U0LPMPL2U : `Maybe.map` is convenient to avoid unwrapping/rewrapping
U0LPMPL2U : If you're just overwriting the old value, you can use `Dict.insert`
U17P7CBFB : I've got a `Dict Int {field:List a}` and I want to set a specific position in that list using `List.Extra.setAt`, which returns a maybe, in case the index is outside the list
U23SA861Y : you might just want to use insert
U17P7CBFB : but I need the other fields in that record to stay the same :confused:
U23SA861Y : insert does that
U23SA861Y : oh nvm
U23SA861Y : record
U23SA861Y : yeah
U17P7CBFB : do I let..in all the way to make it readable?
U23SA861Y : ```Dict.update key (Maybe.map (\x -&gt; {x | field=foo}) dict ```
U0LPMPL2U : Why do you want to set a value at an index in a list?
U17P7CBFB : I'm modeling a problem using a graph, and since I need multiple positions on each node where edges can start/end, I need to encode that data in the node somewhere
U17P7CBFB : not particularly happy with the data structure, but it's ok
U17P7CBFB : I know the general solution is to not write code that lends itself well to using lenses, but here I am
U17P7CBFB : `Maybe.map` and then `case`ing on the inner maybe turned out quite readable, actually
U17P7CBFB : I have two options now on the case that never happens: either return `Nothing`, which will silently ignore any error, or `Debug.crash`. What would you do?
U2GPAEU1L : "Show me the data" is the foundation of all science...
I'm not trying to derail anything, a very interesting claim is being made, and I'm simply asking for the evidence. I'm not quite sure why I'm being "criticized" for asking for the data. Imagine, hypothetically, that the reverse is true, and that naming it with "hackX" actually get's _more_ women to show up. In that case, you're spreading a message that is making it _harder_ for women. The point of the data is to figure out what is _actually true_.

I'm not sure how to respond to "I don't feel like i need to consider the p value of their measurements "....

U23SA861Y : I would look at your data structures and figure out why your updates are so twisted
U0LPMPL2U : Can we combine the operations with `Maybe.andThen` ?
U23SA861Y : it kind a has a bit of a code smell here
U17P7CBFB : <@U0LPMPL2U> that's the silencing option
U17P7CBFB : or actually, that's the error propagation option, a la javascript
U17P7CBFB : since it removes the node from the graph
U17P7CBFB : so it'll fail later, somehow
U17P7CBFB : this is the `Nothing` case inside a function that looks like `Dict.update`
U0LPMPL2U : Do the ids in the list need to be sequential? Given that you're trying to get random access, I'm guessing not?
U0LPMPL2U : Could you use a `Dict` instead of a `List`?
U17P7CBFB : nope, but the index matters
U17P7CBFB : I can, but that doesn't solve the problem
U0LPMPL2U : Now you don't have the issue of setting a value out of bounds
U0LPMPL2U : unless the bounds actually mattered for your application
U17P7CBFB : They do :confused:
U0LPMPL2U : what are the limitations?
U17P7CBFB : but it shouldn't be able to set it out of bounds
U17P7CBFB : sec
U0LPMPL2U : not sure if using an enum as the id might help :neutral_face:
U17P7CBFB : for drawing: I need to know how many things there can be in total, the data of each position 0..nfor the model: I need random access, and to not write out of bounds, and to make sure the data written (Id, Slot) corresponds to an Id that is in the graph, and a slot number that is within 0..k for that other item in the graph

U17P7CBFB : the type system isn't going to help me with those invariants, unfortunately
U17P7CBFB : did you mean enum as in a union type?
U0LPMPL2U : yes
U17P7CBFB : Since I've got any number of inputs, dynamically modified, that won't work :confused:
U0LPMPL2U : if you only allowed 1-6, you can do something like:```
type Key = One | Two | Three | Four | Five | Six
```
```

U0LPMPL2U : right
U17P7CBFB : Imagine a box diagram with edges between the boxes, and on each box there are multiple positions where you can start/end the lines you draw between the boxes
U17P7CBFB : That's pretty much exactly what I'm trying to do
U681TBBUP : Is it essentially a directed graph with edges that can be in different states?
U17P7CBFB : drathier/elm-graph only allows one edge between each pair of nodes, so if I have multiple lines between the same two nodes I'll run into problems. Also, I don't know what position on the box that edge is starting/ending
U681TBBUP : Make that a multigraph
U17P7CBFB : sure, but I still need to encode where on the box each of those edges go, and make sure only one edge goes to/from each position, and that there are no self-edges etc.
U17P7CBFB : if the graph library supported subgraphs, I could model each box as a graph itself, which would make the other invariants easier to check, but it's still not quite what I need
U681TBBUP : Can the edges be in arbitrary positions on the boxes or is it discrete?