บทนำ

วัตถุประสงค์

- เข้าใจว่าทำไมต้องเรียน Java
- เข้าใจความแตกต่างระหว่างคำศัพท์ IDE, และ JDK
- เข้าใจขั้นตอน compile โปรแกรม
- รู้ส่วนประกอบของโปรแกรมภาษา Java
- สามารถเขียนโปรแกรมภาษา Java โปรแกรมอย่างง่ายได้

ทำไมต้องเรียน Java

เพราะ Java เป็นภาษาที่เอื้อให้ผู้เขียนโปรแกรมสามารถเขียนโปรแกรมบนเครื่องแม่ข่าย เครื่องคอมพิวเตอร์ตั้งโต๊ะ และ อุปกรณ์มือถือเล็ก ๆ ได้ ดังนั้น Java จะเป็นภาษาที่มีประโยชน์ในการพัฒนาโปรแกรมประยุกต์ในปัจจุบัน

คุณลักษณะของภาษา Java

- ง่าย: แม้ว่าบางส่วนของ Java จะมีต้นแบบมาจาก C++ แต่ก็ผู้สร้าง Java ก็ออกแบบและปรับปรุงให้มันง่ายและดีขึ้น
- Object-Oriented: Java ถูกออกแบบมาสำหรับการเขียนโปรแกรมแบบ Object ไม่ใช่แบบกระบวนการ (Procedural) ทำให้มีความยืดหยุ่น (Flexibility), มีความเป็นหน่วย (Modularity), มีความชัดเจนของตัวโปรแกรม (Clarity), และ มีการนำกลับมาใช้ได้ใหม่ (Reusability) มากขึ้น
- โปรแกรมแบบกระจาย (Distributed): Java ได้ถูกออกแบบมาให้การติดต่อระหว่างโปรแกรมผ่านเครือข่ายทำได้ง่ายขึ้น
- ผ่านการแปลมาแล้ว (Interpreted): แม้การพัฒนาโปรแกรมภาษา Java นั้นเหมือนกับการพัฒนาภาษาอื่นๆ คือ ผู้พัฒนาต้องเขียนโปรแกรม (source code) ก่อน แล้วแปลโปรแกรมเป็นโปรแกรมที่ CPU จะสามารถ execute ได้ แต่ ภาษา Java ต่างกับภาษาอื่นๆ ตรงที่ source

code ของ Java จะถูกแปลเป็นรูปแบบ (format) ที่สามารถปฏิบัติงานได้ทุก platform (Java Virtual Machine มีหน้าที่เป็น Interpreter รับผิดชอบในการแปลนี้)

