

Bài 2

Môi trường và công cụ lập trình

Môn học: PF-JAVA

Mục tiêu

- Trình bày được ý nghĩa của JDK và JRE
- Trình bày được ý nghĩa của IDE
- Cài đặt được IntelliJ IDEA
- Tạo được ứng dụng Java đầu tiên
- Sử dụng được các câu lệnh nhập và xuất dữ liệu
- Trình bày được các SCM thông dụng
- Sử dụng được các lệnh Git cơ bản
- Tạo được sản phẩm đầu tiên trên code.org

Chương trình máy tính

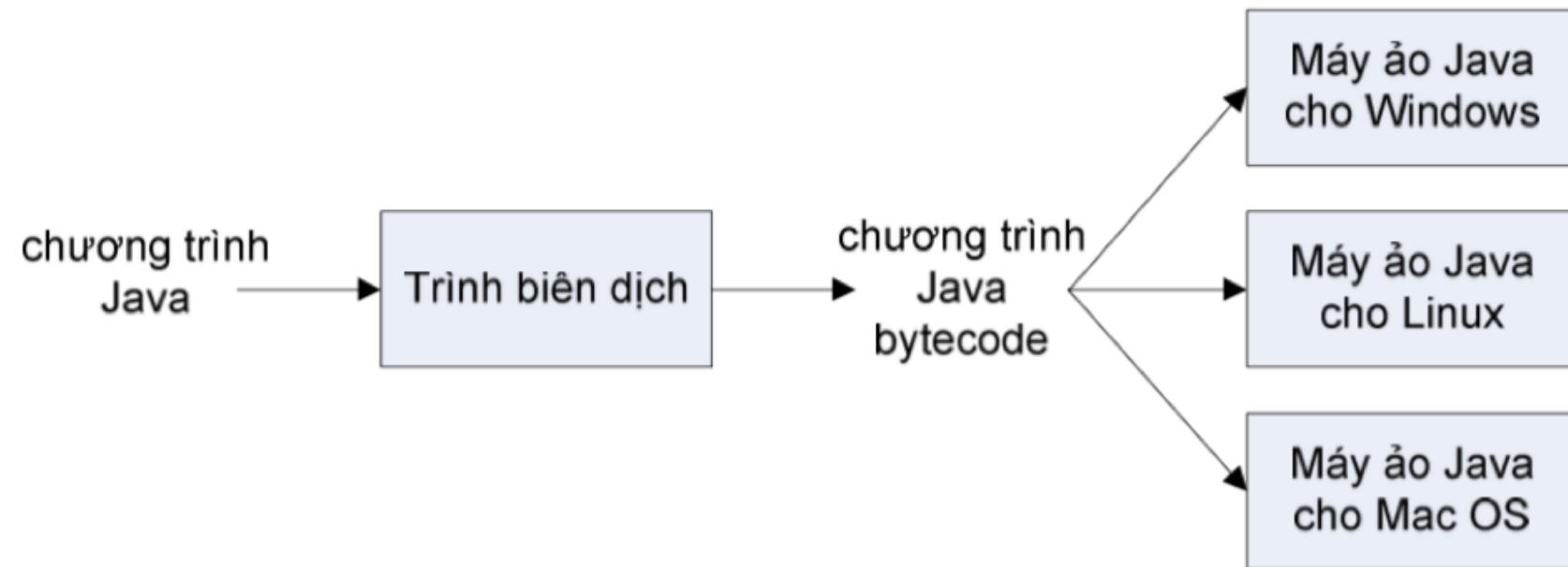
- Chương trình máy tính là các chuỗi **hướng dẫn** (instruction) để máy tính **thực thi** (execute)
- Trong một ngôn ngữ lập trình, các chuỗi hướng dẫn được gọi là **câu lệnh** (statement)

Ngôn ngữ Lập trình Java

- Là một ngôn ngữ lập trình hướng đối tượng
- Có khả năng thực thi ở nhiều loại thiết bị
- Được sử dụng rộng rãi

