

Mục tiêu

- Nắm được các đặc trưng của Java
- Các kiểu chương trình Java
- Định nghĩa về máy ảo Java
- Các nội dung của JDK (Java Development Kit)
- Sơ lược các đặc trưng mới của Java2

2.1 Giới thiệu Java

Java là một ngôn ngữ lập trình được Sun Microsystems giới thiệu vào tháng 6 năm 1995. Từ đó, nó đã trở thành một công cụ lập trình của các lập trình viên chuyên nghiệp. Java được xây dựng trên nền tảng của C và C++. Do vậy nó sử dụng các cú pháp của C và các đặc trưng hướng đối tượng của C++.

Vào năm 1991, một nhóm các kỹ sư của Sun Microsystems có ý định thiết kế một ngôn ngữ lập trình để điều khiển các thiết bị điện tử như Tivi, máy giặt, lò nướng, ... Mặc dù C và C++ có khả năng làm việc này nhưng trình biên dịch lại phụ thuộc vào từng loại CPU.

Trình biên dịch thường phải tốn nhiều thời gian để xây dựng nên rất đắt. Vì vậy để mỗi loại CPU có một trình biên dịch riêng là rất tốn kém. Do đó nhu cầu thực tế đòi hỏi một ngôn ngữ chạy nhanh, gọn, hiệu quả và độc lập thiết bị tức là có thể chạy trên nhiều loại CPU khác nhau, dưới các môi trường khác nhau. "Oak" đã ra đời và vào năm 1995 được đổi tên thành Java. Mặc dù mục tiêu ban đầu không phải cho Internet nhưng do đặc trưng không phụ thuộc thiết bị nên Java đã trở thành ngôn ngữ lập trình cho Internet.

2.1.1 Java là gì

Java là ngôn ngữ lập trình hướng đối tượng, do vậy không thể dùng Java để viết một chương trình hướng chức năng. Java có thể giải quyết hầu hết các công việc mà các ngôn ngữ khác có thể làm được.

Java là ngôn ngữ vừa biên dịch vừa thông dịch. Đầu tiên mã nguồn được biên dịch bằng công cụ JAVAC để chuyển thành dạng ByteCode. Sau đó được thực thi trên từng loại máy cụ thể nhờ chương trình thông dịch. Mục tiêu của các nhà thiết kế Java là cho phép người lập trình viết chương trình một lần nhưng có thể chạy trên bất cứ phần cứng cụ thể.

Ngày nay, Java được sử dụng rộng rãi để viết chương trình chạy trên Internet. Nó là ngôn ngữ lập trình hướng đối tượng độc lập thiết bị, không phụ thuộc vào hệ điều hành. Nó không chỉ dùng để viết các ứng dụng chạy đơn lẻ hay trong mạng mà còn để xây dựng các trình điều khiển thiết bị cho điện thoại di động, PDA, ...

2.2 Các đặc trưng của Java

- Đơn giản
- Hướng đối tượng
- Độc lập phần cứng và hệ điều hành
- Mạnh
- Bảo mật

- Phân tán
- Đa luồng
- Động

2.2.1 Đơn giản

Những người thiết kế mong muốn phát triển một ngôn ngữ dễ học và quen thuộc với đa số người lập trình. Do vậy Java được loại bỏ các đặc trưng phức tạp của C và C++ như thao tác con trỏ, thao tác nạp đè (overload),... Java không sử dụng lệnh "goto" cũng như file header (.h). Cấu trúc "struct" và "union" cũng được loại bỏ khỏi Java.

