Preprocessor

/ 程式到底怎麼被執行的？

執行流程

main.cpp

```
#include <iostream>
using namespace std;

int F(int);

int main() {
  cout << F(5);
}
```

vice.cpp

```
int F(int x){
  return x+1;
};
```

執行流程

Preprocessor

recognize meta-information

Compiler

translate source code into machine-dependent object code

Linker

link together all object files into an application

執行流程

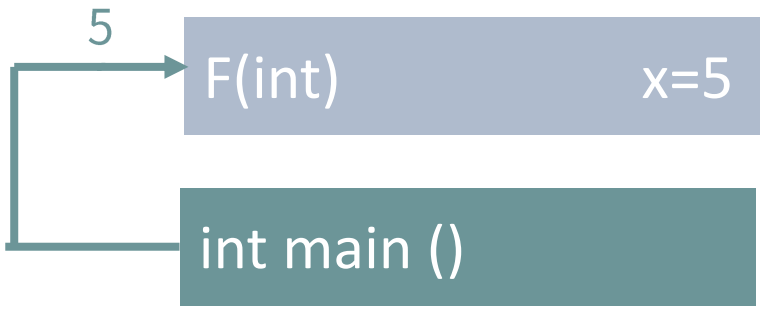

```
g++ -c AirTicket.cpp
g++ -c main.cpp
g++ -o prog AirTicket.o main.o
./prog.exe
```

Memory

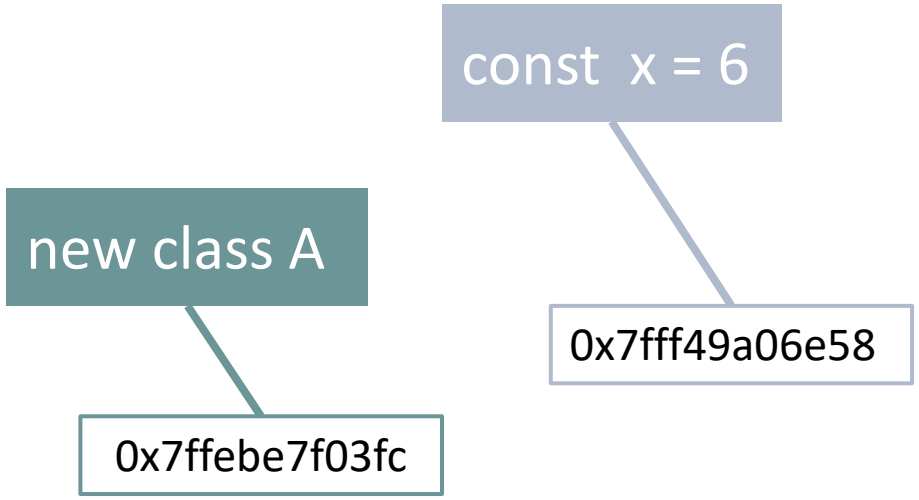
/ 資料存在哪?

基本介紹

Stack - 規律可預測的



Heap- 不規律不可預測的



OOP

/ uses objects to design programs

前言

一個語言如果不改變你的思考方式，就不值得學

- 程式語言的本質究竟是什麼？
- 什麼是「抽象化」？
- 不同系統關注重點不同
  - ➢ 在純物件導向程式語言的世界中，一切都是物件！

基本介紹

Object-Oriented Programming

- Object = data fields + methods
- Program = object + object + ⋯ + object

Features

- 封裝 -- 物件傳遞
- 繼承 -- 多人開發
- 多型 -- 程式維護

類別

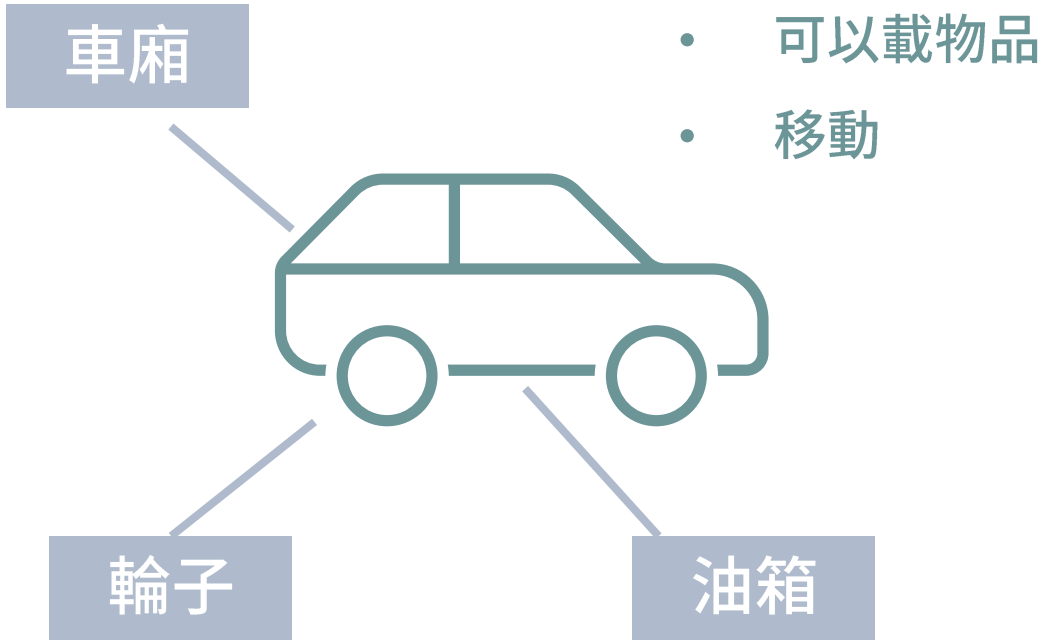
成員變數 (組成)