Write One, Run anywhere

- Java có tính độc lập nền tảng (platform independent)
- Một chương trình Java có thể chạy trên các nền tảng khác nhau mà không phải biên dịch lại



Máy ảo Java (JVM) và byte code

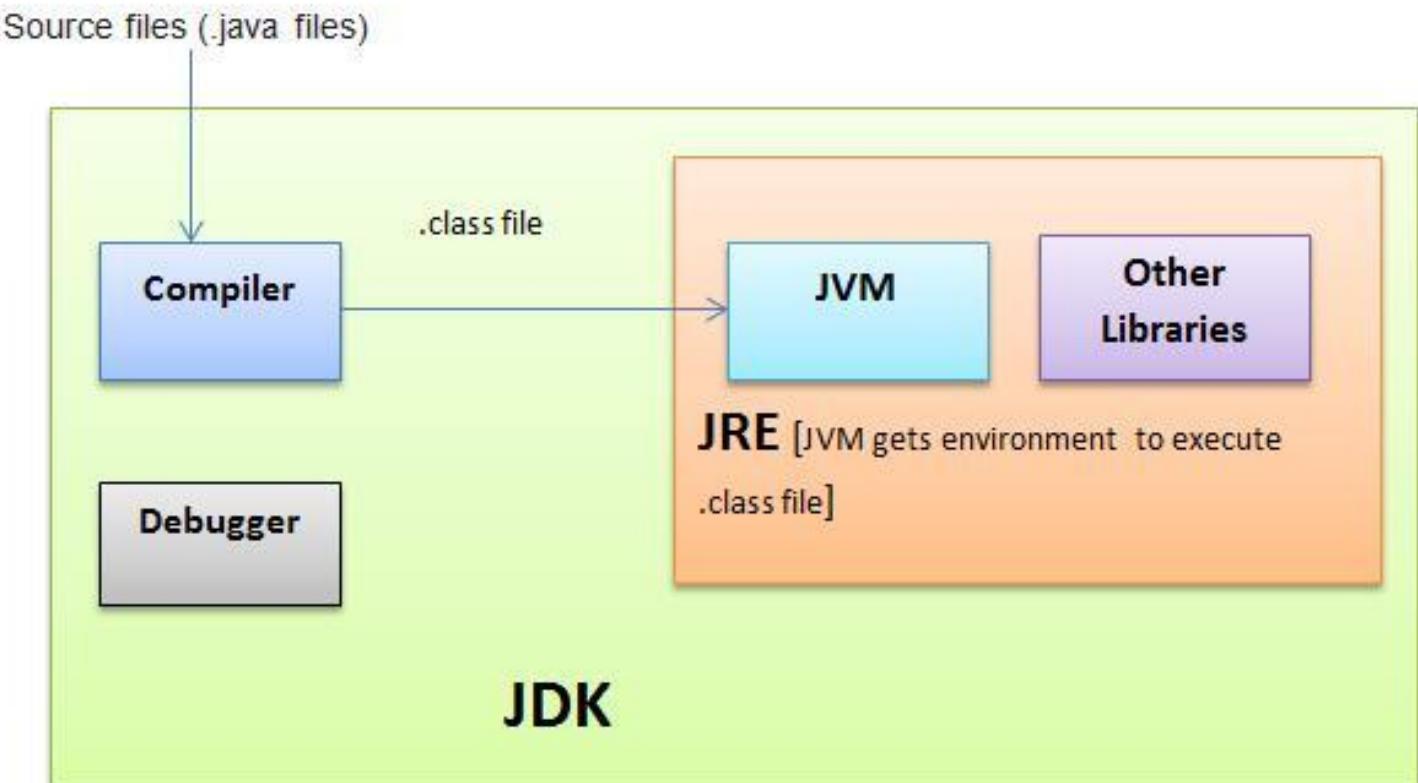
- Một chương trình viết bằng ngôn ngữ bậc cao cần được dịch sang ngôn ngữ máy trước khi có thể được chạy trên máy tính
- Mã nguồn Java không được dịch trực tiếp ra ngôn ngữ máy mà được biên dịch (*compile*) ra byte code
- byte code là ngôn ngữ được thực thi trên một máy tính ảo gọi là JVM (Java Virtual Machine)
- Khi một chương trình Java được thực thi thì JVM sẽ thông dịch (*interpret*) ra ngôn ngữ máy thực sự

