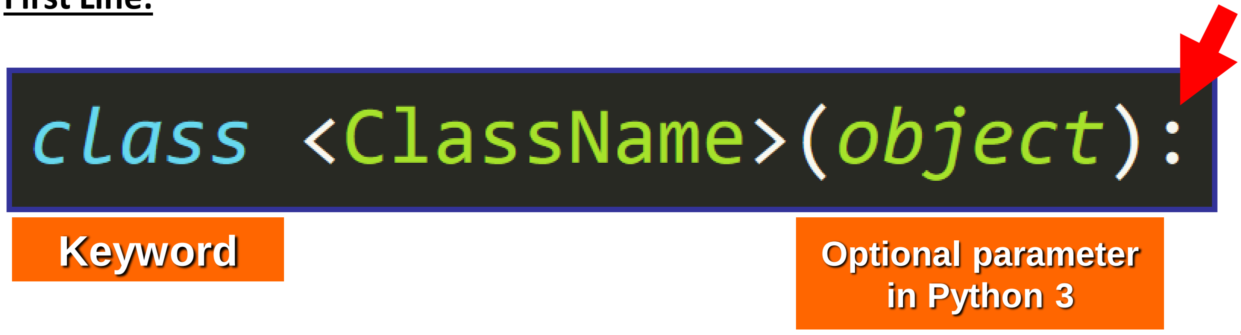
Python OOP: Classes

Introduction to Classes

Key Takeaways

- **Classes**
  - They act like "blueprints" that describe the attributes and functionality of a type of real-world object or abstract concept.
  - They are used to represent real-world objects or entities relevant to the context of a program or system. For example, houses, bank accounts, employees, clients, cars, products.
  - Main Elements:
    - ✓ Class Attributes
    - ✓ Constructor __init__()
    - ✓ Methods
  - Guidelines:
    - ✓ Class names are nouns and they should start with an uppercase letter. For example: **H**ouse, **H**uman, **D**og, **A**ccount.
    - ✓ If the name has more than one word, each word should be capitalized (CamelCase). For example: **S**avings**A**ccount
    - ✓ The body of the class must be indented.
- **First Line:**



```
class <ClassName>(object):
```

Introduction to Classes

Key Takeaways

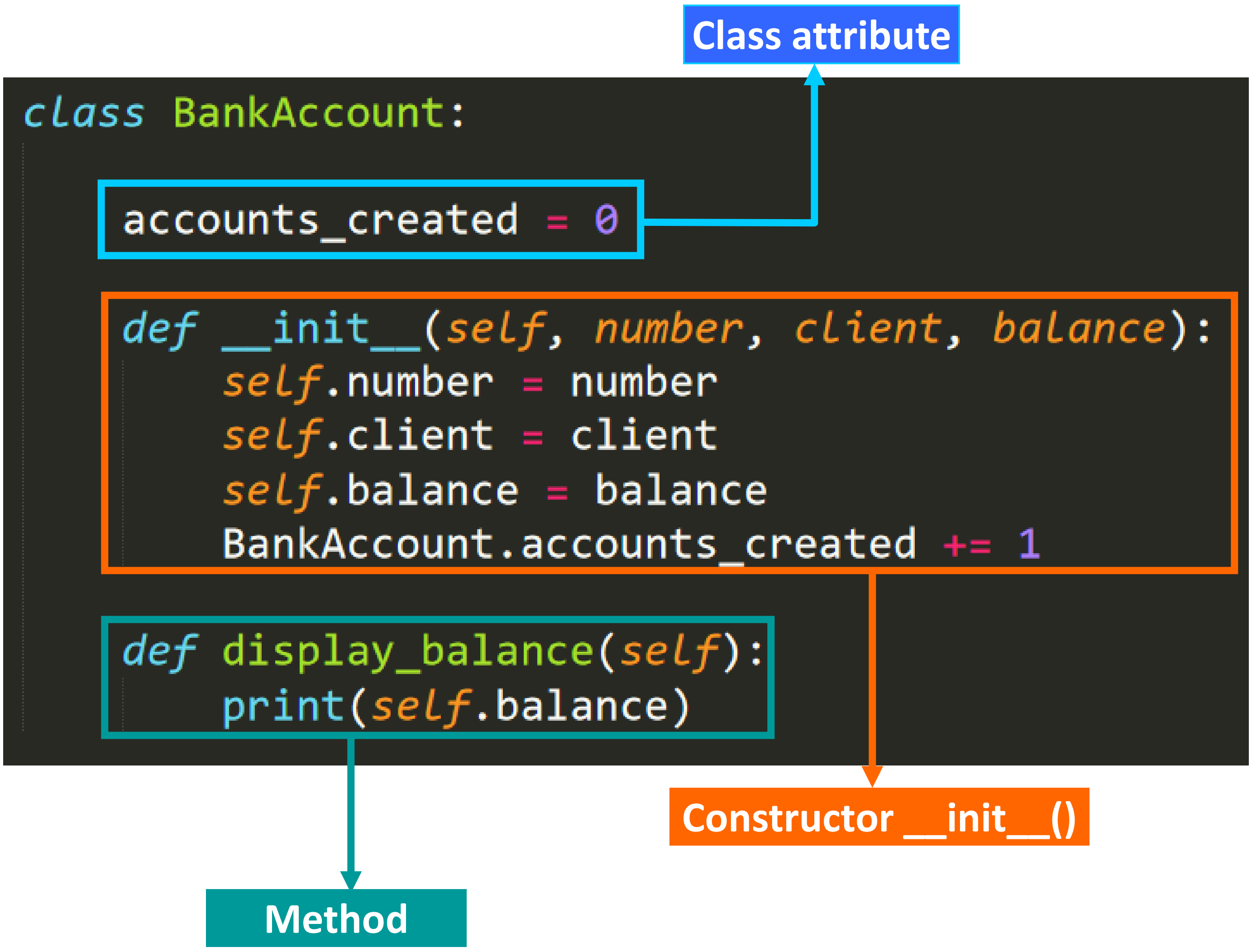
- **General Syntax (Python 3):**

```
class <ClassName>:

    # Class Attributes
    <class_attribute> = <value>

    # Constructor and Instance Attributes
    def __init__(self, <parameters>):
        self.<attr1> = <value1>
        self.<attr2> = <value2>
        ....

    # Methods
    def <method_name>(self, <parameters>):
        # Body
```

Introduction to Classes

Key Takeaways

- **Sample Class:**



```python
class BankAccount:

    accounts_created = 0

    def __init__(self, number, client, balance):
        self.number = number
        self.client = client
        self.balance = balance
        BankAccount.accounts_created += 1

    def display_balance(self):
        print(self.balance)
```