Java → Python Mental Model Guide

Python is interpreted, not compiled.

- To run a Java program we need a significant amount of boilerplate code, and usually we create a project.
- Creating a Python script can be as simple as entering the code and running it, with no boilerplate.

However, you may want to consider creating and activating a virtual environment, to ensure you're using the correct version of Python with the correct modules.

```
(venv) PS:> python ./hello.py
hello world!
```

Dynamic Typing, Not Static

- In statically-typed languages we declare variables with specific types.
- In Python, we focus on values, not variables: variables don't have fixed types.

```
some_value = 9
print(f"Some value: {some_value}")

some_value = "hello!"
print(f"Some value: {some_value}")
```

“Less Code is Better”

- Boilerplate code is normal in Java
- Python has more of a minimalist philosophy. Even the choice of whether to
- write object-oriented or procedural code is entirely yours.

```
def greet(name):
    print(f"Hello there, {name}")

greet("Bob")
```

Indentation *Is* Structure

- In Java we use {} to define code blocks and indentation to improve readability.
- In Python indentation itself defines code blocks. It's mandatory, not optional.

```python
class Dog:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def bark(self):
        print("Woof!")

def main():
    dog = Dog("Mavi", 2)
    dog.bark()

main()
```

Functions First, Classes Second

- Java wraps everything in a class.
- In Python we only need to reach for classes if they're actually needed.

```python
def get_names():
    names = []
    for i in range(3):
        name = input(f"Enter name {i + 1}: ")
        names.append(name)
    return names

def show_names(names):
    print("\nYou entered:")
    for name in names:
        print(name)

def main():
    collected_names = get_names()
    show_names(collected_names)


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

Python's Standard Libary is Huge

- Python has over 300 modules in its standard library.
- It's easy to install new modules in Python, but you often won't need to.

```python
from http.server import SimpleHTTPRequestHandler, HTTPServer

class Handler(SimpleHTTPRequestHandler):
    def do_GET(self):
        self.send_response(200)
        self.end_headers()
        self.wfile.write(b"Hello from Python's standard library!")

server = HTTPServer(("localhost", 8000), Handler)
print("Serving on http://localhost:8000")
server.serve_forever()
```

Data Structures Are Built-In

- In Java, containers like ArrayList, HashMap, etc. are part of the standard library.
- In Python, the most important containers are part of the language itself.

```
def main():
    items = [1, 2, 3]                          # list
    point = (10, 20)                           # tuple
    colours = {"red", "green", "blue"}         # set
    person = {"name": "Alice", "age": 30}      # dict

    print(items, point, colours, person)

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

“Experimentation Is Normal”

- Of course we can and do experiment in compiled languages like Java, but …
- Python’s lack of boilerplate, compilation or static typing really encourages experimentation.
- No waiting for the build to finish!

Friendly Errors

- Python's errors are designed to be easy to understand.

```python
def greet(name):
    print("Hello, " + name)

greet(123)
```

```
(venv) PS:> python ./error.py
Traceback (most recent call last):
  File "/Users/john/Documents/Courses/Marketing/Python vs Java/./er
ror.py", line 6, in <module>
    greet(123)
    ~~~~~^^^^^
  File "/Users/john/Documents/Courses/Marketing/Python vs Java/./er
ror.py", line 4, in greet
    print("Hello, " + name)
          ~~~~~~~~~~^~~~~~
TypeError: can only concatenate str (not "int") to str
```

Multi-Paradigm Freedom

- Python support procedural, object-oriented, and functional-style programming
- We can write complex structured programs in Python, or simple “scripts”.

```
numbers = [x for x in range(0, 10)]

squares_of_even = list(
    map(lambda x: x * x,
        filter(lambda x: x % 2 == 0, numbers))
)

print("Squares of even numbers:", squares_of_even)
print("Sum:", sum(squares_of_even))
```

Java versus Python Example

- Python is often way shorter and simpler!

App.java

```java
import java.util.ArrayList;

class App {
    public static void main(String[] args) {
        ArrayList<String> names = new ArrayList<>();
        names.add("Alice");
        names.add("Bob");
        names.add("Charlie");

        for (String name : names) {
            System.out.println(name);
        }
    }
}
```

```python
names = ["Alice", "Bob", "Charlie"]

for name in names:
    print(name)
```

Conclusion!

To learn Python fast as a Java developer:

- Embrace simplicity
- Use built-ins
- Think in values, not types
- Write top-down
- >> Experiment often! <<

Don’t forget, the the builtin containers are surprisingly powerful:
You can use list and set comprehensions, generators (the tuple equivalent), etc.