Python Tricks

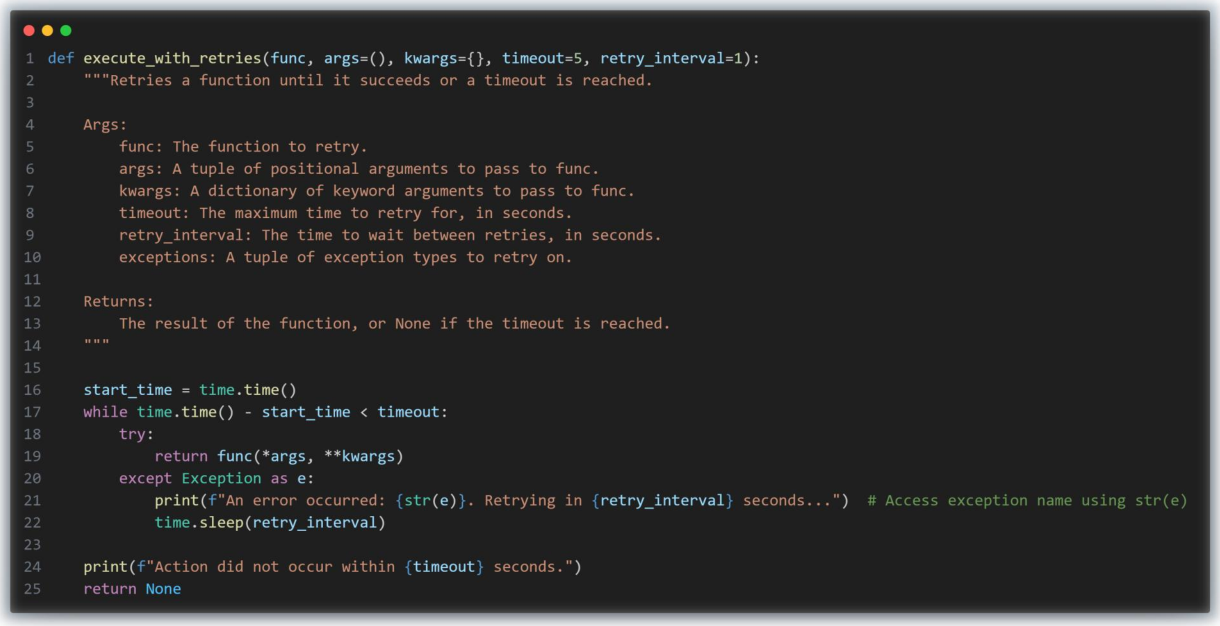
Execute_with_retries() :

- ❖ This function takes a Function and its arguments (positional or keyword) and execute it
- ❖ If it fails , it will try again based on the timeout and retry_interval

- ❖ But the Function must contain something that raise exception

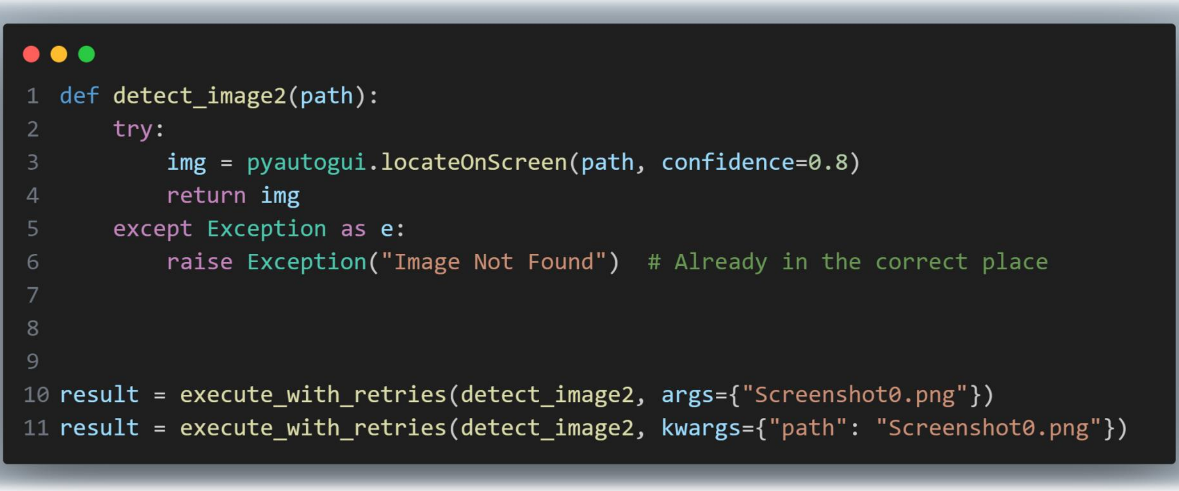
Use case:

- We use it with any function that depend on periodic action that we maybe miss but it will happen again
- We use it with when we work on hardware that resets or need time



```python
def execute_with_retries(func, args=(), kwargs={}, timeout=5, retry_interval=1):
    """Retries a function until it succeeds or a timeout is reached.

    Args:
        func: The function to retry.
        args: A tuple of positional arguments to pass to func.
        kwargs: A dictionary of keyword arguments to pass to func.
        timeout: The maximum time to retry for, in seconds.
        retry_interval: The time to wait between retries, in seconds.
        exceptions: A tuple of exception types to retry on.

    Returns:
        The result of the function, or None if the timeout is reached.
    """

    start_time = time.time()
    while time.time() - start_time < timeout:
        try:
            return func(*args, **kwargs)
        except Exception as e:
            print(f"An error occurred: {str(e)}. Retrying in {retry_interval} seconds...")  # Access exception name using str(e)
            time.sleep(retry_interval)

    print(f"Action did not occur within {timeout} seconds.")
    return None
```

Example :



```python
def detect_image2(path):
    try:
        img = pyautogui.locateOnScreen(path, confidence=0.8)
        return img
    except Exception as e:
        raise Exception("Image Not Found")  # Already in the correct place



result = execute_with_retries(detect_image2, args={"Screenshot0.png"})
result = execute_with_retries(detect_image2, kwargs={"path": "Screenshot0.png"})
```