

# OOP - Inheritance

## Introduction

Inheritance is a subtopic in Object Oriented Programming.

Inheritance allows us to define a new class that will inherit all the properties and methods from another class.

It allows us to have a parent - child relationship between classes.

**Parent class** = This is the class that being inherited from, also called the 'Base Class'.

**Child class** = This is the class will inherit properties and methods from a parent class.

**Logics between Parent and Child classes** = In real life definitions we can have a 'Parent' term, for example **vehicle**.

And have several **child** definitions below it - sort of sub-categories, such as 'private car', 'truck' and 'bus'.

(!) All child classes would "absorb" all the methods and properties of the parent class.

## Inheritance of methods & Properties

For example, Let's assume that the parent class would have 1 method, and 1 property, that is relevant for all child classes as well.

- **'engine\_status()' method**

By mentioning the name of the 'Parent' class, in the 'Child' class declaration (inside the round brackets), we gain access to all 'Parent' methods

- **'self.is\_engine\_running' property**

We will add a 'super().\_\_init\_\_(is\_engine\_running)', to the child \_\_init\_\_ method, and by that, we can inherit the property of the parent class

**- Go ahead to the next page for the example -**

Let's see how Inheritance actually looks like, and create a Parent class and also a Child class.

```
class Vehicle:
    def __init__(self, is_engine_running):
        self.is_engine_running = is_engine_running

    def engine_status(self):
        if self.is_engine_running == True:
            print("Engine is running")

        elif self.is_engine_running == False:
            print("Engine is off")

        else:
            print("Wrong value inserted")
```

• In green- variable inheritance flow

Parent class

```
class PrivateCar (Vehicle):
    def __init__(self, is_engine_running):
        self.is_engine_running = is_engine_running
        super().__init__(is_engine_running)

ford_focus = PrivateCar(True)
ford_focus.engine_status()
```

Child class

Inherits all 'Methods' of Vehicle' class.

`super().__init__`, passes the variable to the parent class

'True' variable is passed to the instance. As it is needed for the parent **Vehicle** class

**output:**

Engine is running

## Example explanation

- ❖ PrivateCar class inherits 'method' and a 'property' from 'Vehicle' class. PrivateCar is the child class, while Vehicle class is the parent
- ❖ By mentioning 'Vehicle', in the declaration of 'PrivateCar' class, we achieve that all methods from 'Vehicle' can be used, by a instance of 'PrivateCar'
- ❖ In the parent class we use a variable that is called 'is\_engine\_running', in a method. In order to pass a property from the 'PrivateCar' class, we use the `super().__init__` phrase