Module I
Java導論

1. Java技術源起
2. Java白皮書與專業術語
3. Java程式語言的關鍵概念
4. Java的執行環境
5. Write once, run anywhere
6. Java技術種類

Java技術源起

- Java Programming Language起源於1991年，由Sun Microsystems (昇陽)公司所開發

  - Java一開始被稱為Oak(橡樹)，而當時開發用途是想創造一種「能在不同CPU的消費性電子產品(如電視、電話)中，能共同使用的程式語言」

  - 專案名稱：Green Project

  - 專案主持人：James Gosling (Java之父)

- 計劃失敗後，Sun公司看見Oak在網際網路(WWW)上應用的前景，於是改造了Oak，於1995年5月以Java的名稱正式釋出。Java伴隨著網際網路的迅速發展而發展，逐漸成為重要的網路程式語言。

- 1995年5月23日，Oak正式更名為Java，同時JDK (Java Development Kits) 1.0a2版正式對外發表

Java白皮書與其專業術語

- Java白皮書 (Java “White Paper”)
  - Java發明者寫了一個有影響力的白皮書，說明他們設計目標與成就
  - May 1996, by James Gosling and Henry McGilton
  - 網址：http://www.oracle.com/technetwork/java/langenv-140151.html
- James Gosling用以下11個專業術語，對Java語言做摘要描述

Simple	Portable
Object Oriented	Interpreted
Network-Savvy	High Performance
Robust	Multithreading
Secure	Dynamic
Architecture Neutral	

Java關鍵概念 (1/9)

- Simple (簡單)
- Object-Oriented (物件導向)
- Distributed (分散式運算)
- Multithreading (多執行緒)
- Secure (安全性)
- Platform-independent (跨平台)

Java關鍵概念 (2/9)

- Simple (簡單)

  - Java程式語言之所以簡單，是因為發明者移除了一些一般程式語言較為複雜的地方與含糊不清的程式結構，如：

    - Java不允許程式設計者直接使用指標(pointer)去操控記憶體位址，這不但複雜而且容易發生錯誤，這是C與C++使用的常見問題
    - Java只允許程式設計者使用物件參考(Object reference)來操控物件，和資源回收機制(Garbage collection)來自動處理已不被參考的物件
    - 另一個簡單的原因，是因為Java的布林(boolean)資料型態可以有true或false的值，不像其它語言只有1與0的值

Java關鍵概念 (4/9)

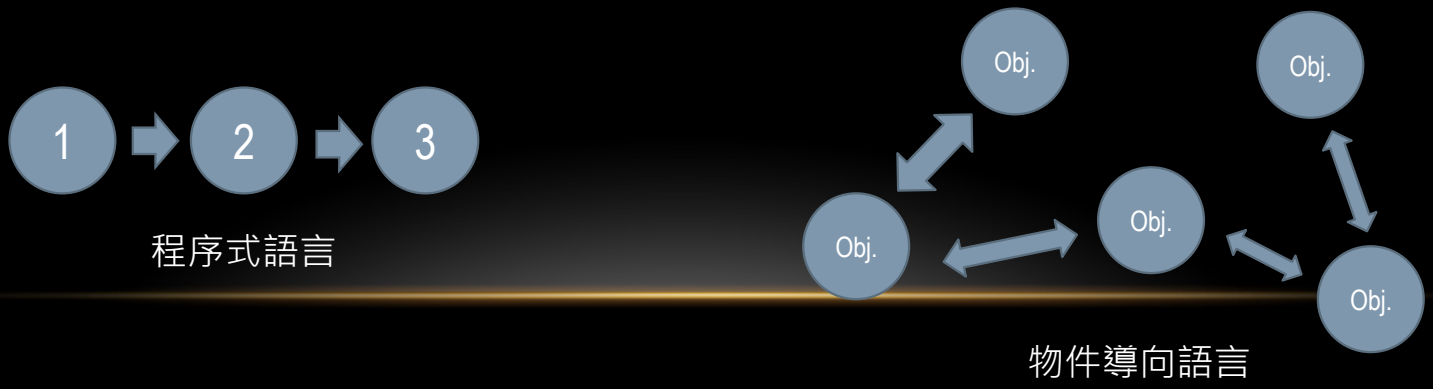
- Object-Oriented (物件導向)

