A tour of Nix

32 / 35 Reimplementation: attrVals

prev next

Write your own implementation of the `attrVals` function.

It consumes a list of `attribute names` and an `attribute set`. It returnes the values of each `attribute name`.

Note: Remember that `attrSet ? "a"` returns `true` and `attrSet ? "j"` => `false`!

Note: Remember that `key = a; attrSet.${a}` returns `1`!

`attrVals` vs `attrValues`:

- `attrVals` -> given a list of names, extract their values from the set and return a list of them

  `attrVals ["a" "b" "c"] attrSet; => should be [1 2 3]`

- `attrValues` -> extract all values in the given set and return a list of them

  `attrValues attrSet; => [1 2 3 4]`

Warning: This is hard!

Note: See video @youtube

```
1  with import <nixpkgs> { };
2  let
3    attrSet = {c = 3; a = 1; b = 2; d=4;};
4
5    #tips: use the map function and access the attribute values
6    attrVals = XXX;
7
8  in
9  rec {
10
11   solution = attrVals ["a" "b" "c"] attrSet; #should be [1 2 3]
12 }
13
```

reset solution run

```
with import <nixpkgs> { };
let
  attrSet = {c = 3; a = 1; b = 2; d=4;};
  attrVals = kys: se: lib.fold (el: c: if se ? "${el}" then [(se.${el})] ++ c else c)
[] kys;

in
rec {

  solution = attrVals ["a" "b" "c"] attrSet; #should be [1 2 3]
}
```