A tour of Nix

22 / 35 inherit/import/with!

prev next

These 3 keywords can be confused pretty easily. Since we are using them a lot in `A tour of Nix` we need to introduce them properly.

With

A with-expression, introduces the `set e1` into the lexical scope of the `expression e2`.

```
let
  as = { x = "foo"; y = "bar"; };
in
  with as; x + y
```

Import

Load, parse and return the Nix expression in the file path. `import` implements Nix's module system: you can put any Nix expression (such as a set or a function) in a separate file, and use it from Nix expressions in other files.

```
rec {
  x = 123;
  y = import ./foo.nix;
}
```

Inherit

The `inherit` keyword causes the specified `attributes` to be bound to whatever attributes with the same name happen to be in scope.

```
let
  x = 123;
in
{
  inherit x;
  y = 456;
}
```

`inherit` takes one or more arguments.

Do this:

- make it work!

Note: See video @youtube

```
1 let
2   myImport = import <nixpkgs> {};
3   x = 123;
4   as = { a = "foo"; b = "bar"; };
5
```

```
 6 in with as; {
 7   inherit x; #example
 8   #fix line below: we want a and b in this scope
 9   inherit X;
10   #also fix this line
11   z = XXX.lib.isBool true;
12 }
13
```

reset solution run

```
let
  myImport = import <nixpkgs> {};
  x = 123;
  as = { a = "foo"; b = "bar"; };

in with as; {
  inherit x;
  inherit a b;
  z = myImport.lib.isBool true;
}
```