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

👉 `__init__()` – the constructor

This is perfect as **Part 2** in your OOP series. Let's break it down clearly 👇

🧱 Python OOP – Part 2: Understanding `__init__()` Constructor

🧠 What is `__init__()` in Python?

> `__init__()` is a **special method** in Python classes that runs **automatically** when you create an object.
> It is used to **initialize the object's data (attributes)**.

✅ It's also called the **constructor** in Python.

🔧 Why Do We Need `__init__()`?

Without it:

- You'd have to manually set data for every object after creating it.
- Risk of forgetting to set attributes → errors

With `__init__()`:

- Object is created **with values** right away
- Clean, safe, and structured code

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

---

💡 Real-Life Analogy:

> Think of `__init__()` like an **ID card form** you fill when you join a school — You enter name, age, roll number just once — and it's stored in your profile automatically.

🧪 Simple Python Example with `__init__()`:

```
class Student:
    def __init__(self, name, age):   # constructor
        self.name = name             # store data in object
        self.age = age

    def display(self):
        print(f"Name: {self.name}, Age: {self.age}")

s1 = Student("Gowtham", 33)
s1.display()
```

☑ Output:

```
Name: Gowtham, Age: 33
```

🔍 Breakdown Line-by-Line:

Line	Explanation
`def __init__(...)`	Constructor method – runs when object is created
`self.name = name`	Saves the name inside the object

`s1 = Student(...)`	Creates an object and auto-calls `__init__()`
`s1.display()`	Accesses object data using `self`

🗨️ Tamil Explanation Snippet for Video:

"`__init__()` na constructor. Object create pannumbodhu, Python automatic-a idha call pannum. Namma inga `name`, `age` madhiri values kudukkumbodhu, andha object-la store aagidum. Apparam ellaa method-layum `self` use pannitu andha data access pannalaam."

🎯 Why `self` is still needed in `__init__()`?

`self` refers to the object being created.
It lets you store the values inside that specific object — so every object can hold different data.

🧪 Bonus: Create 2 Students

```
s1 = Student("Gowtham", 33)
s2 = Student("Priya", 28)

s1.display()  # Gowtham, 33
s2.display()  # Priya, 28
```

Each object keeps its **own data** — thanks to `__init__()` and `self`.

☑️ With `__init__()` and `self` (Clean OOP style):

```
class Employee:
    def __init__(self, name, aadhaar):
        self.name = name
        self.aadhaar = aadhaar

    def enter_office(self):
        print(f"{self.name} enters using Aadhaar {self.aadhaar}.")

    def open_bank_account(self):
        print(f"Bank account opened for {self.name} with Aadhaar
{self.aadhaar}.")

emp1 = Employee("Gowtham", "1234")
emp1.enter_office()
emp1.open_bank_account() # ☑ No need to pass again
```

❌ When NOT to use `__init__()` in Python?

You don't need `__init__()` when your class:

- Doesn't store any data
- Only contains utility methods (like calculations or tools)
- Is used like a helper or toolbox

☑ Example 1: Class without `__init__()` (utility class)

```
class MathTools:
    def square(self, n):
        return n * n

    def cube(self, n):
```

```
        return n * n * n

tool = MathTools()
print(tool.square(4))  # Output: 16
```

- No object-specific data is stored
- So `__init__()` is **not needed**

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

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

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

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