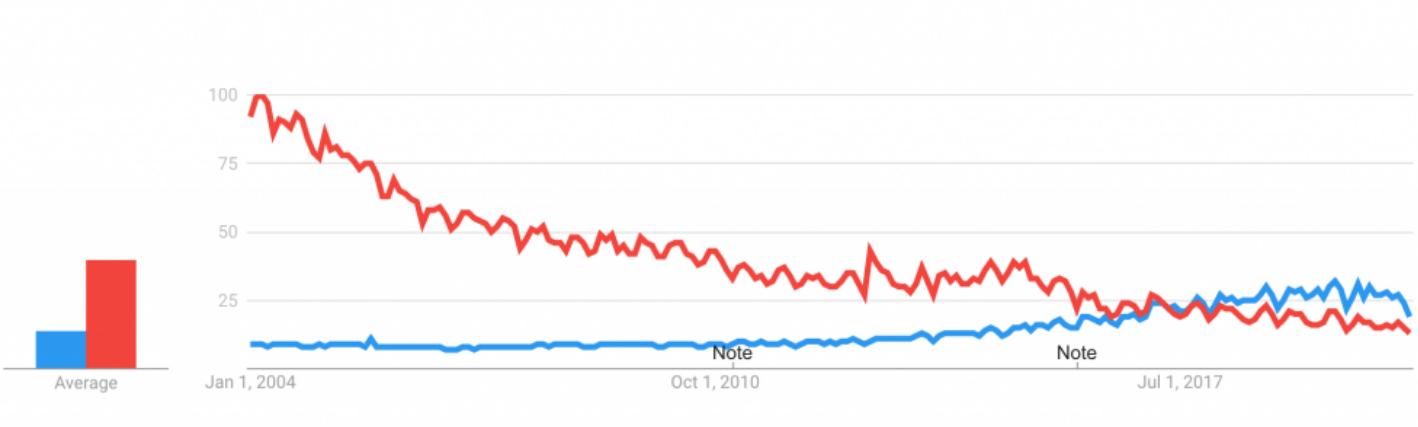
PYTHON VS JAVA: WHAT'S THE DIFFERENCE?

Python has become more popular than Java. Google Trends shows Python's fame rose above Java in 2017:



(*Source*)

The trend is likely caused because of Python's great use for experimentation, and Java's better use for production code. There is more experimentation than production code.

Java is a statically typed and compiled language, and Python is a dynamically typed and interpreted language. This single difference makes Java faster at runtime and easier to debug, but Python is easier to use and easier to read.

Python has gained popularity, in large part, due to its communicativity; people just grasp it easier. With it, the libraries for Python are immense, so a new programmer will not have to start from scratch. Java is old and still widely used, so it also has a lot of libraries and a community for support.

Now, let's take a look at these in depth, including some code examples to illustrate the differences between Python and Java.

Python overview

Python was first released in 1991. It is an interpreted, high-level, general purpose programming language. It is Object-Oriented.

Designed by Guido van Rossum, Python actually has a design philosophy centered around code readability. The Python community will grade each other's code based on how Pythonic the code is.

When to use Python

Python's libraries allow a programmer to get started quickly. Rarely will they need to start from scratch. If a programmer wishes to jump into machine learning, there's a library for that. If they wish to create a pretty chart, there's a library for that. If they wish to have a progress bar shown in their CLI, there's a library for that.

Generally, Python is the Lego of the programming languages; find a box with instructions on how to use it and get to work. There is little that needs to be started from scratch.

Because of its readability, Python is great for:

1. New programmers
2. Getting ideas down fast
3. Sharing code with others

Java overview

Java is old. Java is a general-purpose programming language that utilizes classes and, like Python, is object-oriented.

Java was developed by James Gosling at Sun Microsystems, released in 1995 as a part of Sun Microsystem's Java Platform. Java transformed the web experience from simple text pages to pages with video and animation.

When to use Java

Java is designed to run anywhere. It uses its Java Virtual Machine (JVM) to interpret compiled code.

The JVM acts as its own interpreter and error detector.

With its ties to Sun Microsystems, Java was the most widely used server-side language. Though no longer the case, Java reigned for a long while and garnered a large community, so it continues to have a lot of support.

Programming in Java can be easy because Java has many libraries built on top of it, making it easy to find code already written for a specific purpose.

Who uses Python & Java?

Python is often used with new programmers or junior developers entering a data science role. The big machine learning libraries, TensorFlow and pyTorch, are both written in Python.

Python has excellent data processing libraries with Pandas and Dask, and good data visualization capabilities with packages such as Matplotlib and Seaborn.

