U4872964V : <@U64MK7215> json request code?
U64MK7215 : Yes. I want the complete elm program
U3SJEDR96 : <http://elm-lang.org/examples/http>
U4872964V : are you talking about the JSONRequest object or just general HTTP requests?
U4872964V : <https://guide.elm-lang.org/architecture/effects/http.html>
U64MK7215 : Json request object. I want an exmaple of an elm program,not just a code snippet

U4872964V : <https://github.com/rtfeldman/elm-spa-example> perhaps?
U4872964V : not sure what you are after actually
U4872964V : almost all Elm programs use http requests
U3SJEDR96 : <@U64MK7215> ^that's an example. Perhaps you could clarify what you're looking for, and why :slightly_smiling_face:
U3LUC6SNS : I get the following structure as the result of parsing:```
> m = run latex "\\emph{foo} " |> latexGet
Macro { name = "emph", args = ["foo"] } : LatexParser.Parser.Latex
```
Here `latex` is a parser and `latexGet` is the code
```
latexGet r =
    r |> Result.withDefault defaultLatex
```
I would like to be able to say `m.name`, `m.args`, but this does not work because of a type problem:
```
> m.name
-- TYPE MISMATCH ----------
`m` does not have a field named `name`.
6| m.name
     ^^^^^^
The type of `m` is:
    Latex
Which does not contain a field named `name`.
```
Makes sense.  But `Latex` is a union type with `Macro` as a member:
```
type Latex
    = Macro Macro_
    | Environment Environment_
    | InlineMath InlineMath_
    | DisplayMath DisplayMath_
    | Words Words_
```
where
```
type alias Macro_ =
    { name : String
    , args : List String
    }
```
What should I be doing?

U4872964V : you should do `case` on your value first, I suppose
U4872964V : you have to handle all types of Latex
U57KYFW67 : Yep. Your `Latex` object may (or may no) be a macro. Maybe it's an `Environment`. If it is, the environment doesn't have a "name"
U3LUC6SNS : OK, that makes sense
U3LUC6SNS : Thanks <@U4872964V> and <@U57KYFW67>
U3LUC6SNS : Oops, a difficulty:  I tried this```
latexGetValue result =
    let
        r =
```

```
        latexGet result
    in
        case r of
            Macro v -&gt;
                v

            Environment v -&gt;
                v
            etc.
```

but it does not work because in for `Macro`, `v` is a `Macro_`, for `Environment`, `v` is an `Environment_`, etc. A case statement requires the values for each alternative to be of the same type.

U3LUC6SNS : I'd like, if possible, to have one function that when applied to a parser result, gives me a `Macro_`, an `Environment_`, etc. as the case may be.
U4872964V : well, that's not how the type system works in Elm. A value can only have one type, if you want to represent multiple types you make a union type, like the `Latex` type
U4872964V : so, instead, you do the stuff you want to do inside the `case` statement
U3LUC6SNS : OK, I'll think about how to proceed -- thanks!
U4872964V : first question, what do you want to do with the value you have?
U3LUC6SNS : My first goal is to write tests to make sure that as I change and add things, I haven't broken what is already built. To do that I have to get into the inner structure of the parse results.
U3LUC6SNS : Eventually I will use parse results to produce html
U4872964V : so, you are writing a test, where you parse a given value and want to check if it parses correctly?
U3LUC6SNS : yes