Yes but you'll have to worry about `Result` because the string might not be a valid integer
U0LPMPL2U : You can say:
```
stringInt : Decoder (Result String Int)
stringInt =
  JD.map String.toInt JD.string
```

U0LPMPL2U : You could then use `andThen` to check for `Ok` or `Err` and convert that to `JD.succeed` or `JD.fail`
U0LPMPL2U : or you can use `Json.Decode.Extra.fromResult` which does that for you
U0LPMPL2U : e.g.```
stringInt : Decoder Int
stringInt =
  string |&gt; JD.andThen (String.toInt &gt;&gt; JDE.fromResult)
```

U40QW928G : I guess I'm just not understanding where to put messages
U40QW928G : or for that matter how to import them
U40QW928G : I try `import User.Commands exposing (UserMsg)`
U23SA861Y : if you did that your messages would be imported by update but not user
U40QW928G : but when I try to put `FetchUser` in the case it doesn't work
U40QW928G : ```Cannot find pattern `FetchUsers`

26|        FetchUsers -&gt;
           ^^^^^^^^^^
Maybe you want one of the following?

    User.Commands.FetchUsers
```

U5LFUHH19 : I'm being told by `elm-package` that `Json.Decode.Extra` is not compatible with Elm 0.18.
U48AEBJQ3 : <@U5LFUHH19> `JD.string |&gt; JD.map String.toInt |&gt; JD.andThen Json.Decode.Extra.fromResult` I think should do it.
U23SA861Y : <@U40QW928G>, do you got a repo to point at so we can get a better feel?
U40QW928G : I can create one
U0LPMPL2U : Imagine two files```
module User.Commands exposing (fetchUsers)

fetchUsers : (a -&gt; msg) -&gt; Cmd msg
fetchUsers tagger =
  -- fetch users and tag with tagger
```

```
module Main exposing (main)
import User.Commands exposing (fetchUsers)

type Msg = FetchUser | UserFetched User

update : Msg -&gt; Model -&gt; (Model, Cmd Msg)
updage msg model =
  case msg of

```
      FetchUser -&gt;  (model, fetchUsers UserFetched)
      -- rest
```

U0LPMPL2U : <@U5LFUHH19> are you using `elm-community/json-extra` ?
U5LFUHH19 : Ah, no.
U40QW928G : ok I have that
U40QW928G : ```fetchUser : Cmd msgfetchUser =
    Http.send FetchUsers userEndpoint```