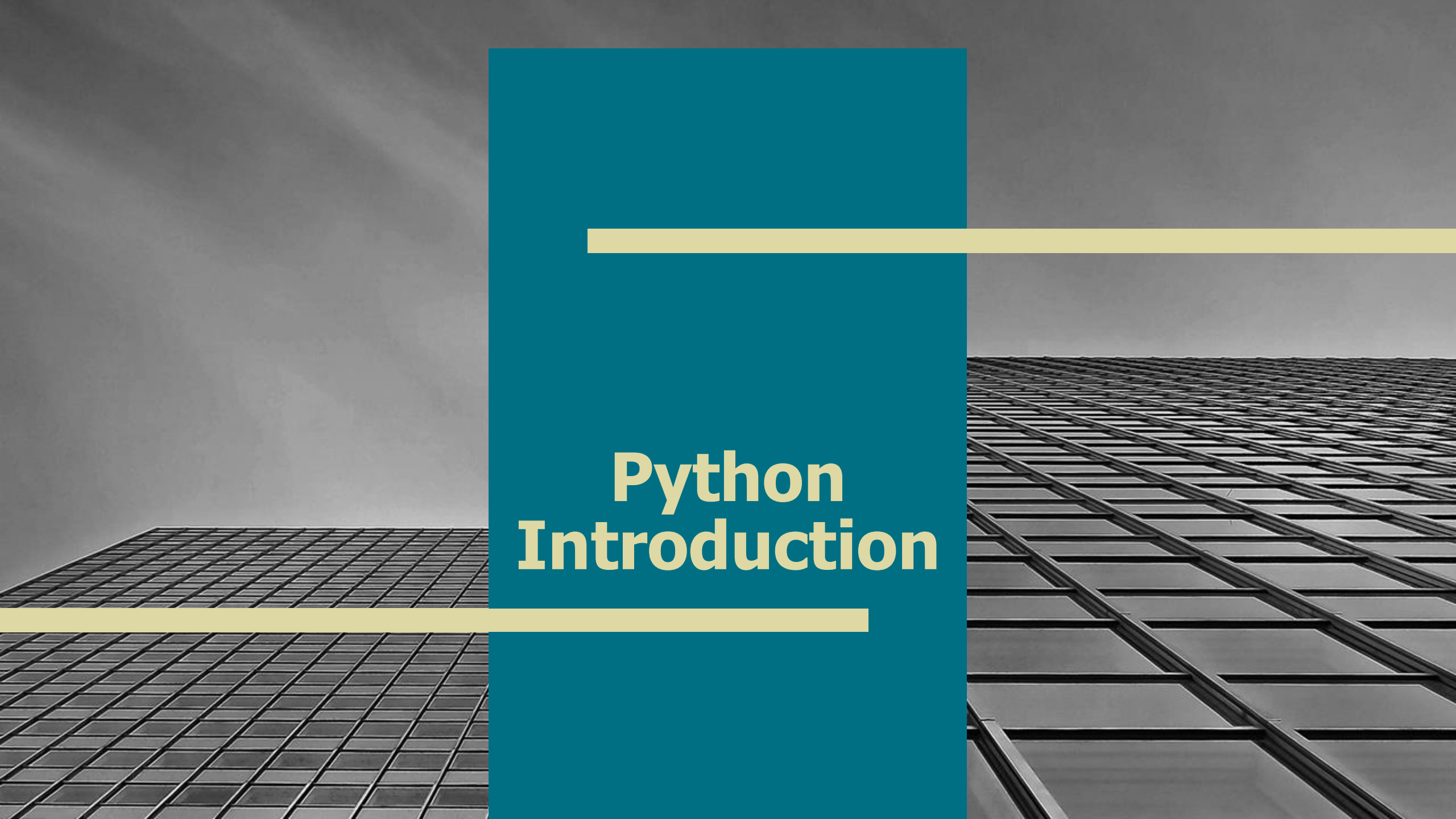


Python Introduction

Python

- Python is an elegant uncomplicated and robust programming language
- It delivers both the power and complexity of traditional compiled languages along with the ease-of-use of simpler scripting and interpreted languages.
- Simple Syntax
- Clear and easy to read
- Powerful programming features
- Youtube, Google, Yahoo, NASA uses Python

History of Python

- Work on Python began in late 1989 by Guido van Rossum, then at CWI (Centrum voor Wiskunde en Informatica, the National Research Institute for Mathematics and Computer Science) in the Netherlands
- Eventually released for public distribution in early 1991
- Van Rossum was a researcher with considerable language design experience with the interpreted language ABC