2.2.2 Hướng đối tượng

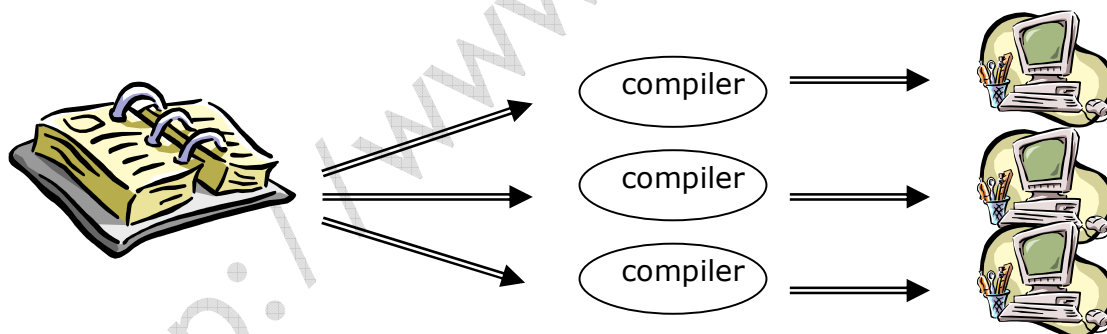
Java được thiết kế xoay quanh mô hình hướng đối tượng. Vì vậy trong Java, tiêu điểm là dữ liệu và các phương pháp thao tác lên dữ liệu đó. Dữ liệu và các phương pháp mô tả trạng thái và cách ứng xử của một đối tượng trong Java.

2.2.3 Độc lập phần cứng và hệ điều hành

Đây là khả năng một chương trình được viết tại một máy nhưng có thể chạy được bất kỳ đâu. Chúng được thể hiện ở mức mã nguồn và mức nhị phân.

Ở mức mã nguồn, người lập trình cần mô tả kiểu cho mỗi biến. Kiểu dữ liệu trong Java nhất quán cho tất cả các hệ điều hành và phần cứng khác nhau. Java có riêng một thư viện các lớp cơ sở. Vì vậy chương trình Java được viết trên một máy có thể dịch và chạy trơn tru trên các loại máy khác mà không cần viết lại.

Ở mức nhị phân, một chương trình đã biên dịch có thể chạy trên nền khác mà không cần dịch lại mã nguồn. Tuy vậy cần có phần mềm máy ảo Java (sẽ đề cập đến ở phần sau) hoạt động như một trình thông dịch tại máy thực thi.



Hình 2.1

Trình biên dịch sẽ chuyển các chương trình viết bằng C, C++ hay ngôn ngữ khác thành mã máy nhưng phụ thuộc vào CPU. Nên khi muốn chạy trên loại CPU khác, chúng ta phải biên dịch lại chương trình.

Hình 2.2

Môi trường phát triển của Java được chia làm hai phần: Trình biên dịch và trình thông dịch. Không như C hay C++, trình biên dịch của Java chuyển mã nguồn thành dạng bytecode độc lập với phần cứng mà có thể chạy trên bất kỳ CPU nào. Nhưng để thực thi chương trình dưới dạng bytecode, tại mỗi máy cần phải có trình thông

dịch của Java hay còn gọi là máy ảo Java. Máy ảo Java chuyển bytecode thành mã lệnh mà CPU thực thi được.

2.2.4 Mạnh mẽ

Java yêu cầu chặt chẽ về kiểu dữ liệu và phải mô tả rõ ràng khi viết chương trình. Chúng sẽ kiểm tra lúc biên dịch và cả trong thời gian thông dịch vì vậy Java loại bỏ các kiểu dữ liệu dễ gây ra lỗi.

2.2.5 Bảo mật

Java cung cấp một số lớp để kiểm tra bảo mật.

Ở lớp đầu tiên, dữ liệu và các phương pháp được đóng gói bên trong lớp. Chúng chỉ được truy xuất thông qua các giao diện mà lớp cung cấp. Java không hỗ trợ con trỏ vì vậy không cho phép truy xuất bộ nhớ trực tiếp. Nó cũng ngăn chặn không cho truy xuất thông tin bên ngoài của mảng bằng kỹ thuật tràn và cũng cung cấp kỹ thuật dọn rác trong bộ nhớ. Các đặc trưng này tạo cho Java an toàn và có khả năng cơ động cao.

