The Syntax and Terminologies

In this lesson, you will learn how to use inheritance syntactically and the terminologies related to it.

WE'LL COVER THE FOLLOWING

The Terminologies

In inheritance, in order to create a new class based on an existing class we use the following terminology:

- **Parent Class (Super Class or Base Class):** This class allows the *re-use* of its **public** properties in another class.
- **Child Class (Sub Class or Derived Class):** This class is the one that *inherits* or *extends* the superclass.

> 💡 A *child* class has **all public** attributes of the *parent* class.

Syntax

In Python, to implement inheritance, the syntax is quite similar to the basic class definition. The syntax is given below:

```
class ParentClass:
    # attributes of the parent class


class ChildClass(ParentClass):
    # attributes of the child class
```

Name of the *parent class* is written in brackets after the name of the *child class* and this is followed by the body of the *child class*.

Example

Let's take an example of a `Vehicle` class as the *parent class* and implement a `Car` class that will extend from this `Vehicle` class. As a *Car **IS A** Vehicle*, hence the implementation of inheritance relation between these classes will stand valid.

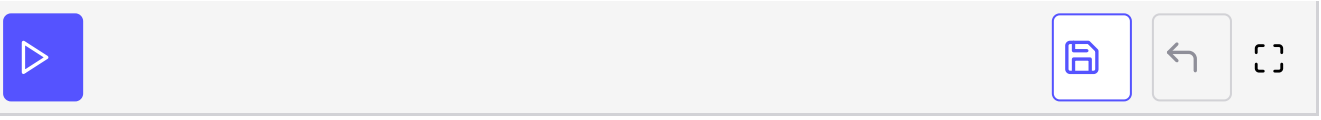
```
class Vehicle:
    def __init__(self, make, color, model):
        self.make = make
        self.color = color
        self.model = model

    def printDetails(self):
        print("Manufacturer:", self.make)
        print("Color:", self.color)
        print("Model:", self.model)


class Car(Vehicle):
    def __init__(self, make, color, model, doors):
        # calling the constructor from parent class
        Vehicle.__init__(self, make, color, model)
        self.doors = doors

    def printCarDetails(self):
        self.printDetails()
        print("Doors:", self.doors)


obj1 = Car("Suzuki", "Grey", "2015", 4)
obj1.printCarDetails()
```



Explanation

- In the code above, we have defined a parent class, `Vehicle`, in **line 1** and a child class, `Car`, in **line 13**.
- `Car` inherits all the properties and methods of the `Vehicle` class and can access and modify them.
- For example in **line 20** of the `Car` class, we have called the

- For example in **line 20** of the `Car` class, we have called the `printDetails()` method, which was actually defined in the `Vehicle` class, in the `printCarDetails()` method.

Before implementing inheritance in depth, let's learn another important concept, `super()`, in the next lesson.