



Java căn bản (Phần 1)

Nguyễn Xuân Nam

ngxnam253@gmail.com

Agenda

- Buổi 1 - Java cơ bản 1: Giới thiệu, cài đặt, hello world, biến, Toán tử
- Buổi 2 - Java cơ bản 2: Chuỗi, ngày giờ, toán học, nhập liệu
- Buổi 3 - Java cơ bản 3: Câu lệnh rẽ nhánh, vòng lặp
- Buổi 4 - Java cơ bản 4: Mảng, phương thức, thực hành
- Buổi 5 - OOP 1: Lớp, đối tượng, thuộc tính, phương thức
- Buổi 6 - OOP 2: Đóng gói, kế thừa, đa hình
- Buổi 7 - OOP 3: Trừu tượng, giao thức
- Buổi 7 - Thực hành lập trình OOP, giao bài tập
- Buổi 8 - Chữa bài tập OOP, giải đáp thắc mắc

Buổi 1: Java cơ bản 1



Nội dung buổi 1

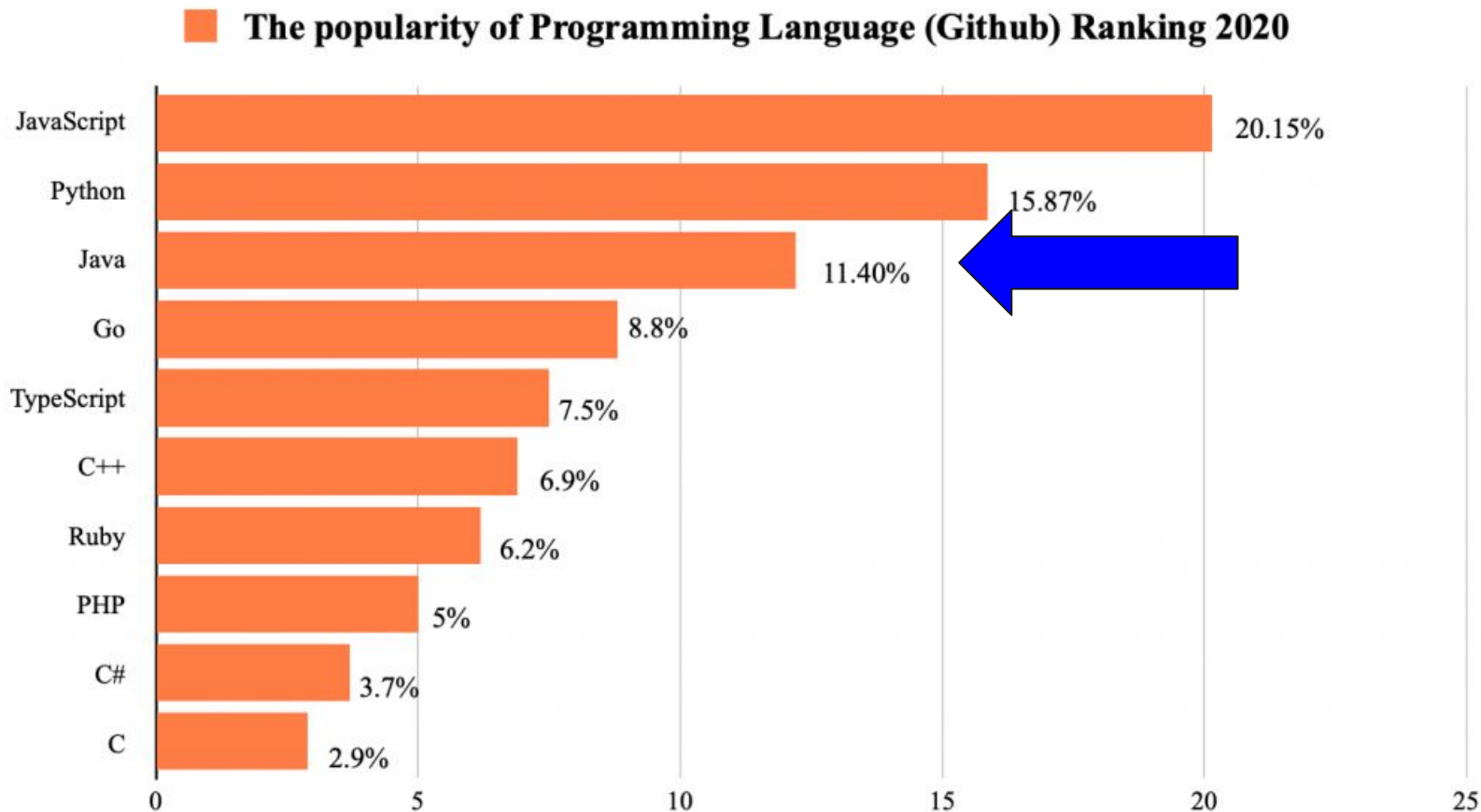
- Giới thiệu về Java
- Cài đặt môi trường
- Hello World
- Biến
- Kiểu dữ liệu
- Toán tử
- Thực hành

Giới thiệu về Java

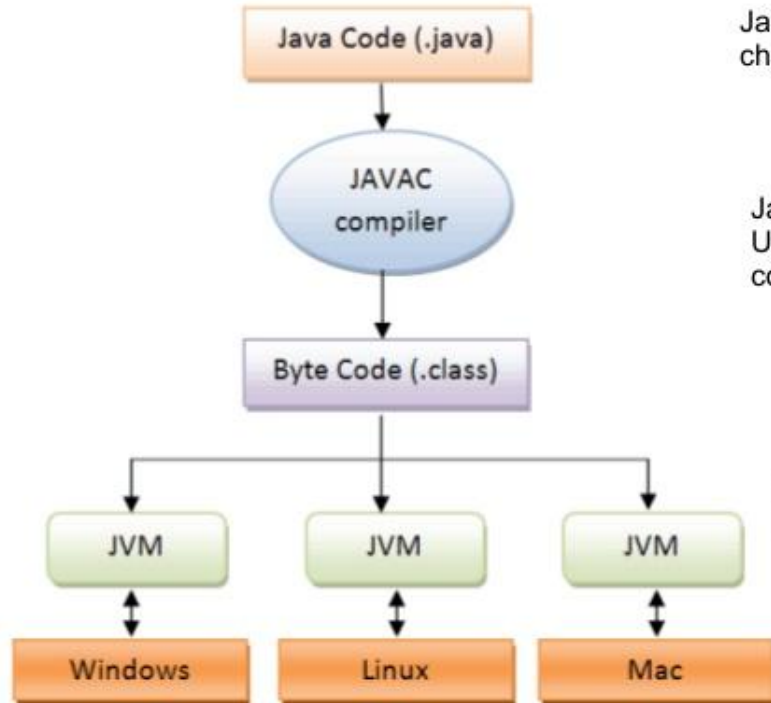
Java là một ngôn ngữ lập trình hướng đối tượng, đa mục đích có khả năng làm việc trong hầu như bất kỳ nền tảng nào mà không cần phải biên dịch lại. Đặc trưng này đã được thể hiện trong câu slogan của nó, "Viết một lần, chạy mọi nơi."



Giới thiệu về Java



Giới thiệu về Java



Java class is written in Unicode characters.

Code java được viết bằng các ký tự unicode

Java compiler convert these Unicode characters into Byte code.