Trong lớp thứ hai, trình biên dịch kiểm soát để đảm bảo mã an toàn. Lớp thứ ba được đảm bảo bởi trình thông dịch. Chúng kiểm tra xem bytecode có đảm bảo các quy tắc an toàn trước khi thực thi. Lớp thứ tư kiểm soát việc nạp các lớp lên bộ nhớ để giám sát việc vi phạm giới hạn truy xuất trước khi nạp vào hệ thống.

2.2.6 Phân tán

Java có thể dùng để xây dựng các ứng dụng có thể làm việc trên nhiều phần cứng, hệ điều hành và giao diện đồ họa. Java được thiết kế cho các ứng dụng chạy trên mạng. Vì vậy chúng được sử dụng rộng rãi trên Internet, nơi sử dụng nhiều nền tảng khác nhau.

2.2.7 Đa luồng

Chương trình Java sử dụng kỹ thuật đa tiến trình (Multithread) để thực thi các công việc đồng thời. Chúng cũng cung cấp giải pháp đồng bộ giữa các tiến trình. Đặc tính hỗ trợ đa tiến trình này cho phép xây dựng các ứng dụng trên mạng chạy uyển chuyển.

2.2.8 Động

Java được thiết kế như một ngôn ngữ động để đáp ứng cho những môi trường mở. Các chương trình Java bổ xung các thông tin cho các đối tượng tại thời gian thực thi. Điều này cho phép khả năng liên kết động các mã.

2.3 Các kiểu chương trình Java

Chúng ta có thể xây dựng các loại chương trình Java như sau:

2.3.1 Applets

Đây là chương trình chạy trên Internet thông qua các trình duyệt hỗ trợ Java như IE hay Netscape. Bạn có thể dùng các công cụ của Java để xây dựng Applet. Applet được nhúng bên trong trang Web hoặc file HTML. Khi trang Web hiển thị trong trình duyệt, Applet sẽ được nạp và thực thi.

2.3.2 Ứng dụng thực thi qua dòng lệnh

Các chương trình này chạy từ dấu nhắc lệnh và không sử dụng giao diện đồ họa. Các

thông tin nhập xuất được thể hiện tại dấu nhắc lệnh.

2.3.3 Ứng dụng đồ họa

Đây là các chương trình Java chạy độc lập cho phép người dùng tương tác qua giao diện đồ họa.

2.3.4 Servlet

Java thích hợp để phát triển ứng dụng nhiều lớp. Applet là chương trình đồ họa chạy trên trình duyệt tại máy trạm. Ở các ứng dụng Web, máy trạm gửi yêu cầu tới máy chủ. Máy chủ xử lý và gửi ngược kết quả trở lại máy trạm. Các chương trình Java API chạy trên máy chủ giám sát các quá trình tại máy chủ và trả lời các yêu cầu của máy trạm. Các chương trình Java API chạy trên máy chủ này mở rộng khả năng của các ứng dụng Java API chuẩn. Các ứng dụng trên máy chủ này được gọi là các Servlet. hoặc Applet tại máy chủ. Các xử lý trên Form của HTML là cách sử dụng đơn giản nhất của Servlet. Chúng còn có thể được dùng để xử lý dữ liệu, thực thi các transaction và thường được thực thi qua máy chủ Web.

2.3.5 Ứng dụng cơ sở dữ liệu

Các ứng dụng này sử dụng JDBC API để kết nối tới cơ sở dữ liệu. Chúng có thể là Applet hay ứng dụng, nhưng Applet bị giới hạn bởi tính bảo mật.

2.4 Máy ảo Java (JVM-Java Virtual Machine)

Máy ảo Java là trái tim của ngôn ngữ Java. Môi trường Java bao gồm năm phần tử sau:

- Ngôn ngữ
- Định nghĩa Bytecode
- Các thư viện lớp Java/Sun
- Máy ảo Java (**JVM**)
- Cấu trúc của file .class

Các phần tử tạo cho Java thành công là

- Định nghĩa Bytecode
- Cấu trúc của file .class
- Máy ảo Java (**JVM**)

Khả năng cơ động của file .class cho phép các chương trình Java viết một lần nhưng chạy ở bất kỳ đâu. Khả năng này có được nhờ sự giúp đỡ của máy ảo Java.

