Learn you ALL Programming Languages in 30 minutes!

Which Languages should I use?

I get to asked the same question by a lot of people everyday: Which programming languages should I use? Sometimes I will discuss this same question with my friends or classmates as well. Since there are so many different languages we can choose from. Moreover, there are so many new languages born every year. If you watch some tech news you should know MIT recently designed a new language called Julia focus on data analysis, and it's believed to combine Python, R, and C++.

Understanding the question

I have to say, it is very usual for anyone to ask this question, whether you just begin to coding or even you've been programming for several years

Answer?

So, what's the answer to this question? The answer to this question is the fundamental answer to all engineering-related question, any guess? It depends! It depends on many factors, like how much time do you have, and most importantly: What's your goal?

What's your goal?

Depends on your goal, you may have different options. If you need a long term use language you may choose…, if you would like some new perspective on coding, you may try…, if you need to develop a product on a special platform…

How to

When you make the decision about which language to use, it's time to learn the stuff

How to

But instead of spend several weeks learn one new language, I would like to show you how to learn all the programming languages at once, fast

Language Features vs. Languages

The first thing we need to understand is language features. A language is like a computer brand, while the actual hardware under the hood is what we actually cares, which is all the language features

Language Features

Several language features we need to consider when learning languages are: variables & types, control flow, structure and some other things to consider

Language Comparision

Here I list three language examples doing exactly the same thing, print out 10 HELLO on the screen when the variable x is greater than 3, as we can see, common modern languages have quite similar syntax in central features, like control flow, function definition, or variable definition. It's not hard to imagine, when you learned one language, do some basic programming with other ones shouldn't be too hard.

Special Language Features

The more time-consuming part is to understand the unqiue features in each languages. This is usually the part that makes languages different from each other.

Programming Paradigm

Another hard part is when you learn a language with a different programming paradigm, for example, C uses Procedural Programming, Java uses object-oriented programming while Haskell uses functional programming. I do want to spend some time talk about functional programming, cuz it's been like a fashion in computer science area in recent years, developers want more performance yet the hardware has reached a bottleneck. And OSU does not have courses that involve functional programming.

Functional Programming

The biggest difference about functional programming from other paradigms is declarative programming. In declarative programming you tell the program what something is instead of doing something. This is a Haskell code snippet that prints out the 10th fibonacci number start from 0. As you can see, we tell Haskell the 0th number is 0, the first number is one, the nth number is the n-1 th number plus the n-2 th number

Procedural Programming

In traditional procedural programming, you would tell the things you want to do. Like make it 0 when n is 0, make it one when n is one, give me the result of f(n-1) + f(n-2) otherwise.

Function as variable

In functional programming languages and some modern languages, functions can be used as variables in your program, and you can assign them, call them, this is a pedagogical example since this one is not actually C++ syntax. And many modern languages have integrated something called lambda expression to serve a similar purpose.