What are Built-in Modules in Python?

Built-in modules are standard Python libraries that are automatically installed with Python. You don't need to install them separately — just import and use them.

- These modules provide pre-written functions to perform common tasks like:
  - Math operations (`math`)
  - Working with files (`os`)
  - Date and time (`datetime`)
  - System-related info (`sys`)
  - Platform-specific details (`platform`) ➞ *(explained below)*

You import them using:

```
import module_name
```

`platform` Module – Explained

The **`platform`** module is a built-in module in Python used to access **information about the underlying platform (OS + Python environment)**.

Common Use Cases:

- Identify the OS (Windows, Linux, macOS)
- Get Python version
- Find machine type or processor info
- Useful in cross-platform development

How to use:

```
import platform

print(platform.system())        # 'Windows', 'Linux', or 'Darwin' (for
macOS)
print(platform.release())       # OS release (e.g., '10' for Windows 10)
print(platform.version())       # Detailed version
print(platform.machine())       # e.g., 'x86_64'
print(platform.processor())     # CPU info
print(platform.python_version())# Python version as a string
```

Example Output on Windows:

```
import platform

print("System:", platform.system())
print("Release:", platform.release())
print("Version:", platform.version())
print("Machine:", platform.machine())
print("Processor:", platform.processor())
print("Python Version:", platform.python_version())
```

```
System: Windows
Release: 10
Version: 10.0.19044
Machine: AMD64
Processor: Intel64 Family 6 Model 158 Stepping 10
Python Version: 3.10.0
```

Feature	`platform` module helps you get...
`system()`	OS name (e.g., Windows/Linux/macOS)
`release()`	OS release version
`version()`	OS build version
`machine()`	Machine type (e.g., x86_64)
`processor()`	CPU architecture
`python_version()`	Current Python version

Programs

1. `import module_name`
2. `from module_name import function_name`

Example:

Let's say we create a module file named **`mymath.py`.**

mymath.py (user-defined module)

```
def add(a, b):
    return a + b

def multiply(a, b):
    return a * b

def square(n):
    return n * n
```

---

1. Using `import module_name`

```
import mymath

print("Addition:", mymath.add(5, 3))
print("Multiplication:", mymath.multiply(4, 2))
print("Square:", mymath.square(6))
```

Output:

```
Addition: 8
```

```
Multiplication: 8
Square: 36
```

2.Using `from module_name import function_name`

```
from mymath import add, square

print("Addition:", add(10, 2))
print("Square:", square(5))
```

Output:

```
Addition: 12
Square: 25
```

---

3. Using `from module_name import *`

```
from mymath import *

print("Addition:", add(1, 2))
print("Multiplication:", multiply(2, 3))
print("Square:", square(7))
```

Output:

```
Addition: 3
Multiplication: 6
Square: 49
```

Example 1: String utilities

`string_utils.py`

```
def reverse_string(s):
    return s[::-1]

def capitalize_words(s):
    return ' '.join(word.capitalize() for word in s.split())
```

Usage: `import string_utils`

```
import string_utils

print(string_utils.reverse_string("hello"))
print(string_utils.capitalize_words("welcome to python"))
```

Output:

```
olleh
Welcome To Python
```

Example 2: Temperature conversion

temperature.py

```
def celsius_to_fahrenheit(c):
    return (c * 9/5) + 32

def fahrenheit_to_celsius(f):
    return (f - 32) * 5/9
```

Usage: `from temperature import celsius_to_fahrenheit`

```
from temperature import celsius_to_fahrenheit, fahrenheit_to_celsius

print(celsius_to_fahrenheit(25))  # 77.0
print(fahrenheit_to_celsius(98.6))  # 37.0
```

Output:

```
77.0
37.0
```

Example 3: Area calculations

geometry.py

```
def area_circle(radius):
    return 3.1416 * radius * radius

def area_rectangle(length, width):
    return length * width
```

Usage: `import geometry as geo`

```
import geometry as geo

print("Circle Area:", geo.area_circle(3))
print("Rectangle Area:", geo.area_rectangle(5, 2))
```

Output:

```
Circle Area: 28.2744
Rectangle Area: 10
```

Example 4: Time utilities

time_utils.py

```
def hours_to_minutes(h):
    return h * 60

def minutes_to_seconds(m):
    return m * 60
```

Usage: `from time_utils import *`

```
from time_utils import *

print("Hours to Minutes:", hours_to_minutes(2))
print("Minutes to Seconds:", minutes_to_seconds(30))
```

Output:

```
Hours to Minutes: 120
Minutes to Seconds: 1800
```

Example 5: Math constants and helper

`math_constants.py`

```
PI = 3.1416
E = 2.7183

def circle_circumference(r):
    return 2 * PI * r
```

Usage:

```
from math_constants import PI, circle_circumference

print("Pi value:", PI)
print("Circumference:", circle_circumference(4))
```

Output:

```
Pi value: 3.1416
Circumference: 25.1328
```
