



ADDIS ABABA UNIVERSITY

**DEPARTMENT OF CONSTRUCTION
TECHNOLOGY MANAGEMENT**

PROGRAMMING ASSIGNMENT

PREPARED BY :-

ABRHAM G/GIORGIS

ETR/1528/09

Python vs. C#: Comparison of the Programming Languages

Python Programming



Python was released in 1991 by Guido van Rossum. Python was to serve as a successor to the ABC language and is a general-purpose, object oriented programming language. It was developed by an individual and is completely open-source and has been for a while. Python code was also designed for readability, with increased white space and an object-oriented approach.

Python also doesn't have a compile step. It's an interpreted language, so its debugger is built in; with bad code or input causing the interpreter to raise an exception, print a stack trace, and feed a source-level debugger. This makes Python a very good choice for quick testing and debugging. Python is also a dynamically typed language, as well as a scripting language.

C# Programming



C#, pronounced 'C-sharp' (like the musical note), is also a very powerful language. It was developed by Microsoft as a modern alternative to C and C++. Because of that, C# is structurally similar to those languages, while gaining updated features and a more gentle learning curve. C# is a high-level, structured, object-oriented, static language (that is, it's statically typed). It can be compiled on different platforms.

A program coded in C# has a similar basic structure to C++: You get a namespace declaration, a class definition for variables and methods, and then a main method.

Comparing C# vs Python

1. C# vs Python: Speed

When we talk about speed, here, we mean your speed, not the program's speed (we'll get to that in performance). To start, Python was designed to be coded. That might sound odd (as all languages are meant to be coded), but Python really takes the programmer into account. Python has a lot of white space and easy readability.

It also has a much simpler syntax than C#. For example, printing out your name in C# takes around ten lines of code. The same task can be accomplished in two with Python. Also, Python doesn't require you to end every line with a semicolon as C languages do. The differences are mostly small things, but they are a huge help when writing code. On the flip-side, C# is familiar. If you know Java or any other C language then learning C# is only a step to the left. While Python operates on many similar structural principles—Like being object-oriented and a high-level language—the syntax is a lot different than C# or other C languages. If you're not familiar with Python, its syntax might slow you down as you learn it.

2. C# vs Python: Readability

I mentioned before that Python is readable (maybe more than once), but there's a good reason why. C# uses what many languages use to delineate its blocks of code: nested curly braces and brackets. While this kind of code can be made readable, it doesn't have to be. The interpreter doesn't require any indentations. So you could end up with lines and lines of brackets and braces; like a painfully unreadable coding hellscape.

Python, on the other hand, has whitespace built into its DNA. It uses whitespace to delineate blocks of code. This means that instead of a grassy field of curly braces, you either code with neat indents or your code just won't run. So, while both languages can make for neat, readable, code, Python basically forces it. Whether Python is more attractive without the curly brace fringe is up to the reader, but many prefer it over C#.

3. Python vs C#: Performance

When it comes to performance there is a clear distinction between C# and Python. C# is a compiled language and Python is an interpreted one. Python's speed depends heavily on its interpreter; with the main ones being CPython and PyPy. Regardless, C# is much faster in most cases.

For some applications, it can be up to 44 times faster than Python. This is for a number of reasons—from Python's garbage collector to its dictionary lookups. It's also partly due to

C# being a compiled language: it takes a bit more work to write but runs more efficiently because of it.

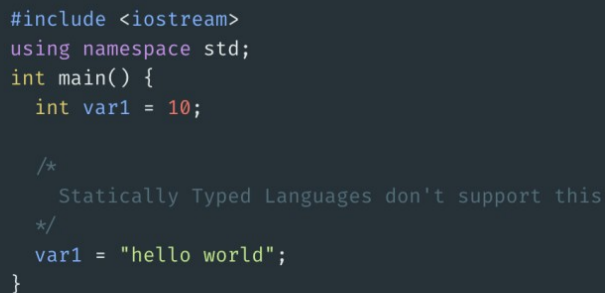
4. Python vs C#: static & dynamic

Python is a more dynamic language than C#. But what does that imply for your project?

When it comes to dynamic languages, the development process is relatively fast and easy. That's why they require the expertise of a team leader, who will oversee the process to make sure that developers build a robust and scalable application.

Since C# is a static language, it includes a **build/compile step**, which some developers aren't fond of. The build process adds a step to the entire web development process and impacts its productivity. But the compiler also detects syntax errors before they become a problem.

Static Typing

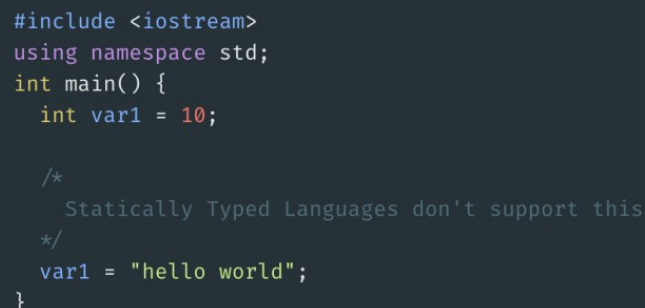


```
#include <iostream>
using namespace std;
int main() {
    int var1 = 10;

    /*
        Statically Typed Languages don't support this
    */
    var1 = "hello world";
}
```

Initially, we declare variable **var1** as an **int** and assign it an integer value. But later, we re-assign it a string. C++ is a statically typed language and doesn't support this behaviour.

Dynamic Typing



```
#include <iostream>
using namespace std;
int main() {
    int var1 = 10;

    /*
        Statically Typed Languages don't support this
    */
    var1 = "hello world";
}
```

Initially, we declare variable **var1** as an **int** and assign it an integer value. But later, we re-assign it a string. C++ is a statically typed language and doesn't support this behaviour.