  - 程序性程式語言 (Procedural Programming)

    · 所著重的是利用撰寫程式的先後次序來解決問題

  - 物件導向程式語言 (Object-Oriented Programming, OOP)

    · 著重於物件之間可以相互作用的關係來解決問題

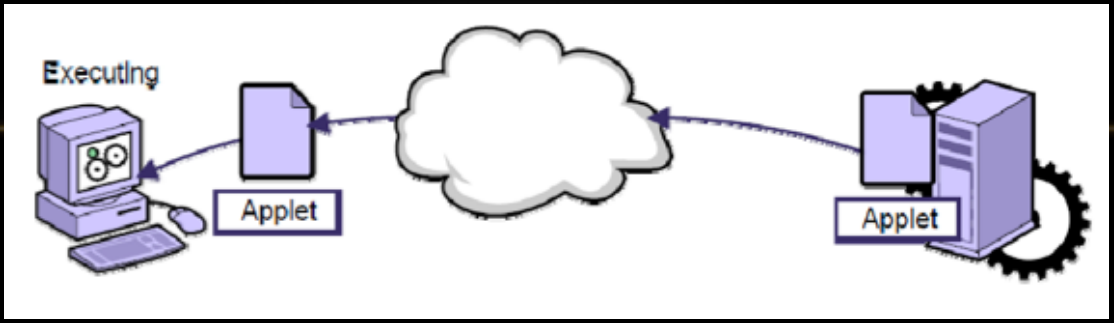


Java關鍵概念 (5/9)

- Distributed (分散式運算)

  - Java是一種分散式程式語言，支援許多分散式網路技術，如：
    - RMI (Remote Method Invocation)
    - CORBA (Common Object Request Broker Architecture)
    - URL (Universal Resource Locator)

  - 另外，Java的動態類別載入功能允許部份的程式碼可由網路下載，並在個人電腦執行，如Applet



Java關鍵概念 (6/9)

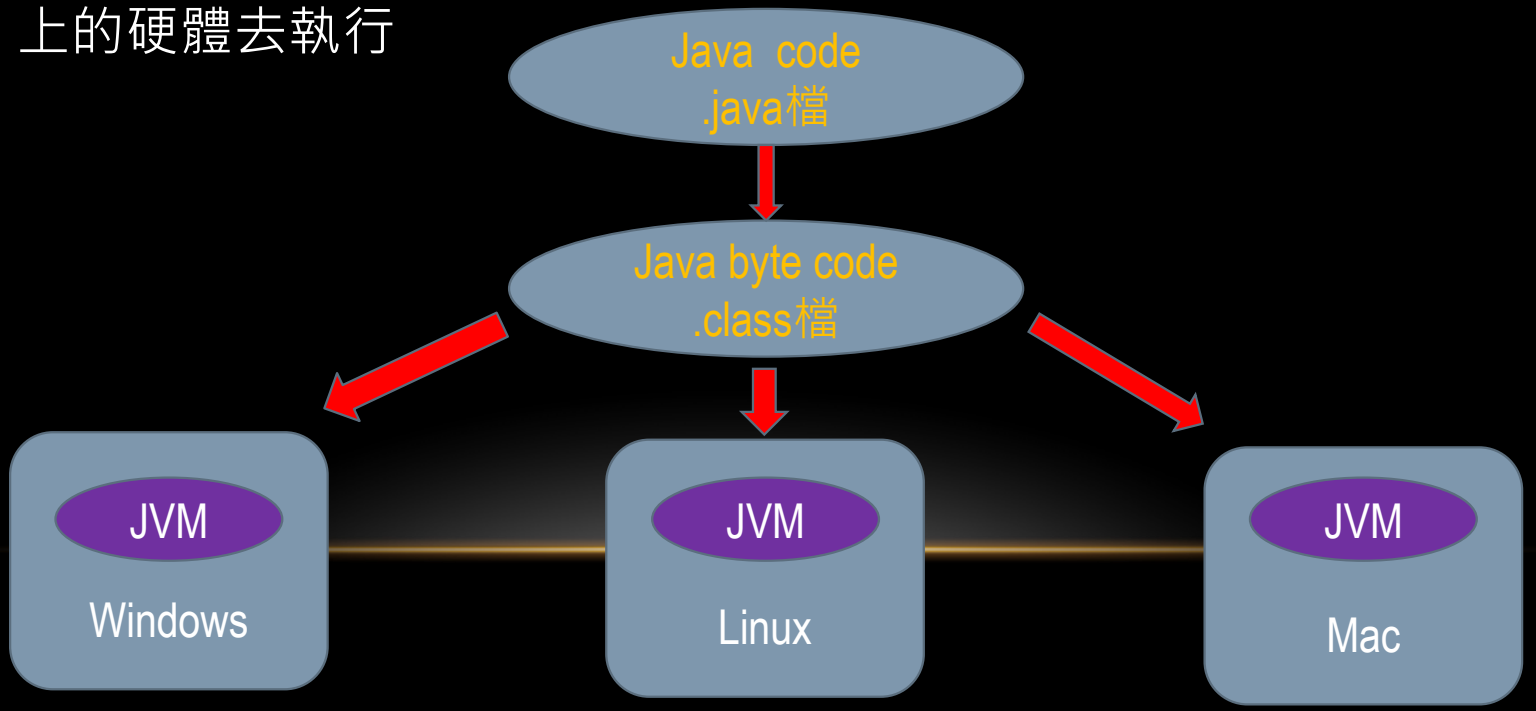
- Multithreading (多執行緒)
  - 可以在同一時間執行多個程序 (Multi-processor)
  - 這代表在使用系統資源時，可以用Java寫出非常有效率的程式