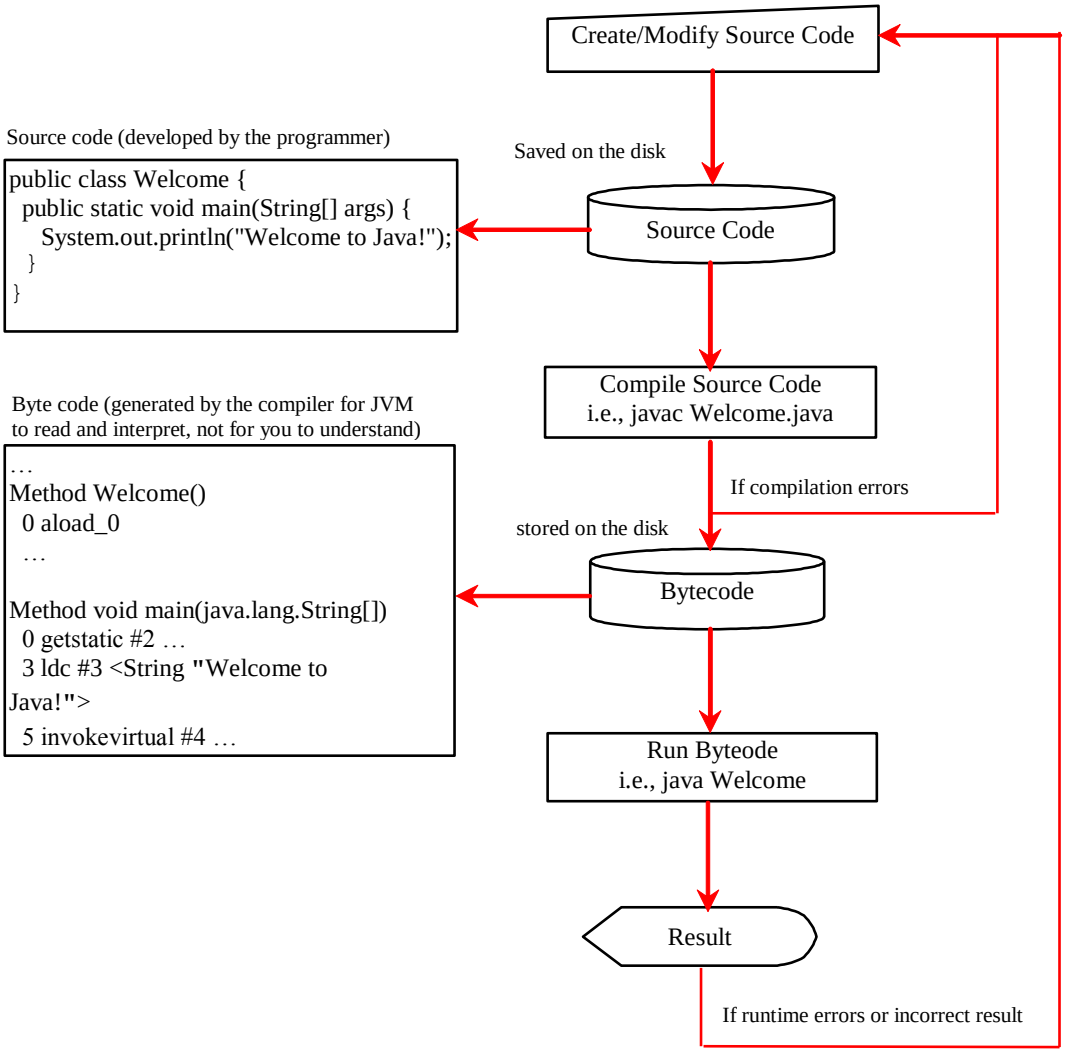
- ทนทานต่อความผิดพลาด (Robust): Java สามารถพบปัญหาหรือ bug ได้หลายชนิดก่อนที่จะ run โปรแกรม นอกจากนี้ Java ยังมีกลไกจัดการกับ exception เวลามีปัญหาเกิดขึ้นในระบบ
- ปลอดภัย (secure): Java มีกลไกเกี่ยวกับความปลอดภัยเพื่อป้องกันระบบจากการทำงานของโปรแกรมแปลกหน้า
- เป็นกลางไม่ขึ้นกับสถาปัตยกรรมตัวเครื่อง (Architecture-neutral and portable): โปรแกรมที่แปลด้วย Java Virtual Machine แล้ว (กลายเป็น Java bytecode) จะสามารถปฏิบัติงานได้ในทุกสถาปัตยกรรมคอมพิวเตอร์
- เคลื่อนย้ายได้ (Portable): bytecode (โปรแกรมที่ถูกแปลแล้ว) สามารถถูกนำไป run ที่เครื่องแบบใดก็ได้ โดยไม่ต้องถูกแปลใหม่
- มีสายการทำงานหลายสายพร้อมกันในโปรแกรมเดียว (Multithreaded): Java ได้ถูกออกแบบให้เป็นการเขียนโปรแกรมแบบ Multithread จึงทำให้ Java สามารถทำงานได้หลายๆ สายงานในเวลาเดียวกัน
- พลวัต (Dynamic): ปัญหาการพัฒนาโปรแกรมที่พบได้บ่อยคือ เวลามีการเปลี่ยนแปลงในตัวภาษาอย่างใดอย่างหนึ่งแล้ว เช่นมีการเปลี่ยนรุ่นของ compiler อาจทำให้ต้องแปลโปรแกรมทั้งหมดอีกครั้งเสมอ ภาษา Java ช่วยแก้ปัญหาตรงนี้ โดยที่เมื่อมีส่วนที่เพิ่มเข้ามาใหม่ Java จะทำการเพิ่มให้เองอัตโนมัติ นักพัฒนาไม่ต้องทำการ compile ใหม่ทุกครั้ง

