Functional programming (with pytohn)

Bibek Pandey

[PYTHON MEETUP #13]

About me

A khalti Developer[Python/Django/Javascript]

Computer Engineering Graduate

NLP, ML, Mathematics Enthusiast

Python, Javascript, Java, Haskell

Learning Functional Programming

About This presentation

Not about making you functional programmers right away.

A Gentle Introduction to FP.

Introducing you to the flavors of FP that python provides.

An attempt to share what I’ve learned so far in FP.

PROGRAMMING PARADIGM

The way programs are made of.

How the building blocks of program interact.

In FP, program is made of many functions which interact with each other through composition.

Functional Programs: What's So special?

Pure Functions

Immutable data structures

Functions are 'First Class Citizens'

Higher Order Functions

Lazy Evaluation/Objects

Pattern Matching

SOME TERMS

Arity

Number of arguments a function takes.

If 1 argument: 1-Ary

If 2 argument: 2-Ary

And so on..

SIDE EFFECT

Modification of the state of something outside the scope.

EXAMPLES: reading/writing stdin/stdout/network/files/db

Side effects create unnecessary complexity and undesired behaviors hard to debug

Pure functions

Does not have any side effects.

The return value just depends on its arguments.

Does not modify the arguments.

Depends only on the arguments.

Just like mathematical functions like sine, cosine, log, etc.

Anonymous functions

Functions which do not have names

Can be created on the fly without assigning to anything

Called ‘lambda’ expressions in Python

Higher order functions

Take other functions as arguments

Or/And return functions as return values

Make our code concise and composable

COLLECTION TRANSFORMATION TOOLS

Operate on lists, and transfer them to other lists/values

MAP

REDUCE

FILTER

COMPOSITION

Passing the return value of a function to other function

Just like we did FoG, GoF in Mathematics

FoG is G composed with F and equivalent to: F(G(x))

Partial application

Applying a function to only a few of its arguments

Helps to create new useful functions out of existing ones.

Currying

Function(arg1, arg2, ..., argN) -> Function(arg)

It has nothing to do with the curry that we eat/cook.

Named after matematician Haskell Curry

Also useful to create new function out of existing ones.

Recursion

Function calling itself

Needs to have a base condition, because otherwise...

Each recursive call solves smaller problem

Lazy evaluation/Objects

Expressions evaluation only when needed

Allows awesome things like infinite lists

Advantage: Only necessary values are computed

Generators in Python

Why functional programming?

High reusability because no “I wanted a banana but got a Gorilla holding a banana” problem[OOP]

Very easy to test and debug

Makes Parallel Computation less error prone

Talk is cheap. Show me the code.

Now, I will..

```
REDUCE (
    Questions_to_answers,
    Your_questions
)
```

Finally..

```
MAP (
    THANK_YOU_VERY_MUCH,
    EVERYONE_HERE_AND_ORGANIZERS
)
```

[Reach Me: https://bewakes.com , https://github.com/bewakes]