Java is used a lot for web development. It is more common among senior-level programmers. It allows for asynchronous programming, and has a decent Natural Language Processing community.

Both languages can be used in API interactions and for machine learning. Java is better developed for building web applications. Python's Flask library is still only able to build the basics to a Python-based UI but is great for creating a Python back-end with an API endpoint.

Python vs Java in code

Let's see how Java and Python work differently.

Syntax

Because Python is an interpreted language, its syntax is more concise than Java, making getting started easier and testing programs on the fly quick and easy. You can enter lines right in the terminal, where Java needs to compile the whole program in order to run.

Type python and then 3+2 and the computer responds with 5.

```
python

3+2
5
```

Consider doing this with Java. Java has no command line interpreter (CLI), so, to print 5 like we did above, we have to write a complete program and then compile it. Here is Print5.java:

```
public class Print5 {

        public static void main(String[] args) {
         System.out.println("3+2=" + (Integer.toString(3+2)));
        }
}
```

To compile it, type javac Print5.java and run it with java Print5.

```
java Print5
3+2=5
```

With Java, we had to make a complete program to print 5. That includes a class and a main function, which tells Java where to start.

We can also have a main function with Python, which you usually do when you want to pass it arguments. It looks like this:

```
def main():
  print('3+2=', 3+2)

if __name__== "__main__":
  main()
```

Classes

Python code runs top to bottom—unless you tell it where to start. But you can also make classes, like possible with Java, like this:

Python Class

```
class Number:
  def __init__(self, left, right):
      self.left = left
      self.right = right

number = Number(3, 2)

print("3+2=", number.left + number.right)
```

The class, **Number,** has two member variables **left** and **right**. The default constructor is __init__. We instantiate the object by calling the constructor **number = Number(3, 2)**. We can then refer to the variables in the class as **number.left** and **number.right**. Referring to variables directly like this is frowned upon in Java. Instead, getter and setter functions are used as shown below.

Here is how you would do that same thing In Java. As you can see it is wordy, which is the main complaint people have with Java. Below we explain some of this code.

Java Class with Getter and Setter functions

```
class PrintNumber {
      int left;
      int right;

      PrintNumber(int left, int right) {
          this.left = left;
          this.right = right;
      }

      public int getleft() {
```

```
            return left;
        }
        public int getRight() {
            return right;
        }
}

public class Print5 {

        public static void main(String[] args) {
            PrintNumber printNumber = new PrintNumber (3,2);
            String sum = Integer.toString(printNumber.getleft()
                  + printNumber.getRight() );
            System.out.println("3+2=" + sum);
        }
}
```

Python is gentle in its treatment of variables. For example, it can print dictionary objects automatically. With Java it is necessary to use a function that specifically prints a dictionary. Python also casts variables of one type to another to make it easy to print strings and integers.

On the other hand, Java has strict type checking. This helps avoid runtime errors. Below we declare an **array of Strings** called **args**.

```
String[] args
```

You usually put each Java class in its own file. But here we put two classes in one file to make compiling and running the code simpler. We have:

```
class PrintNumber {

    int left;
    int right;
}
```

That class has two member variables **left** and **right**. In Python, we did not need to declare them first. We just did that on-the-fly using the **self** object.

In most cases Java variables should be **private**, meaning you cannot refer to them directly outside of the class. Instead you use getter functions to retrieve their value. Like this.

```
public int getleft() {
    return left;
}
```

So, in the **main** function, we instantiate that class and retrieve its values:

```
public int getleft() {
    return left;
}
```

```
public static void main(String[] args) {
    PrintNumber printNumber = new PrintNumber (3,2);
    String sum = Integer.toString(printNumber.getleft()
         + printNumber.getRight() );
}
```

Where Python is gentle in its treatment of variables, Java is not.

For example, we cannot concatenate and print numbers and letters like **"3+2=" + 3 + 2**. So, we have to use the function above to convert each integer to a string **Integer.toString(),** and then print the concatenation of two strings.

Learn both Java & Python

Both programming languages are suitable for many people and have large communities behind them. Learning one does not mean you can't learn the other—many programmers venture into multiple languages. And learning multiple can reinforce the understanding of programming languages altogether.

By many measures, Python is the simpler one to learn, and migrating to Java afterwards is possible.

Related reading

- BMC DevOps Blog
- Using Python for Big Data & Analytics
- Java Developer Roles and Responsibilities
- 10 Must-Read Books for Java Developers
- Top Programming & Software Dev Conferences of 2020
- Java vs Go: What's The Difference?