Hello World!

Python is a very simple language, and has a very straightforward syntax. It encourages programmers to program without boilerplate (prepared) code. The simplest directive in Python is the "print" directive - it simply prints out a line.

So lets go through the video 1 – which talks more about creating your First HelloWorld program in Python.

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.

Why Python?

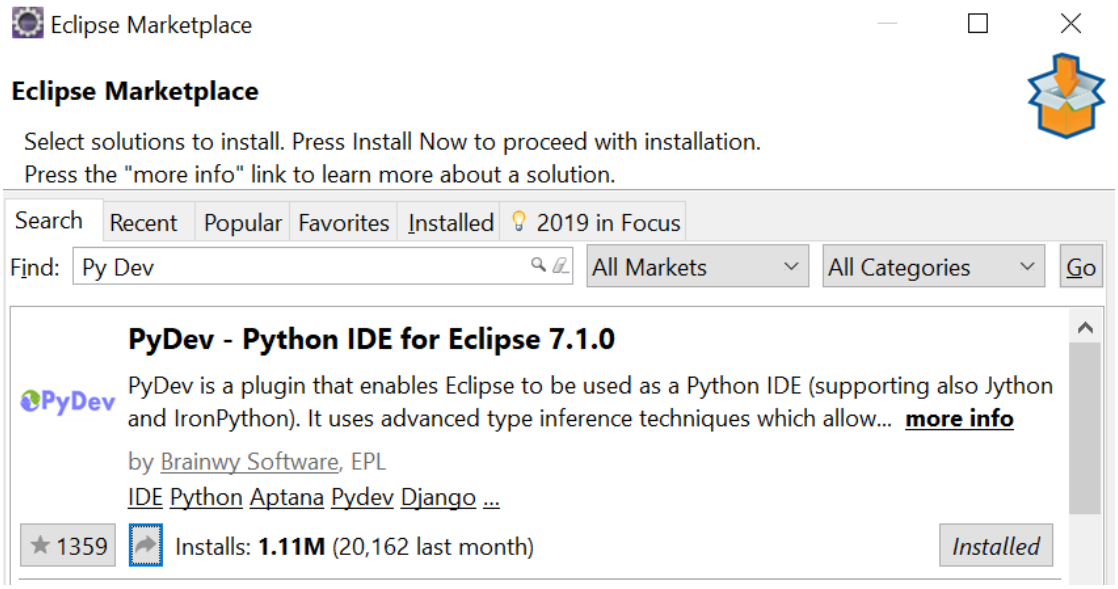
- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This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

Important Points to Remember:

- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

First Step to Start:

1. Download and Unzip Python : https://www.python.org/downloads/
2. Download and Unzip Eclipse: https://www.eclipse.org/downloads/
3. Install Python Development Plugin in Eclipse: http://marketplace.eclipse.org/content/pydev-python-ide-eclipse
4. Download and install PyCharm: https://www.jetbrains.com/pycharm/download/



Indentations

Python uses indentation to indicate a block of code. Python will give you an error if you skip the indentation.

```
if 100 > 20:
    print("100 is Bigger than 20")
```

Comments:

Comments start with a #.

Eg. `# You Cannot skip Indentations`

Docstring:

- Python also has extended documentation capability, called docstrings.
- Docstrings can be one line, or multiline.
- Python uses triple quotes at the beginning and end of the docstring:

```
""" Now this String is Called DocString
    Which can be Multiline also """
```

Variables

Unlike other programming languages, Python has no command for declaring a variable. A variable is created the moment you first assign a value to it.

```
age = 5
name = "John"
print(age)
print(name)
```

Tip: Your variable name indicates your programming experience. Always use elaborated, camelCase variables.

A variable can have a short name (like x and y) or a more descriptive name (age, carName, total_volume). Rules for Python variables:

- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive (age, Age and AGE are three different variables)

There are three numeric types in Python:

- int
- float
- complex

```
x = 1    # int
y = 2.8  # float
z = 1j   # complex
```

- Int, or integer, is a whole number, positive or negative, without decimals, of unlimited length.
- Float, or "floating point number" is a number, positive or negative, containing one or more decimals
- Complex numbers are written with a "j" as the imaginary part.

Casting

There may be times when you want to specify a type on to a variable. This can be done with casting.

Casting in python is therefore done using constructor functions:

- `int()` - constructs an integer number from an integer literal, a float literal (by rounding down to the previous whole number), or a string literal (providing the string represents a whole number)
- `float()` - constructs a float number from an integer literal, a float literal or a string literal (providing the string represents a float or an integer)
- `str()` - constructs a string from a wide variety of data types, including strings, integer literals and float literals

```
x = str("s1") # x will be 's1'

y = int(2.8) # y will be 2

z = float("3")   # z will be 3.0
```

Python Strings

String literals in python are surrounded by either single quotation marks, or double quotation marks. Strings in Python are arrays of bytes representing unicode characters.

`'hello'` is the same as `"hello"`.

```
a = "Hello, World!"

print(a[1])

print(a[2:5])

print(a.strip()) # removes any whitespace from the beginning or the end!"

print(len(a)) #returns the length of a string

print(a.lower()) # returns the string in lower case

print(a.upper()) #returns the string in upper case

print(a.replace("H", "J")) # replaces a string with another string

print(a.split(",")) # returns ['Hello', ' World!']
```

Python allows for command line input.

That means we are able to ask the user for input.

```
print("Enter your name:")
x = input()
print("Hello, " + x)
```

Python Operators

Python divides the operators in the following groups:

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Identity operators
- Membership operators
- Bitwise operator

Arithmetic operators

Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y