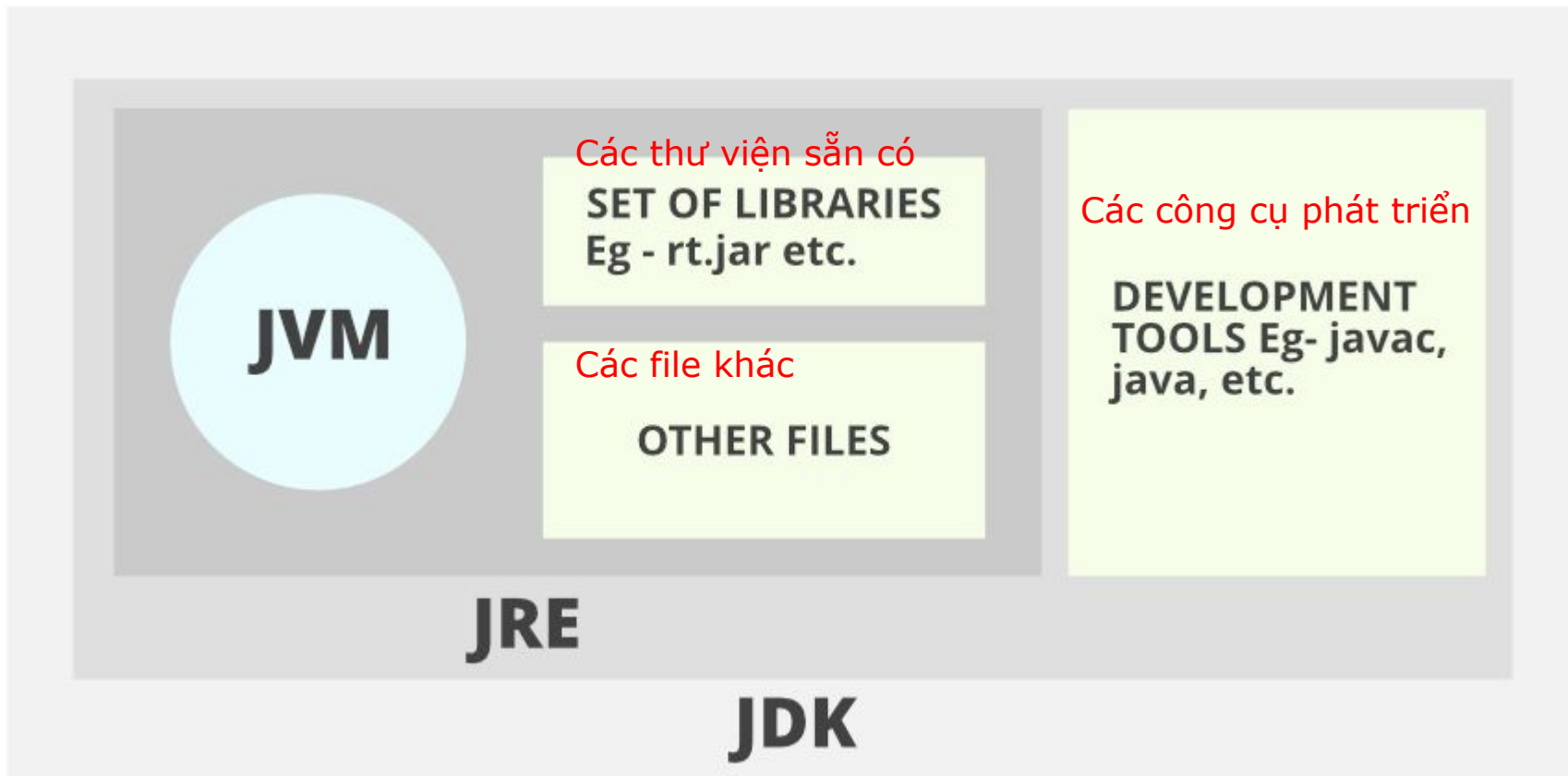
Trình biên dịch chuyển các ký tự unicode sang Byte code

Java Byte code can only be understandable by JVM.

Java Byte code được chạy trên JVM, mà mỗi hệ điều hành đều có một phiên bản JVM và tất cả đều chạy được Byte code.

JVM is native code and specific to OS

Giới thiệu về Java



Cài đặt môi trường

1. Tải Java JDK 11

Truy cập: <https://www.oracle.com/java/technologies/javase/jdk11-archive-downloads.html>









ORACLE

Products Industries Resources Customers Partners Developers Events

🔍

👤 View Accounts

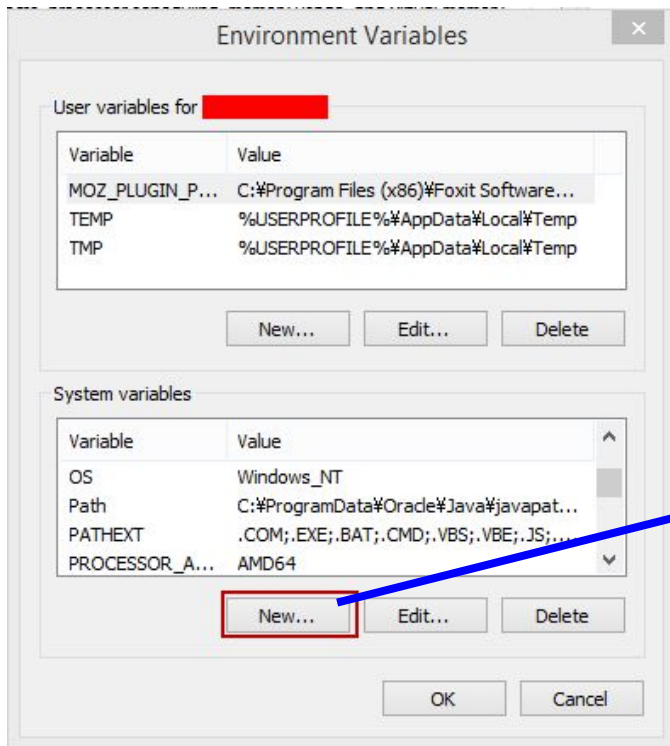
📞 Contact Sales

Linux x64 Debian Package	138.35 MB	 jdk-11.0.15_linux-x64_bin.deb
Linux x64 RPM Package	144.53 MB	 jdk-11.0.15_linux-x64_bin.rpm
Linux x64 Compressed Archive	161.0 MB	 jdk-11.0.15_linux-x64_bin.tar.gz
macOS x64 DMG Installer	154.92 MB	 jdk-11.0.15_osx-x64_bin.dmg
macOS x64 Compressed Archive	155.43 MB	 jdk-11.0.15_osx-x64_bin.tar.gz
Solaris SPARC Compressed Archive	184.57 MB	 jdk-11.0.15_solaris-sparcv9_bin.tar.gz
Windows x64 Installer	140.3 MB	 jdk-11.0.15_windows-x64_bin.exe
Windows x64 Compressed Archive	157.93 MB	 jdk-11.0.15_windows-x64_bin.zip

