A tour of Nix

25 / 35 Assertions

prev next

`Assertions` are generally used to check that certain requirements on or between features and dependencies hold. They look like this:

```
assert e1; e2
```

where `e1` is an expression that should evaluate to a `boolean` value. If it evaluates to `true`, `e2` is returned; otherwise expression evaluation is aborted and a backtrace is printed.

Note: that **->** is the **logical implication** Boolean operation.

See also https://nixos.org/manual/nix/stable/language/constructs#assertions.

Note: See video @youtube

```
1  with import <nixpkgs> {};
2  let
3    func = x: y: assert (x==2) || abort "x has to be 2 or it won't work!"; x +
4    n = "-5"; # only modify this line
5  in
6
7  assert (lib.isInt n) || abort "Type error since supplied argument is no int!
8
9  rec {
10   ex00 = func (n+3) 3;
11 }
```

reset solution run

```
with import <nixpkgs> {};
let
  func = x: y: assert (x==2) || abort "x has to be 2 or it won't work!"; x + y;
  n = -1;
in

assert (lib.isInt n) || abort "Type error since supplied argument is no int!";
assert (lib.isInt n) -> (n > -5);

rec {
  ex00 = func (n+3) 3;
}
```