FP Lang. The Documented Story

By Cameron Stocks

Inspiration

I’ve been very motivated recently to write and design a programming language of my own. I do however have a lack of knowledge and am going to watch a couple *tutorials* to assist me in this process.

All information will be added and referenced in the bibliography at the end.

I have liked the idea of *Python* for a long time, yet I personally feel it lacks the performance of a compiled language such as *C/C++* and there are many languages that are newer that are attempting to match the versatility of *Python* with the speed of *C/C++*.

I have decided to begin my own personal adventure on the development of a new language that will initially be an interpreted language and will eventually become a compiled language. I am going to attempt it in multiple languages, yet my end goal is to have a   
stand-alone language with possibly very basic *C/C++* back ends and possible integration with tools such as *NASM, LLVM,* or any of the C compilers such as *GNU-GCC, Clang* or *MinGW*.

I want to make a language with versatile high-level functionality and incredible low-level speeds.

The Name

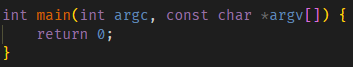
The name of my language will be ‘FP’ also referred to as the *FP Lang*. It will be will completely independent with a compiler and possibly even a linker. I do not know if I want to have tools similar to the *Link.exe* or the *ld.exe* functions as I feel it would be a copy of the *C/C++* language where I intend to re-create it.

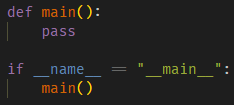
Table of contents

Chapter 0: The ideas and structure of FP Lang.

FP Lang. The newest kid on the block. FP is designed to be fast, accurate and have high performance. FP is going to revolutionize the software development field. It will be   
high-level in structure yet low-level in speed.

What are the main ideas of FP? FP will be a functional programming language; this is where it got its name *FP (Functional Programming)*. I as a developer do love the concepts and capabilities of *OOP (Object Orientated Programming)*, yet I prefer the functional Side of development. There are programs I write where I prefer to design a specific method yet do not want to wrap it within a class. I know there are a lot of functional programming languages, yet it has been a dream of mine to develop my own.

FP lang will be syntactically like *C* yet it will differ.  
Here we have a defined a main method in a *C* program

FP will likewise have a main method as I feel it is important to define an entry point in FP as we don’t want a convention like we do in *Python* such as:  


*Python* does not require a main method, yet current convention is for it to be done like so.

I feel like this could be eliminated entirely with a the implementation of a *main* method as the entry point.