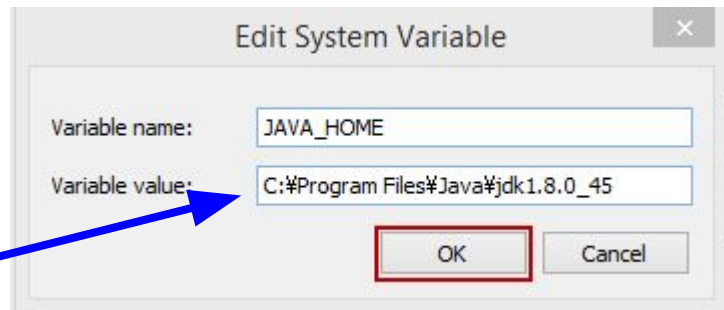
Cài đặt môi trường

2. Sau khi cài đặt trên windows, thực hiện thiết lập JAVA_HOME cho java

MyComputer --> Properties -> Advanced system settings -> Environment Variables

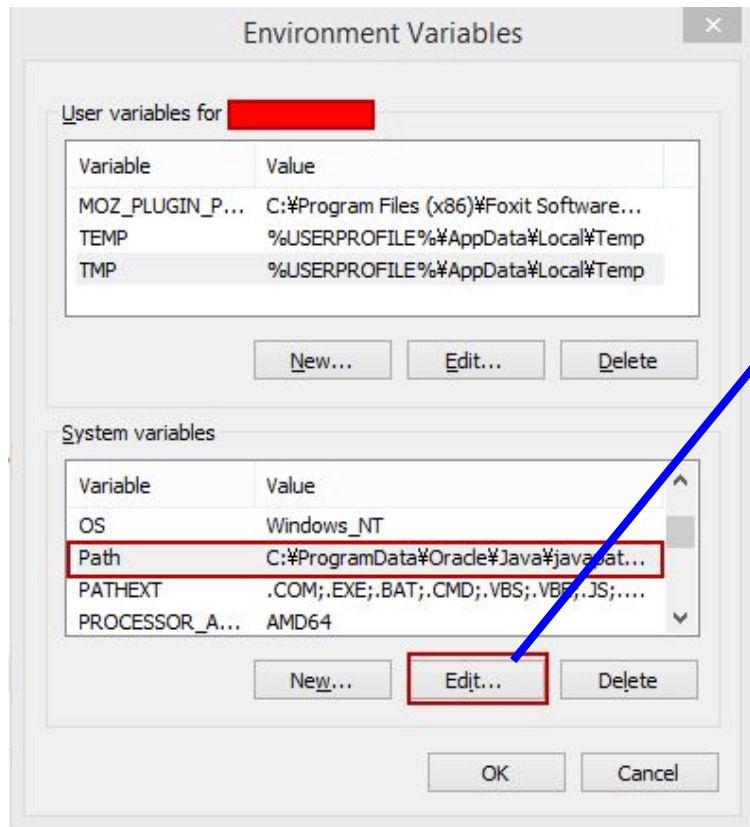


Nhập đường dẫn cài đặt java vào đây



Cài đặt môi trường

3. Sửa biến Path



Thêm đường dẫn trỏ đến thư mục bin trong thư mục cài đặt java.

Ví dụ:

C:\Program Files\Java\jdk1.8.0_45\bin

Chú ý cần có dấu ; để phân biệt các đường dẫn.

Thiết lập xong cần logout hoặc khởi động lại máy. Sau đó kiểm tra bằng câu lệnh sau trên terminal.

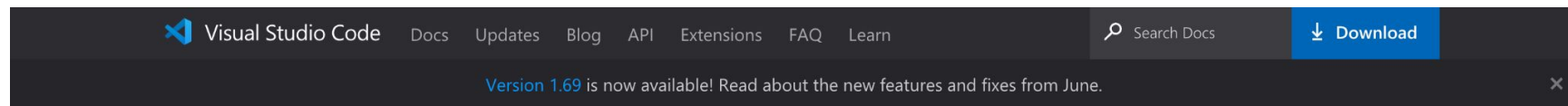
4. Kiểm tra version

```
(base) namnx@Nams-MacBook-Air TM_JavaCore % java -version
java version "11.0.15" 2022-04-19 LTS
Java(TM) SE Runtime Environment 18.9 (build 11.0.15+8-LTS-149)
Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.15+8-LTS-149, mixed mode)
(base) namnx@Nams-MacBook-Air TM_JavaCore %
```

Cài đặt môi trường


5. Cài đặt Visual Studio Code

Download URL: <https://code.visualstudio.com/download>



Download Visual Studio Code


Free and built on open source. Integrated Git, debugging and extensions.



↓ Windows

Windows 8, 10, 11

User Installer	64 bit	32 bit	ARM
System Installer	64 bit	32 bit	ARM
.zip	64 bit	32 bit	ARM




↓ .deb

Debian, Ubuntu

.deb	64 bit	ARM	ARM 64
.rpm	64 bit	ARM	ARM 64
.tar.gz	64 bit	ARM	ARM 64

↓ .rpm

Red Hat, Fedora, SUSE



↓ Mac

macOS 10.11+

.zip Universal Intel Chip Apple Silicon

Cài đặt môi trường

6. Cài đặt Java Extension Pack trong VSCODE

The screenshot shows the Visual Studio Code interface with the Extensions Marketplace open. The left sidebar lists several extensions, including 'Java Extension Pack', 'Maven for Java', 'Debugger for Java', 'Language Support for Java(TM)', 'Java Test Runner', 'Anaconda Extension Pack', and 'Chinese (Simplified) Language Pack'. The 'Java Extension Pack' is highlighted, and its details are shown in the main panel. The extension is by Microsoft, has 5,223,426 downloads, and a 4.5-star rating. It is currently in the 'Installing' state. Below the details, there is a section titled 'Extensions Included' which lists 'Language Support for Java™ by Red Hat', 'Code Navigation', and 'Auto Completion'. A notification dialog is displayed in the bottom right corner, stating: 'Cannot activate the 'Java Test Runner' extension because it depends on the 'Debugger for Java' extension, which is not loaded. Would you like to reload the window to load the extension?'. A 'Reload Window' button is visible in the dialog. The status bar at the bottom shows 0 errors, 0 warnings, and 0 info messages.

Extension: Java Extension Pack — ptc-angular-structure

EXTENSIONS: MARKETPLACE

Java Extension Pack

Java Extension Pack 0.7.1
Popular extensions for Java development a...
Microsoft Installing

Maven for Java 0.17.1
Manage Maven projects, execute goals, ge...
Microsoft Installing

Debugger for Java 0.19.0
A lightweight Java debugger for Visual Stu...
Microsoft Installing

Language Support for Java(TM) ... 0.46.0
Java Linting, Intellisense, formatting, refact...
Red Hat Installing

Java Test Runner 0.18.1
Run and debug JUnit or TestNG test cases
Microsoft Installing

Anaconda Extension Pack 1.0.1
The Anaconda Extension Pack is a set of ex...
Microsoft Install

Chinese (Simplified) Language Pa... 1.36.1
中文(简体)
Microsoft Install

Extension: Java Extension Pack x

Java Extension Pack vscjava.vscode-java-pack Preview

Microsoft | 5,223,426 | ★★★★★ | Repository | License

Popular extensions for Java development and more.

Installing

Details Contributions Changelog

Java Extension Pack

Java Extension Pack is a collection of popular extensions that can help write, test and debug Java applications in Visual Studio Code. Check out [Java in VS Code](#) for more information.

Extensions Included

By installing Java Extension Pack, the following extensions will be installed:

- Language Support for Java™ by Red Hat
 - Code Navigation
 - Auto Completion

Cannot activate the 'Java Test Runner' extension because it depends on the 'Debugger for Java' extension, which is not loaded. Would you like to reload the window to load the extension?