JDK, SDK และ IDE

- SDK (ย่อมาจาก Software Development Kit) เป็นชุดเครื่องมือช่วยในการพัฒนาซอฟต์แวร์ชนิดต่าง ๆ
- JDK (ย่อมาจาก Java Development Kit) เป็น SDK ชนิดหนึ่งที่ช่วยในการพัฒนาซอฟต์แวร์ที่เป็นภาษาจาวา มีหลายรุ่น เช่น รุ่นมาตรฐาน (Standard Edition, J2SE) รุ่นสำหรับบริษัท (Java Enterprise Edition, J2EE) ฯลฯ

- IDE (ย่อมาจาก Integrated Development Environment) เป็นโปรแกรมประยุกต์ที่อำนวยความสะดวกให้นักพัฒนาโปรแกรมโดย IDE ส่วนใหญ่จะประกอบไปด้วย compiler, debugger, และ source code editor ตัวอย่าง IDE สำหรับ Java ที่เป็นที่นิยมเช่น Borland, JBuilder, Netbeans, และ Eclipse

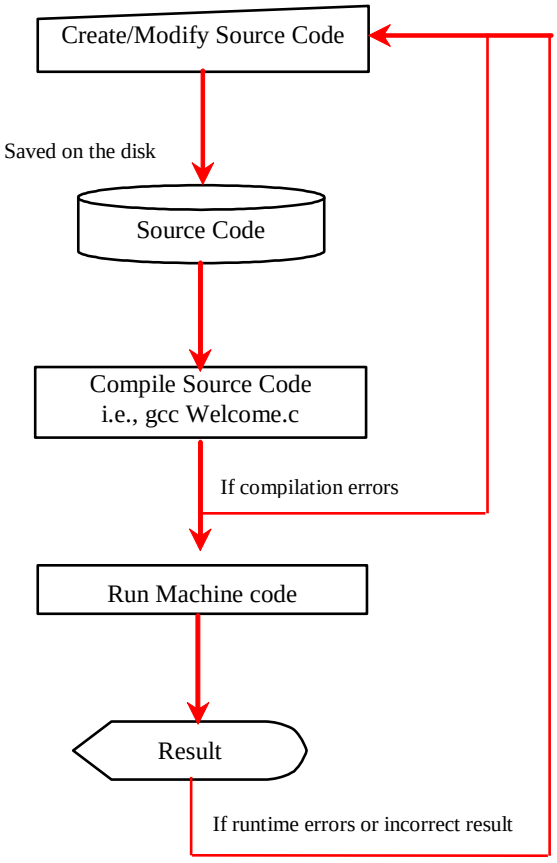
ขั้นตอนการ compile และ run โปรแกรม



ขั้นตอนการ compile และ run โปรแกรม Java สามารถแสดงได้ดังแผนผังด้านบน เริ่มจากผู้เขียนเริ่มเขียน source code ให้เสร็จ (อาจใช้ editor ใน IDE, Notepad, Wordpad หรือ vi ก็ได้) จากนั้นจะเป็นขั้นตอนการ compile เพื่อให้ได้เป็น bytecode (file นามสกุล java) ออกมา