Java關鍵概念 (7/9)

- Secure (安全性)
  - 禁止使用指標 (pointer) 操控記憶體
  - 驗證所有Java程式是否擁有合法的程式碼
  - 支援數位簽章 (Digital Signatures):
    - Java技術的程式碼可以由公司具有的數位簽章或個人所擁有的數位簽章來簽署

Java關鍵概念 (8/9)

- Platform-independent (跨平台)
  - Java程式設計者先使用Java編譯器 (Java Compiler)，將Java程式碼譯成與平台無關的位元碼 (byte code)
  - 位元碼透過各系統專有的Java虛擬機器 (JVM, Java Virtual Machine)上的硬體去執行



Java關鍵概念 (9/9)

- JVM (Java Virtual Machine)目前已有多種平台版本，如Linux, Mac, Windows, Solaris…等，除了這些大型作業統外，還有針對各種小型系統設計的JVM, 如PDA, 手機等…

- 因此Java可以在不同的CPU或不同作業系統環境上執行，**JVM**即是Java能跨平台的主要原因

想想看：跨國商業合作時，契約簽定的模式是什麼呢？

Java執行環境 (1/2)

- Java程式的執行環境
  - 一個Java程式只需要一個Java虛擬機器(JVM)去執行
  - 一個Java程式也需要一套針對此平台所設計的標準**Java類別函式庫** (Java class libraries)

- JVM與Java類別函式庫的組合被稱為：
  Java執行環境 (Java Runtime Environment, JRE)

Write once, run anywhere

- Sun Microsystems (昇陽)在開發Java同時，對許多常見的系統平台都提供了Java執行環境，使程式設計師在撰寫Java程式時，無需考慮到系統與硬體平台的問題，專心致力於程式功能與邏輯實現

- Write once, run anywhere已成為Java程式設計師們的精神指標！

Java技術產品種類

- Java Standard Edition (Java SE)
  - 標準版
  - 適用於開發用戶端程式
- Java Enterprise Edition (Java EE)
  - 企業版
  - 適用於開發伺服器端程式
- Java Micro Edition (Java ME)
  - 手持裝置版
  - 適用於開發手機、無線設備等程式

✔自Java 6以後，已取消了J2SE, J2EE等用法

章節整理

- Java並非一炮而紅的程式語言，藉由網際網路興起而找到自己的一片天，瞭解Java其技術演進與發展歷史，才能掌握Java程式語言有趣之處
- Java為物件導向程式語言，因此，我們將會在後續學習如何操作物件撰寫程式，並懂得如何用程式來解決問題
- 熟悉並瞭解Java關鍵特性與功能概念
- Java因為有了JVM, 而能達到跨平台執行的功能，而Write once, run anywhere也成為了所有程式設計師們的共同目標與精神
- 寫程式不只是一份工作的技能，更是強化自己邏輯觀念與培養解決問題能力的好工具