U5QJW0DDE : <@U3SJEDR96> inspecting Chrome's network console in the two apps appears to reveal the Elm version is querying for more information on clicks, so I guess that explains it
U5QJW0DDE : <@U3SJEDR96> although, the total amount of data transferred during a session with each app and with the same click flow appears similar, so..
U3SJEDR96 : yeah, by "resources" I mean that each page has an `init` function that returns a task to fetch all the resources like JSON's and whatnot the page needs, so the elm version tries to prevent a flash of partial content by fetching those json's before doing the initial render of the next page, which should a complete render the first time around
U5QJW0DDE : i see
U3SJEDR96 : so rather than rendering (and re-rendering) while data comes in, the elm version does that in two separate steps. Less snappy on fast connections, much nicer on slow connections - seeing a partial page while it's still fetching json data is not particularly nice.
U3SJEDR96 : so rather than optimizing only for fast connections, it tries to give a nice experience across the board
U3SJEDR96 : might not be optimal for some scenario's, but it was a design goal in this implementation :slightly_smiling_face:
U5QJW0DDE : actually the re-frame version has an infinite scroll and continues to load new pages as you scroll down, so it's just a different design decision
U3SJEDR96 : with different trade-offs, indeed. That's kind of the thing, though - both solutions come with trade-offs, but those come as a result of design decisions rather than being anything inherent to the languages
U5QJW0DDE : right, thanks for the explanation
U3LUC6SNS : I just installed elm-tools/parser but get this error message in the repl:```
> import Parser exposing (Parser, (|.), (|=), succeed, symbol, float, ignore, zeroOrMore)
elm-make: Map.!: given key is not an element in the map
```

U3SJEDR96 : might want to `rm -rf elm-stuff &amp;amp;&amp;amp; elm-package install --yes`
U663M2MB7 : I basically aliased that.
U3SJEDR96 : I've used that maybe three times :thinking_face: Installing everything through `elm-package install` rather than manually editing elm-package.json seems to prevent it from happening, most of the time
U663M2MB7 : As someone coming from a backend background, not familiar with anything web. Does this <http://package.elm-lang.org/packages/elm-lang/svg/2.0.0/Svg> expect me to know a lot about how browsers implement Svg?
U663M2MB7 : e.g. I created a line like so `line [ y1 xt, y2 xb, stroke "#FF0000 " ] []` - Curious about what `stroke` means, or where I can find it's definition,
U663M2MB7 : Is it perhaps a CSS property?
U3SJEDR96 : stroke is an svg property, <https://www.w3schools.com/graphics/svg_stroking.asp>
U3SJEDR96 : and yeah, that package does assume knowledge of SVG, though this isn't a browser-specific thing; tools like inkscape or illustrator can also work with that format
U663M2MB7 : Nice, thank you. I felt a bit lost reading the package documentation.
U4J14E7F1 : Does anyone has any experience using Elm with some kind of cms API? I'm wondering how one would build front end in Elm without knowing the texts and titles in advance. Inject it through flags or request from Elm?
U4F64AKQV : Using flags sounds like a reasonable idea.
U4F64AKQV : Or if you have the raw data for the posts stored in some DB, you could potentially make a GET request with Http.
U2AHAPQUV : you can look for examples with firebase or contentful. I just googled on packages, there seems to be a very simple/minimalistic CMS package: <http://package.elm-lang.org/packages/peterszerzo/elm-cms/1.0.3>
U48AEBJQ3 : jxxcarlson: I prefer to write reusable view functions to take as little information as possible. I don't even like to think of it as a parent/child relationship most of the time. Instead, I think in terms of functions which take parameters.
'Container' elements shouldn't know how to construct their children beyond information that only they specifically know. The children should be built at or above the level where the container function is called.

This has a flattening effect on the code where a function is taking data out of its arguments (e.g. the model) and using them to construct HTML elements. We can then rein-in complexity by exporting portions of this into functions.

I end up with functions that are each performing small steps of extracting data from a data structure and passing them onto other functions. Decisions are made at the last point all of the required information to make the decision is available. If something needs information from data structures on multiple levels, the higher data is either passed down via parameters, either directly, or by partially applying it to a function which is being passed.
```

U663M2MB7 : Can we set functions to `undefined` in the same way one can in Haskell, while composing functions?
U3SJEDR96 : `Debug.crash "todo"`
U1922CGQH : In case the docs don't warn enough, I should stress that elm-cms is not quite flushed out on the security side (it requires the auth to happen outside of elm and creds be passed down in flags, and it doesn't react to changes). That said, if you poke around with it, <@U2AHAPQUV>, let me know, I'd love to hear some thoughts on it :).U4J14E7F1
Flags: But wouldnt that make the app crash / not start properly if the data is corrupted (say its a bug in back end) without me getting the chance to handle it properly?

Http-request: one downside that comes to mind is that client has to wait for another request though the data was known already when app was sent. You wouldn't notice on a fast network of course.

Anyway, so there is no common server side way to inject text before app was sent?