อย่างที่ได้กล่าวแล้วว่า bytecode สามารถนำไปรันบนเครื่อง หรือ Platform ต่าง ๆ ได้เลย ถ้าเครื่องดังกล่าวมี JVM (Java Virtual Machine) ที่เหมาะสม เพราะ JVM ที่เหมาะกับเครื่องนั้น จะสามารถแปลง (Interpret) bytecode ให้เป็นคำสั่งที่เครื่องนั้น ๆ นำไปรันได้เอง แม้ว่า bytecode นั้นจะเป็นผลมาจากการ compile ที่เครื่องอื่น

เพื่อเปรียบเทียบกับภาษาอื่น ให้พิจารณาตัวอย่างขั้นตอนการ compile และ run โปรแกรมภาษา C ดังรูปด้านล่าง (ให้สังเกตว่า หลังขั้นตอนการ compile, C compiler จะให้ Machine code ที่ CPU ใช้รันได้เลย แทนที่จะเป็น bytecode ที่ต้องใช้ JVM มาแปลงอีกทีหนึ่ง)



ข้อเสียสำหรับขั้นตอนการ compile และ รันของภาษา C คือ Machine code จะผูกติดกับเครื่องชนิดเดียวกับที่ใช้ compile source code เท่านั้น ไม่สามารถนำ Machine code ไปรันที่เครื่องชนิดอื่นได้ นอกเสียจากจะ compile source code เพื่อให้ได้ Machine code ใหม่เท่านั้น

ส่วนประกอบของโปรแกรมภาษา Java

- Comments: ส่วนที่เป็นคำอธิบาย source code ของผู้เขียน source code นั้น เพื่อให้ ผู้เขียนสามารถกลับมาทำความเข้าใจ code ของตัวเองได้ในภายหลัง หรือ เพื่อให้ผู้อื่นที่ไม่ใช่ผู้เขียนโปรแกรมสามารถทำความเข้าใจ source code นั้นเพื่อศึกษา หรือ ทำการแก้ไขได้ถูกต้อง comment มี 3 ลักษณะ ดังนี้:-
    - Line comment ใช้สัญลักษณ์ // ที่หน้าคำอธิบายของโปรแกรม เพื่อให้ compiler รู้ว่าข้อความหลังสัญลักษณ์นี้ไปจนจบบรรทัดนี้เป็น comment ของผู้เขียนโปรแกรม และ ไม่นำมาเป็นส่วนของโปรแกรมที่จะใช้ compile
    - Paragraph comment ใช้สัญลักษณ์ /* .... */ คร่อม comment ที่มีความยาวมากกว่าหนึ่งบรรทัด
    - Javadoc Comment: เริ่มด้วย /** จบด้วย */ คล้ายกับ Paragraph comment แต่มีประโยชน์มากกว่า เมื่อเราจะสร้างเอกสารประกอบการใช้ class ข้อมูล และ method ต่าง ๆ ใน source code ของเราโดยใช้คำสั่ง javadoc

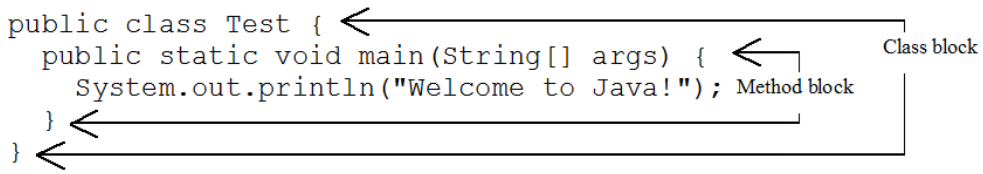
- Package: ใช้บอกชื่อ package ของโปรแกรม เช่น package chapter1; จะทำให้รู้ว่า class ที่อยู่ใน file source code นี้เป็นของ package ชื่อ chapter1 (package เป็นการจัดการ source code ที่ทำงานด้วยกัน หรือ มีจุดประสงค์คล้ายกัน เข้าไว้ด้วยกัน)