Reload Window

Hello world

The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left and the Main.java editor on the right. The Explorer sidebar shows a project named 'TM_JAVACORE' with a subfolder 'src'. The 'src' folder is expanded, and 'Main.java' is selected. The 'Main.java' file is highlighted with a green box. The editor shows the following code:

```
1 public class Main {  
    Run | Debug  
2     public static void main(String[] args) {  
3         System.out.println(x: "Hello World");  
4     }  
5 }
```

Arrows indicate the steps: a red arrow points from the text 'B1. Tạo folder TM_JAVACORE, sau đó tạo folder src' to the 'TM_JAVACORE' folder in the Explorer; a green arrow points from the text 'B2. Tạo file Main.java trong folder src' to the 'Main.java' file in the Explorer; and an orange arrow points from the text 'B3. Nhập nội dung trong file Main.java như hình' to the code in the editor.

B1. Tạo folder TM_JAVACORE, sau đó tạo folder src

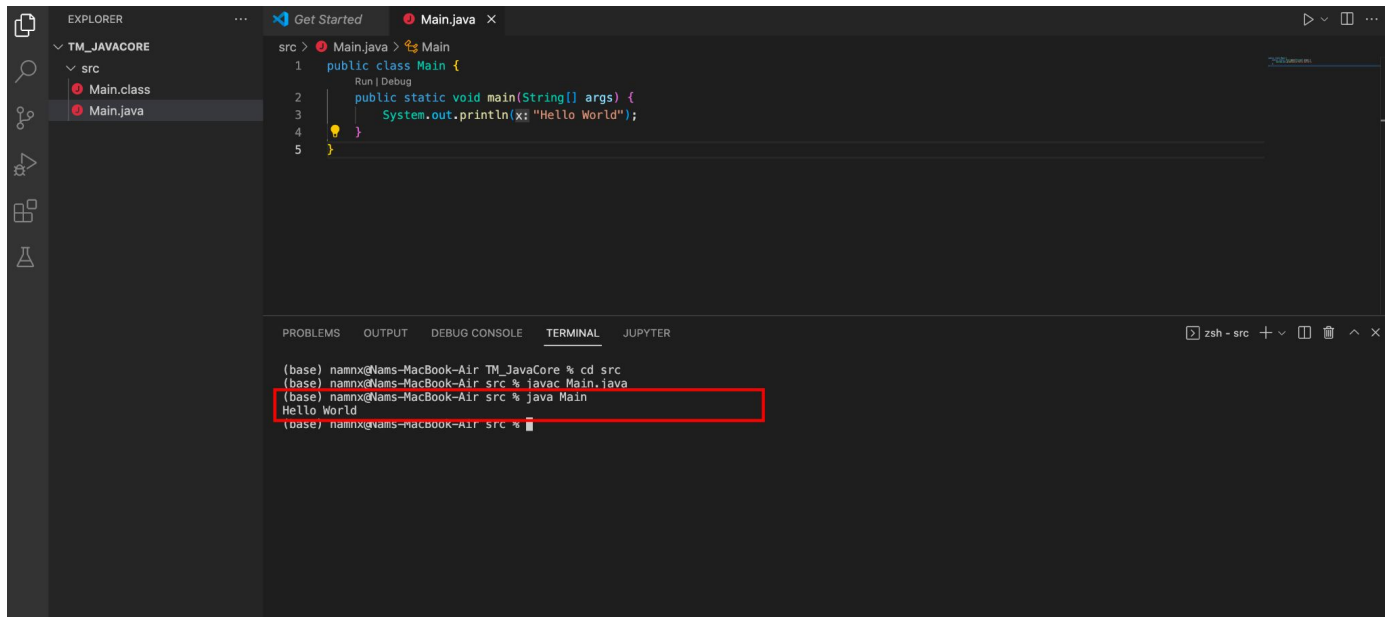
B2. Tạo file Main.java trong folder src

B3. Nhập nội dung trong file Main.java như hình

Hello world

1. Run bằng command line

- > javac Main.java
Tạo ra file Main.class
- > java Main
Hello World



The screenshot shows an IDE with a dark theme. On the left, the 'EXPLORER' sidebar shows a project named 'TM_JAVACORE' with a subfolder 'src' containing 'Main.class' and 'Main.java'. The main editor displays 'Main.java' with the following code:

```
src > Main.java > Main
1 public class Main {
2     public static void main(String[] args) {
3         System.out.println(x; "Hello World");
4     }
5 }
```

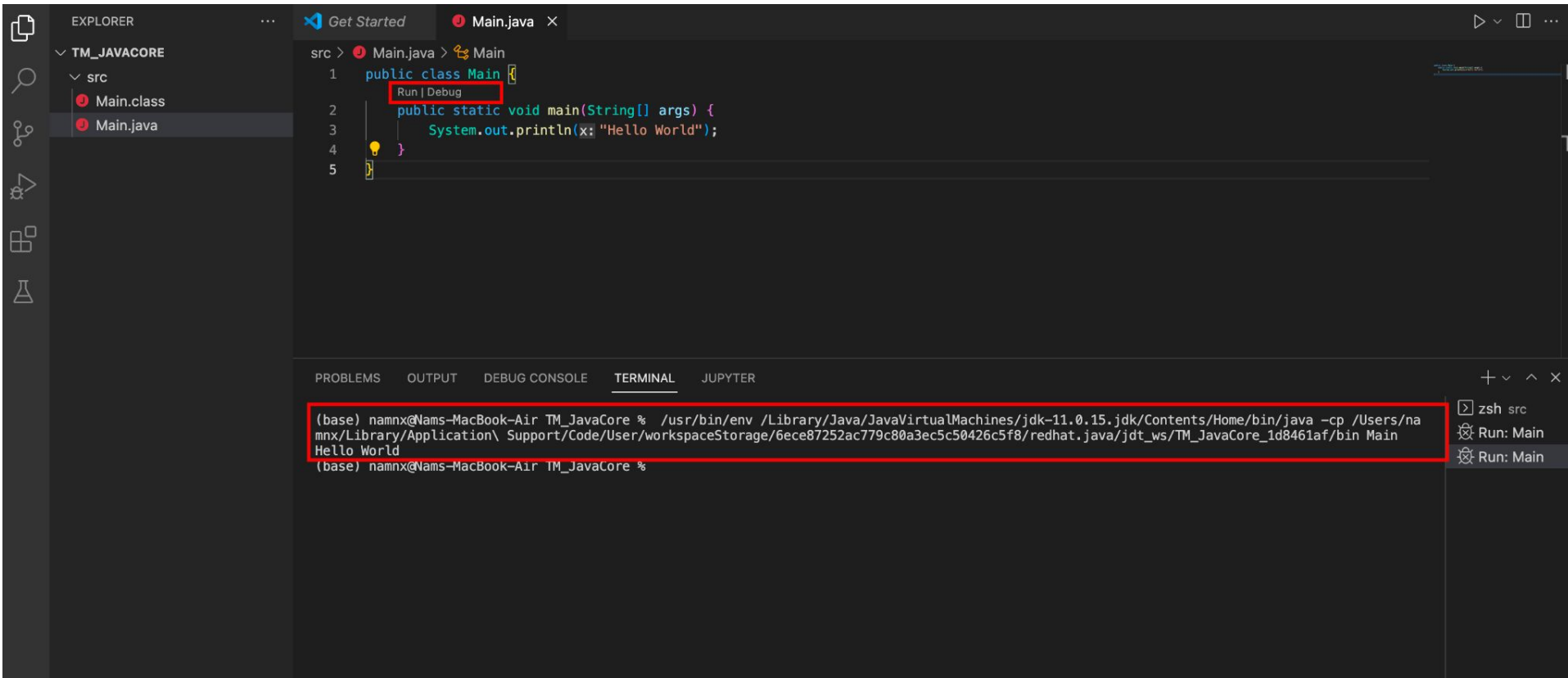
Below the editor is a 'TERMINAL' panel. It shows the following commands and output:

```
(base) namnx@Nams-MacBook-Air TM_JavaCore % cd src
(base) namnx@Nams-MacBook-Air src % javac Main.java
(base) namnx@Nams-MacBook-Air src % java Main
Hello World
(base) namnx@Nams-MacBook-Air src %
```

