Script **generated by TTT**

Title: Seidl: Functional Programming and Verification (09.11.2018)

Date: Fri Nov 09 08:30:13 CET 2018

Duration: 90:20 min

Pages: 6

2.2 Expressions

```
# 3+4;;
- : int = 7
# 3+
  4;;
- : int = 7
#
```

→ At #, the interpreter is waiting for input.

→ The ;; causes evaluation of the given input.

→ The result is computed and returned together with its type.

Advantage: Individual functions can be tested without re-compilation !

Another definition of seven does not assign a new value to seven, but creates a new variable with the name seven.

```
# let seven = 42;;
val seven : int = 42
# seven;;
- : int = 42
# let seven = "seven";;
val seven : string = "seven"
```

The old variable is now hidden (but still there)!

Apparently, the new variable may even have a different type.

Simultaneous Definition of Variables

```
# let (x,y) = (3,4.0);;
val x : int = 3
val y : float = 4.0

# let (3,y) = (3,4.0);;
val y : float = 4.0
```

Simultaneous Definition of Variables

```
# let (x,y) = (3,4.0);;
val x : int = 3
val y : float = 4.

# let (3,y) = (3,4.0);;
val y : float = 4.0
```

(1+2)

(1)
unit

Records: Example

```
# type person = {given:string; sur:string; age:int};;
type person = { given : string; sur : string; age : int; }
# let paul = { given="Paul"; sur="Meier"; age=24 };;
val paul : person = {given = "Paul"; sur = "Meier"; age = 24}
# let hans = { sur="kohl"; age=23; given="hans"};;
val hans : person = {given = "hans"; sur = "kohl"; age = 23}
# let hansi = {age=23; sur="kohl"; given="hans"}
val hansi : person = {given = "hans"; sur = "kohl"; age = 23}
# hans=hansi;;
- : bool = true
```

Remark

... Records are tuples with named components whose ordering, therefore, is irrelevant.

... As a new type, a record must be introduced before its use by means of a type declaration.

... Type names and record components start with a small letter.