

# What is Inheritance?

In this lesson, you will be introduced to Inheritance, a powerful concept in Object-Oriented Programming.

We'll cover the following ^

- Definition
- The IS A Relationship
- The Python Object class

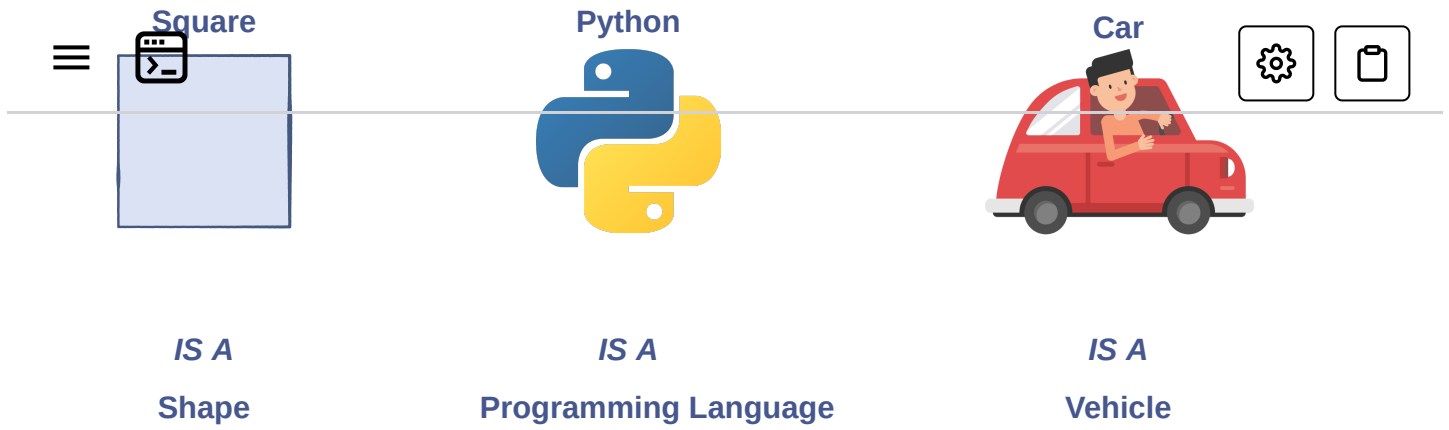
Now that you are familiar with the concepts of *objects* and *classes*, let's discuss **inheritance** which is another key concept in *Object-Oriented Programming*.

## Definition #

**Inheritance** provides a way to create a new class from an existing class. The new class is a specialized version of the existing class such that it inherits all the *non-private* fields (*variables*) and *methods* of the existing class. The existing class is used as a starting point or as a *base* to create the new class.

## The *IS A* Relationship #

After reading the above definition, the next question that comes to your mind is *when do we use inheritance?* Well, the answer is that wherever we come across an **IS A** relationship between objects, we can use inheritance.



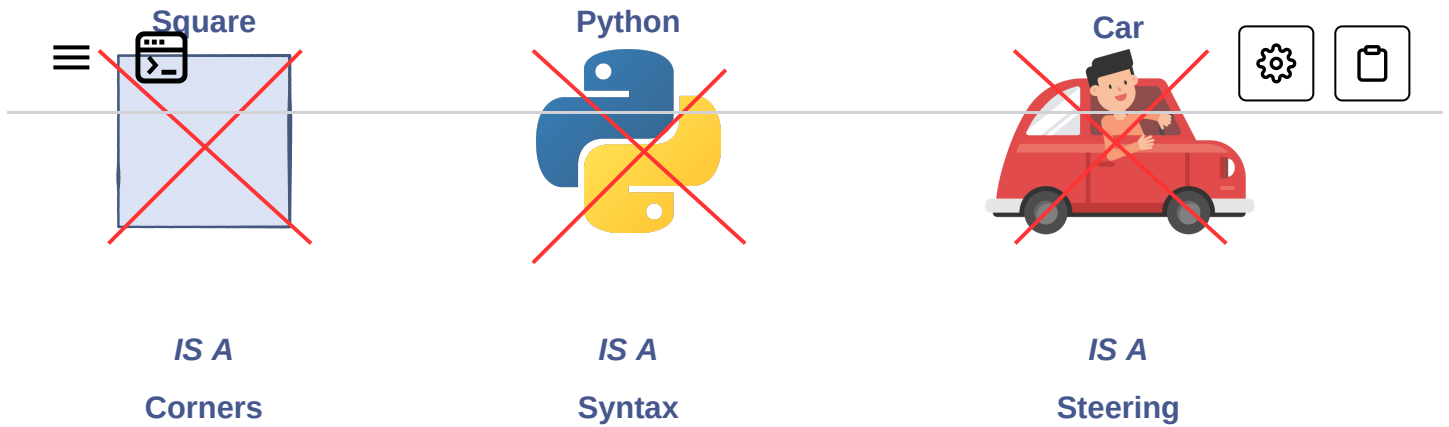
In the above illustration, we can see the objects have a **IS A** relationship between them. We can write it as:

- Square **IS A** shape
- Python **IS A** programming language
- Car **IS A** vehicle

So, from the above descriptions regarding *inheritance*, we can conclude that we can build new classes by extending *existing classes*.

Existing Class	Derived Class
Shape	Square
Programming Language	Python
Vehicle	Car

Let's find out where an **IS A** relationship doesn't exist.



The above illustration shows that we cannot use inheritance whenever an **IS A** relationship doesn't exist between the classes.


## The Python Object class #


The primary purpose of object-oriented programming is to enable a programmer to model the *real-world objects* using a programming language.

In Python, whenever we create a `class`, it is, by default, a subclass of built-in Python `object class`. This makes it an excellent example of inheritance in Python. This class has very few properties and methods but does provide a strong basis for Object-Oriented Programming in Python.

Are things getting interesting? So, let's move to the next lesson in which we will discuss the syntax and terminologies related to inheritance.

[← Back](#)[Next →](#)[Solution Review: Implement the Comp...](#)[The Syntax and Terminologies](#)☒ Completed

 Report an Issue

 Ask a Question  
([https://discuss.educative.io/tag/what-is-inheritance\\_\\_inheritance\\_\\_learn-object-oriented-programming-in-python](https://discuss.educative.io/tag/what-is-inheritance__inheritance__learn-object-oriented-programming-in-python))

