

PHP 物件導向 - 基礎觀念篇

瞭解 PHP 物件導向的開發方式

物件導向@PHP

- 類別？物件？怎麼分？
- 抽象？繼承？多型？
- 變數還是屬性？函式還是方法？
- PHP5 在物件導向上還有哪些特色？

類別
Class

```
class Person() {}
```

類別只是一堆函式的集合嗎？

- 凝聚力
- 物件的藍圖
- 包裝商業邏輯
- 適應性良好
- 會佔記憶體空間

物件
Object

```
$jace = new Person();
```

物件和一般變數有什麼不同？

- 能管理自身的狀態
- 可以調用方法或取用屬性
- 物件變數只放位址

成員

Member

```
$jace->height & $jace->eat();
```

屬性即變數？方法即函式？

- 類別成員，用 self
- 物件成員，用 this
- 屬性可以有常數預設值

封装

Encapsulation

\$jace->height = 0 ?

不能任意更改物件內部狀態

- setter 可以對參數值做檢查
- getter 讓我們能在輸出狀態前有改變的機會

作用域

Scope

private, protected, public

定義變數可視範圍

- private: 我的就是我的
- protected: 我和我的子孫可以用
- public: 大家都能用
- static: 影響同型態的所有物件實體
- 類別 static 方法中，不可用 this ！
- 方法 local 變數

抽象

Abstraction

'Person' === get_class(\$jace)

特意簡化問題

- Neo 開 Ferrari 跑車
- Jace 開 Toyota 小車
- 人開車
- 概略地來表達事物

繼承

Inheritance

Manager extends Person

不僅僅是復用程式碼

- 共用相同的邏輯
- 覆寫 (override) 不同的邏輯

多型

Polymorphism

\$employee->doWork()

一視同仁

- 強型別：必須是同一個型別或其子類別
- 弱型別：Duck Type (只考慮能不能做什麼)

介面

Interface

Car implement Drivable

告訴使用者可以怎麼做

- 只提供方法的宣告
- 類別必須「實作」介面
- 可以實作多個介面

異常

Exception

\$jace->eat(\$neo)

處理異常錯誤的好方法

- throw
- try ... catch
- 一層一層往上拋

PHP5 特色

Special

不同於其他語言的特色

- `__constructor()`, `__destructor()`
- `clone`
- magic method (`__set`, `__get`, `__call`, ...)
- type hint (通常是強型別語言才有)

SPL

Standard PHP Library

More Power

延展 PHP5 的功能

- Iterators
- Directories and Files
- Array Overloading
- Exceptions

謝謝