JDK vs JRE

JRE giúp thực thi các chương trình Java trong JVM

JDK giúp phát triển và biên dịch mã Java thành chương trình Java

JRE cũng được kèm theo trong JDK



IDE

IDE – Integrated Development Environment

- Integrated Development Environment: Môi trường phát triển tích hợp là một loại phần mềm máy tính có công dụng giúp đỡ các lập trình viên trong việc phát triển phần mềm.
- IDE làm những công việc:
 - Cung cấp trình soạn thảo (*source code editor*) và hỗ trợ soạn thảo.
 - Kết nối với trình biên dịch (*compiler*) và/hoặc trình thông dịch (*interpreter*).
 - Kết nối với trình dịch để *chạy chương trình* nhằm hỗ trợ việc phát triển.
 - Kết nối với cổng *debug* của trình dịch để hỗ trợ *tầm lỗi*.

Các IDE phổ biến cho Java



NetBeans



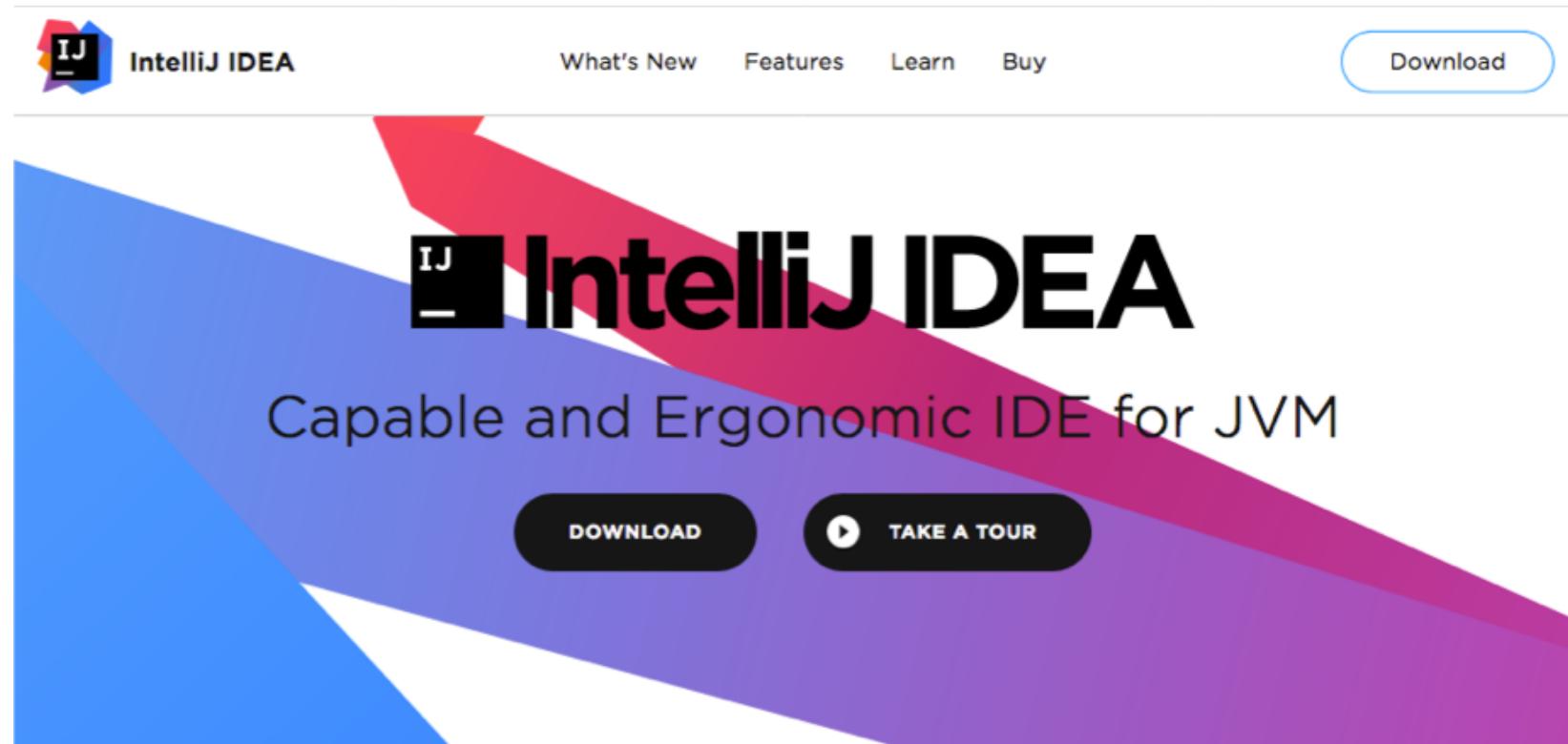
eclipse



IntelliJ IDEA

IntelliJ IDEA

- Tải IntelliJ IDEA tại: <https://www.jetbrains.com/idea/>



Sử dụng IntelliJ IDEA

- IntelliJ IDEA là IDE dùng để soạn thảo mã nguồn
- Tìm bất kỳ lệnh/menu nào:
 - Help/Find Action...
 - ⌘ + ⌘ + A
 - ⌘ + ⌘ + A
- Tìm kiếm trong tất cả các file: ⌘ + ⌘
- Reformat mã:
 - ⌘ + ⌘ + L
 - ⌘ + ⌘ + L

Demo: Tạo ứng dụng Java - 1

- Bước 1: Tạo project mới trên IntelliJ IDEA đặt tên helloworld
- Bước 2: Tạo lớp HelloWorld với nội dung:

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello world");  
    }  
}
```

Có thể sử dụng các cách gõ tắt sau:

- psvm + TAB: viết nhanh hàm main()
- sout + TAB: viết nhanh hàm System.out.println()

Demo: Tạo ứng dụng Java - 2

- Bước 3: Chạy ứng dụng: Chọn mục “Run ‘HelloWorld.main()’”.



Có thể sử dụng phím tắt để chạy ứng dụng.

Câu lệnh (statement)

- Câu lệnh được tạo nên từ các thành phần:
 - Giá trị (value)
 - Toán tử (operator)
 - Biểu thức (expression)
 - Từ khoá (keyword)
 - Chú thích (comment): Không được máy tính thực thi
- Ví dụ đây là 4 câu lệnh trong Java:

```
int x, y, z;  
x = 5;  
y = 6;  
z = x + y;
```

Nhập và xuất dữ liệu

Giao diện người dùng

- Giao diện người dùng (UI – User Interface) là nơi cho phép người dùng tương tác với máy tính
- Các phần mềm máy tính thường có giao diện người dùng
- Có một số loại giao diện người dùng thông dụng như:
 - Command-line interface: Tương tác thông qua dòng lệnh
 - Giao diện người dùng đồ họa (GUI – Graphical User Interface): Tương tác thông qua các thành phần đồ họa
 - Giao diện dựa trên web (Web-based User Interface)
 - Voice User Interface: Tương tác thông qua giọng nói

Hiển thị lên màn hình command-line

Nhập dữ liệu từ bàn phím

- Java cung cấp một số thư viện có sẵn cho phép nhận dữ liệu từ bàn phím
- Đối với giao diện command-line, cách thông thường để nhập dữ liệu từ bàn phím là sử dụng lớp `java.util.Scanner`.
- Chẳng hạn, đoạn mã sau cho phép nhập một số nguyên từ bàn phím:

```
Scanner scanner = new Scanner(System.in);
int x = scanner.nextInt();
```

Nhập dữ liệu từ bàn

- Một số cách để hiển thị kết quả thực thi trong Java:
 - Sử dụng thuộc tính **innerHTML**
 - Sử dụng **document.write()**
 - Sử dụng hàm **alert()**
 - Sử dụng hàm **console.log()**

Source Code Management System (SCMs) – Hệ thống quản lý mã nguồn

Câu hỏi mở

- Tại sao lại cần quản lý mã nguồn?
- Tại sao lại cần quản lý phiên bản mã nguồn?
- Quy trình hoạt động cơ bản của Git là như thế nào?
- Có cách khác ngoài Git để quản lý mã nguồn không?

Tại sao cần quản lý mã nguồn - 1

- Công việc nhiều
- Khó kiểm soát dữ liệu lớn
- Khi các dữ liệu lặp lại, khó có thể theo dõi và phục hồi lại
- Ví dụ:
 - Cá nhân
 - Bạn đang làm việc với một số tập tin và thư mục nào đó. Hàng ngày, bạn tiến hành nhiều thay đổi trên các files này. Vào một ngày xấu trời nào đó, bạn cần quay lại thời điểm của những files này cách đây 2 ngày thì bạn phải làm sao?
 - Nhóm dự án
 - Bạn và vài người bạn của bạn đang làm việc trên một số files (project), làm sao để nhiều người làm việc hiệu quả trên các files này (không có những trường hợp ghi đè nội dung của nhau, xóa nhầm files, quản lý được ai thao tác trên các files nào...)

Tại sao cần quản lý mã nguồn - 2

- Cần quản lý mã nguồn để trợ giúp cho cá nhân và nhóm:
 - Cộng tác thuận lợi
 - Lưu trữ dữ liệu tập chung
 - Dễ dàng chỉnh sửa
 - Tránh trùng lặp, xung đột thông tin

SCMs là gì

- Source Code Management System– Hệ thống quản lý mã nguồn là một phần mềm cung cấp:
 - Cách phối hợp và các dịch vụ giữa các thành viên trong một nhóm phát triển phần mềm
 - Cách quản lý tập tin và kiểm soát phiên bản
 - Cho các nhà phát triển khả năng làm việc đồng thời trên các tập tin, hợp nhất với các thay đổi khác của nhà phát triển khác
