### C++ 快速導覽 - 類別 繼承

### 節錄自

### <http://pydoing.blogspot.tw/2013/02/cpp-inheritance.html>

以下C++檔案，請參考 t15\_9.cpp

繼承 (inheritance) 為物件導向程式設計 (object-oriented programming) 的特性之一，衍生類別 (derived class) 可以承接基礎類別 (base class) 的 **public** 或 **protected** 成員 (member) ，使之具有共通的 **public** 或 **protected** 成員。

衍生類別也常被稱為子類別 (subclass) ，基礎類別也常被稱為父類別 (superclass) 。

「繼承」是物件導向程式設計中一個重要也是核心的觀念，動詞原文為 inherit 。雖然 inherit 在英文的意思泛指從什麼得到什麼，用作中文「繼承」說得通，也能用作「遺傳」。然而中文的「繼承」隱含某物不再，另物將起的意思，譬如「我繼承某某的精神」，雖然某某不見得已死，未來將要付出努力的卻是我而非某某。因而這裡的意思中文用「遺傳」比較恰當，子代會從親代遺傳某些生物特性，子代與親代也會並存一段時間，這就沒有某物不再的意含了。然而這裡我們仍沿用程式設計常用的「繼承」一詞，但仍提出意見以免讀者混淆。

|  |  |
| --- | --- |
|  | #include <iostream> class Demo {public:    Demo() {        a = 10;        b = 33;    } protected:    int a;    int b;}; class Demo2: **public** Demo {public:    int do\_something() {        return a + b;    }   }; int main(void) {    Demo2 d;    std::cout << d.do\_something() << std::endl;     return 0;}  |

*Demo2* 繼承自 *Demo* ，繼承的寫法是在衍生類別後面接冒號，然後存取標籤 (access label) 加基礎類別

|  |  |
| --- | --- |
|  | class Demo2: **public** Demo {public:    int do\_something() {        return a + b;    }   }; |

### C++ 快速導覽 - 類別 多重繼承

節錄自

<http://pydoing.blogspot.tw/2013/02/cpp-multiple-inheritance.html>

以下C++檔案，請參考 t15\_10.cpp

衍生類別 (derived class) 可以繼承超過一個的基礎類別 (base class) ，這樣的特性被稱為多重繼承 (multiple inheritance) 。

衍生類別也常被稱為子類別 (subclass) ，基礎類別也常被稱為父類別 (superclass) 。

|  |  |
| --- | --- |
|  | #include <iostream> class Demo {public:    Demo() {        a = 702;        b = 631;    } protected:    int a;    int b;}; class Demo2 {public:    Demo2() {        c = 548;        d = 255;    } protected:    int c;    int d;};class Demo3: **public** Demo, **public** Demo2 {public:    int do\_something() {        return a + b + c + d;    }   }; int main(void) {    Demo3 d;    std::cout << d.do\_something() << std::endl;     return 0;}  |

*Demo3* 繼承自 *Demo* 與 *Demo2* ，用逗號 , 間隔兩個類別

|  |  |
| --- | --- |
|  | class Demo3: public Demo, public Demo2 {public:    int do\_something() {        return a + b + c + d;    }   }; |