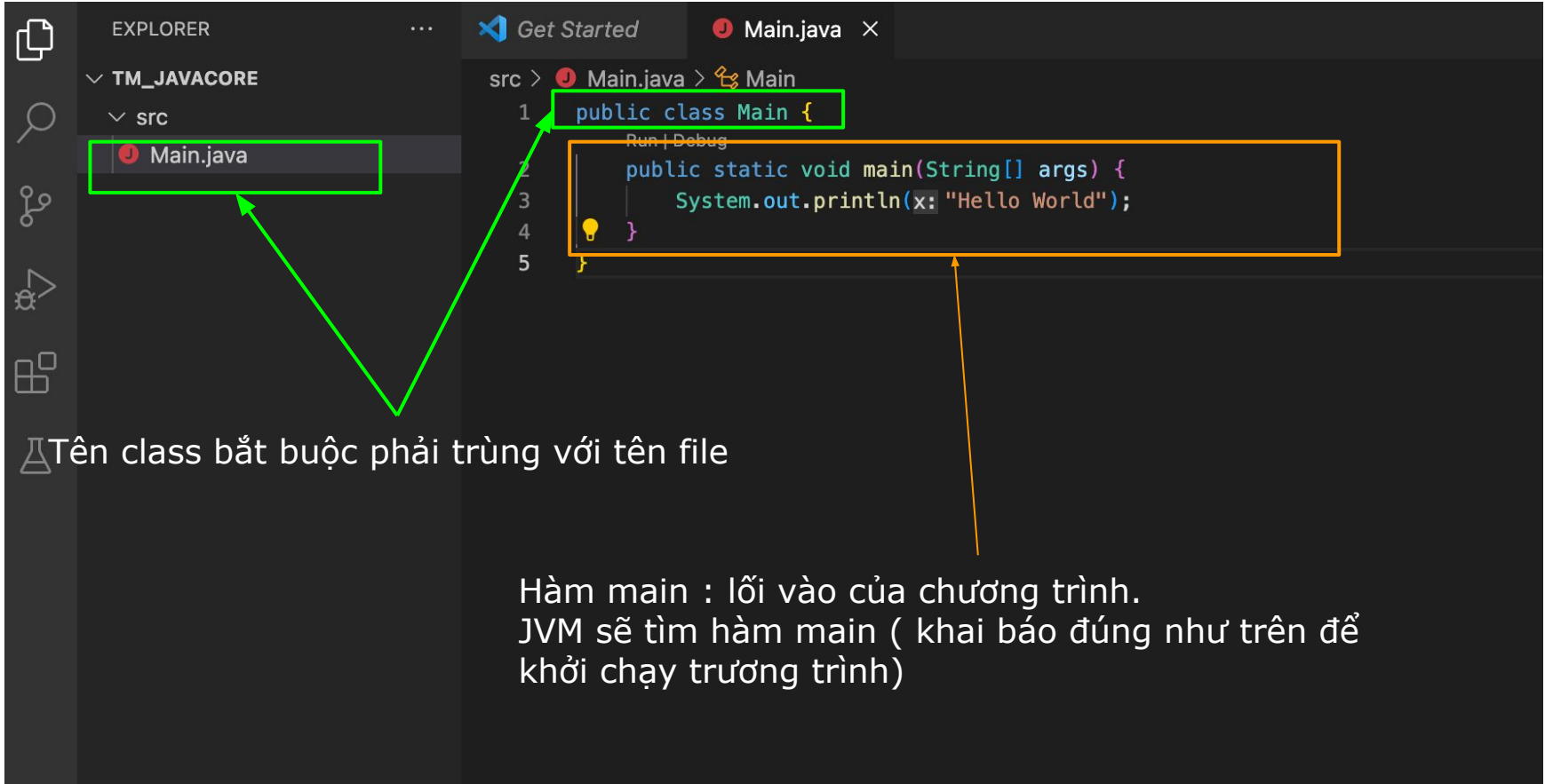
The output 'Hello World' is highlighted with a red rectangle.

Hello world

1. Run hoặc debug bằng vscode



Hello world

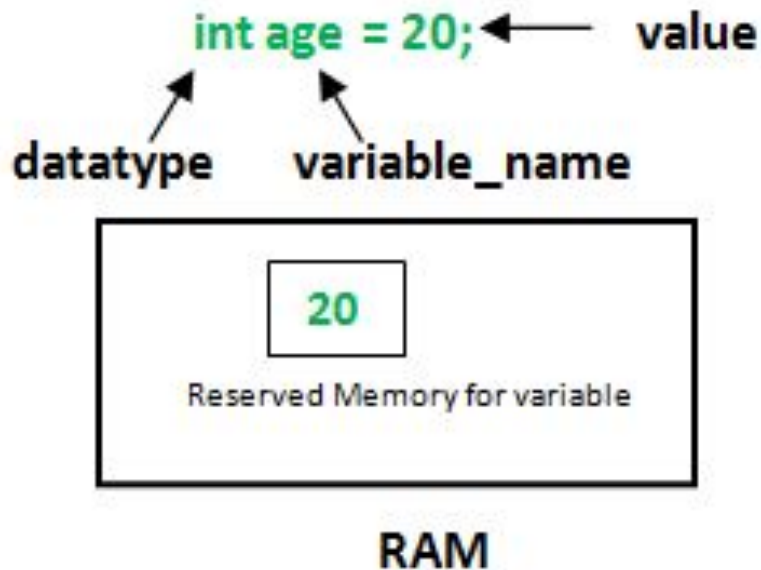


```
src > Main.java > Main
1 public class Main {
2     public static void main(String[] args) {
3         System.out.println(x: "Hello World");
4     }
5 }
```

Tên class bắt buộc phải trùng với tên file

Hàm main : lối vào của chương trình.
JVM sẽ tìm hàm main (khai báo đúng như trên để khởi chạy chương trình)

Biến (Variables)



Biến để lưu giá trị, dữ liệu.

datatype: Kiểu dữ liệu của biến

Ví dụ:

- **String** : Lưu trữ chuỗi các ký tự, ví dụ: "Hello world". Giá trị của chuỗi được đặt trong dấu "" (double quotes)
- **int** : Lưu trữ các số nguyên, không lưu thập phân, ví dụ 123 hoặc -123
- **float** : Lưu trữ các số thập phân, ví dụ 19.99 hoặc -19.99
- **char** : Lưu các ký tự, ví dụ 'a' hoặc 'B'. Các ký tự được đặt dấu ' ' (single quote)
- **boolean** : Lưu các giá trị có 2 trạng thái: **true** hoặc **false**

Biến (Variables)

In giá trị của biến (để kiểm tra dữ liệu)

```
String name = "Nam";  
System.out.println("Hello " + name);  
  
int a = 3;  
System.out.println("giá trị biến a: " + a);
```

