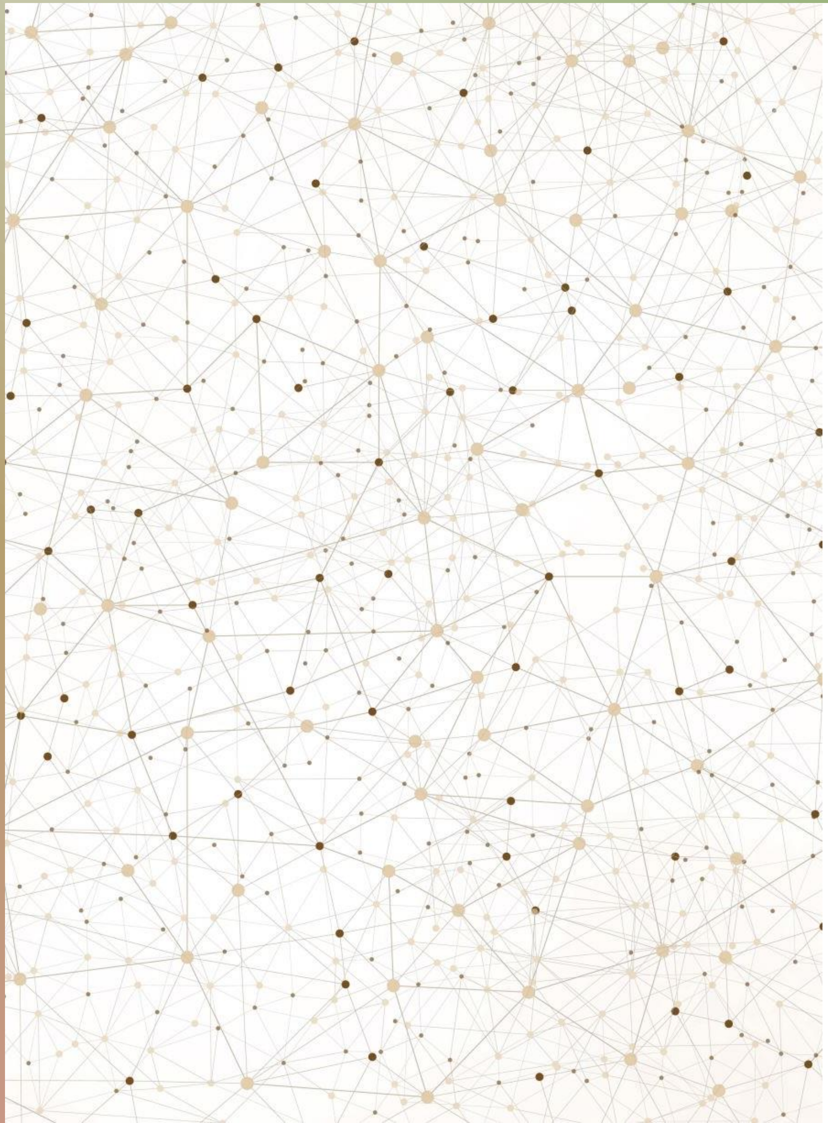


ELM

- Group 4: Rui Miao, Xiaoqian Sun

Name, Binding, Scopes

- Name
  - must begin with a lower-case letter
  - use camelCase, never use snake_case or SCREAMING_SNAKE_CASE
- Binding
  - ++: combines strings, +: addition
  - Map
- Scopes
  - Local scope
  - Global scope

Data Types

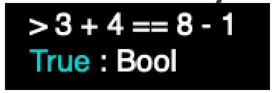
- Primitive Types
  - Number: Int, Float
  - Appendable: String, List a
  - Comparable: Int, Float, Char, String, comparable
  - Compappend: String, List comparable

- Structured Types
  - Tuple, List, Record

- Custom Types

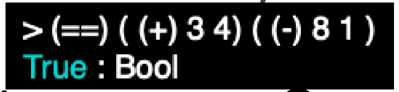
Expressions & Assignment Statements

- Expressions
  - Infix-style expression: first +, second -, third ==



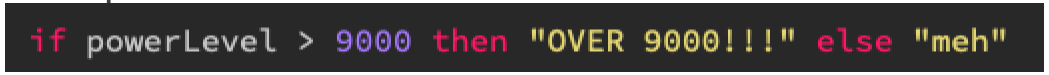
```
> 3 + 4 == 8 - 1
True : Bool
```

  - Prefix-style expression: first ==, second +, third -



```
> (==) ( (+) 3 4) ( (-) 8 1 )
True : Bool
```