 - Theo dõi và kiểm tra các thay đổi được yêu cầu và thực thi
 - Theo dõi tình trạng sửa lỗi và thực thi

VCS là gì

- Version Control System (VCS) – Hệ thống quản lý phiên bản mã nguồn là một phần mềm cung cấp:
 - Khôi phục lại phiên bản cũ của các file
 - Khôi phục lại phiên bản cũ của toàn bộ dự án
 - Xem lại các thay đổi đã được thực hiện theo thời gian
 - Xem ai là người thực hiện thay đổi cuối cùng có thể gây ra sự cố
 - Khôi phục lại các file vô tình xoá mất

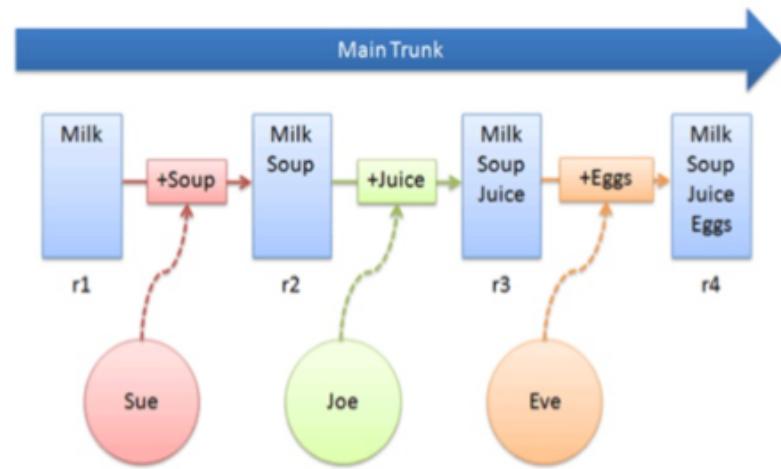
Ai là người sử dụng?

- Nhà phát triển dự án
- Kiểm thử viên dự án
- Người sử dụng

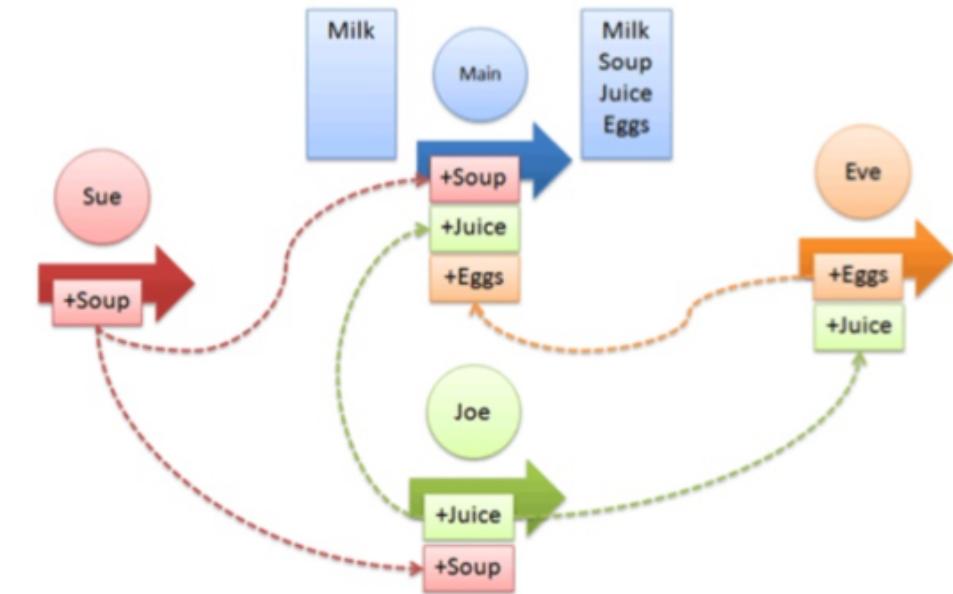
Một vài công cụ quản lý mã nguồn

- Subversion
- Git
- Mercurial
- Bazaar
- CVS

Các kiểu của hệ thống quản lý phiên bản (VCS)



Centralized – VCS



Distributed- VCS

Git và GitHub

- Git là một hệ thống điều khiển phiên bản (version control system)
- Git được sử dụng để quản lý mã nguồn (source code) và ghi nhận các thay đổi
- GitHub là một dịch vụ Git được cung cấp miễn phí
- GitHub có phiên bản trả phí dành cho các doanh nghiệp

Các thuật ngữ

- Commit: một phiên bản, được đánh ID, chứa toàn bộ nội dung các file tại thời điểm được đánh phiên bản.
- Tag: “thẻ” được đính vào commit.
- Repository: kho chứa, chứa toàn bộ các commit và các thẻ
- Remote: kho chứa không nằm ở máy tính hiện tại

Các hoạt động cơ bản khi làm việc với Git

- git clone: nhân bản (không phải sao chép) một repository
- git init: Khởi tạo một repository
- git add: ghi nhận file vào commit kế tiếp
- git commit: đánh phiên bản dựa trên các file đã được ghi nhận
- git push: đồng bộ các commit mới tạo tới kho chứa remote



[Thực hành] Tạo dự án mới và đồng bộ tới Github