TDAH Talks 2016 - E01

Fabricio C Zuardi (@fczuardi)

Assunto: Currying

Currying

Curry

photo: curry-powder-590 by Oregon State University

Curry

photo: MacTutor History of Mathematics archive

Haskell Brook Curry

There are three programming languages named after him, Haskell, Brook and Curry, as well as the concept of currying, a technique used for transforming functions in mathematics and computer science.

source: Haskell_Curry @ Wikipedia

Currying is. . .

> "…the technique of translating the evaluation of a function that takes multiple arguments (or a tuple of arguments) into evaluating a sequence of functions, each with a single argument. It was introduced by Gottlob Frege, developed by Moses Schönfinkel, and further developed by Haskell Curry." -- Wikipedia

source: Currying - Wikipedia

Currying is. . .

> "…the process of transforming a function that takes multiple arguments into a function that takes just a single argument and returns another function if any arguments are still needed." -- Haskell Wiki

source: Currying - Haskell Wiki

. . . a technique for postponing a problem.

"…um jeito de empurrar problemas pra frente."

Curried function

> "A function that will return a new function until it receives all it's arguments" -- Brian Lonsdorf

source: Hey Underscore, You're Doing It Wrong!

Partially-aplied function

> "A function that will return a new function until it receives all it's arguments" -- Brian Lonsdorf

source: Hey Underscore, You're Doing It Wrong!

Example (javascript)

```javascript
const sumAB = function (a, b) {
  return a + b;
}
sumAB(1, 2); // 3
```

Example (javascript)

```javascript
const curriedSumAB = function (a) {
  return function (b) {
    return a + b;
  }
}
curriedSumAB(3)(4); //7
```

Example (javascript)

```javascript
const sumAB = (a, b) => a + b;
sumAB(1, 2); // 3

const curriedSumAB = (a) => (b) => a + b;
curriedSumAB(3)(4); //7
curriedSumAB(1)(1); //2
```

(0)(0)

```
wtf(0)(0)
```

Example step-by-step (javascript)

```
const curriedSumAB = (a) => ((b) => a + b);
const sum3 = curriedSumAB(3);
typeof sum3; // function
sum3; // function (b) { return a + b;  }
sum3(4); // 7
```

curry (Ramda.js)

Returns a curried equivalent of the provided function. The curried function has two unusual capabilities. First, its arguments needn't be provided one at a time. If f is a ternary function and g is R.curry(f), the following are equivalent:

- g(1)(2)(3)
- g(1)(2, 3)
- g(1, 2)(3)
- g(1, 2, 3)

curriedSumAB (Ramda.js)

```
const sumAB = (a, b) => a + b;
const curriedSumAB = curry(sumAB);

curriedSumAB(1, 2)
curriedSumAB(3, 4)
curriedSumAB(3)(4)
```

E pq isso é útil?

Function Composition

Obrigado!