- Assignment Statements
  - "=" : assignment
  - if-expression instead of if-statement



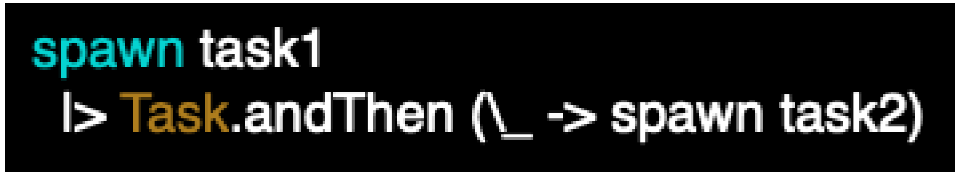
```
if powerLevel > 9000 then "OVER 9000!!!" else "meh"
```

Support OO Programming

- Purely functional

Concurrency

- Processes
  - Spawn



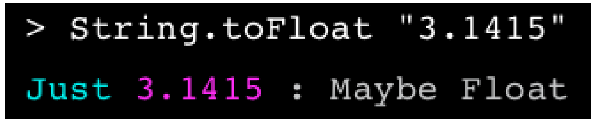
```
spawn task1
  |> Task.andThen (\_ -> spawn task2)
```

- More flexibility in later release

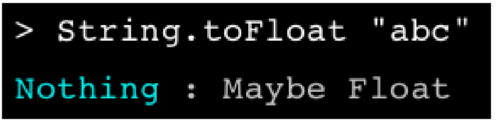
Exception Handling & Event Handling

+ Error handling

  - MayBe

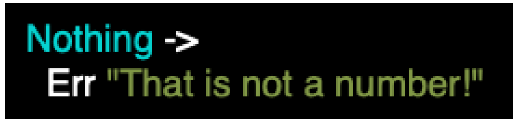


```
> String.toFloat "3.1415"
Just 3.1415 : Maybe Float
```

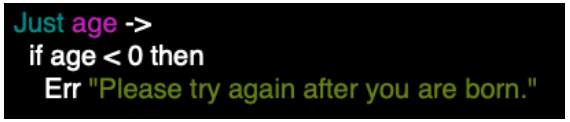


```
> String.toFloat "abc"
Nothing : Maybe Float
```

  - Result



```
Nothing ->
  Err "That is not a number!"
```



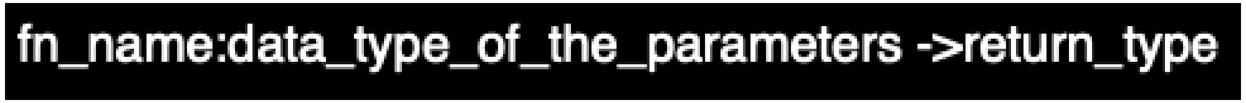
```
Just age ->
  if age < 0 then
    Err "Please try again after you are born."
```

+ Event handling

  - Html.Events

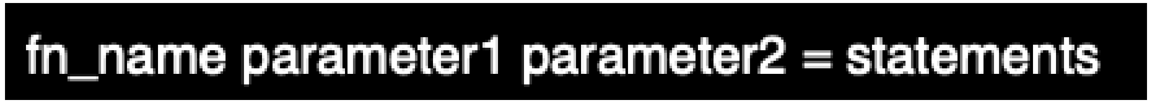
Functional Programming

- Building blocks
- 3 steps:
  - Function Declaration



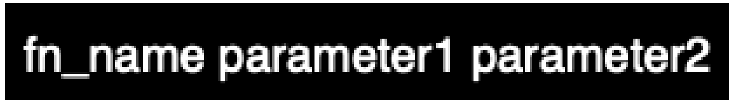
```
fn_name:data_type_of_the_parameters ->return_type
```

  - Function Definition



```
fn_name parameter1 parameter2 = statements
```

  - - Call



```
fn_name parameter1 parameter2
```

Q&A

Thank you.

- Group 4 : Rui Miao, Xiaoqian Sun