C++ Inheritance

In C++, inheritance is a process in which one object acquires all the properties and behaviors of its parent object automatically. In such way, you can reuse, extend or modify the attributes and behaviors which are defined in other class.

In C++, the class which inherits the members of another class is called derived class and the class whose members are inherited is called base class. The derived class is the specialized class for the base class.

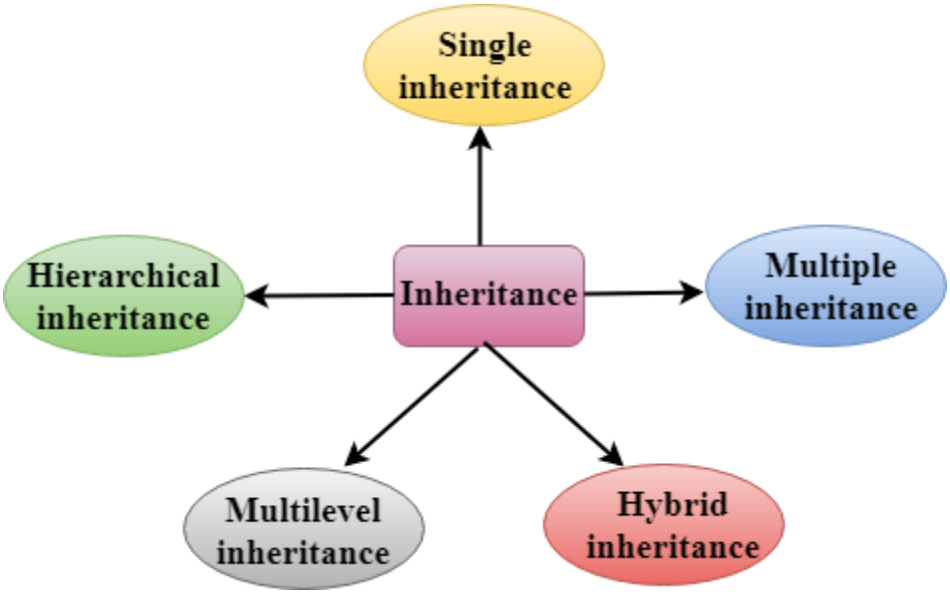
Advantage of C++ Inheritance

Code reusability: Now you can reuse the members of your parent class. So, there is no need to define the member again. So less code is required in the class.

Types Of Inheritance

C++ supports five types of inheritance:

- Single inheritance
- Multiple inheritance
- Hierarchical inheritance
- Multilevel inheritance
- Hybrid inheritance



Derived Classes

A Derived class is defined as the class derived from the base class.

The Syntax of Derived class:

```
class derived_class_name :: visibility-mode base_class_name
```

```
{
   // body of the derived class.
}
```

Where,

derived_class_name: It is the name of the derived class.

visibility mode: The visibility mode specifies whether the features of the base class are publicly inherited or privately inherited. It can be public or private.

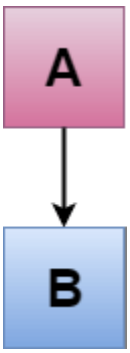
base_class_name: It is the name of the base class.

Note:

- In C++, the default mode of visibility is private.
- The private members of the base class are never inherited.

C++ Single Inheritance

Single inheritance is defined as the inheritance in which a derived class is inherited from the only one base class.



Where 'A' is the base class, and 'B' is the derived class.

C++ Single Level Inheritance Example: Inheriting Fields

When one class inherits another class, it is known as single level inheritance. Let's see the example of single level inheritance which inherits the fields only.

```
#include <iostream>
using namespace std;
 class Account {
   public:
   float salary = 60000;
 };
   class Programmer: public Account
 {
```

```
  public:
  float bonus = 5000;
  };
int main(void) {
    Programmer p1;
    cout<<"Salary: "<<p1.salary<<endl;
    cout<<"Bonus: "<<p1.bonus<<endl;
   return 0;
}
```

Output:

```
Salary: 60000
Bonus: 5000
```

Let's see another example of inheritance in C++ which inherits methods only.

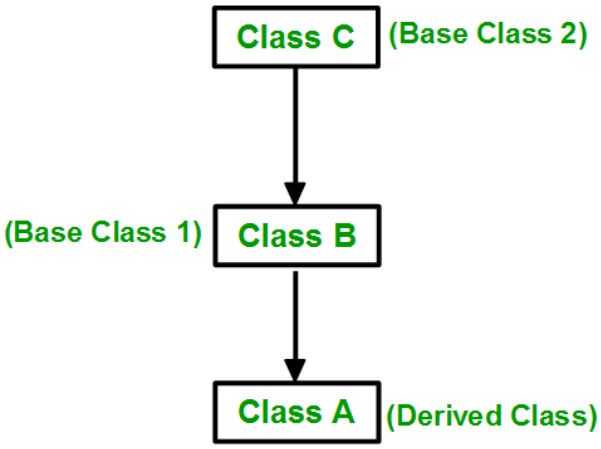
```
#include <iostream>
using namespace std;
 class Animal {
   public:
 void eat() {
    cout<<"Eating..."<<endl;
 }
   };
   class Dog: public Animal
 {
     public:
    void bark(){
   cout<<"Barking...";
    }
 };
int main(void) {
   Dog d1;
   d1.eat();
   d1.bark();
   return 0;
}
```

Output:

```
Eating...
Barking...
```

Multilevel Inheritance

A class can also be derived from one class, which is already derived from another class.



Example

```
// Base class (parent)
class A {
  public:
    void display() {
      cout << "Some content in parent class." ;
    }
};

// Derived class (child)
class B: public A {

    void msg() {
      cout << "Content in Class B." ;
    }

};

// Derived class (grandchild)
class C: public A {

    void Show() {
      cout << " content in  class C." ;
```

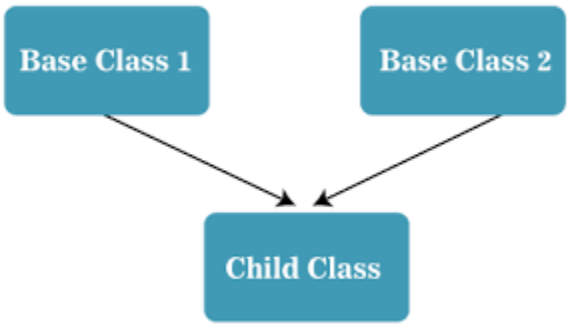
```
    }

};

int main() {
  C obj;
  obj.display();
  return 0;
}
```

Multiple Inheritance

A class can also be derived from more than one base class, using a **comma-separated list:**



Example

```
// Base class
class MyClass {
  public:
    void myFunction() {
      cout << "Some content in parent class." ;
    }
};

// Another base class
class MyOtherClass {
  public:
    void myOtherFunction() {
      cout << "Some content in another class." ;
    }
};

// Derived class
class MyChildClass: public MyClass, public MyOtherClass {
};

int main() {
  MyChildClass myObj;
```

```
  myObj.myFunction();
  myObj.myOtherFunction();
  return 0;
}
```