Khai báo nhiều biến

```
int a = 1, b = 2, c = 3;  
System.out.println(a + b + c);  
  
int x, y, z;  
x = y = z = 4;  
System.out.println(x + y + z);
```

Biến (Variables)

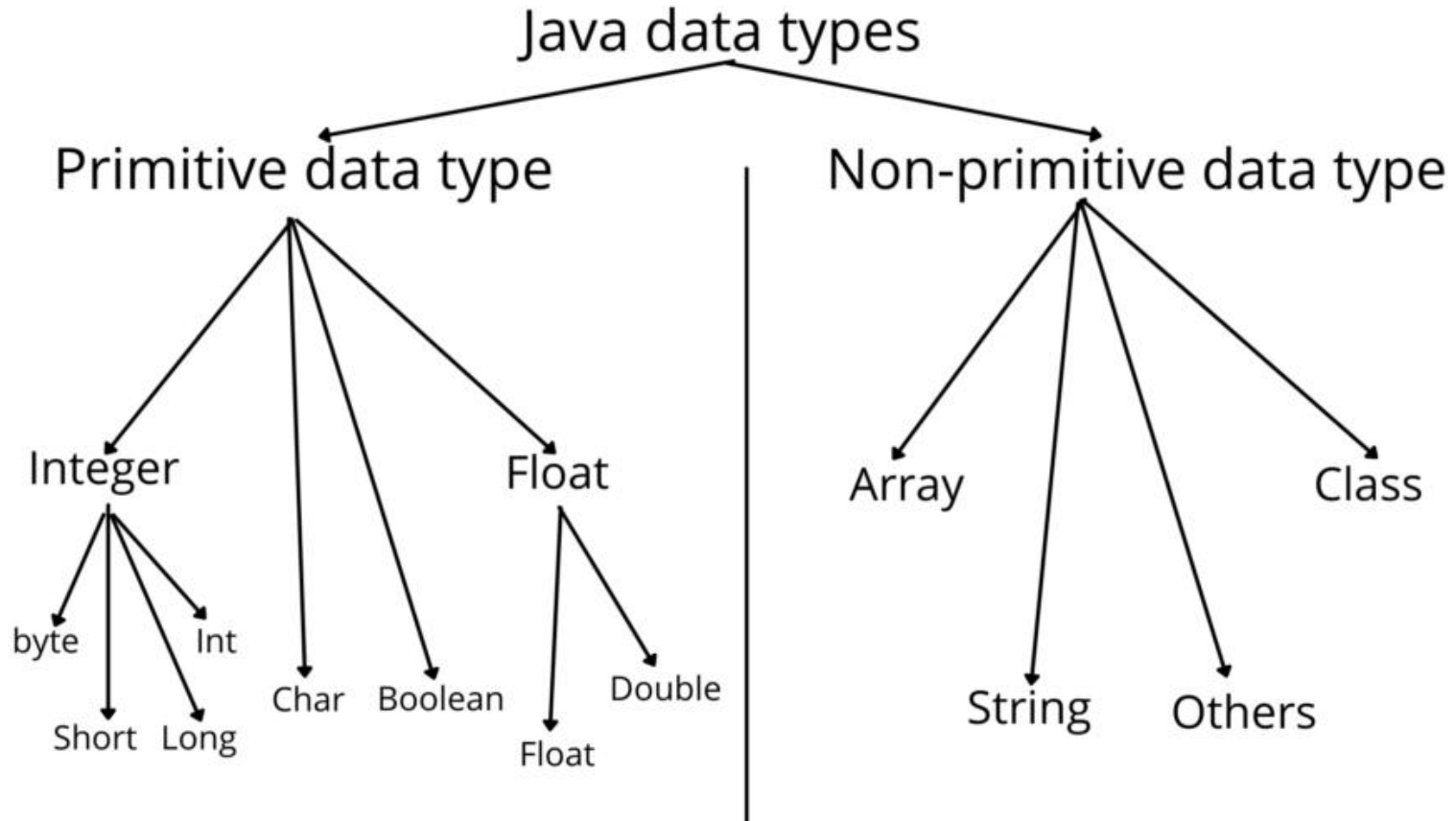
Biến cố định, không thay đổi được giá trị

```
final int var = 10;  
var = 15; //Lỗi
```

Quy tắc đặt tên biến

- Chỉ được bắt đầu bằng một ký tự(chữ), hoặc một dấu gạch dưới(_), hoặc một ký tự dollar(\$)
- Tên biến không được chứa khoảng trắng
- Bắt đầu từ ký tự thứ hai, có thể dùng ký tự(chữ), dấu gạch dưới(_), hoặc ký tự dollar(\$)
- Không được trùng với các từ khóa
- Có phân biệt chữ hoa và chữ thường

Kiểu dữ liệu (Data Types)



Kiểu dữ liệu (Data Types)

Kiểu số nguyên (Integer)

Kiểu dữ liệu	Miền giá trị	Giá trị mặc định	Kích cỡ mặc định
byte	-128 đến 127	0	1 byte
short	-32768 đến 32767	0	2 byte
int	-2^{31} đến $2^{31}-1$	0	4 byte
long	-2^{63} đến $2^{63}-1$	0L	8 byte

Ví dụ:

```
int age;
```

```
age = 35;
```

```
long salary = 4000000L;
```

Kiểu dữ liệu (Data Types)

Kiểu số thực

Kiểu dữ liệu	Miền giá trị	Giá trị mặc định	Kích cỡ mặc định
float	1.40239846 x 10 ⁻⁴⁵ 3.40282347 x 10 ³⁸ ,	0.0f	4 byte
double	4.9406564584124654 x 10 ⁻³²⁴ 1.7976931348623157 x 10 ³⁰⁸	0.0d	8 byte

Ví dụ:

```
float weight;
```

```
weight = 40f;
```

```
double height = 1.6; //Vì double là kiểu mặc định cho kiểu số thực, nên có thể viết gọn hơn
```

Kiểu dữ liệu (Data Types)

Kiểu ký tự (char)

- Kích thước 2 byte
- Lưu được $2^16 = 65536$ ký tự
- Giá trị mặc định **null**
- Có thể khai báo bằng ký tự hoặc bằng số

Ví dụ:

```
char a = 'a';
```

```
char b = '5';
```

```
char c = 65; //theo bảng mã ASCII c == 'A'
```

Kiểu ký luận lý (boolean)

```
boolean isCheck = false;
```

```
int x = 10;
```

```
boolean flag = x % 2 == 0; //Trả về true do x chia hết cho 2
```


Kiểu dữ liệu (Data Types)

Tương quan giữa kích thước các kiểu dữ liệu và RAM

byte	1
short	1 2
int	1 2 3 4
long	1 2 3 4 5 6 7 8
float	1 2 3 4
double	1 2 3 4 5 6 7 8
RAM (1GB)	1 2 3 4 5 6 7 8 9 10 ... 1.073.741.824

Kiểu dữ liệu (Data Types)

Ép kiểu dữ liệu

- Widening Casting(Implicit)

```
int myInt = 10;  
double myDouble = myInt; // Tự động chuyển từ số nguyên sang số thập phân  
System.out.println(myInt); // Kết quả: 10  
System.out.println(myDouble); // Kết quả: 10.0
```

byte → short → int → long → float → double



- Narrowing Casting(Explicitly done)

double → float → long → int → short → byte



```
double myDouble = 1.23d;  
int myInt = (int) myDouble; // ép kiểu thủ công từ double -> int  
System.out.println(myDouble); // Kết quả: 1.23  
System.out.println(myInt); // Kết quả: 1
```

