A tour of Nix

14 / 35 Attribute sets: rec

prev next

`Sets` or `attribute sets` are really **the core of the Nix language**, since ultimately the language is all about creating derivations, which are really just `sets of attributes` to be passed to build scripts.

Sets are just a list of name/value pairs (called attributes) enclosed in curly brackets, where each value is an arbitrary expression terminated by a semicolon. For example:

```
{ x = 123;
  text = "Hello";
  y = f { bla = 456; };
}
```

This defines a set with attributes named x, text, y. The order of the attributes is irrelevant. An attribute name may only occur once.

Now:

- Find out what the `rec` keyword does and what the difference to an `attribute set` without `rec` is.
- Delete the `rec` before the `attribute set` and observe the different behaviour of the nix interpreter.

Hint: You find the answer within the solution section.

Note: That is all for now, but you can continue reading in the nix documentation.

Note: See video @youtube

```
1 rec {
2   x = "a";
3   y = x;
4 }
5
```

reset solution run

```
rec {
  x = "a";
  y = x;
}
#Recursive sets are just normal sets, but the attributes can refer to each other.
#Be aware of infinite recursions. They are not possible!
#rec {
#  x = y;
#  y = x;
#} Does not Work.
```