2.4.1 Máy ảo Java là gì ?

Máy ảo là một phần mềm dựa trên cơ sở máy tính ảo. Nó có tập hợp các lệnh logic để xác định các hoạt động của máy tính. Người ta có thể xem nó như một hệ điều hành thu nhỏ. Nó thiết lập các lớp trừu tượng cho: Phần cứng bên dưới, hệ điều hành, mã đã biên dịch.

Trình biên dịch chuyển mã nguồn thành tập các lệnh của máy ảo mà không phụ thuộc vào phần cứng cụ thể. Trình thông dịch trên mỗi máy sẽ chuyển tập lệnh này thành chương trình thực thi. Máy ảo tạo ra một môi trường bên trong để thực thi các lệnh bằng cách:

- Nạp các file .class
- Quản lý bộ nhớ
- Dọn "rác"

Việc không nhất quán của phần cứng làm cho máy ảo phải sử dụng ngăn xếp để lưu trữ các thông tin sau:

- Các "Frame" chứa các trạng thái của các phương pháp.
- Các toán hạng của mã bytecode.
- Các tham số truyền cho phương pháp.
- Các biến cục bộ.

Khi JVM thực thi mã, một thanh ghi cục bộ có tên "Program Counter" được sử dụng. Thanh ghi này trở tới lệnh đang thực hiện. Khi cần thiết, có thể thay đổi nội dung thanh ghi để đổi hướng thực thi của chương trình. Trong trường hợp thông thường thì từng lệnh một nối tiếp nhau sẽ được thực thi.

Một khái niệm thông dụng khác trong Java là trình biên dịch "Just In Time-JIT". Các trình duyệt thông dụng như Netscape hay IE đều có JIT bên trong để tăng tốc độ thực thi chương trình Java. Mục đích chính của JIT là chuyển tập lệnh bytecode thành mã máy cụ thể cho từng loại CPU. Các lệnh này sẽ được lưu trữ và sử dụng mỗi khi gọi đến.

2.4.2 Quản lý bộ nhớ và dọn rác

Trong C, C++ hay Pascal người lập trình sử dụng phương pháp nguyên thủy để cấp phát và thu hồi bộ nhớ ở vùng "Heap". Heap là vùng bộ nhớ lớn được phân chia cho tất cả các thread.

Để quản lý Heap, bộ nhớ được theo dõi qua các danh sách sau:

Danh sách các vùng nhớ chưa cấp phát.

Danh sách các vùng đã cấp.

Khi có một yêu cầu về cấp phát bộ nhớ, hệ thống xem xét trong danh sách chưa cấp phát để lấy ra khối bộ nhớ đầu tiên có kích cỡ sát nhất. Chiến thuật cấp phát này giảm tối thiểu việc phân mảnh của heap.

"Coalescing" là kỹ thuật khác cũng giảm thiểu việc phân mảnh của heap bằng cách gom lại các vùng nhớ chưa dùng liền nhau. Còn kỹ thuật sắp xếp lại các phần đã dùng để tạo vùng nhớ rảnh lớn hơn gọi là "Compaction".

Java sử dụng hai heap riêng biệt cho cấp phát vùng nhớ tĩnh và vùng nhớ động. Một heap (heap tĩnh) chứa các định nghĩa về lớp, các hằng và danh sách các phương pháp. Heap còn lại (heap động) được chia làm hai phần được cấp phát theo hai chiều ngược nhau. Một bên chứa đối tượng còn một bên chứa con trỏ trỏ đến đối tượng đó.

"Handle" là cấu trúc bao gồm hai con trỏ. Một trỏ đến bảng phương pháp của đối tượng, con trỏ thứ hai trỏ đến chính đối tượng đó. Chú ý rằng khi "compaction" cần cập nhật lại giá trị con trỏ của cấu trúc "handle".