Toán tử (Operators)

Toán tử số học

Toán tử	Ý nghĩa	Ví dụ
+	Cộng	$a + b = 40$
-	Trừ	$a - b = 20$
*	Nhân	$a * b = 300$
/	Chia lấy nguyên	$a / b = 3$
%	Chia lấy dư	$a \% b = 0$
++	Tăng 1	$a++ = 31$
--	Giảm 1	$b-- = 9$

Toán tử (Operators)

Toán tử quan hệ

Toán tử	Ý nghĩa	Ví dụ
==	So sánh bằng	$a == b \Rightarrow \text{false}$
!=	So sánh khác	$a != b \Rightarrow \text{true}$
>	So sánh lớn hơn	$a > b \Rightarrow \text{true}$
<	So sánh nhỏ hơn	$a < b \Rightarrow \text{false}$
>=	So sánh lớn hơn hoặc bằng	$a >= b \Rightarrow \text{true}$
<=	So sánh nhỏ hơn hoặc bằng	$a <= b \Rightarrow \text{false}$

Toán tử (Operators)

Toán tử logic

Toán tử	Ý nghĩa	Ví dụ
&&	Toán tử và	<code>c && d => false</code>
	Toán tử hoặc	<code>c d => true</code>
!	Toán tử phủ định	<code>!c => c = false</code>

Toán tử (Operators)

Toán tử gán

Toán tử	Ý nghĩa	Ví dụ
=	Toán tử đơn giản, gán giá trị toán hạng bên phải cho toán hạng trái	$a = 10$
+=	Thêm giá trị toán hạng phải tới toán hạng trái và gán giá trị đó cho toán hạng trái	$c += a \Rightarrow c = c + a$
-=	Trừ đi giá trị toán hạng phải từ toán hạng trái và gán giá trị này cho toán hạng trái	$c -= a \Rightarrow a = c - a$
*=	Nhân giá trị toán hạng phải với toán hạng trái và gán giá trị này cho toán hạng trái	$c *= a \Rightarrow c = c * a$
/=	Chia toán hạng trái cho toán hạng phải và gán giá trị này cho toán hạng phải	$c /= a \Rightarrow c = c / a;$
%=	Lấy phần dư của phép chia toán hạng trái cho toán hạng phải và gán cho toán hạng trái	$c \% = a \Rightarrow c = c \% a$

Toán tử (Operators)

Toán tử ba ngôi (Ternary operator)

Cú pháp:

<điều kiện> ? <Biểu thức 1> : <Biểu thức 2>;

Điều kiện ở đây phải có kết quả là **true** hoặc **false**

Ví dụ:

```
int a = 4;
```

```
int b = 2;
```

```
String s = (a % b == 0) ? "a chia hết cho b" : "a không chia  
hết cho b";
```

```
System.out.println(s);
```

Thực hành

Bài tập

Buổi 2: Java cơ bản 2



Nội dung buổi 2

- Chuỗi (String)
- Hằng số (Constant)
- Ngày giờ (Date Time)
- Toán học (Math)
- Nhập dữ liệu với Scanner
- Thực hành

Chuỗi (Strings)

Định nghĩa

String str = "Tôi là coder";

- Là tập hợp các ký tự
- Đặt trong dấu nháy kép (double quote)
- Đánh chỉ mục từ 0 (zero-based index)
- Chiều dài (length) là số ký tự

0	1	2	3	4	5	6	7	8	9	10	
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	
T	ô	i		l	à		c	o	d	e	r

Truy xuất ký tự trong chuỗi:

- str[0] -> ký tự 'T'
- str[3] -> ký tự ' ' (ký tự trắng/space)
- str[10] -> ký tự 'r'
- str[11] -> **Lỗi**

Có 2 cách khai báo String:

//literal

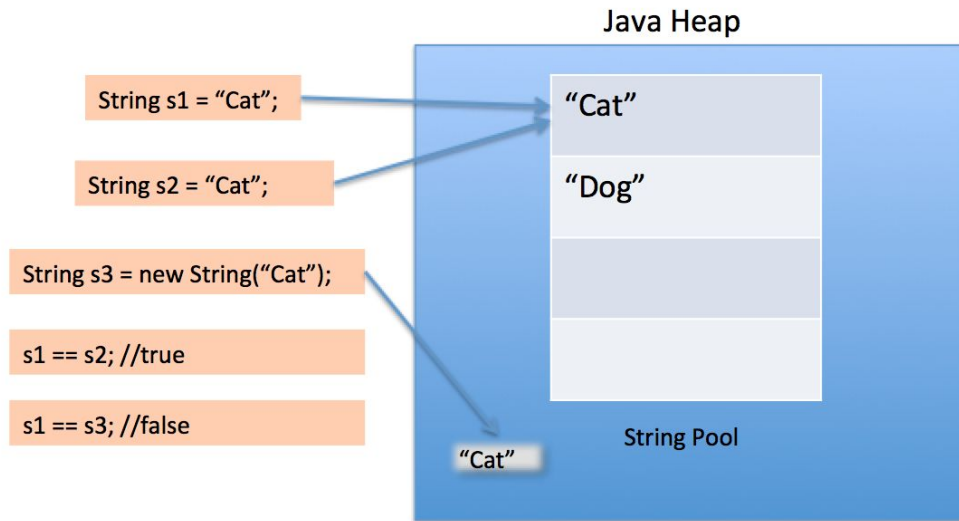
String str = "Tôi là coder";

//object

String str = new **String**("Tôi là coder");

Chuỗi (Strings)

Literal String vs String Object



CHÂN LÝ so sánh chuỗi: Sử dụng phương thức **equals**

- `s1.equals(s2); //true`
- `s1.equals(s3); //true`
- `s3.equals(s1); //true`

Chuỗi (Strings)

Một số phương thức cơ bản

Phương thức	Mô tả
<code>toUpperCase()</code>	Chuyển đổi chuỗi thành chữ hoa
<code>toLowerCase()</code>	Chuyển đổi chuỗi thành chữ thường
<code>trim()</code>	Xóa khoảng trắng ở đầu và cuối chuỗi
<code>length()</code>	Trả về độ dài của chuỗi
<code>equals()</code>	So sánh nội dung 2 chuỗi
<code>equalsIgnoreCase()</code>	So sánh nội dung 2 chuỗi nhưng không phân biệt chữ hoa, thường
<code>charAt()</code>	Lấy một ký tự tại vị trí index được chỉ định
<code>indexOf()</code>	Trả về index của ký tự được chỉ định xuất hiện đầu tiên
<code>substring()</code>	Trả về đối tượng chuỗi mới là chuỗi con của chuỗi đã cho tính từ <code>startIndex</code> đã nhập đến cuối cùng hoặc đến <code>endIndex</code>

Chuỗi (Strings)

Nối chuỗi

Sử dụng phép cộng

```
String firstName = "Tech";  
String lastName = "Master";  
String fullName = firstName + " " + lastName  
System.out.println(fullName); //Tech Master
```

Sử dụng phương thức concat

```
String firstName = "Tech";  
String lastName = "Master";  
System.out.println(firstName.concat("Master")); //TechMaster  
System.out.println(firstName.concat(" " + "Master")); //Tech Master
```

Chuỗi (Strings)

Ký tự đặc biệt

Nháy kép trong nháy kép

```
String text1 = "Voldemort được gọi là "kẻ là ai cũng biết là ai đấy""; //Lỗi
```

Vì dấu nháy kép dùng để xác định chuỗi nên khi đặt nháy kép trong một chuỗi thì java sẽ không hiểu.

