Types of Inheritance

In this lesson, we'll learn about the types of inheritance which includes multiple inheritance and multilevel inheritance.

WE'LL COVER THE FOLLOWING

- Multiple Inheritance
  - Example
  - Implementation
- Multilevel Inheritance
  - Example
  - Implementation

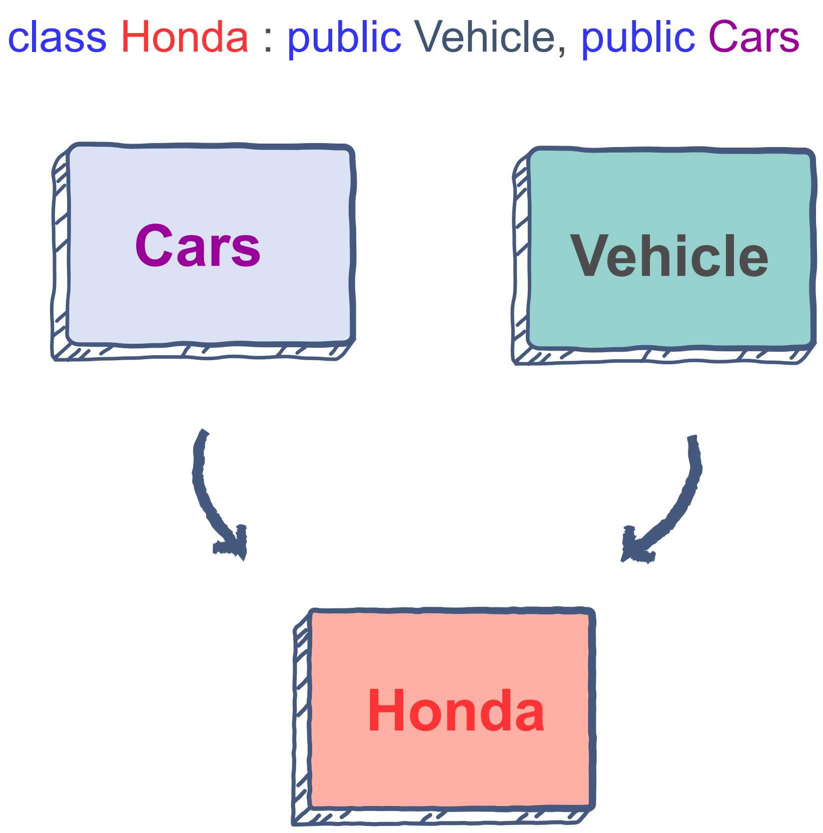
Multiple Inheritance

We can inherit the base class attributes to the derived class if we want derived class to have access data members and member functions of the base class. But to inherit multiple classes data members and member functions to the derived, the concept of *multiple inheritance* comes in. We can inherit multiple classes as base classes separated by `,`

```
class Derived : public Base1 , public Base2 , ...
```

Example

Let's take the example of `Vehicle` and `Cars` classes which acts as the base classes of the `Honda` class:



Multiple Inheritance

Implementation

Implementation of the `Honda` class is given below:

```
class Vehicle{
  protected:
  string Make;
  string Color;
  int Year;
  string Model;

  public:
  Vehicle(){
    Make = "";
    Color = "";
    Year = 0;
    Model = "";
  }

  Vehicle(string mk, string col, int yr, string mdl){
    Make = mk;
    Color = col;
    Year = yr;
    Model = mdl;
  }

  void print_details(){
    cout << "Manufacturer: " << Make << endl;
    cout << "Manufacturer: " << Make << endl;
    cout << "Color: " << Color << endl;
    cout << "Year: " << Year << endl;
    cout << "Model: " << Model << endl;
  }
};

class Cars{
  string trunk_size;

  public:
  Cars(){
    trunk_size = "";
  }

  Cars(string ts){
    trunk_size = ts;
  }

  void car_details(){
    cout << "Trunk size: " << trunk_size << endl;
  }
};

class Honda: public Vehicle, public Cars{
  int top_speed;

  public:
  Honda(){
    top_speed = 0;
  }

  Honda(string mk, string col, int yr, string mdl, string na, int ts)
  :Vehicle(mk, col, yr, mdl), Cars(na){
    top_speed = ts;
  }

  void Honda_details(){
    print_details();
    car_details();
    cout << "Top speed of the car: " << top_speed << endl;
  }
};

int main(){
  Honda car("Honda", "Black", 2006, "Accord", "14.7 cubic feet", 260);
  car.Honda_details();
}
```

Now, the `Honda` class object has access to all member functions of `Cars` and `Vehicle` classes as they're now base classes of `Honda` class. The highlighted lines in the code indicate how multiple inheritance is achieved.

Multilevel Inheritance

If we want to inherit data members and member functions of the base class which is already inherited from another class, the concept of `multilevel inheritance` comes in. This contains a more hierarchical approach.