- 車廂
- 輪子
- 油箱

成員函數(功能)

- 載物
- 移動

物件



- 可以載物品
- 移動

Instantiation

Constructor

- invoke when an object of your class is created

Destructor

- is invoked automatically when the object goes out of scope

This-Pointer

Pointer

- its own address
- Only member functions have a this-pointer

An implicit parameter

```
name = 'YU,XIN-SHENG';

this->name = 'YU,XIN-SHENG';
```

Encapsulation

Like Function

- hide the actual content , so that the user only needs to know how to use it

Level

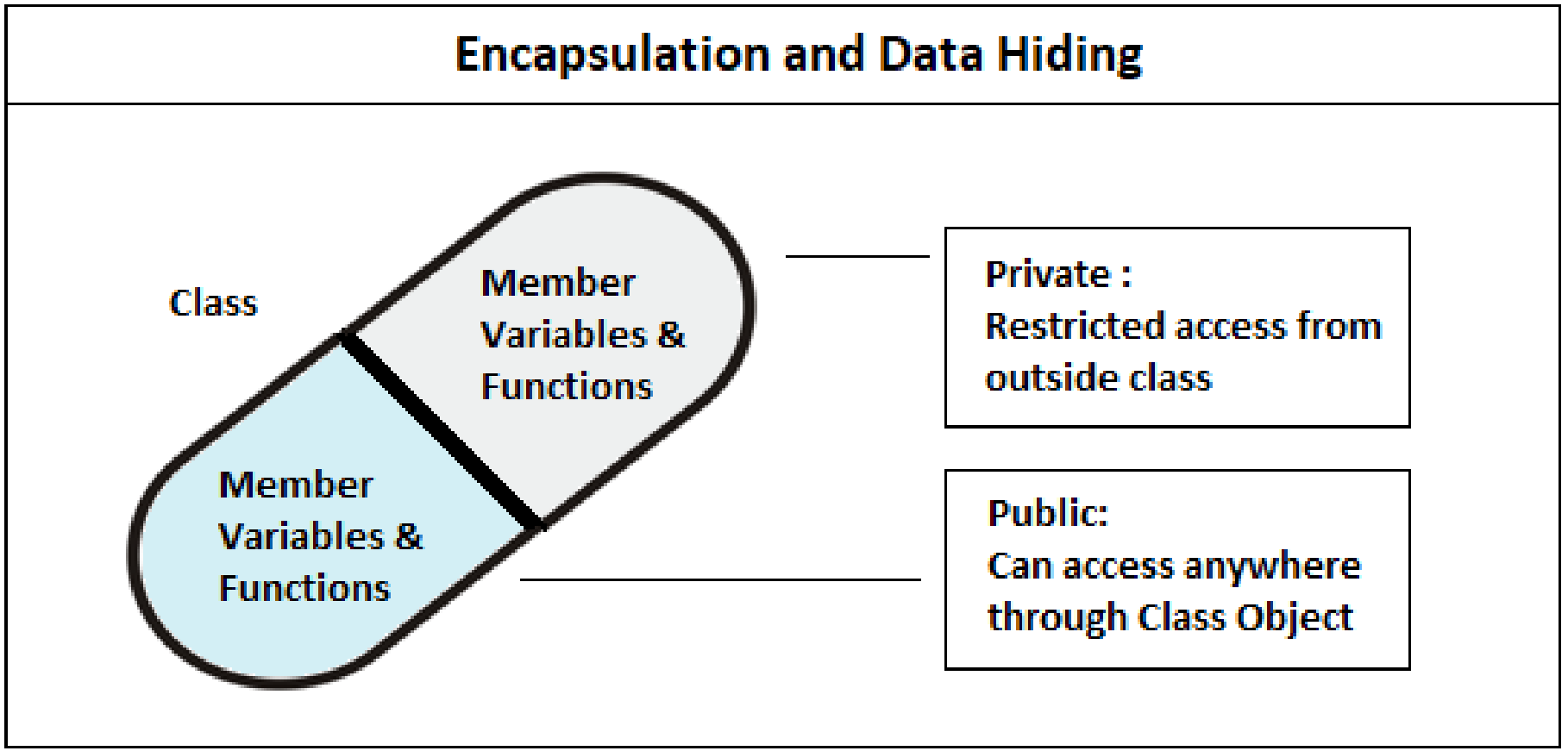
① Public

② Protected

③ Private

① ② 3

Encapsulation



- 圖片出自 : https://www.cpp.thiyagaraaj.com

Inheritance

Introduction

- This also provides an opportunity to reuse the code
- implement the is a relationship

Shape Problem

- circle is a kind of oval (radius)
- oval is a kind of circle (area overwrite)

Inheritance

Base and Derived Classes

- derived class ~~(child)~~
- base class ~~(parent)~~
- 繼承享受了取用基底類別內容的好處，卻也必須背負牽一髮動全身的風險。

Overload

Overload

- In a class, define same method , but parameters are different

```
class A {
    public:
        print(int x);
        print(int x, int y);
        print(int x, string z);
}
```

Override

Override

- A derived class can override the contents of a method in the base class

```
class Computer {
    public output(){
        1、Displayed on the screen
    }
};

class ModernComputer extend Computer{
    public output(){
        1、Displayed on the screen
        2、Displayed on the TV
    }
};
```