Khắc phục bằng cách sử dụng ký tự **backslash** \ (gọi là **escape**)

```
String text1 = "Voldemort được gọi là \"kẻ là ai cũng biết là ai đấy\"";  
//OK
```

Vậy nếu trong chuỗi muốn dùng dấu backslash thì sao?

```
String text1 = "Dấu backslash \\ được sử dụng trong chuỗi"; //OK
```

Ký tự	Ý nghĩa
\"	Nháy kép
\'	Nháy đơn
\\	Dấu backslash

Hằng số (Constants)

Khái niệm

Hằng là một giá trị được khởi tạo và không thay đổi trong suốt thời gian chạy chương trình.

```
final <Kiểu dữ liệu> <TÊN_HẰNG> = <Giá trị>;
```

```
final double PI = 3.14;
```

```
final int WORKING_HOURS_A_DAY = 8;
```

```
PI = 3.15; //Lỗi
```

```
WORKING_HOURS_A_DAY = 9; //Lỗi
```


Ngày giờ (Date & Time)

LocalDate vs LocalTime vs LocalDateTime

Đối tượng	Mục đích	Khởi tạo	Phép toán cộng / trừ	So sánh
LocalDate	Lưu thông tin về ngày, tháng, năm	<code>LocalDate.now();</code> <code>LocalDate.of(2022, 4, 30);</code>	<code>plusYears();</code> <code>plusMonths();</code> <code>plusDays();</code> <code>minusYears();</code> <code>minusMonths();</code> <code>minusDays();</code>	<code>isEqual();</code> <code>isBefore();</code> <code>isAfter();</code> <code>compareTo();</code>
LocalTime	Lưu thông tin về giờ, phút giây	<code>LocalTime.now();</code> <code>LocalTime.of(10, 20, 30)</code>	<code>plusHours();</code> <code>plusMinutes();</code> <code>plusSeconds();</code> <code>minusHours();</code> <code>minusMinutes();</code> <code>minusSeconds();</code>	<code>isEqual();</code> <code>isBefore();</code> <code>isAfter();</code> <code>compareTo();</code>
LocalDateTime	LocalDate + LocalTime	<code>LocalDateTime.now()</code> <code>LocalDateTime.of(</code> <code>LocalDate.of(2022, 7, 10),</code> <code>LocalTime.of(17, 05, 10)</code> <code>);</code>	Tất cả các phương thức trên	<code>isEqual();</code> <code>isBefore();</code> <code>isAfter();</code> <code>compareTo();</code>

Ngày giờ (Date & Time)

Định dạng (format) hiển thị ngày, giờ

```
LocalDateTime now = LocalDateTime.now();  
System.out.println("Thời gian hiện tại: " + now);  
Kết quả:
```

Thời gian hiện tại: **2022-07-12T13:29:54.435878**

Mặc định khi in đối tượng LocalDateTime sẽ có format như sau:

năm-tháng-ngàyT**giờ:phút:giây**.nano-giây

Chú ý: chữ 'T' chỉ là ký tự để phân tách ngày và giờ.

Để in ra những định dạng khác nhau (ví dụ: ngày/tháng/năm) thì cần **DateTimeFormatter**.

```
LocalDateTime now = LocalDateTime.now();  
DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd/MM/yyyy HH:mm:ss");  
String currentTimeStr = now.format(formatter);  
System.out.println("Thời gian hiện tại: " + currentTimeStr);  
Kết quả:
```

Thời gian hiện tại: **12/07/2022 13:40:28**

Formatter được định nghĩa dựa trên **pattern**.

Giải thích pattern:

d = day (ngày), dd = in ra ngày định dạng 2 chữ số.
M = month (tháng), MM = in ra tháng định dạng 2 chữ số.
y = year (năm), yyyy = in ra năm có định dạng 4 chữ số

Giải thích pattern:

H = hour (giờ), HH = in ra giờ có định dạng 2 chữ số
m = minute (phút), mm = in ra phút có định dạng 2 chữ số
s = second (giây), ss = in ra giây có định dạng 2 chữ số

Ngày giờ (Date & Time)

Một số ký hiệu thông dụng

Xem trong giáo trình

Chuyển từ String sang DateTime

String str = "20/03/2022 00:10:30";

Đây là một chuỗi, có thông tin ngày giờ, vậy làm sao để chuyển sang được đối tượng DateTime?

```
LocalDateTime localDateTime = LocalDateTime.parse(  
    "20/03/2022 00:10:30",  
    DateTimeFormatter.ofPattern("dd/MM/yyyy HH:mm:ss")  
);  
System.out.println(localDateTime);
```

Chú ý:

Trong các dự án thực tế, các phương thức **format** và **parse** là các phương thức được sử dụng rất nhiều để hiển thị và tạo khởi tạo đối tượng DateTime.

Toán học (Math)

Java Math

Tích hợp sẵn trong java, chỉ việc gọi ra và sử dụng.

Các phương thức thông dụng:

- `Math.PI`: hằng số PI
- `Math.abs()`: trả về giá trị tuyệt đối của tham số
- `Math.ceil()`: trả về giá trị `double` là số làm tròn tăng bằng giá trị số nguyên gần nhất
- `Math.floor()`: trả về giá trị `double` là số làm tròn giảm
- `Math.max()`: trả về số lớn nhất trong hai số
- `Math.min()`: trả về số nhỏ nhất trong hai số
- `Math.pow(ơ số, số mũ)`: lấy lũy thừa
- `Math.sqrt()`: Khai căn
- `Math.sin()`, `Math.cos()` tính sin, cos của đơn vị góc
- `Math.random()`: sinh số `double` ngẫu nhiên từ 0 đến 1
- `Math.toDegrees()`: đổi góc radian thành độ
- `Math.toRadians()`: đổi góc đơn vị độ sang radian

Ví dụ:

```
System.out.println(Math.sqrt(9)); //3.0
```

Nhập dữ liệu với Scanner

Mục đích

Lớp **Scanner** của package **java.util** được sử dụng để để đọc dữ liệu đầu vào từ các nguồn khác nhau như từ bàn phím, đọc file,...

Trong phạm vi khóa học lần này ta sẽ dùng Scanner để đọc dữ liệu nhập từ console để phục vụ cho các bài học tiếp theo.

Cách sử dụng

```
Scanner scanner = new Scanner(System.in);
```

Khai báo đối tượng scanner, trong đó **System.in** là console mà scanner sẽ đọc dữ liệu.

```
System.out.println("Nhập tên : ");  
String name = scanner.nextLine();
```

nextLine() để đọc nội dung trên console và gán nội dung đó về kiểu String.

```
System.out.println("Nhập tuổi : ");  
int age = scanner.nextInt();
```

nextInt() để đọc nội dung trên console và gán nội dung đó về kiểu số nguyên (Integer).

```
System.out.println("Nhập email : ");  
String email = scanner.nextLine();
```

nextLine() để đọc nội dung trên console và gán nội dung đó về kiểu String.

```
scanner.close();  
System.out.println("--- Thông tin user ---");  
System.out.println("Họ : " + name);  
System.out.println("Tuổi : " + age);  
System.out.println("Email : " + email);
```

Dùng scanner xong nhớ gọi phương thức **close()**

Thực hành

Mục đích