Features of Python

- **High-level**
  - Machine code → Assembly Lang→ Fortran, C, Pascal → C++,Java → Interpreted Scripting Lang(Tcl, Perl, Python)
  - Each of these languages has higher-level data structures that reduce the "framework" development time that was once required.
  - Python's lists (resizable arrays) and dictionaries (hash tables) are built into the language.
- **Object-oriented**
  - Data and logic are discrete elements of programming.
  - OOP allows for associating specific behaviors, characteristics, and/or capabilities with the data that they execute on or are representative of.
  - Python is not just an OO language like Java or Ruby. It is actually a pleasant mix of multiple programming paradigms.

Features

- **Scalable**
  - Can grow our code from project to project, add other new or existing Python elements, and reuse code at our whim.
  - Python encourages clean code design, high-level structure, and “packaging” of multiple components, all of which deliver the flexibility, consistency, and faster development time required as projects expand in breadth and scope.
  - Python’s pluggable and modular architecture allows the project to flourish as well as maintain manageability.

Features

- **Extensible**
  - Logical organization is possible by separating your code into multiple files, or modules, and be able to access code from one module and attributes from another.
  - Python's syntax for accessing modules is the same for all modules, whether you access one from the Python standard library, one you created just a minute ago, or even an extension you wrote in another language.
  - Python extensions can be written in C and C++ for the standard implementation of Python in C(Cpython).
  - The Java language implementation of Python is called Jython.
  - IronPython, the C# implementation for the .NET or Mono platforms.
- **Portable**
  - Python is written in C, and because of C's portability, Python is available on practically every type of platform that has an ANSI C compiler.
  - Portability applies across multiple architectures as well as operating systems.

Features

- **Easy-to-learn**
  - Python has relatively few keywords, simple structure, and a clearly defined syntax.
- **Easy-to-read**
  - Absent from the Python syntax are the usual mandatory symbols found in other languages for accessing variables, code block definition, and pattern-matching.
  - These include dollar signs ($), semicolons (;), tildes (~), and so on.
  - Python code is fairly understandable even to a reader who has never seen a single line of Python before.
- **Easy-to-maintain**
  - Maintaining source code is part of the software development lifecycle.
  - Much of Python’s success is that source code is fairly easy to maintain, dependent, of course, on size and complexity.

Features

- **Robust**
  - Python provides “safe and sane” exits on errors.
  - The interpreter dumps out a “stack trace” full of useful information such as why your program crashed and where in the code
  - Python’s robustness is beneficial for both the software designer and the user. There is also some accountability when certain errors occur that are not handled properly.
- **Effective as a Rapid Prototyping Tool**
  - Python has so many different interfaces to other systems that it is powerful enough in features and robust enough that entire systems can be prototyped completely in Python.
  - Also, numerous external libraries have already been developed for Python,

Features

- **A Memory Manager**
  - The biggest pitfall with programming in C or C++ is that the responsibility of memory management is in the hands of the developer.
  - memory management is performed by the Python interpreter, the application developer is able to steer clear of memory issues
  - This leads to fewer bugs, a more robust application, and shorter overall development time.
- **Interpreted and (Byte-) Compiled**
  - Python is classified as an interpreted language, meaning that compile-time is no longer a factor during development.
  - Like Java, Python is actually byte-compiled, resulting in an intermediate form closer to machine language.
  - This improves Python's performance, yet allows it to retain all the advantages of interpreted languages.

