🎬 Python Guide: Callback Functions

✅ What is a Callback Function?

A **callback function** is a function that you **pass as an argument to another function**,
so that it can be **called (executed) later**, usually **after some action is completed**.

🎯 Common in:

- UI actions (like button clicks)
- Async programming
- Threading
- Event handling

🔧 Step-by-Step Example 1: Simple Callback with `print_status()`

```
def process_data(data, callback):
    print("Processing:", data)
    # Once processing is done, call the callback
    callback()

def print_status():
    print("✅ Data processed successfully!")

process_data("user-info", print_status)
```

🎯 Explanation:

- `process_data()` accepts a function `callback` as a parameter
- After its job is done, it **calls** `callback()`
- `print_status()` is **passed, not executed** (`print_status`, not `print_status()`)

⚙️ Example 2: Callback in UI Simulation (Button Click)

```
def on_button_click(callback):
    print("🖱️ Button clicked")
    callback()

def show_message():
    print("👋 Hello Gowtham, welcome!")

on_button_click(show_message)
```

✅ Whenever the button is “clicked,” `show_message()` is called through the callback.

⚙️ Example 3: Callback in Threading

```
import threading
import time

def long_task(callback):
    print("🚀 Task started...")
    time.sleep(2)
    print("✅ Task finished")
    callback()

def after_done():
    print("📢 Callback: Now update the UI!")

# Running long task in a separate thread
thread = threading.Thread(target=long_task, args=(after_done,))
```

thread.start()

🧠 After the background task is done, it **calls `after_done()` automatically**
🎯 This is how many UI frameworks or APIs **update progress bars or show popups** after completion.

📦 Real-World Use Cases

Use Case	What the Callback Does
Button click	Run a function when user clicks
File upload	Call `on_success()` after upload
Background thread	Call `on_complete()` after processing
API request	Call `on_response(data)` after fetch

🧠 Why Use Callback Functions?

Benefit	Explanation
Decouples logic	Main function doesn't need to know what happens next
Reusable	Can plug different callbacks for different actions
Required in async/threading	Wait → then call callback
Event-driven design	Perfect for UI & real-time systems

⚠️ Common Mistake to Avoid

```
# WRONG: You’re calling the function immediately
process_data("info", print_status())  # ❌ this executes the function, not passes it

# RIGHT
process_data("info", print_status)  # ✅ this passes the function as callback
```

🔁 Callback vs HOF

Feature	Callback Function	Higher-Order Function
Purpose	Execute after something	Compose logic dynamically
Who controls call	Main function	Caller function
Examples	`on_click`, `on_done`, etc	`map()`, `filter()`, decorators

✅ TL;DR in 1 Line:

A **callback** is a function you **pass to another function**, so it can be **called back later** — often after an event or async task.

About the Author

Gowtham SB is a **Data Engineering expert, educator, and content creator** with a passion for **big data technologies, as well as cloud and Gen AI** . With years of experience in the field, he has worked extensively with **cloud platforms, distributed systems, and data pipelines**, helping professionals and aspiring engineers master the art of data engineering.

Beyond his technical expertise, Gowtham is a **renowned mentor and speaker**, sharing his insights through engaging content on **YouTube and LinkedIn**. He has built one of the **largest Tamil Data Engineering communities**, guiding thousands of learners to excel in their careers.

Through his deep industry knowledge and hands-on approach, Gowtham continues to **bridge the gap between learning and real-world implementation**, empowering individuals to build **scalable, high-performance data solutions**.

Socials

🎥**YouTube** - https://www.youtube.com/@dataengineeringvideos

📸**Instagram** - https://instagram.com/dataengineeringtamil

📸**Instagram** - https://instagram.com/thedatatech.in

🤝**Connect for 1:1** - https://topmate.io/dataengineering/

💼**LinkedIn** - https://www.linkedin.com/in/sbgowtham/

🌐**Website** - https://codewithgowtham.blogspot.com

💻**GitHub** - http://github.com/Gowthamdataengineer

Gowtham SB
www.linkedin.com/in/sbgowtham/ Instagram - @dataengineeringtamil

🗨 **Whats App** - https://lnkd.in/g5JrHw8q

📧 **Email** - atozknowledge.com@gmail.com

📱 **All My Socials** - https://lnkd.in/gf8k3aCH