Thuật toán dọn rác có thể áp dụng cho các đối tượng đặt trong heap động. Khi có yêu cầu về bộ nhớ, trình quản lý heap trước tiên kiểm tra danh sách bộ nhớ chưa cấp phát. Nếu không tìm thấy khối bộ nhớ nào phù hợp (về kích cỡ) thì trình dọn rác sẽ được kích hoạt khi hệ thống rảnh. Nhưng khi đòi hỏi bộ nhớ cấp bách thì trình dọn rác sẽ được kích hoạt ngay.

Trình dọn rác gọi hàm Finalize trước khi dọn dẹp đối tượng. Hàm này sẽ dọn dẹp các tài nguyên bên ngoài như các file đang mở. Công việc này không được trình dọn rác thực thi.

2.4.3 Quá trình kiểm tra file .class

Việc kiểm tra được áp dụng cho tất cả các file .class sắp được nạp lên bộ nhớ để đảm bảo tính an toàn. Trình "Class Loader" sẽ kiểm tra tất cả các file .class không thuộc hệ điều hành với mục đích giám sát sự tuân thủ các nghi thức để phát hiện các file .class có nguy cơ gây hư hỏng đến bộ nhớ, hệ thống file cục bộ, mạng hoặc hệ điều hành. Quá trình kiểm tra sẽ xem xét đến tính toàn vẹn toàn cục của lớp.

File .class bao gồm ba phần logic là:

- Bytecode
- Thông tin về Class như phương pháp, giao diện và các giá trị được tập hợp trong quá trình biên dịch.

➤ Các thuộc tính về lớp.

Các thông tin của file .class được xem xét riêng rẽ trong các bảng sau:

- Bảng Field chứa các thuộc tính
- Bảng Method chứa các hàm của class
- Bảng Interface chứa các giao diện và các hằng số

Quá trình kiểm tra file .class được thực hiện ở bốn mức:

- Mức đầu tiên thực hiện việc kiểm tra cú pháp để đảm bảo tính cấu trúc và tính toàn vẹn cú pháp của file .class được nạp.
- Mức thứ hai sẽ xem xét file .class để đảm bảo các file này không vi phạm các nguyên tắc về sự nhất quán ngữ nghĩa.
- Mức thứ ba sẽ kiểm tra bytecode. Trong bước này các thông tin so sánh sẽ là số thông số truyền của hàm, khả năng truy xuất sai chỉ số của mảng, chuỗi, biểu thức.
- Mức thứ tư sẽ kiểm tra trong thời gian thực thi để giám sát các việc còn lại mà ba bước trên chưa làm. Ví dụ như liên kết tới các lớp khác trong khi thực thi, hay kiểm tra quyền truy xuất. Nếu mọi điều thỏa mãn, lớp sẽ được khởi tạo.

2.5 Bộ công cụ phát triển JDK (Java Development Kit)

Sun Microsystem đưa ra ngôn ngữ lập trình Java qua sản phẩm có tên là Java Development Kit (JDK). Ba phiên bản chính là:

Java 1.0 - Sử dụng lần đầu vào năm 1995

Java 1.1 – Đưa ra năm 1997 với nhiều ưu điểm hơn phiên bản trước.

Java 2 – Phiên bản mới nhất

JDK bao gồm Java Plug-In, chúng cho phép chạy trực tiếp Java Applet hay JavaBean bằng cách dùng JRE thay cho sử dụng môi trường thực thi mặc định của trình duyệt.

JDK chứa các công cụ sau:

2.5.1 Trình biên dịch, 'javac'

Cú pháp:

`javac [options] sourcecodename.java`

2.5.2 Trình thông dịch, 'java'

Cú pháp:

`java [options] classname`

2.5.3 Trình dịch ngược, 'javap'

Cú pháp:

`javap [options] classname`

2.5.4 Công cụ sinh tài liệu, 'javadoc'

Cú pháp:

`javadoc [options] sourcecodename.java`

2.5.5 Chương trình tìm lỗi - Debug, 'jdb'

Cú pháp:

`jdb [options] sourcecodename.java`

OR

`jdb -host -password [options] sourcecodename.java`

2.5.6 Chương trình xem Applet , 'appletviewer'

Cú pháp:

`appletviewer [options] sourcecodename.java / url`

2.6 Java Core API

Nhân Java API được đưa ra qua phiên bản JFC 1.1. Một số package thường dùng được liệt kê như sau:

2.6.1 java.lang

Chứa các lớp quan trọng nhất của ngôn ngữ Java. Chúng bao gồm các kiểu dữ liệu cơ bản như **Character, Integer,...** Chúng cũng chứa các lớp làm nhiệm vụ xử lý lỗi và các lớp nhập xuất chuẩn. Một vài lớp quan trọng khác như **String** hay **StringBuffer**.

2.6.2 java.applet

Đây là package nhỏ nhất chứa một mình lớp Applet. Các lớp Applet nhúng trong trang Web đều dẫn xuất từ lớp này.

2.6.3 java.awt

Package này được gọi là Abstract Window Toolkit (AWT). Chúng chứa các tài nguyên dùng để tạo giao diện đồ họa. Một số lớp bên trong là: **Button, GridBagLayout, Graphics**.

2.6.4 java.io

Cung cấp thư viện nhập xuất chuẩn của ngôn ngữ. Chúng cho phép tạo và quản lý dòng dữ liệu theo một vài cách.

2.6.5 java.util

Package này cung cấp một số công cụ hữu ích. Một vài lớp của package này là: **Date, Hashtable, Stack, Vector** và **StringTokenizer**.

2.6.6 java.net

Cung cấp khả năng giao tiếp với máy từ xa. Cho phép tạo và kết nối với Socket hoặc URL.

2.6.7 java.awt.event

Chứa các lớp dùng để xử lý các sự kiện trong chương trình như chuột, bàn phím.

2.6.8 java.rmi

Công cụ để gọi hàm từ xa. Chúng cho phép tạo đối tượng trên máy khác và sử dụng các đối tượng đó trên máy cục bộ.

2.6.9 java.security

Cung cấp các công cụ cần thiết để mã hóa và đảm bảo tính an toàn của dữ liệu truyền giữa máy trạm và máy chủ.

2.6.10 java.sql

Package này chứa Java DataBase Connectivity (JDBC), dùng để truy xuất cơ sở dữ liệu quan hệ như Oracle, SQL Server.

2.7 Các đặc trưng mới của Java 2

Các phiên bản trước của Java chỉ thích hợp để viết các ứng dụng nhỏ trên Web hơn là xây dựng các ứng dụng chạy trên mạng để đảm nhiệm toàn bộ các công việc của của một

công ty hoặc hệ thống phân tán. Java 2 đáp ứng yêu cầu này. Một vài đặc trưng của chúng là:

- **Swing**

Đây là một tập các lớp và giao diện mới dùng để tạo giao diện ứng dụng đồ họa bằng thiết kế "Nhìn và cảm giác" (Look and Feel)

- **Kéo và thả**

Đây là khả năng di chuyển thông tin giữa các ứng dụng hay các phần khác nhau của chương trình.

- **Java 2D API**

Chứa các tập hợp các lớp hỗ trợ cho ảnh và đồ họa hai chiều.

- **Âm thanh**

Tập hợp các đặc trưng âm thanh hoàn toàn mới cho Java.

- **RMI**

RMI (Remote Method Invocation) cho phép các ứng dụng gọi các phương pháp của đối tượng tại máy từ xa và cho phép giao tiếp với chúng.

Tóm tắt

- Java là ngôn ngữ biên dịch và thông dịch
 - Các đặc trưng của Java

Đơn giản

Hướng đối tượng

Độc lập phần cứng

Mạnh

Bảo mật

Phân tán

Đa luồng

Động

- Máy ảo Java
- Java Development Kit
- Các đặc trưng mới của Java 2

Bài tập

Cài đặt Java 2

Gõ các lệnh sau tại dấu nhắc và liệt kê các tham số khác nhau của chúng:

```
javac
```

```
java
```