U48AEBJQ3 : <@U4J14E7F1> With flags and ports you can accept something as a `Json.Decode.Value` and then decode using `Json.Decode.decodeValue` to avoid crashing and handle errors.
U4J14E7F1 : <@U48AEBJQ3> Of course, thanks!
U5ABF3BH7 : Hello. From my Elm records, I want to talk to the back end (Rails) to create/update objects. Are there any examples on that?
U5ABF3BH7 : I know how to get data with Http.get Http.send, but Post is not clear to me.
U4J14E7F1 : <@U5ABF3BH7>, its almost like get, you just pass the function a body (your parameters) too. Check out http-docs <http://package.elm-lang.org/packages/evancz/elm-http/1.0.0/Http>
U2U94G0QG : <@U5ABF3BH7> take a look at this article, do a search for 'createDocumentCmd : String -> Cmd Msg'<https://css-tricks.com/structure-elm-application/>

U2U94G0QG : there is a nice clean example of building a POST
U5ABF3BH7 : <@U2U94G0QG> Thanks a lot!
U2U94G0QG : no prob! there's also elm-http-builder which people say good things about, but I've never used it so can't offer advice there beyond maybe look into that
U3SJEDR96 : knutandersstokke: Just so you know, that's an outdated package - `elm-lang/http/latest` is going to give better results :wink:
U4J14E7F1 : <@U3SJEDR96> That's right, I always end up on the wrong one :joy: It should have red background or something haha
U0FP80EKB : I really like http builder, although it is important to understand the underlying http module, as well, since it relies on stuff there
U5ABF3BH7 : <@U0FP80EKB> Do you have examples using the Http builder to post things?
U0FP80EKB : There's an example in the README here <https://github.com/lukewestby/elm-http-builder>
U0FP80EKB : The key parts are the `withJsonBody` and `withExpect`, then you do `send` with the appropriate response handler
U5ABF3BH7 : <@U0FP80EKB> Ok, I will look into it. thanks
U3HQVHERX : It has a warning at the top. But it's in a calming blue color...
U3HQVHERX : Also check out <http://package.elm-lang.org/packages/krisajenkins/remotedata/latest> if you want to easily show "loading" states in your view
U61JQ4F4J : Hi! How do you recommend that I parse an arbitrary JSON payload, consisting of heterogeneous values and with undefined keys?
U23SA861Y : if you don't know the data you a receiving, then it is useless to you
U23SA861Y : There are some things you can do, but it is ultimately akin to runtime type reflection.
U64FYS317 : jonf, we can't really assume that. For instance, he could be building a json inspector. However, it would seem that that sort of use isn't really in line with the benefits of elm at all
U23SA861Y : like I said, runtime type inspection
U23SA861Y : when you start to go down that route you are effectively programming in json
U23SA861Y : what are you doing with this json <@U61JQ4F4J> ?
U61JQ4F4J : <@U23SA861Y> <@U64FYS317> it's an API response related to a file system directory listing... with the special case of actual JSON files, those are embedded in the response... for normal files I get its hash as value...
U61JQ4F4J : the JSON files themselves can be anything
U61JQ4F4J : I can't change the API behaviour since it's third party
U61JQ4F4J : I want to display this listing as a file browser list layout
U61JQ4F4J : with the ability to expand the JSON files
U4WH8STNX : I'd suggest using a port, doing this in JS shouldn't be a problem
U23SA861Y : if you don't need them, then just keep them as strings

U61JQ4F4J : <@U4WH8STNX> that's what I thought actually, but didn't look clean to me...
U64FYS317 : hrm. I'm not familiar with elm's reflection module(s) as I'm quite new to elm myself. However, my first (probably naive) thought would be to create a large recursive union type that you could attempt to unmarshal into?
U64FYS317 : <@U61JQ4F4J> probably has the right of it there
U23SA861Y : elm doesn't have a reflection model persay, you would have find a library or write it yourself
U3SJEDR96 : So write a generic JSON decoder?
U4WH8STNX : The thing with the the *clean* part is that this sort of thing isn't really in the domain of Elm as a language - you'd have to work around every security measure in place that is there to protect you from the messy JS world. I would send the stuff through a port and normalize the data into something the Elm type system can handle and then push it back into the Elm world and be happy with it
U23SA861Y : seriously, if they are just reading the generic files just keep them as values and don't parse them at all
U3SJEDR96 : <https://ellie-app.com/3JyyCNNZrQga1/1>
U4WH8STNX : or you could do what <@U23SA861Y> suggests :slightly_smiling_face:
U0RPQMZ9S : I think <@U64FYS317>'s approach, big a big recursive union type representing JSON structure, is the most straightforward way to go
U61JQ4F4J : <@U4WH8STNX> yeah, there's actually no way I could predict the adequate type for the payload values...
U3SJEDR96 : It's not even that big of a datastructure, really. And the `Other` type is just in case someone decides to throw a `function () {}` in there
U61JQ4F4J : the recursive union type sounds like a good option though... I thought they wasn't supported in Elm though but I was just confused with records
U0RPQMZ9S : something like this ```
type JsonValue = Str String | Number Float| Boolean Bool | Arr (List JsonValue) | Object (Dict String JsonValue)
```

U3SJEDR96 : <@U0RPQMZ9S> click on that ellie a few lines up :slightly_smiling_face:
U0RPQMZ9S : I just felt the urge to model the domain so strongly
U64FYS317 : :slightly_smiling_face: i know that feel
U64FYS317 : (I was writing one myself, albeit more slowly)
U3SJEDR96 : It's pretty straightforward to represent generic JSON in Elm, type-safely, and allows you to quite easily make a pretty UI for it, too.
U0RPQMZ9S : this could easily be a pulled out into a library
U3SJEDR96 : So what I'd do, <@U61JQ4F4J>, is make a nice datastructure for your directory tree, `type FSTree = Directory String (List FSTree) | File String JSVal` and have a nice little decoder for it
U4WH8STNX : <@U3SJEDR96>, never thought of that, good point. I wouldn't want to use something like this because it loses all domain meaning. JSON decoders might be tedius but I think this is a plus in the sense that it makes you think about the necessities in API design :slightly_smiling_face:
U3SJEDR96 : Arbitrary JSON *is* the domain. The idea would be to use that _within_ a semantic structure.
```