- Reserved words (หรือ keywords): เป็นคำที่ถูก complier จองไว้แล้ว เช่นคำว่า class, public, static, void, int, double ฯลฯ มีความหมายเฉพาะตัว ไม่สามารถใช้สำหรับความหมายอย่างอื่นได้ เช่น ถ้าประกาศ double เป็นชื่อของตัวแปร (int double;) compiler จะแจ้ง error

- Modifiers: เป็น reserved words ชนิดหนึ่ง เช่น public static private final abstract และ protected จะเป็นคำที่บอกคุณสมบัติของข้อมูล method หรือ class ว่ามันควรจะถูกใช้อย่างไร

- Statements: ใช้แทน action หรือการกระทำ ที่ผู้เขียนตั้งใจจะให้โปรแกรมทำ เช่น a = b + c; System.out.println("Hello world!"); ข้อสังเกต ทุก ๆ statement จะจบลงด้วยเครื่องหมาย ;

- Blocks: คู่เปิดปิดของวงเล็บปีกกา ที่จัดให้ส่วนต่าง ๆ ของโปรแกรมเป็นกลุ่ม เช่น กลุ่มของ class กลุ่มของ method ดังแสดงในรูปด้านล่าง



- Classes: เป็นส่วนประกอบของโปรแกรมที่มีความสำคัญ source code ที่ไม่มี class จะไม่สามารถถูก run ได้ (ให้นึกถึง class เหมือนพิมพ์เขียว ที่ใช้เวลาเราจะผลิตอะไร เราต้องร่างแบบในพิมพ์เขียวก่อนการผลิตออกมาเป็นชิ้นงานจริง)

- Methods: เป็นชุดของ statement ที่ถูกรวมอยู่ใน block ของ method ซึ่งชุดของ statement ดังกล่าวถูกพัฒนาเพื่อให้โปรแกรมสามารถเรียกใช้ statement ได้ทั้งชุดต่อการเรียกหนึ่งครั้ง บางครั้งผู้พัฒนาโปรแกรมอาจออกแบบให้ method รับ input เพื่อประมวลผล ซึ่ง input นั้นถูกเรียกว่า argument ของ method เช่น statement System.out.println("Welcome to Java!"); มี method ชื่อ println และ มี string "Welcome to Java!" เป็น argument

- The main method: เป็น method ที่ชื่อ main จะทำหน้าที่ควบคุมขั้นตอนการทำงานของโปรแกรม (Java จะรันโปรแกรมโดยเรียกใช้งาน method นี้ก่อน ดังนั้น โปรแกรม Java ทุกโปรแกรมต้องมี method นี้)

ตัวอย่างโปรแกรมอย่างง่าย 1

```
1:  // This program prints Welcome to Java!
2: public class Welcome {
3:     public static void main(String[] args) {
4:         System.out.println("Welcome to Java!");
5:     }
6: }
```

บรรทัดที่

1: comment ไม่ได้ถูกใช้ในการ compile และไม่เกี่ยวกับการทำงานของโปรแกรม ถูกเขียนเพื่ออธิบาย source code เท่านั้น

2: เป็นการประกาศ class ชื่อ Welcome

3: เป็นการประกาศ method ชื่อ main ซึ่งมี argument เป็น array ของ String (รายละเอียดของ argument นี้จะกล่าวภายหลัง)

4: เป็น Statement หรือ สิ่งที่ผู้เขียนโปรแกรมอยากให้โปรแกรมทำ เมื่อ method main ถูกเรียก (ในที่นี้คือ ให้ print คำว่า Welcome to Java! ออกหน้าจอ

5. เครื่องหมาย } เป็นการบอก compiler ว่าขอบเขตของ method main ได้สิ้นสุดที่ตำแหน่งของเครื่องหมายนี้

6. เครื่องหมาย } เป็นการบอก compiler ว่าขอบเขตของ class Welcome ได้สิ้นสุดที่ตำแหน่งของเครื่องหมายนี้