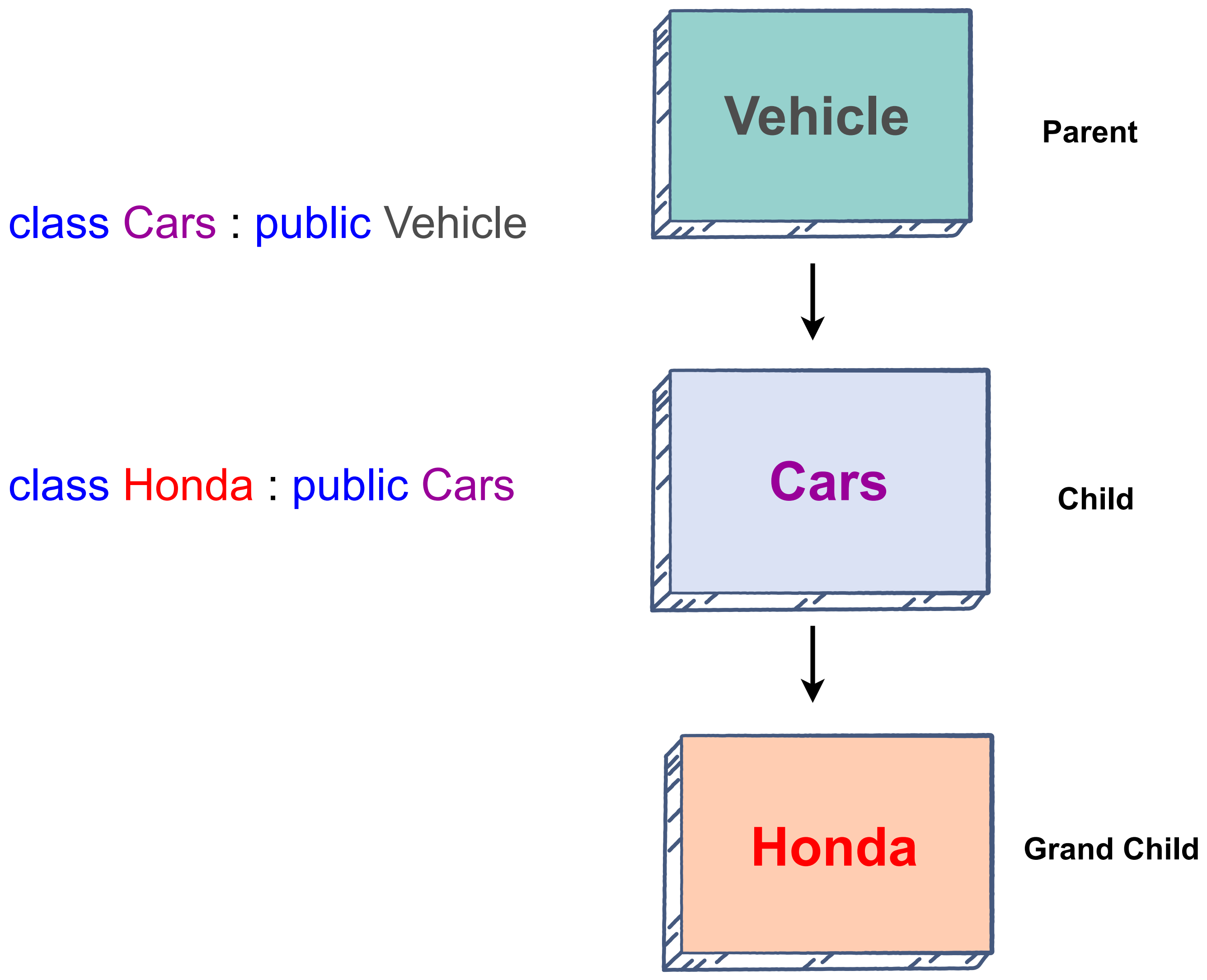
```
class parent

class child : public parent

class grandChild : public child
```

Example

Let’s take the example of `Vehicle` class which acts as a parent to `Cars` class. Now `Cars` class acts as a parent to `Honda` class.



Multilevel Inheritance

Implementation

Implementation of the `Honda` class is given below:

```
class Vehicle {
  protected:
  string Make;
  string Color;
  int Year;
  string Model;

  public:
  Vehicle(){
    Make = "";
    Color = "";
    Year = 0;
    Model = "";
  }

  Vehicle(string mk, string col, int yr, string mdl){
    Make = mk;
    Color = col;
    Year = yr;
    Model = mdl;
  }

  void print_details(){
    cout << "Manufacturer: " << Make << endl;
    cout << "Color: " << Color << endl;
    cout << "Year: " << Year << endl;
    cout << "Model: " << Model << endl;
  }
};

class Cars: public Vehicle{
  string trunk_size;

  public:
  Cars(){
    trunk_size = "";
  }

  Cars(string mk, string col, int yr, string mdl, string ts)
    :Vehicle(mk, col, yr, mdl){
      trunk_size = ts;
    }

  void car_details(){
    cout << "Trunk size: " << trunk_size << endl;
  }
};

class Honda: public Cars{
  int top_speed;

  public:
  Honda(){
    top_speed = 0;
  }

  Honda(string mk, string col, int yr, string mdl, string na, int ts)
    :Cars(mk, col, yr, mdl, na){
      top_speed = ts;
    }

  void Honda_details(){
    print_details();
    car_details();
    cout << "Top speed of the car: " << top_speed << endl;
  }
};

int main(){
  Honda car("Honda", "Black", 2006, "Accord", "14.7 cubic feet", 260);
  car.Honda_details();
}
```

Now, `Honda` class object has access to all member functions of `Cars` class and the `Cars` class has access to all members functions of the `Vehicle` class as they're now base classes of `Honda` class. The highlighted lines in the code indicate how multilevel inheritance is achieved.

In the next lesson, we'll learn about the advantages of inheritance.