Composition

In this lesson, we'll learn how can we achieve composition in C++.

WE'LL COVER THE FOLLOWING ^

- Example
- Implementation

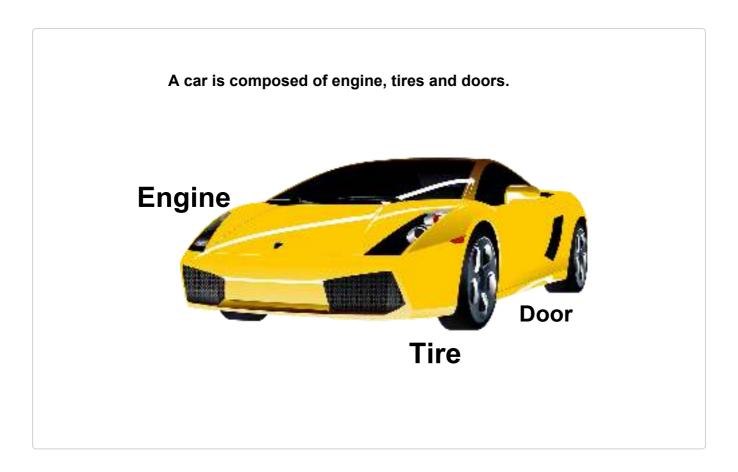
Composition is accessing other classes objects in your class and *owner* class owns the object and is responsible for its lifetime. Composition relationships are **Part-of** relationships where the part must constitute part of the whole object. We can achieve composition by adding smaller parts of other classes to make a complex unit.

So, what makes composition unique?

In composition, the lifetime of the owned object depends on the lifetime of the owner.

Example

A car is composed of an *engine*, *tires*, and *doors*. In this case, a Car owned these objects so a Car is an *Owner* class and tires, doors and engine classes are *Owned* classes.

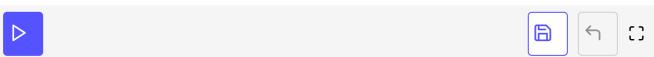


Implementation

Let's look at the implementation of Car class for better understanding:

```
#include <iostream>
using namespace std;
class Engine{
  int capacity;
  public:
  Engine(){
    capacity = 0;
  Engine(int cap) {
    capacity = cap;
  }
  void Engine_details() {
    cout << "Engine details: " << capacity << endl;</pre>
  }
};
class Tires{
  int No_of_tires;
  public:
  Tires(){
    No_of_tires = 0;
```

```
Tires(int nt) {
    No_of_tires = nt;
 void Tire_details() {
   cout << "Number of tyres: " << No_of_tires << endl;</pre>
};
class Doors{
  int No_of_doors;
  public:
 Doors(){
   No_of_doors = 0;
 Doors(int nod) {
   No_of_doors = nod;
 void Door_details() {
   cout << "Number of Doors: " << No_of_doors << endl;</pre>
};
class Car{
  Engine Eobj;
  Tires Tobj;
 Doors Dobj;
  string color;
  public:
  Car(Engine eng, Tires tr, int dr, string col)
    : Eobj(eng), Tobj(tr), Dobj(dr){
   color = col;
  }
  void Car_detail(){
    Eobj.Engine_details();
   Tobj.Tire_details();
   Dobj.Door_details();
    cout << "Car color: " << color << endl;</pre>
 }
};
int main(){
  Engine Eobj(1600);
  Tires Tobj(4);
  Doors Dobj(4);
  Car Cobj(Eobj, Tobj, 4, "Black");
  Cobj.Car_detail();
}
```



Doors classes. Car class owns the objects and is responsible for their lifetime. When Car dies, so does *tire*, *engine* and *doors* too.

In the next lesson, we'll learn about **Aggregation**, the very important concept in C++.