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

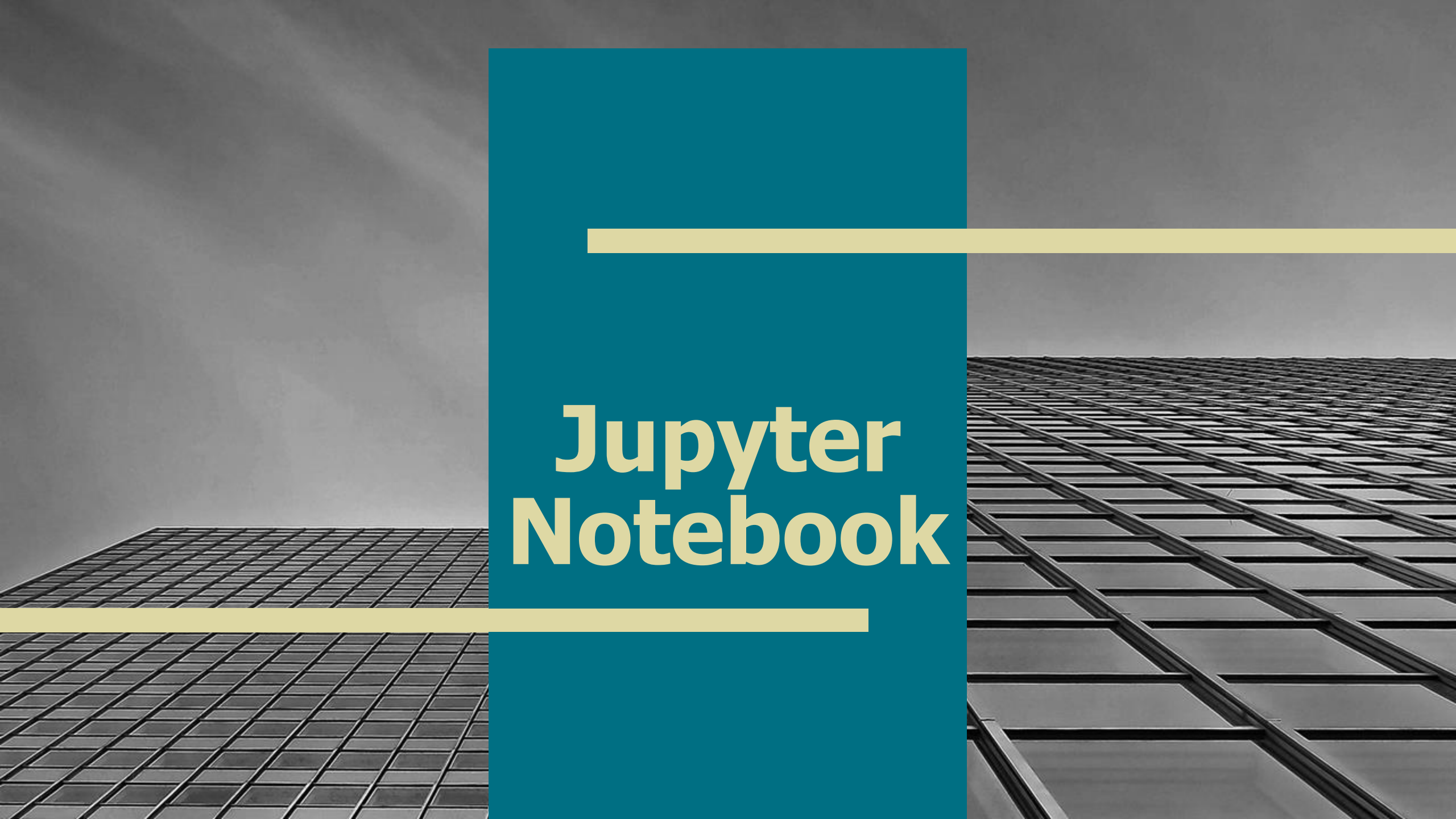
Why Python?

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-orientated way or a functional way.


Running
Python

Running Python

- Three different ways to start Python.
- The simplest way by starting the interpreter interactively, entering one line of Python at a time for execution.
- Another way to start Python is by running a script written in Python. This is accomplished by invoking the interpreter on your script application.
- Finally, you can run from a graphical user interface (GUI) from within an integrated development environment (IDE). IDEs typically feature additional tools such as debuggers and text editors.



Jupyter Notebook

Jupyter Notebook

- The Jupyter Notebook is an incredibly powerful tool for interactively developing and presenting python projects.
- A notebook integrates code and its output into a single document that combines visualizations, narrative text, mathematical equations, and other rich media.
- The intuitive workflow promotes iterative and rapid development, making notebooks an increasingly popular choice at the heart of contemporary data science, analysis, and increasingly science at large.

Jupyter Notebook

- Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself.
- The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R.
- Jupyter ships with the IPython kernel, which allows you to write your programs in Python,

What are notebooks?

A *notebook* combines the functionality of

- a word processor — handles formatted text
- a "shell" or "kernel" — executes statements in a programming language and includes output inline
- a rendering engine — renders HTML in addition to plain text

Online Python Editors

- Python.org
- https://www.python.org/shell/
- Programiz Python Compiler
- https://www.programiz.com/python-programming/online-compiler/
- W3Schools.com
- https://www.w3schools.com/python/python_compiler.asp

References

Wesley J. Chun, “Core Python Applications Programming”, 3rd Edition, Pearson Education, 2016

Thank You